

# CTE-STEM

5TH APSCE INTERNATIONAL  
CONFERENCE ON COMPUTATIONAL  
THINKING AND STEM EDUCATION

# 2021

**Conference  
Proceedings**  
**2nd - 4th June 2021**

Organised by:



Hosted by:



Supported by:

An Institute of



School of  
Computing



香港教育大學  
The Education University  
of Hong Kong

**Eco/Think @ JC >**  
賽馬會運算思維教育  
Inspiring digital creativity 激發數碼創意



# **Proceedings of Fifth APSCE International Conference on Computational Thinking and STEM Education 2021**

**2<sup>nd</sup> - 4<sup>th</sup> June 2021**

**Singapore**

**Organized by**

Asia-Pacific Society for Computers in Education

**Hosted by**

National Institute of Education

Nanyang Technological University, Singapore

Copyright 2021

All rights reserved

Publication of Asia Pacific Society for Computers in Education

ISSN 2664-5661



## *Preface*

The 5th APSCE International Conference on Computational Thinking and STEM Education 2021 (CTE-STEM 2021) is organized by the Asia-Pacific Society for Computers in Education (APSCE). CTE-STEM 2021 is hosted by the National Institute of Education, Nanyang Technological University (NIE/NTU). This conference continues from the success of the previous four international Computational Thinking conferences organised by the Education University of Hong Kong (EdUHK) and JC@Coolthink in Hong Kong. In addition to Computational Thinking, we will be expanding the conference to invite STEM researchers and practitioners to share their findings, processes and outcomes in the context of computing education or computational thinking.

CTE-STEM 2021 is a forum for worldwide sharing of ideas as well as dissemination of findings and outcomes on the implementation of computational thinking and STEM development. The conference will comprise keynote speeches, invited speeches, panel discussions, workshops and paper presentations. All accepted papers will be published in ISSN-coded proceedings.

The International Teachers Forum is organized for teaching practitioners to share their practices in teaching Computational Thinking, Computing and STEM in the classroom. We believe bringing all these would create enriching experiences for educators and researchers to share, learn and innovate approaches to learning through Computational Thinking and STEM education. This year, teachers can participate in Lightning Talks to share ideas about teaching and learning CT.

The Students Forum (BuildingBloCS) is organized by students, for students. It is Singapore's annual Computing education outreach programme. Started back in 2017, it is not only a national computing education outreach programme, but also a platform for leadership development, innovation programme, EVIA (Education & Values In Action) and student-friendly social network. We have been very encouraged by the strong support given by Ministry of Education (Singapore) and many other community and industry partners.

On behalf of APSCE and the Conference Organizing Committee, we would like to express our gratitude towards all speakers, panelists, as well as paper presenters for their contribution to the success of CTE-STEM 2021.

We sincerely hope everyone enjoys and get inspired from CTE-STEM 2021.

With Best Wishes,

Professor LOOI, Chee-Kit

*Conference Chair,  
CTE-STEM 2021  
National Institute of Education  
Nanyang Technological  
University, Singapore*

A/P WADHWA, Bimlesh

*Conference Co-Chair,  
CTE-STEM 2021  
National University of  
Singapore, Singapore*

Professor DAGIENÉ, Valentina

*Conference Co-Chair,  
CTE-STEM 2021  
Vilnius University, Lithuania*

## **Main Theme and Sub-themes**

**“Computational Thinking and STEM Education”** is the main theme of CTE-STEM 2021 which aims to keep abreast of the latest development of how to facilitate students’ computational thinking abilities and STEM development, in the context of computing education or computational thinking. The conference also aims to disseminate findings and outcomes on the implementation of CT development in school and STEM education. There are 19 sub-themes under CTE-STEM 2021, namely:

Computational Thinking and Coding Education in K-12

Computational Thinking and Unplugged Activities in K-12

Computational Thinking and Subject Learning and Teaching in K-12

Computational Thinking and Teacher Development

Computational Thinking and IoT

Computational Thinking and STEM/STEAM Education

Computational Thinking and Data Science

Computational Thinking and Artificial Intelligence Education

Computational Thinking Development in Higher Education

Computational Thinking and Special Education Needs

Computational Thinking and Evaluation

Computational Thinking and Non-formal Learning

Computational Thinking and Psychological Studies

Computational Thinking in Educational Policy

STEM Learning in the Classroom

STEM Activities in Informal Contexts

STEM Education Policies

STEM Pedagogies and Curriculum

STEM Teacher Education and Professional Development

### **Paper Submissions to CTE-STEM 2021**

The conference received a total of 47 submissions (29 full papers, 14 short papers and 4 poster papers) by 116 authors from 21 countries/regions (see Table 1)

*Table 1: Distribution of Paper Submissions for CTE-STEM 2021*

<b>Country/ Region</b>	<b>No. of Authors</b>	<b>Country/Region</b>	<b>No. of Authors</b>
Canada	4	Lithuania	2
China	19	Malaysia	5
Cyprus	1	Mexico	4
Estonia	1	Netherlands	1
Finland	4	Peru	2
Greece	2	Singapore	11
Germany	2	Spain	1
Hong Kong	14	Sweden	5
India	4	Taiwan	9
Italy	4	United States	18
Japan	3	Total	116

The International Programme Committee (IPC) is formed by 74 members and 13 co-chairs worldwide. Each paper with author identification anonymous was reviewed by at least three IPC Members or co-chairs. Meta-reviewers then made recommendation on the acceptance of papers based on IPC Members' reviews. With the comprehensive review process, 35 accepted papers are presented (10 full papers, 15 short papers and 10 poster papers) (see Table 2) at the conference.

*Table 2: Paper Presented at CTE-STEM 2021*

<b>Sub-themes</b>	<b>Full Paper</b>	<b>Short Paper</b>	<b>Poster Paper</b>	<b>Total</b>
Computational Thinking and Coding Education in K-12	2	1	2	5
Computational Thinking and Unplugged Activities in K-12	0	0	1	1
Computational Thinking and Subject Learning and Teaching in K-12	3	2	0	5
Computational Thinking and Teacher Development	1	1	0	2
Computational Thinking and IoT	0	0	0	0
Computational Thinking and STEM/STEAM Education	0	0	1	1
Computational Thinking and Data Science	0	0	2	2
Computational Thinking and Artificial Intelligence Education	0	0	0	0
Computational Thinking Development in Higher Education	1	2	1	4
Computational Thinking and Special Education Needs	0	1	0	1
Computational Thinking and Evaluation	1	0	0	1
Computational Thinking and Non-formal Learning	2	0	0	2
Computational Thinking and Psychological Studies	0	1	0	1
Computational Thinking in Educational Policy	0	0	0	0

Sub-themes	Full Paper	Short Paper	Poster Paper	Total
STEM Learning in the Classroom	0	3	0	3
STEM Activities in Informal Contexts	0	1	0	1
STEM Education Policies	0	1	0	1
STEM Pedagogies and Curriculum	0	2	1	3
STEM Teacher Education and Professional Development	0	0	2	2
Total	10	15	10	35

## **Editors**

Chee Kit LOOI

Nanyang Technological University

Bimlesh WADHWA

National University of Singapore

Valentina DAGIENĖ

Vilnius University

Peter SEOW

Nanyang Technological University

Ying Hwa KEE

Nanyang Technological University

Long Kai WU

Nanyang Technological University



# Table of Contents

## COMPUTATIONAL THINKING AND CODING EDUCATION IN K-12

### *Full Paper*

Exploring the Effectiveness of Pair Programming in Developing Students' Computational Thinking Skills through Scratch

Wee Meng Frankie LEOW, Wendy HUANG .....2

Achievement and Effort in Acquiring Computational Thinking Concepts: A Log- based Analysis in a Game-based Learning Environment

Shuhan ZHANG, Gary K.W. WONG, Peter C.F. CHAN .....8

### *Short Paper*

Cultivating Computational Thinking through Game-based Scratch Programming

Xiaoqian LI, Jing LI, Jiansheng LI .....14

### *Poster Paper*

Developing Girls' Computational Thinking by Playing Programming Games

Jing LI, Jiansheng LI .....18

Programming Socio-scientific Games: A Computational Thinking Approach to Real-world Problems

Marianthi GRIZIOTI, Chronis KYNIGOS .....20

## COMPUTATIONAL THINKING AND UNPLUGGED ACTIVITIES IN K-12

### *Poster Paper*

Research on the Design of Unplugged Computer Science Teaching Activities in Elementary School—Taking the Fruit Delivery Game Course as an Example

Bingqing YANG .....23

## COMPUTATIONAL THINKING AND SUBJECT LEARNING AND TEACHING IN K-12

### *Full Paper*

A Hybrid Approach to Teaching Computational Thinking at a K-1 and K-2 Level

Damien ROMPAPAS, Steven YOON, Jonothan CHAN .....26

Using the Beginners Computational Thinking Test to Measure Development on Computational Concepts Among Preschoolers

Maria ZAPATA-CÁCERES, Nardie FANCHAMPS .....32

Storytelling through Programming in Scratch: Interdisciplinary Integration in the Elementary English Language Arts Classroom

Emrah PEKTAŞ, Florence R. SULLIVAN .....38

### *Short Paper*

Students' Learning of Computational Thinking in Schools with Different Curriculum Approaches Including Individual Student Characteristics

Amelie LABUSCH, Birgit EICKELMANN .....43

A Standard Decomposition Process to Inform the Development of Game-Based Learning Environments Focused on Computational Thinking

Elizabeth L. ADAMS, Ching-Yu TSENG, Paul FOSTER, Vinson LUO, Leanne R. KETTERLIN-GELLER, Eric C. LARSON, and Corey CLARK .....47

## COMPUTATIONAL THINKING AND TEACHER DEVELOPMENT

### *Full Paper*

Different Paths, Same Direction: How Teachers Learn Computational Thinking in STEM Practices through Professional Development

Sally WU, Amanda PEEL, Connor BAIN, Michael HORN, Uri WILENSKY .....52

### *Short Paper*

An Experience of Conducting Online Teacher Development for Computational Thinking Teaching in a Primary School Context

Siu-Cheung KONG.....58

## COMPUTATIONAL THINKING AND STEM/STEAM EDUCATION

### *Poster Paper*

ARTEC Logic Puzzle: The Role of Computational Thinking with Extension to Extended Logic

Chung-Oi KOK.....63

## COMPUTATIONAL THINKING AND DATA SCIENCE

### *Poster Paper*

Infusing Computational Thinking into the Accounting Practice Course

Tao WU, Maiga CHANG .....66

VizBlocks: A Data Visualization Literacy Education Tool

TRAVIS Jia Yea CHING, Bimlesh WADHWA .....68

## COMPUTATIONAL THINKING DEVELOPMENT IN HIGHER EDUCATION

### *Full Paper*

Making the Thinking Results of Programming Visible and Traceable with a Multi-layer Board Game

YungYu ZHUANG, Andito SAPUTRO, Mahesh LIYANAWATTA, Jen-Hang WANG, Su-Hang YANG, Gwo-Dong CHEN .....71

### *Short Paper*

A Framework for Integrating Computational and Design Thinking Processes

Riccardo CHIANELLA, Diego REITANO, Ettore MORDENTI, George BARITSCH.....77

The Effects of an AR Programming Game on Students' Different Prior Computational Thinking Skills

Huai-Hsuan HUANG, Vandit SHARMA, Kaushal Kumar BHAGAT, Wen-Min HSIEH, Nian-Shing CHEN.. 81

### *Poster Paper*

A Systematic Review of Distributed Pair Programming Based on the Team Effectiveness Model

Fan XU, Ana-Paula CORREIA .....85

## COMPUTATIONAL THINKING AND SPECIAL EDUCATION NEEDS

### *Short Paper*

Proposal for the Production of Virtual Reality Environments in Elementary Education with a Constructivist Approach

José E. GUZMÁN-MENDOZA, Héctor CARDONA-REYES, M. Lorena BARBA-GONZÁLEZ, Klinge O. VILLALBA-CONDORI, Dennis ARIAS-CHAVEZ, M. Luisa Fernanda RÁBAGO-GONZÁLEZ .....88

## COMPUTATIONAL THINKING AND EVALUATION

### *Full Paper*

A Preliminary, Systematic Review of Teaching and Learning Computational Thinking in Early Childhood Education

Anika SAXENA, Gary WONG .....93

## COMPUTATIONAL THINKING AND NON-FORMAL LEARNING

### *Full Paper*

Bringing Physical Computing to an Underserved Community in an Informal Learning Space

Chin-Lee KER, Bimlesh WADHWA, Peter Sen-Kee SEOW, Chee-Kit LOOI.....101

Combining Maker Technologies to Promote Computational Thinking and Heart-ware skills through Project-based Activities: Design Considerations and Empirical Outputs

Ali HAMIDI, Sepideh TAVAJOH, Marcelo MILRAD .....107

## COMPUTATIONAL THINKING AND PSYCHOLOGICAL STUDIES

### *Short Paper*

Influential Factors of Hong Kong Secondary School Students' Intrinsic Motivation to Coding Education during the COVID-19 Epidemic: A Correlational Analysis

Xin ZHANG, Gary K.W. WONG, Qiaobing WU, Bill Y.P. TSANG.....114

## STEM LEARNING IN THE CLASSROOM

### *Short Paper*

An Evolving Definition of Computational Thinking in Science and Mathematics Classrooms

Amanda PEEL, Sugat DABHOLKAR, Sally WU, Michael HORN, Uri WILENSKY .....119

Action Research on Engineering Design-oriented and Project-based STEM Teaching Model

Hong YU, Lu ZOU .....123

A Case Study of 7<sup>th</sup> Grade Students Learning Programming to Solve Mathematics Problems

Wendy HUANG, Chee-Kit LOOI, Mi Song KIM.....127

## STEM ACTIVITIES IN INFORMAL CONTEXTS

### *Short Paper*

Developing STEM Makers with Mentoring and Authentic Problem-Solving Strategies

Xiaojing WENG, Thomas K.F. CHIU, Morris S.Y. JONG.....132

## STEM EDUCATION POLICIES

### *Short Paper*

Euro-Asia Collaboration for Enhancing STEM Education

Anders BERGLUND, Valentina DAGIENE, Mats DANIELS, Vladimiras DOLOGOPOLOVAS, Siegfried ROUVRAIS, Miriam TARDELL.....136

## **STEM PEDAGOGIES AND CURRICULUM**

### *Short Paper*

Designing an Interdisciplinary Social-scientific STEM Curriculum on Students' Empathy, Efficacy, and Interest

Biyun HUANG, Morris Siu-Yung JONG, Ching Sing CHAI, Yun DAI, Darwin LAU ..... 141

A Co-design Approach for Developing Computational Thinking Skills in Connection to STEM Related Curriculum in Swedish Schools

Rafael ZEREGA, Ali HAMIDI, Sepideh TAVAJOH, Marcelo MILRAD ..... 144

### *Poster Paper*

Analysis of the Development Direction of STEM Curriculum in China

Lihua PENG ..... 148

## **STEM TEACHER EDUCATION AND PROFESSIONAL DEVELOPMENT**

### *Poster Paper*

Teacher Sensemaking on Computational Thinking in a Community of Math Teachers

Chung Yiu SIU, Mi Song KIM, Wendy HUANG, Chee-Kit LOOI..... 151

A Systematic Review of Teachers' Preparedness towards Computational Thinking Integration in Mathematics

Shiau-Wei CHAN, Chee-Kit LOOI, Shivani MAHEDIRATA, Mi Song KIM ..... 153

# **Computational Thinking and Coding Education in K-12**

# Exploring the Effectiveness of Pair Programming in Developing Students' Computational Thinking Skills through Scratch

Wee Meng Frankie LEOW<sup>1</sup>, Wendy HUANG<sup>2</sup>

<sup>1</sup>Bedok Green Secondary School, Singapore

<sup>2</sup>National Institute of Education, Nanyang Technological University, Singapore

leow\_wee\_meng\_frankie@moe.edu.sg, wendy.huang@nie.edu.sg

## ABSTRACT

Pair programming (PP) is a useful strategy to promote computational thinking (CT) among students. Studies have shown that PP under appropriate conditions can enhance student achievement and increase their motivation in learning programming. Furthermore, studies have also shown that Scratch, a graphical block-based programming language, enables student learning in programming to become more interesting, more challenging and more creative. This study explored the effectiveness of PP in developing students' CT skills through Scratch in a Singapore secondary school. The findings suggest that PP is more effective than the solo programming, both in supporting and enhancing students' learning and understanding of basic programming concepts and CT skills, as well as on improving students' motivation toward programming. Limitations of this study and implications for teaching are also discussed.

## KEYWORDS

Pair Programming, Scratch, Computational Thinking, Computer Applications, K-12

## 1. INTRODUCTION

To nurture students to be future-ready and contribute effectively in an increasingly complex and interconnected world shaped by computer technologies, the Singapore Ministry of Education (MOE) has strengthened digital literacy among students through the Smart Nation Initiative (Smart Nation, 2014) and the National Digital Literacy Programme (MOE, 2020). As developing computational capabilities is one of the key enablers for these national initiatives, secondary schools and junior colleges computer education curriculum were also revised to introduce computational thinking (CT) and its related concepts such as abstraction, algorithmic thinking and decomposition to students through programming in subjects such as Computer Applications (CPA) and O-Level Computing (MOE, 2017, 2019). Secondary students who took CPA are introduced to programming at secondary two through Scratch 2.0 (Scratch), a graphical block-based programming language, using Scratch editor.

The secondary two CPA students in a typical public co-educational school (it is called "School A" in this paper) initially learned Scratch through solo programming. While students worked independently to complete the Scratch projects, the teachers observed that students struggled to correctly apply the knowledge they have learned to create the projects and got frustrated as a result when the codes did not work as intended. Students may know the function of each graphical block but they did not know how to combine those blocks in order to produce valid and correct

programs. Students also faced difficulties in the use of variables, operators blocks, event blocks and blocks that encapsulate other blocks (e.g. loops). For example, students commonly have misconceptions regarding variable initialisation and loop conditions during the creation of their scratch projects. Hence, despite the ease in using Scratch to learn programming, many students tend to find programming difficult to learn and get frustrated when they are unable to get their programs to work as intended (Choo et al., 2017; Rahmat et al., 2012).

To explore the effectiveness of PP in developing students' CT skills and in motivating students to learn programming through Scratch, the secondary two CPA students in School A attended three PP lessons. This study explored the effectiveness of pair programming (PP) in developing students' CT skills, measured by their learning achievement in PP. The study focused on answering the following research questions:

1. What is the effectiveness of PP in developing students' CT skills through Scratch?
2. How motivated are students to learn programming through Scratch when using PP?

## 2. LITERATURE REVIEW

### 2.1. Pair Programming

PP involves two people working side by side each other at one computer and collaborate closely to create a program. One acts as the driver who is responsible for controlling the shared resources (e.g., computer, mouse, keyboard) and actively involved in the programming task such as using the mouse to input the codes. The other acts as the navigator who is responsible for observing the driver's work and providing support by pointing out errors in the codes and/or offering suggestions on how to solve a problem (Williams & Kessler, 2002). During the program completion process, the driver and navigator roles are switched after a period of time (Williams & Kessler, 2002).

Studies have shown that students regularly perform better with PP than with solo programming in CT (Lye & Koh, 2014; Werner & Denning, 2009). Paired students were more likely to hand in solutions for their programming tasks that were of higher quality than students who programmed independently (McDowell et al., 2002). Furthermore, various studies have also shown that PP can

(1) improve individual programming skills (Braught, Eby, & Wahls, 2008; Cliburn, 2003) and (2) reduces frustration experienced by novice programmers, increases their satisfaction, enjoyment; and promote positive attitudes in programming in them (Bishop-Clark, Courte, Evans, & Howard, 2006; Preston, 2005).

## 2.2. Computational Thinking

During programming, students are exposed to CT. Wing (2017, p. 8) defines CT as the “thought processes involved in formulating a problem and expressing the solution(s) in ways that a computer—human or machine—can effectively carry out.” The “computer” here refers to an information processing agent that can be a human or a computer, or a combination of both. Selby (2014) further elaborates that CT as cognitive processes, involves thinking in abstractions, algorithmically and in terms of decomposition, generalization and evaluation. Binkley et al. (2012) and Yadav et al. (2014) also posit that CT has the potential to foster creativity and problem-solving skills among students. Hence, CT is not just about problem formulation, but also about problem solving where students are encouraged to think in new ways to come up with solutions. Therefore, CT equips and empowers the students with knowledge, skills and programming competencies to move beyond being consumers of technology to becoming creative thinkers and problem-solvers in a tech-driven world.

## 3. METHODS

### 3.1. Participants

The participants were 40 secondary two CPA students in School A. They were introduced to Scratch prior to the PP lessons and had some basic knowledge and skills about Scratch programming. 12 students were female and 28 students were male.

### 3.2. The Learning Platform: Scratch

Scratch is a graphical block-based programming language suitable for students to learn programming because of its low floor (easy for novice programmers to get started), high ceiling (opportunities for expert programmers to create complex projects) and wide walls (supporting different types of projects that grow out of the programmers’ own interests and learning profiles) (Resnick et al., 2009). Studies have shown that using Scratch improves students’ motivation in learning programming (Ouahbi et al., 2015) and understanding of basic programming concepts (Saez-Lopez et al., 2016).

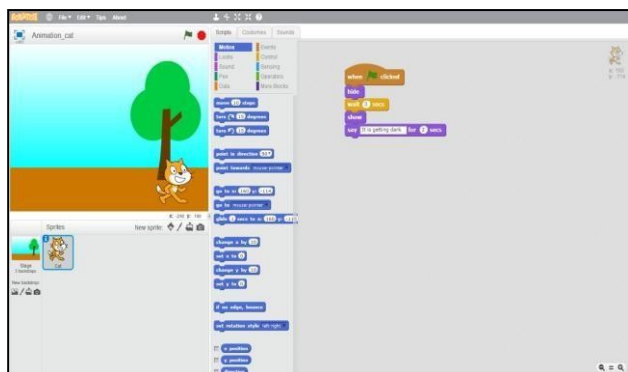


Figure 1. Scratch user interface

Writing a program is done by dragging and dropping the graphical Scratch blocks to connect them to each other vertically. These blocks are color-coded and grouped into different categories based on their functions (e.g., event blocks, control blocks), thereby allowing programmers to see the relationship between the different blocks easily.

Accordingly, students can create programs, which in Scratch are called projects, such as stories, animations, games, simulations, songs, etc. by connecting the blocks in the correct sequence. Figure 1 shows the Scratch user interface while Figure 2 shows an example of a program written using Scratch blocks.

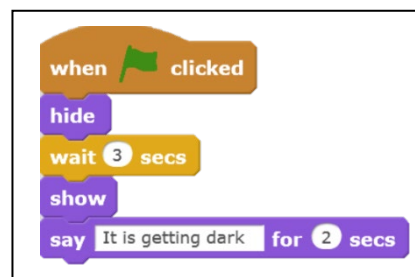


Figure 2. Example of a program written using Scratch blocks

### 3.3. Procedure

Prior to the intervention, students completed a solo programming project over one hour and 30 minutes. Thereafter, they attended three PP lessons (four hours thirty minutes in total). In each PP lesson, students shared one computer to work through the scenario, design and develop their Scratch project, with one driving (controlling the mouse and keyboard) and the other navigating (checking for errors and bugs, and providing support and feedback). The pairs must switch their roles every 10 minutes during PP.

Pairs were assigned based on student choice. All students chose a same-gender classmate to work with for all the three PP lessons. There was a total of 6 pairs of girls and 14 pairs of boys. However, for each subsequent lesson, every student was required to choose a new partner.

After students reviewed earlier lessons on Scratch programming, they were introduced to the Scratch project that they need to complete and the rubrics for the project as well as PP and the accompanying PP expectations. Table 1 further shows a summary of the activities for each PP lesson.

### 3.4. The Scratch Programming Projects

Over the three classes, students were given two programming projects to assess their programming knowledge and capability during PP. They consisted of students’ choice of two semi-open projects with a defined outcome and an undefined process (see Table 2) and were to be completed by the paired students within lesson one (for PP project 1) and within lessons two and three (for PP project 2).

### 3.5. Data collection

In this paper, data was collected during PP by observing students’ behaviors and interactions (including the questions asked by students when seeking help, frequency of seeking help from teachers, and verbatim comments by students during PP) as they designed, coded and implemented their Scratch projects. We observed how students applied CT skills such as evaluation when they encountered bugs and algorithmic thinking when conceptualising and implementing the projects. We also examined these projects based on the rubrics and compared

their scores with the Scratch projects done earlier through solo programming.

Table 1. Sample PP lesson

Lesson	Time/min	Description of lesson activities
1	5	Revision of last Scratch lesson's concepts.
	5	Students are introduced to PP (includes the showing of PP video in lesson 1. But the showing of PP video will not be implemented in lessons 2 and 3) or reminded of PP expectations in lessons 2 and 3.
	5	Students get into pairs (each pair will need to have a different partner for each lesson) and are introduced to the different scenarios for the Scratch project that they need to complete within the lesson (includes implementation details and rubrics for this project).
	10	Each pair decides on their preferred task scenario for the project. Thereafter, the paired students will prepare the script and storyboard for their Scratch project.
	60	The paired students carry out PP to complete their project and will switch roles after every 10 minutes.
	5	Summary of the concepts learned in the lesson.

Table 2. Overview of the pair programming project

Type	Project
Solo	Solo Project: Two sprites having a conversation at the basketball court, with one sprite introducing himself/herself to and having a conversation with the other sprite to get to know him/her better.
PP	PP Project 1: Choose 1 out of 3 scenarios Animation 1: Two sprites having a conversation, with one sprite sharing a riddle with his/her friend. Animation 2: Two sprites having a conversation, with one sprite sharing his/her favourite Korean drama and why he/she likes this Korean drama to his/her friend. Animation 3: Two sprites having a conversation, with one sprite sharing his/her favourite game that he/she plays with his/her friend.
	PP Project 2: Choose 1 out of 3 scenarios Game 1: A cat appears and it is supposed to catch doughnuts as they fall from the sky. Game 2: A cat appears and it is supposed to jump

onto blocks to collect stars.

Game 3: A mouse appears at the start of the maze and it is supposed to find the cheese. A cat will forever chase after the mouse.

## 4. FINDINGS AND DISCUSSIONS

### 4.1. Comparison of students' scores for solo programming and PP

To evaluate the effectiveness of PP in developing students' CT skills, measured by their learning achievement in PP, paired samples t-tests were conducted to determine whether the mean of students' scores for PP project 1 ( $M=9.48$ ,  $SD=7.542$ ) and PP project 2 ( $M=6.65$ ,  $SD=7.150$ ) significantly differed from the mean of students' scores for solo programming task ( $M=5.18$ ,  $SD=7.542$ ).

A pair programming session is considered effective to enhance students' performance if their mean score for either PP projects 1 or 2 is higher than their score for a similar solo programming project and the improvement is statistically significant. In general, students working in pairs performed better compared to programming alone as both the mean scores for the PP projects were higher than the mean score for the solo programming project.

The results for the paired t-tests indicated that the difference between PP project 1 and solo programming project was significant,  $t(39)=-3.61$ ,  $p<.001$ . Therefore, this could mean that PP may positively affect the students' learning performance.

However, results for the paired t-tests showed that the difference between PP project 2 and solo programming project was not significant,  $t(39)=-1.30$ ,  $p>0.001$ . It may be caused by three possible reasons.

Firstly, it may be due to the increasing difficulty on PP project 2, which was a Scratch game in contrast to a Scratch animation in PP project 1. Studies have shown that task complexity influences the effectiveness of PP and in turn, student learning (Hannay et al., 2010).

Secondly, it may be due to a change in partners in PP project 2. Factors that had been identified to influence the effects of PP include partners' personalities and temperaments (Hannay et al., 2010; Katira et al., 2004); and social factors such as gender, partnership and culture (Zhong, Wang, & Chen, 2016). As students had to change partners, this meant that they may be paired with a less desirable partner and therefore, having compatibility of pairs issues and resulted in lower motivation to persevere and complete the project. In this case, as the pairs were of the same gender, the social factor that is likely to contribute to the insignificant difference between PP project 2 and solo programming project is partnership between the pairs being affected by the partners' personalities and temperaments.

Thirdly, it may be due to the partner's skills, knowledge and experiences (Hannay et al., 2010; Lui & Chan, 2006). For example, if a low progress student is paired with another low progress student, the improvement in the learning achievement for both students may not be as



greater as the learning achievement of a pair that consists of a high progress student and a low progress student. In the latter, the high progress student will gain more knowledge and competencies in the CT skills since each time he/she teaches, he/she re-learns the materials while the low progress student will benefit from peer teaching. Therefore, PP can be beneficial even when partners bring different levels of prior programming experience, but the improvement in learning achievement may not always be the same for both partners. This suggests that when students work with a partner who has relatively more experience, they can still learn.

It can be therefore stated that PP may positively affect the students' academic performance. Results of these analyses are shown in Table 3.

Table 3. Results of paired t-tests for the different tasks

Comparisons between tasks	Mean difference	t	df	Sig
Solo project vs PP project 1	4.30	-3.61	39	0.0087
Solo project vs PP project 2	1.48	-1.30	39	0.1997

#### 4.2. Teachers' Observations of Students' Behaviors and Interactions during PP

We observed three categories of pair behaviors during the completion of the projects: collaborative, exploratory and off-task. Pairs engaged collaboratively when they interact verbally and non-verbally to share their thoughts and ideas during the creation of their projects, and willingly switches roles after each 10 minutes interval. Exploratory behavior

goes beyond students engaging collaboratively. Pairs constructively challenge each other's thoughts, ideas and programming decisions. On the other hand, off-task behavior involves pairs or individual student within the pairs being disengaged and holds up the programming process. For example, pairs engage in verbal or non-verbal exchanges not about their Scratch project or programming. Further description and examples are shown in Table 4.

Table 4. Pair behaviors during game interaction

Category	Description	Example
Collaborative	Partner gives and receives suggestions, ask questions and responds by carrying out the suggestions. Switches roles willingly.	Driver adds certain blocks of codes; Navigator offers comments that identify errors; Driver makes the changes.
Exploratory	Pair listens and engages constructively around suggestions. Verbalises reasons	Navigator spots errors and offers suggestions; Driver disagrees and navigator explains

	and reflect.	his/her reasoning. Driver becomes convinced and makes the changes, then test the game/animation.
Off-task	Partner makes decisions that disregard/dismisses the other partner's input, without any explanation. Or driver is programming while Navigator is not tracking what is happening on monitor (e.g., leave the computer station).	One partner insists the other change the sequence of codes and no reason was given. The other ignores partner.

Majority of the pairs did not engage in planning during the 10 minutes designated for the planning of the script and drawing of the storyboard. Instead, they engaged directly with Scratch Editor to plan and input the Scratch blocks for their projects. To further explore what PP looks like, we analysed the distribution of specific pair behaviors by gender pairs that were happening most of the time during the creation of the two PP projects. While collaborative behavior was the most common across pairs; a few pairs spent their time in off-task or exploratory behavior. The results are shown in Table 5.

Table 5. Distribution of pair behaviors by gender pairs most of the time while completing PP projects 1 and 2

Category of pair behavior	Gender pair	
	Girl-girl	Boy-boy
Collaborative	4	10
Exploratory	1	2
Off-task	1	2

We observed that during PP, students took the initiative to ask the teachers questions on whether their suggested codes are workable or whether their sequence of algorithmic thinking or of decomposition is correct. This contrasted with solo programming when more students either gave up or asked the teachers what are the codes to input in order to complete the solo project.

Furthermore, while most pairs spent most of their time in collaborative behavior, we observed that most female pairs spent proportionally more time on collaborative behavior and a smaller proportion of their time in exploratory behavior, while some male pairs spent a greater proportion of their time in exploratory and off-task behaviors. This suggests that gender can be an issue in PP context. Although studies showed that males tend to be more

assertive in their views and focus on independence (Leaper & Smith, 2004); and females try to avoid conflict and seek support, consensus and suggestions (Sullivan et. al., 2015), the issue of gender in PP in secondary school context needs further exploration.

The findings showed that students behaviors and interactions varied across pairs, and differences could be due to the level of confidence (either individually or as a pair) in completing the projects based on their knowledge and skills in Scratch programming. Furthermore, the findings also showed that majority of the students were more motivated to learn and engage in programming through Scratch while doing PP. Overall, the result of this paper is consistent with other studies that PP could reduce frustration experienced, enhance student enjoyment, and promote positive attitudes in programming ((Bishop-Clark et al., 2006; McDowell et al., 2002; Preston, 2005).

## 5. LIMITATIONS

The findings in this paper are limited in several ways. First, we did not measure the quality of the relationship between partners as a factor affecting the students' behaviors and interactions during PP. Studies have shown that one partner can dominate the interactions (Deitrick, Shapiro, & Gravel, 2016). Second, we did not measure the class collaborative culture and the extent to which collaboration supported PP. Future work involving rich observational data could help describe the classroom culture regarding collaboration. Third, we did not have mixed gender pairing of students of which may have yielded additional insight into pair behaviors. Lastly, we did not investigate the time factor: period of switching roles. The period of switching roles in this study was a fixed time interval of 10 minutes. We did not investigate whether if fixing a longer time interval of 15 to 20 minutes or having pairs switched their roles according to their own needs as and when they chose, would help in the learning of CT skills and achievement. This would provide additional insights on the effect of period of switching roles in PP on student learning.

## 6. CONCLUSION AND IMPLICATIONS FOR TEACHING

Overall, our findings suggest that students who programmed with a partner learned more than when they programmed alone. PP also seemed to motivate students to acquire CT skills. Hence, our finding supports prior studies that show the benefits of PP for learning and provide some detail on the factors that relate to those benefits.

The findings in this paper also have implications for teaching. Firstly, the findings can help teachers understand what PP looks like in a secondary school classroom and the different variability in how pairs interact. Therefore, teachers must plan to create effective pairs. When pairs possess different levels of experience of programming knowledge and skills, both students will benefit, but in different ways. However, it is disadvantageous to pair students possessing very different attitudes toward collaboration together. For example, having a partner who prefers to programme alone can undermine the more collaborative student's learning and lead to pair behaviors

that hold up the progress of their Scratch project completion.

Future research can examine the period of switching roles between the driver and the navigator and how this impacts the learning of CT skills and motivation in learning programming through Scratch. Additional research is needed in order to determine the extent to which the quality of the relationship between partners affects the students' behaviors and interactions during PP, and in turn their learning achievement in programming.

## 7. DECLARATION OF CONFLICTING INTERESTS

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

## 8. REFERENCES

- Binkley, M., Erstad, O., Herman, J., Raizen, S., Ripley, M., Miller-Ricci, M., & Rumble, M. (2012). Defining twenty-first century skills. In P. Griffin, B. McGaw, & E. Care (Eds.), *Assessment and teaching of 21st century skills* (pp. 17-66). Dordrecht, Netherlands: Springer.
- Bishop-Clark, C., Courte, J., & Howard, E. V. (2006). Programming in pairs with Alice to improve confidence, enjoyment, and achievement. *Journal of Educational Computing Research*, 34(2), 213-228.
- Braught, G., Eby, L. M., & Wahls, T. (2008). The Effects of pair-programming on individual programming skill. *ACM SIGCSE Bulletin*, 40(1), 200-204.
- Choo, G. K., Leow, W. M. F., Kaur, S., Yee, W. L. C. (2017, October 31). *Nurturing independent learners through teaching debugging* [Seminar session]. Computing Teachers Seminar 2017, Singapore.
- Cliburn, D. C. (2003). Experiences with pair programming at a small college. *Journal of Computing Sciences in Colleges*, 19(1), 20-29.
- Deitrick, E., Shapiro, R. B., & Gravel, B. (2016). *How do we assess equity in programming pairs?* Singapore: International Society of the Learning Sciences.
- Hannay, J. E., Arisholm, E., Engvik, H., & Sjøberg, D. I. (2010). Effects of personality on pair programming. *IEEE Transactions on Software Engineering*, 36(1), 61-80.
- Katira, N., Williams, L., Wiebe, E., Miller, C., Balik, S., & Gehringer, E. (2004). On understanding compatibility of student pair programmers. *ACM SIGCSE Bulletin*, 36(1), 7-11.
- Leaper, C., & Smith, T. E. (2004). A meta-analytic review of gender variations in children's language use: Talkativeness, affiliative speech, and assertive speech. *Developmental Psychology*, 40(6), 993.
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51-61.
- Lui, K. M., & Chan, K. C. (2006). Pair programming productivity: Novice-novice vs. expert-expert.

- International Journal of Human-computer studies*, 64(9), 915-925.
- McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2002). The effects of pair programming on performance in an introductory programming course. *Proceedings of the Thirty-Third Technical Symposium on Computer Science Education*, (pp. 38-42). ACM Press.
- MOE. (2020, March 4). *Learn for Life – Ready for the Future: Refreshing Our Curriculum and Skillsfuture for Educators* [Press release]. <https://www.moe.gov.sg/news/press-releases/learn-for-life-ready-for-the-future--refreshing-our-curriculum-and-skillsfuture-for-educators>
- MOE. (2017). *O-Level Computing Syllabus*. Retrieved December 20, 2020, from <https://www.moe.gov.sg/docs/default-source/document/education/syllabuses/sciences/files/o-level-computing-teaching-and-learning-syllabus.pdf>
- MOE. (2019). *N-Level Computer Applications Syllabus*. Retrieved December 20, 2020, from <https://www.moe.gov.sg/docs/default-source/document/education/syllabuses/sciences/files/2019-computer-applications-syllabus.pdf>
- Ouahbi, I., Kaddari, F., Darhmaoui, H., Elachqar, A., & Lahmine, S. (2015). Learning basic programming concepts by creating games with scratch programming environment. *Procedia-Social and Behavioral Sciences*, 191, 1479–1482.
- Preston, D. (2005). Pair programming as a model of collaborative learning: A Review of the research. *Journal of Computing Sciences in colleges*, 20(4), 39-45.
- Rahmat, M., Shahrani, S., Latih, R., Yatim, N. F. M., Zainal, N. F. A., & Rahman, R. A. (2012). Major problems in basic programming that influence student performance. *Procedia– Social and Behavioral Sciences*, 59, 287-296.
- Resnick, M., Maloney, J., Monroy-Hernandez, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., & Kafai, Y. (2009). Scratch: Programming for all. *Communications of the ACM*, 52(11), 60-67.
- Saez-Lopez, J. M., Román-González, M., & Vázquez- Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two year case study using scratch in five schools. *Computers & Education*, 97, 129–141.
- Selby, C. (2014). *How can the Teaching of Programming be Used to Enhance Computational Thinking Skills?* The United Kingdom: University of Southampton.
- Smart Nation. (2014). *Why Smart Nation*. Retrieved December 07, 2020, from <https://www.smartnation.sg/about-smart-nation>
- Sullivan, F. R., Kapur, M., Madden, S., & Shipe, S. (2015). Exploring the role of gendered discourse styles in online science discussions. *International Journal of Science Education*, 37(3), 484–504.
- Werner, L., & Denning, J. (2009). Pair programming in middle school: What does it look like? *Journal of Research on Technology in Education*, 42(1), 29-49.
- Williams, L. A., & Kessler, R. R. (2002). *Pair programming illuminated*. Boston, MA: Addison-Wesley Longman Publishing Company.
- Wing, J.M. (2017). Computational thinking's influence on research and education for all. *Italian Journal of Educational Technology*, 25(2), 7-14.
- Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational Thinking in Elementary and Secondary Teacher Education. *ACM Transactions on Computing Education*, 14(1), 1–16.
- Zhong, B., Wang, Q., & Chen, J. (2016). The Impact of social factors on pair programming in a primary school. *Computers in Human Behavior*, 64, 423-431.

# Achievement and Effort in Acquiring Computational Thinking Concepts: A Log-based Analysis in a Game-based Learning Environment

Shuhan ZHANG<sup>1\*</sup>, Gary K. W. WONG<sup>2</sup>, Peter C. F. CHAN<sup>3</sup>

<sup>1,2</sup> Faculty of Education, The University of Hong Kong, Hong Kong

<sup>3</sup> NetDragon, Hong Kong

shuhan@connect.hku.hk, wongkgw@hku.hk, peterchan@edmodo.com

## ABSTRACT

Numerous attempts have been made to apply coding games in computational thinking (CT) education, and using log data to explore CT learning is an emerging field. This paper explored the acquirement of CT concepts (sequences, loops, and conditionals) by primary and secondary school students who used a digital coding game called *Coding Galaxy*. It aims to investigate (1) whether secondary school students outperform primary school students, and (2) whether playing easy game missions is a scaffold for completing hard missions. Participants (N=188) were sampled from local schools in Hong Kong. Students were divided into three groups (A, B, C). Primary school students constituted Group A and B, while Group C consisted of secondary school students. Group A was assigned with only hard missions while easy missions were locked, whereas Group B and C were given access to both easy and hard missions. Data were extracted from students' log files, and 6599 records were analyzed using learning analytics techniques. Students' performance was evaluated based on game achievements and the effort they made to get the achievement. The results indicate that (1) students performed best in sequences, followed by loops and conditionals; (2) While secondary students shared the same pattern with primary students regarding the difficulty of acquiring CT concepts, secondary students performed better; and (3) While Group A shared similar game achievements with Group B, Group B made less effort in getting the achievements, indicating that easy missions can scaffold hard missions. The implications of the findings to various educational stakeholders are discussed.

## KEYWORDS

Computational thinking, K-12 education, game-based learning, log data, learning analytics

## 1. INTRODUCTION

Computational thinking (CT) has become a heated topic since 2006 when Jeanette Wing proposed the term as “an approach to solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science” (Wing, 2006, p. 33). Later in 2014, Wing further gave a more descriptive definition, stating that CT involves “formulating a problem and expressing its solution(s) in such a way that a computer-human or machine-can effectively carry out” (Wing, 2014, p.1). Wing's call for the importance of CT has aroused great effort in incorporating CT into educational practices (Martins-Pacheco et al., 2019), and programming education has become the main context for CT development (Grover & Pea, 2013).

Programming for young children was originated from the term “Constructionism” (Papert, 1980) which argues that students build knowledge more effectively when they actively engage in creating their own projects. Papert developed a constructionist programming environment, the *LOGO* programming tool, to provide a place where students can represent their abstract ideas through concrete constructions (Papert 1980). With the popularity of CT education, programming tools have become the vehicle for numerous initiatives developed for supporting CT education, among which visual programming tools, represented by Scratch (Resnick et al., 2009), have widely applied for its low complexities in programming syntax (Zhao & Shute, 2019).

CT learning environment can be categorized regarding its programming language and the nature of the task it displays (Manske et al., 2019). Regrading programming language, they can be classified into *text-based* programming tools, *block-based visual* programming tools, and *arrow-based visual* programming tools (Manske et al., 2019; Moreno-León, 2018). While text-based tools support users to create programs in textual programming languages, block-based programming platforms share the features of “low floor” (easy to begin with) and “high ceiling” (allow complex projects) (Grover & Pea, 2013). Further, to support younger children to engage in programming activities, arrow-based programming environment, represented by Scratch Jr (Bers & Resnick, 2015), was created, where representations that are analogous to objects (eg. arrows) are used (Moreno-León, 2018; Manske et al., 2019). As for the nature of the task, CT learning environments can be classified into *open task* environments and *goal-oriented* environments (Manske et al., 2019). In open task environments (eg. Scratch), users can author the design of their projects, with the flexibility of creating their own storyline, whereas goal-oriented platforms, represented by digital games, impose constraints on learning progression, providing explicit tasks for learners to complete. (Manske et al., 2019).

For students, CT learning environments offer a playground to practice CT skills (Lockwood & Mooney, 2017), whereas for teachers, these tools provide a way to measure students' learning progression (Shute et al., 2017). Students' acquisition of CT concepts and skills can be measured through evaluating their programming projects, from which different levels of performance can be indicated (Tang et al., 2020). Yet there are some main concerns of this approach--the *absence* of an element does not necessarily indicate that the students lack the knowledge, while the *presence* of a code construct is not always an accurate indicator of how much the students

grasp the concept (Kurland et al., 1985; Brennan & Resnick, 2012). To tackle these challenges, digital games can serve as an effective tool to assess concept acquisition. As a goal-oriented learning tool, CT games are designed with tasks that cover certain CT concepts, and learners' knowledge can be assessed through evaluating their performance in solving the task.

To ensure the CT games can effectively support learning, appropriate instructional design is critical. Instructional design of a game refers to how the game affords players' learning and playing (Laporte & Zaman, 2018), of which one important dimension is the organization of learning tasks, represented by the sequencing of the tasks (Merriënboer & Kirschner, 2017). Thus, for the design of CT games, it is vital to consider the sequencing of displaying tasks of different CT concepts and the sequencing of implementing different knowledge points for each concept. Although there have been numerous attempts in exploring the content taught by CT games, limited is known about how the concepts are delivered via game tasks (Laporte & Zaman, 2018), and studies focusing on the sequencing of CT concepts and knowledge points within concepts are still scarce.

This paper will introduce a case study on K-12 students using a coding game to learn CT concepts. It aims to explore the sequencing of concept acquisition and knowledge points within a concept. As this is the first paper focusing on this particular coding tool, we start by investigating the three fundamental CT concepts, which are *sequences*, *loops*, and *conditionals*. The case study takes place in a self-regulated learning context where students were assigned game tasks to complete at home during the COVID-19 pandemic, involving both primary and secondary students. Students' knowledge acquisition of CT concepts was assessed based on game performance, and the results of different cohorts were compared. The study aims to answer the following research questions:

1. How does students' game performance characterize the difficulty of acquiring CT concepts (sequences, loops, and conditionals)?
2. Do primary and secondary students share the same order of difficulty of acquiring CT concepts?
3. Is completing easy missions a scaffold for completing hard missions?

## 2. METHOD

### 2.1. Sample

Participants were selected from local schools in Hong Kong. A total of 188 students consented to participate in this study, with 101 from Grade 6 in primary school (age 10-12) and 87 from Grade 2 in secondary school (age 12-

14). According to the school curriculum, these groups of students have learned the basic CT concepts at school, so they were expected to be able to play the coding game under a self-regulated learning context.

### 2.2. Apparatus

The game adopted by this study, *Coding Galaxy* (CG hereafter), is designed based on an *arrow-based visual*

programming language where arrows are used as commands for players to manipulate directly. This context is developmentally appropriate for novice learners, because it could prevent syntax errors and have no requirement on children's reading skills (Bers, 2018). Each mission is a puzzle in which the learner can control the character (an astronaut) to solve the puzzle using simple visual programming language. In doing so, the learner must identify viable routes and use available commands to work out the solution (See Figure 1). Additionally, the learners are encouraged to use the fewest commands for the solution in order to obtain the mission reward.



Figure 1. Coding Galaxy Puzzle Map.

The mission reward is presented as one, two, or three stars upon finishing a mission. Three stars are awarded for the optimal solution to the puzzle, involving correct identification of patterns and accurate use of commands, to achieve the destination with the fewest commands while collecting all crystals. Two stars are awarded for partially fulfilling these criteria. One star is awarded for those who only solve the puzzle but fail to fulfill other criteria. Also, there is no limit on time spent on each task, and multiple attempts are allowed for each mission.

### 2.3 Research Design

Participants were divided into three groups (see Table 1). Primary school students constituted Group A and B, while Group C consisted of secondary school students. All the students were assigned game chapters of sequences, loops, and conditionals. Group A was assigned with only hard missions while easy missions were locked on the platform, whereas Group B and C were given access to both easy and hard missions. All students were given two weeks to complete the tasks. Table 2 illustrate the design of game missions in terms of knowledge points and the mapping with easy and hard missions respectively.

Table 1. Information of Each Group.

	Grade	Task
Group A (n=50)	Primary school	Hard missions
Group B (n=51)	Primary school	Easy missions, hard missions
Group C (n=87)	Secondary school	Easy missions, hard missions



Table 2. Map of Game Missions and Knowledge Points

CT Concept	Knowledge point	Description	Easy*	Hard*
Sequences	Simple sequence	Sequence with fewer than 10 commands	√	
	Relative position	Basic spatial awareness, tracking positions to move with commands	√	√
	Relative direction	Basic spatial awareness, imagining relative directions from the view of the character	√	√
	Complex sequence	Sequence with more than 10 commands		√
Loops	Apply preset loops	A loop has already been completed in the solution, need to put it in with other commands to complete the whole solution.	√	
	Loop preset commands	Some commands are already in the incomplete solution. Complete the loop by setting the loop time or inserting new commands.	√	
	Loop one command	Solution contains loop with 1 command	√	√
	Loop multiple commands	Solution contains loop with more than 1 command		√
	Nested loops	Solution contains loop within loop		√
Conditionals	Add action under condition	The condition has been preset, only need to add action through inserting commands	√	
	Create conditional command	Create conditional commands through selecting right conditions and inserting related commands		√

\*easy: easy missions, assigned to Group B and C    \*hard: hard missions, assigned to all groups

## 2.4 Data Analysis

Two variables for assessing student knowledge acquisition were defined and used for analysis, namely *achievement*, and *effort*. *Achievement* refers to the average stars student get in each game mission, and *effort* is reflected by the number of attempts before achieving the highest number of stars for each mission. As completion of each mission is rewarded with three possibilities of number of stars, *effort* is presented in three dimensions, namely, 1-, 2-, and 3- star attempts. To be more specific, if the highest achievement a player reached in a mission is two stars, which is the third attempt for trying, then the value of *effort* for this mission is “2-star attempts equal to 3”.

Log files of each participant were extracted from the game backstage, after which the dataset was processed based on the defined variables. While *achievement* was calculated with math formulas, data for *effort* was extracted with a Python script.

## 3. RESULT AND DISCUSSION

### 3.1 RQ1: How do students' game performance characterize the difficulty of acquiring CT concepts?

A total of 6599 records were extracted. Students' performance of each concept was compared. As for achievement, students got the highest stars in sequences, followed by loops and conditionals (see Figure 2), indicating a growing difficulty level of the three concepts. Yet regarding effort, the trend was mixed for three dimensions (see Figure 3). For 1-star attempts, the same order of difficulty was found, whereas the results of 3-star attempts showed that players used the most attempts to get the best solution in loops missions.

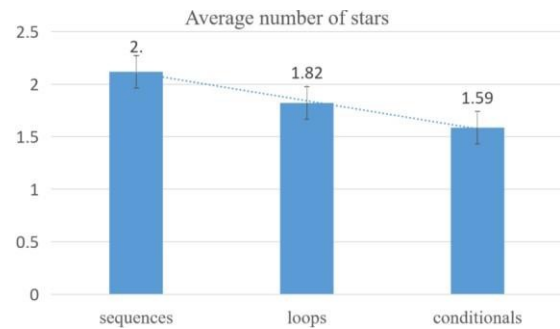


Figure 2. Achievement of Each Concept.

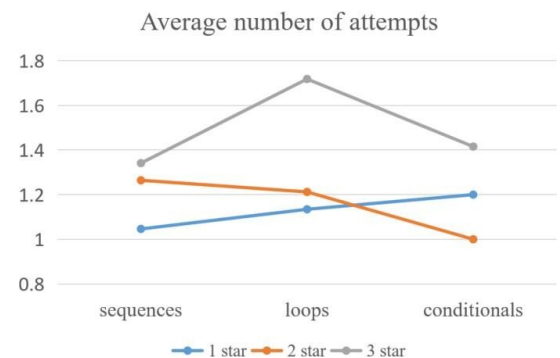


Figure 3. Effort of Each Concept.

These findings indicated that achievement and effort reflected different trends in terms of the order of difficulty of the three concepts. This may be explained by the order students follow when they play the game. Since the game missions are displayed in chapters, with each chapter focusing on one CT concept, most students played the game following the order of chapters, which is sequences, loops, and conditionals, according to the timestamp from the log files. Thus, it is likely that playing sequences and

loops familiarized them with how to solve the puzzles, which can be a scaffold for playing conditionals chapters afterward.

Further, both achievement and effort indicated that sequences is the easiest to learn while loops is comparatively difficult. This result is in line with the findings reported by Israel-Fishelson & HersHKovitz (2019) who used another coding game on primary school students. Based on indicators of the concept achievement and the number of attempts, they demonstrated that sequences was generally easiest for students while loops tended to be challenging. This provides some implications for educational practitioners who intend to teach programming to novices. It is recommended to start with introducing the concept of sequences, and more room for practice can be provided when teaching loops and conditionals.

### 3.2 RQ2: Do primary and secondary students share the same order of difficulty of acquiring CT concepts?

As Group B and C were assigned with the same game tasks, the performance of the tasks generated from the groups were compared regarding their achievement in the game (see Figure 4). Results showed that primary students (Group B) shared the same order of difficulty of concept acquisition with secondary students (Group C), with sequences as the easiest concept, followed by loops and conditionals. Moreover, secondary students outperformed primary students in all three concepts, with secondary students getting more than two stars on average for each concept, implying that the design of the arrow-based programming language may be too easy for students belong to this age bracket. Thus, for designers of CT learning environments, it is suggested to consider the age of potential users and their acceptance of different programming languages.

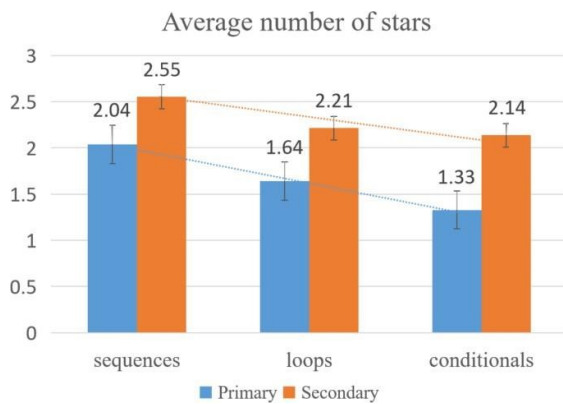


Figure 4. Achievement of Primary and Secondary students.

### 3.3 RQ3: Is completing easy missions a scaffold for completing hard missions?

Students' performance between Group A and B was compared. Figure 5 displayed the results of game achievement. It is indicated that the two groups performed similarly regarding the average number of stars. Yet the results for effort yielded different results (see Figure 6). For each concept, Group A had a lower value in 3-star attempts, implying that in cases where players were able to solve the puzzles with the optimal solutions, fewer

attempts were made by those who played easy missions beforehand. This indicates that playing easy missions could possibly scaffold students to solve harder problems.

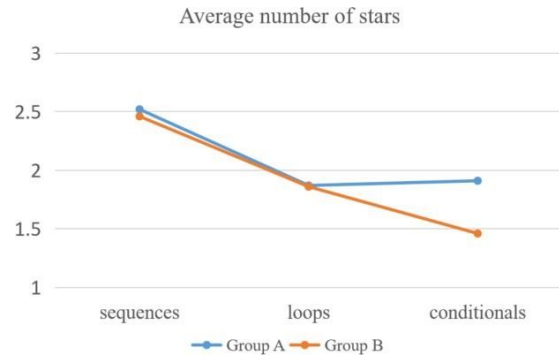


Figure 5. Achievement of Group A and Group B.

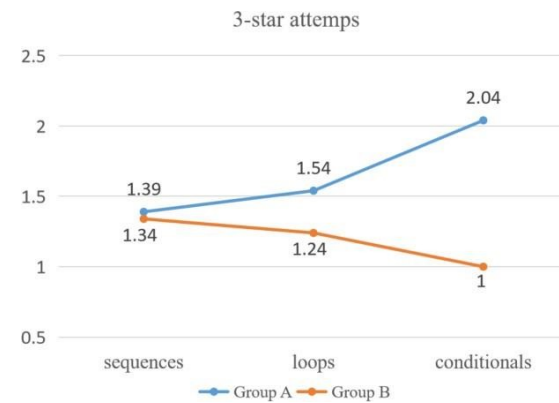


Figure 6. Effort of Group A and Group B.

These results can provide rich implications for the design of programming games and CT learning environments. Referring to Table 2, suggestions of the design of programming tasks for novices are as follows.

- ★ For *sequences*, initial tasks can be designed with solutions less than ten commands, accompanied with some basic spatial awareness (relative position, relative direction), after which more complex sequence tasks can be introduced.
- ★ For *loops* (see Figure 7), learners can be exposed to applying preset loops in the tasks first where they can test how loops work. Also, loop preset commands can be used to support novices. This can be reached by giving the access to modifying a preset loop in terms of *either* setting loop times *or* inserting new commands inside a loop. This would help learners get a deeper understanding of how loops can be applied through trials and errors. After these warming-up tasks, students can be given the opportunity to try creating loop commands from single loops to nested loops. These designs click with the model of “use-modify-create” proposed by Lee et al. (2011) for supporting the design of CT practical activities. The model suggests a learning progression to lead students to go from user to modifier to creator of programming projects (Grover & Pea, 2013), and it has been successfully applied in many CT learning platforms (eg. Zhao & Shute, 2019).

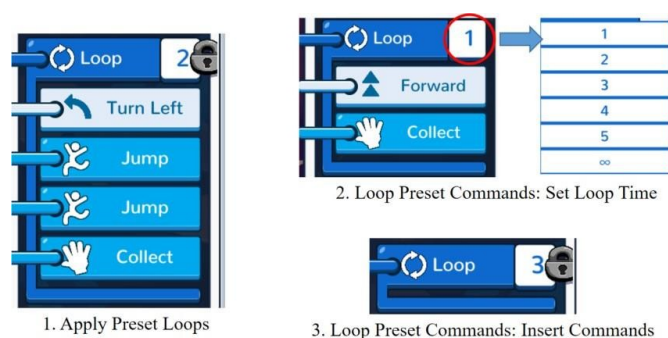


Figure 7. Designs for Loops Tasks for Novices.

- ★ As for *conditionals* (see Figure 8), designers can start with offering a conditional command with a preset condition where players can add actions by inserting commands. This can reduce the cognitive load caused by choosing the right condition. In addition, to expose students to learning how to implement the right condition, some choices of conditions can be offered first. After these practices, learners can be introduced to tasks that require creating conditional commands from scratch.

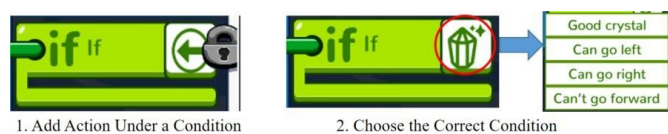


Figure 8. Designs for Conditional Tasks for Novices.

#### 4. CONCLUSION

This paper presented a case study on how students performed in a programming game in a self-regulated learning context during the COVID-19 pandemic. Results indicated that sequences was the easiest concept to acquire, while loops and conditionals were comparatively challenging, suggesting that instructors can provide more support when teaching these two concepts to novices. While primary and secondary students displayed the same order of difficulty in acquiring the three concepts secondary students outperformed primary counterparts, with an average of more than two stars throughout the game, which indicates that arrow-based programming language may be too easy for secondary students. Plus, those who played both easy missions and hard missions used less effort to achieve the same performance compared to those who only had access to hard missions, implying that some scaffolding task designs (eg. apply preset loops) may lay a foundation for more challenging tasks (eg. nested loops). Suggestions for CT game design for the concept of sequences, loops, and conditionals were given, which were elaborated with examples.

Limitations of the study are as follows. First, since playing the game was not a compulsory assignment for these students, it is likely that students' motivation to complete these tasks was driven by their interest in programming. Thus, the results of performance may be more positive than the reality, as those who were capable of completing the tasks were probably more motivated to do so. For future research, it is suggested that external forces (eg. rewards)

can be imposed to encourage more students to get involved. Second, the background information we collected from secondary students may not be enough to explain their higher performance than primary students. As the information was collected from their current secondary schools, how much they have learned before entering their current schools was unknown. Therefore, age may not be the only factor that resulted in the difference in performance. It would be more comprehensive if the difference can also be explained from their previous programming experience. For future research about CT knowledge acquisition, it is recommended to collect information about students' prior programming knowledge and extracurricular programming experience.

#### 5. REFERENCES

- Bers, M. U. (2018). *Coding, playgrounds and literacy in early childhood education: The development of KIBO robotics and ScratchJr*. Paper presented at the 2018 IEEE Global Engineering Education Conference (EDUCON).
- Bers, M. U., & Resnick, M. (2015). *The official ScratchJr book: Help your kids learn to code*: No Starch Press.
- Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. Paper presented at the Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada.
- Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational Researcher*, 42(1), 38-43.
- Israel-Fishelson, R., & Hershkovitz, A. (2019). Persistence in a Game-Based Learning Environment: The Case of Elementary School Students Learning Computational Thinking. *Journal of Educational Computing Research*, 58(5), 718-918. doi:10.1177/0735633119887187
- Kurland, D. M., & Pea, R. D. (1985). Children's mental models of recursive LOGO programs. *Journal of Educational Computing Research*, 1(2), 235-243.
- Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., . . . Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, 2(1), 32-37.
- Manske, S., Werneburg, S., & Hoppe, H. U. (2019). Learner Modeling and Learning Analytics in Computational Thinking Games for Education. In *Data Analytics Approaches in Educational Games and Gamification Systems* (pp. 187-212): Springer.
- Martins-Pacheco, L. H., von Wangenheim, C. A. G., & da Cruz Alves, N. (2019). *Assessment of Computational Thinking in K-12 Context: Educational Practices, Limits and Possibilities-A Systematic Mapping Study*. Paper presented at the Proceedings of the 11th International Conference on Computer Supported Education (CSEDU 2019).



- Moreno-León, J. (2018). *On the development of computational thinking skills in schools through computer programming with Scratch*. Doctoral dissertation
- Laporte, L., & Zaman, B. (2018). A comparative analysis of programming games, looking through the lens of an instructional design model and a game attributes taxonomy. *Entertainment Computing*, 25, 48-61.
- Lockwood, J., & Mooney, A. (2017). Computational Thinking in Education: Where does it fit? A systematic literary review. *arXiv preprint arXiv:1703.07659*.
- Papert, S. (1980). Children, computers and powerful ideas. In: New York: Basic Books.
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., . . . Silverman, B. (2009). Scratch: programming for all. *Communications of the ACM*, 52(11), 60-67.
- Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142-158.
- Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, 103798.
- Van Merriënboer, J. J., & Kirschner, P. A. (2017). *Ten steps to complex learning: A systematic approach to four-component instructional design*: Routledge.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
- Wing, J. M. (2014). Computational thinking benefits society. *40th Anniversary Blog of Social Issues in Computing*, 2014, 26.
- Zhao, W., & Shute, V. J. (2019). Can playing a video game foster computational thinking skills? *Computers & Education*, 141, 103633.

# Cultivating Computational Thinking through Game-based Scratch Programming

Xiaoqian LI<sup>1</sup>, Jing LI<sup>2</sup>, Jiansheng LI<sup>3\*</sup>

<sup>1,2,3</sup> Nanjing Normal University, China

2543404638@qq.com, 2587361612@qq.com, 2869753244@qq.com

## ABSTRACT

Computational thinking has become a necessary skill for students in the 21st century. Programming teaching is an effective way to cultivate computational thinking. However, programming is difficult and boring for some students. In this paper, it is explored whether game-based Scratch programming improves students' computational thinking and programming self-efficacy. In addition, the paper also explores whether individual differences of students affect computational thinking. The results showed that game-based Scratch programming could effectively improve the computational thinking skills, especially logical thinking. Secondly, playing Scratch games could improve students' programming self-efficacy. Finally, it was found that students' preference for games and computer operation skills would not affect the effect of programming games to cultivate computational thinking.

## KEYWORDS

computational thinking, game-based programming, Scratch, self-efficacy, secondary vocational students

## 1. INTRODUCTION

Under the wave of artificial intelligence, computational thinking, as a key ability of individuals in the artificial intelligence society, has been paid attention to and has become a necessary skill for students in the 21st century. In recent years, the cultivation of computational thinking has been incorporated into the instructional framework of information technology and other courses. For example, the UK has implemented a complete set of computational thinking courses in all disciplines, including computer science, information technology and digital literacy (Brown, Sentance, Crick, & Humphreys, 2014). Besides, computational thinking has been set up in the primary and secondary school courses as one of its national instructional courses in Australia (Falkner, Vivian, & Falkner, 2014).

Computer programming education was introduced into the basic education more and more. The research found that the computer teaching content of secondary vocational school is single and traditional. Most of them stay in the teaching of basic computer operation and common office software, even though the computer major has been added Python or other programming language. Due to the difficulty and dullness of programming itself and the lack of basic computer knowledge of secondary vocational students, many students have a fear of programming. Therefore, the training effect of computational thinking is not satisfactory.

Prensky (2003) pointed out that the mode of integrating entertainment and teaching was really suitable for teenagers. Game-based programming teaching can make abstract problems vivid and let students master the use of

basic sentences of programming language in the process of accomplishing practical tasks. In this process, students can improve their programming ability and computational thinking ability. In addition, it combines game elements and game scenes, so it can help students be more interested and motivated to complete programming tasks. As a graphical programming software, Scratch programming has become a powerful tool for game-based learning due to its modularity, interactivity, entertainment. Therefore, this paper attempts to apply Scratch programming game to secondary vocational students to improve their computational thinking ability and programming self-efficacy. Combined with students' personality characteristics, such as students' preference for the games and computer operation skills, this study puts forward the following research questions:

- (1) Can game-based Scratch programming significantly improve computational thinking? If so, which sub dimensions would be improved?
- (2) Can game-based Scratch programming significantly improve programming self-efficacy?
- (3) Will individual differences such as students' preference for games and computer operation skills affect the effect of cultivating computational thinking?

## 2. CONCEPTS AND DIMENSIONS OF COMPUTATIONAL THINKING

In 2006, Professor Wing first proposed the concept of computational thinking (referred to as "CT"). She explained that computational thinking is a series of thinking activities covering the breadth of computer science, such as problem solving, system design, and human behavior understanding, using the basic concepts of computer science (Wing, 2006). There are many definitions about the dimensions of computational thinking. International Society for Technology in Education and Computer Science Teachers Association defined computational thinking as abstraction, algorithm design, automation, data representation, data collection and data analysis (ISTE & CSTA, 2011); Shute, Sun, & Asbell-Clarke (2017) defined computational thinking as parallelism, algorithm thinking, problem decomposition, debugging, iteration and generalization. There is also a widely used way of classification. Romero, Lepage, & Lille (2017) divided computational thinking into five aspects: algorithmic thinking, abstraction, decomposition, evaluation and generalization. In addition, Brennan & Resnick (2012) defined computational thinking as consisting of computational concepts, practices, and perspectives from the perspective of practical activities, which is also a highly operational definition in the cultivation of computational thinking. Among them, concepts refer to the concepts used in programming, including: sequences, loops, events, parallelism,

conditions, operators and data. Practices refer to the behaviors carried out when creating a programming project, including: increment and iteration, testing and debugging, reuse and mixing, abstraction and modularity. Perspectives refer to the understanding of oneself, the relationship with others and the surrounding technological world, including expression, connection and query.

### 3. COMPUTATIONAL THINKING AND GAME-BASED PROGRAMMING

The earliest game programming language is logo language, followed by Scratch, Hopscotch, Code Combat, APP Inventor, Switch Playgrounds and so on. Based on the unique advantages of game programming, it is often used in the basic teaching of computational thinking. For example, the Greek researcher used the educational game Run Marco to teach basic programming concepts in primary school. The results showed that the use of educational games can help students understand basic programming concepts, and students also showed strong enthusiasm in using this game (Giannakoulas & Xinogalos, 2018). P. Rose, Habgood, & Jay (2020) developed a game based on Scratch programming called "Pirate Pluser". It was found that playing games can enhance the understanding of program abstraction for children aged 10-11 effectively. Furthermore, integrating Scratch into classroom activities has been shown to improve students' attitudes towards coding and computer programming (Korkmaz, 2016). In addition to video games, plug-in games are also a good choice. For example, the board game code monkey is developed for 8-year-old and above players. Players move the monkey pattern on the board to the destination by applying the computing concept. The game aims at helping players learn computational concepts, such as conditional, loops, boolean operators, logical operators, etc. In fact, in order to make the game successful, players need to decompose the problem to get the solution plan. Then via testing various plans in the system, players find the most effective strategy to overcome the challenge of the game. Therefore, the game needs a series of skills, such as problem decomposition, system testing and debugging, which are also important parts of CT. In addition, Jiang & Huang (2019) constructed a framework of children's programming game based on the cultivation of computational thinking in their research, which corresponded the steps of using computer to solve problems with the game elements. Combined with the dimensions of computational thinking, the relationship among the dimensions of computational thinking and game elements can correspond as shown in Table 1.

Table 1. Correspondence between CT and programming game elements

CT	programming game elements
Problem decomposition and representation	Core mechanism
Algorithm construction	Rule challenge
Debugging	Game objectives

It can be found that game-based programming has an impact on students' problem decomposition ability, programming concepts, logic and abstraction, operation and debugging. Because the participants are beginners, the researchers divided the game into seven levels in this study. The process of problem decomposition was weakened. Therefore, combined with the definition of computational thinking and the specific content of Scratch programming game, this paper mainly reflects the development of students' cognitive and non cognitive level of Computational Thinking from the three elements of algorithm, logical thinking and debugging, in which the algorithm contains the basic algorithm concepts, namely sequences, conditions and loops.

## 4. METHODS

### 4.1 Participants

The participants of this study were 36 nursing students from a secondary vocational college in Nanjing, Jiangsu Province, China. The participants were all girls, and they were all programming beginners. Before the experiment, the operation of office software was still taught on the information technology course.

### 4.2 Instructional Design

The researcher conducted a three-week tutorial on the Scratch programming game. In the teaching process, the teacher's explanation is the auxiliary, and the student's operation is the main. After explaining the basic game interface, the teacher gives the students the task to break through. The teacher assigned a total of 1-7 levels to pass. With the progress of the course, the difficulty of breakthrough is constantly upgrading. Figure 1 shows the interface of the fifth level breakthrough interface.

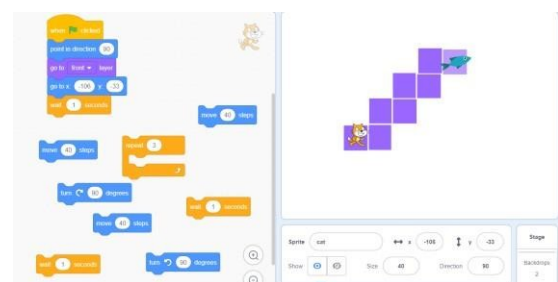


Figure 1. Level 5 of Scratch game.

### 4.3 Measures and Tools

Combined with the definition of Computational Thinking and the key dimensions of computational thinking, the Computational Thinking test compiled by Korkmaz (2016) is used in this paper. This set of test questions is based on the basic concept of calculation and logic, grammar of programming language. It consists of 28 single topics and is suitable for students from Grade 5 to Grade 10. We have a pretest and a post test before and after the teaching.

The computer programming self-efficacy scale was

adapted from Kukul & Karatas (2019) to evaluate the programming self-efficacy of the experimental subjects. The reliability coefficient for this questionnaire was 0.957. The question "Do you like to play games" was used to divide students into three types: "like", "general" and "not like". What's more the computer basic ability test scores

were used to divide the students' computer operation skills level. According to the average scores, the students were divided into low level group and high level group.

## 5. RESULTS

After the experiment, 36 valid questionnaires were collected, and the collected data were imported into SPSS.19.0 for data analysis. According to the research problems of this study, the analysis results are as follows:

### 5.1 After playing the game, the results of Computational Thinking Test improved significantly

First of all, in order to verify the impact of playing Scratch programming game on computational thinking performance, a paired sample T-test was conducted on the pretest and post test scores. The results showed that the average score of computational thinking increased significantly, and there was a significant difference between pretest and post test ( $P < 0.01$ ), indicating that the overall level of computational thinking improved after

playing Scratch game. Secondly, we continued to explore how different elements of computational thinking were improved. The algorithm dimension, logic dimension and debugging dimension were tested by a paired sample T- test, and the results were shown in Table 2. It can be seen that students' algorithm performance, logic performance and debugging performance have been significantly improved after playing Scratch programming game ( $P < 0.01$ ). According to the effect size (d), It was found that Scratch programming game improved logical thinking most obviously, followed by algorithm, and finally debugging. This can be explained from the characteristics of Scratch game. Beginners pay most attention to how to pass the game when they play the game, so the training effect of programming logic thinking is the most significant. However, students can not master the programming algorithm in a short time, and debugging errors need to be completed on the basis of the algorithm

Table 2. Difference between pretest and post test of CT

	Pretest		Post test		T	P	d
	Mean	SD	Mean	SD			
CT	56.25	16.19	75.14	10.45	-6.77	0.000**	
Algorithm	18.75	11.11	18.72	8.43	-10.49	0.000**	2.12
Logic	21.81	8.96	39.03	6.19	-10.85	0.000**	2.27
Debugging	10.42	5.65	16.39	5.43	-5.16	0.000**	1.08

\*  $p < .05$ .

\*\*  $p < .01$ .

### 5.2 Playing Scratch programming game can effectively improve programming self-efficacy

In order to accurately explore whether playing programming games can improve students' programming self-efficacy, the reliability analysis ( $\alpha = 0.096 > 0.9$ ) KMO and Bartlett's test ( $KMO = 0.84 > 0.7$ ) of the self- efficacy questionnaire used in this paper were conducted. The results show that the questionnaire has high reliability. Then, a paired sample t-test was carried out on the pretest and post test results of self-efficacy questionnaire, and the

results were shown in Table 3. As described in the table, there was a significant difference in the scores of students'

programming self-efficacy between pretest and post test ( $P < 0.01$ ). Due to the difficulty of programming, students often have a fear of difficulties in programming, especially for beginners. The results of this study show that Scratch programming games can help students improve their cognition of programming and increase their programming confidence to a certain extent.

Table 3. Difference between pretest and post test of programming self-efficacy

	Mean	N	SD	SE	T	P
Pretest	32.64	36	8.37	1.39	-	
Posttest	36.72	36	6.68	1.11	2.97	0.005**

\*  $p < .05$ .

\*\*  $p < .01$ .

### 5.3 The degree of game preference and computer foundation will not affect the performance of computational thinking

First of all, the degree of preference for game and computational thinking post test results were tested. The preference degree was divided into three levels, namely "like", "not like" and "general". The results were shown in Table 4. The relationship between the preference degree and computational thinking performance was not significant. One of the possible reasons is that the subjects are all girls, and there is no great difference in their preference for games.

Table 4. The correlation between game liking and CT

		Preference for game	Post test results
	Pearson correlation	1	.07
Preference for game	Significance (double tail)		.71
	Number of cases	36	36
Post test results	Pearson correlation	.07	1
	Significance (double tail)	.71	
	Number of cases	36	36

Secondly, the computer skill level was divided into high level group and low level group according to the average score. A paired sample T-test was used to test the results of the two groups. The results showed that there were significant differences between pretest and post test of

Computational Thinking Test of the two groups. In order to further explore the influence of computer level on the post test results of computational thinking, T-test was conducted on the post test results of the two groups. Results were shown in Table 5 and the level of computer had no significant effect on the post test results of computational thinking ( $P>0.05$ ). This may be because Scratch programming is a game based on block and does not need a high level of computer operation.

*Table 5. The correlation between computer level and CT*

	M	SD	T	P
high level	76.50	10.40		
low level	73.44	10.60	0.09	0.39

## 6. CONCLUSION AND DISCUSSION

With the advent of the era of artificial intelligence, computational thinking is not a unique way of thinking in the field of computer science, but has become a way of thinking in all social fields. This requires reseraches and teachers to shoulder the important task of cultivating computational thinking, and constantly explore the methods and strategies of cultivating computational thinking.

The results show that the game-based programming is an effective method to cultivate computational thinking. Besides, the results also show that the improvement of logic is the most significant, followed by algorithm and debugging. A study showed that Scratch users often produce code with 'code smells' such as duplicate blocks and long scripts which impact how they understand and debug projects (P. Rose, Habgood, & Jay, 2020). Additionally, debugging is to identify and repair errors when the algorithm can not provide the expected solution, so debugging needs a good algorithm foundation. Therefore, educators should balance the entertainment and education of programming games, and pay attention to the learning of sequence, loops, conditionals and other algorithms. This study also explores whether playing Scratch programming game can effectively improve the secondary vocational students' programming self-efficacy, and the result is positive. Scratch's graphical, building block programming method shields the grammar rules, algorithm structure and other learning obstacles, greatly reducing the cognitive difficulty of students. Therefore, to a certain extent, game-based Scratch programming can improve the confidence of programming. Finally, the study found that students' preference for the game and computer operation skills will not affect the effect of game-based programming to cultivate computational thinking.

However, the cultivation of human thinking is a continuous process. Limited by the research sample and cycle, the conclusion of this study inevitably has some limitations. Therefore, how to design the instruction and research of computational thinking, and how to combine the proper learning strategies with the subject are the problems worthy of further study in the future.

## 7. REFERENCES

Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational*

*thinking*. Paper presented at the Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada.

Brown, N. C., Sentance, S., Crick, T., & Humphreys, S. (2014). Restart: The resurgence of computer science in UK schools. *ACM Transactions on Computing Education (TOCE)*, 14(2), 1-22

Falkner, K., Vivian, R., & Falkner, N. (2014). *The Australian digital technologies curriculum: challenge and opportunity*. Paper presented at the Proceedings of the Sixteenth Australasian Computing Education

Conference-Volume 148.

Giannakoulas, A., & Xinogalos, S. (2018). A pilot study on

the effectiveness and acceptance of an educational game for teaching programming concepts to primary school students. *Education and Information Technologies*, 23(5), 2029-2052.

ISTE & CSTA. (2011). *Operational Definition of Computational Thinking for K-12 Education*. Retrieved January 1, 2021, from <http://www.iste.org/docs/pdfs/Operational-Definition-of-Computational-Thinking.pdf>

JIANG Xi-na, HUANG Xin-yuan (2019). The Design of Programming Games for Kids Pointed to Cultivating Computational Thinking Ability[J].*Modern Educational Technology*,29(03):119-126.

Korkmaz, Z. (2016). The Effects of Scratch-Based Game Activities on Students' Attitudes, Self-Efficacy and Academic Achievement. *International Journal of Modern Education & Computer Science*, 8(1), 16-23.

Kukul, V., & Karatas, S. (2019). Computational Thinking Self-Efficacy Scale: Development, Validity and Reliability. *Informatics in Education*, 18(1), 151-164

P. Rose, S., Habgood, M. P. J., & Jay, T. (2020). Designing a Programming Game to Improve Children's Procedural Abstraction Skills in Scratch. *Journal of Educational Computing Research*, 58(7), 1372-1411.

Prensky, M. (2003). Digital game-based learning. *Computers in Entertainment (CIE)*, 1(1), 21.

Román-González, M., Pérez-González, J., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, 72, 678-691. doi: 10.1016/j.chb.2016.08.047.

Romero, M., Lepage, A., & Lille, B. (2017). Computational thinking development through creative programming in higher education. *International Journal of Educational Technology in Higher Education*, 14(1), 42.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142-158.

Wing, J. M. (2006). Computational thinking. *Communications of the Acm*, 49(3), 33-35.



# Developing Girls' Computational Thinking by Playing Programming Games

Jing LI<sup>1</sup>, Jiansheng LI<sup>2\*</sup>

<sup>1,2</sup> Nanjing Normal University, China  
2587361612@qq.com, 2869753244@qq.com

## ABSTRACT

The purpose of this study is to explore the specific impact of playing programming games on each dimension of girls' computational thinking through playing Cat Eat Fish, a game designed based Scratch. The results showed that playing programming games can promote girl beginners' computational concepts and perspectives, but the role of playing programming games in promoting the girls' computational practices did not be found.

## KEYWORDS

computational thinking, game-based learning, girls

## 1. INTRODUCTION

Computational thinking (CT), a basic skill in the 21st century, has been incorporated into K12 education and higher education in many countries. According to Wing (2006), computational thinking covers a series of thinking activities in the field of computer science, specifically, it refers to the use of basic concepts of computer science for problem solving and system design. The dimensions of computational thinking in this study are based on Brennan and Resnick's definition (Brennan & Resnick,2012), in which the concepts of computation include sequences, repetitions, cycles, conditionals, and selection, the computational practice is to solve practical computational problems, and computational perspective involves the attitude and perspectives of computational thinking.

It has been evidenced that the education of introductory programming can be supported by playing games, but it takes longer for girls to acquire the same computational thinking skills as boys (Atmatzidou & Demetriadis,2016). Playing games can promote students' understanding computational concepts (Kazimoglu, Kiernan, Bacon, & MacKinnon,2012). Some studies showed that playing games can improve attitude toward computational thinking, but others demonstrated that playing games had no effect on computational perspectives (Zhao & Shute,2019).

There is an urgent need to explore whether playing games can promote the computational thinking of beginners, especially girls, and if the answer is yes, what aspects of computational thinking can be advanced by playing

programming games. Thus, the research questions of this study are as follows:(1) Which aspects of computational concepts are more effective in playing games? (2) Can

playing games promote girls' computational practice? (3) Can playing games improve girls' computational perspectives? If the answer is yes, what dimensions of computational perspectives would be improved?

## 2. METHOD

### 2.1. Participants and design

The participants were 48 secondary school students from Nanjing Health School in China. They were all girls who had no programming experience and the average age of them was 16. The whole experiment lasted for 3 weeks.

### 2.2. Materials

#### 2.2.1. Testing questionnaires

The Computational Thinking test (CTt; Moreno-León, & Robles,2018) were selected to measure students' computational concepts. The testing questionnaire for computing practice was selected from the International Challenge on Informatics and Computational Thinking. The Computational Thinking Scales (CTS; Korkmaz, Çakir, & Özden,2017) was used to survey computational perspectives.

#### 2.2.2. The Cat Eat Fish game

The game used in this study was designed based on Scratch called Cat Eat Fish, in which students were asked to combine the code blocks scattered in the code editing area to make the cat eat the fish. It contained seven levels and the one who took the least time and can successfully passes the game won.

## 3. RESULTS

### 3.1. Which Aspects Of Computational Concepts Are More Effective After Playing the Game?

The analysis results of the computational concepts scores, presented in Table 1, revealed a significant difference between pre-test scores and post-test scores. Cohen's effect size ( $d = 1.39$ ) suggested a large effect of playing the programming game (Cohen,1988). Table 2 and Table 3 indicated that there were significant differences in the pre-test and post-test results of sequences, cycles, repetitions, conditions and selection, and the effect of cycles is the largest.

Table 1. Results of paired t-test for girls' computational concepts.

	Mean	N	SD	SE	t	p
Pre-t	54.57	47	16.64	2.43	-	
Post-t	74.04	47	11.31	1.65	8.44	.000

Table 2. Statistical description for each dimension of girls' computational concepts.

	Pretest		Posttest	
	Mean	SD	Mean	SD
sequences	15.74	4.30	18.72	2.20
cycles	18.51	5.61	21.49	4.65

repetitions	7.66	4.65	14.04	4.38
conditions	4.47	3.79	9.04	3.99
selection	8.19	5.05	10.43	5.09

Table 3. Results of paired t-test and effects sizes for each dimension of girls' computational concepts.

	t	p	d
sequences	-4.95	0.000	0.92
cycles	-3.33	0.002	1.72
repetitions	-7.70	0.000	1.41
conditions	-6.76	0.000	1.01
selection	-2.33	0.024	0.44

### 3.2. Can Playing the Programming Game Promote Girls' Computational Practices?

The results, shown in Table 4, presented that there was no significant difference in the pre-test and post-test scores of girls' computational practices.

Table 4. Results of paired t-test for girls' computational practices.

	Mean	N	SD	SE	t	p	Pre-
t	20.32	47	9.17	1.34			
Post-t	22.77	47	7.79	1.14	-1.84	0.073	

### 3.3. Can Playing the Programming Game Improve Girls' Computational Perspectives?

A paired sample t-test was used to test the results of girls' computational perspectives. The results, shown in Table 5, presented that there were significant differences between pre-test and post-test surveys. Table 6 and Table 7 showed statistically significant differences in the means of creativity, problem solving and critical thinking. From the size of the effect, creativity ( $d=0.50$ ) and problem-solving ( $d=0.42$ ) had a larger effect.

Table 5. Results of paired t-test for girls' computational perspectives.

	Mean	N	SD	SE	t	p
Pre-t	80.30	47	13.05	1.90	-	
Post-t	87.47	47	12.12	1.77	4.52	.000

Table 6. Statistical description for each dimension of girls' computational perspectives.

	Pretest		Posttest	
	Mean	SD	Mean	SD
creativity	25.34	7.13	28.32	4.73
problem solving	17.87	5.16	19.89	4.47
critical thinking	17.28	4.65	18.34	3.22
algorithmic thinking	19.81	5.17	20.91	4.25

Table 7. Results of paired t-test and effects sizes for each dimension of girls' computational perspectives.

	t	p	d
creativity	-2.88	0.006	0.50
problem solving	-2.52	0.015	0.42

critical thinking	-2.09	0.043	0.29
algorithmic thinking	-1.78	0.082	0.23

## 4. DISCUSSION AND CONCLUSION

The results of this study indicated that playing programming games can improve girls' computational concepts in a short period of time, and it improves girls' mastery of computational concepts such as sequences, circulations, repetitions, conditions and selection. Furthermore, the Cat Eat Fish game in this study cannot promote girls' computational practices. Girls' computational perspectives were significantly improved after playing the Cat Eat Fish programming game, especially creativity, problem-solving ability and critical thinking ability, while the algorithm thinking dimension of computational attitude was not significantly improved. In practice, teachers can design some simple programming games like Cat Eat Fish to promote girl beginners to foster computational thinking skills. Due to the short duration of this study, future researches can further explore whether playing programming games for a long time can improve the computational practices dimension of girls' computational thinking.

## 5. REFERENCES

- Atmatzidou, S., & Demetriadis, S. (2016). Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences. *Robotics and Autonomous Systems*, 75, 661-670.
- Brennan, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. Paper presented at the Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada.
- Cohen, J. (1988). Statistical Power Analysis for the Behavioral Sciences. *Journal of the American Statistical Association*, 31(334), 499-500.
- Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning Programming at the Computational Thinking Level via Digital Game-Play. *Procedia Computer Science*, 9, 522-531.
- Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior*, 72, 558-569.
- Román-González, M., Pérez-González, J., Moreno-León, J., & Robles, G. (2018). Extending the nomological network of computational thinking with non-cognitive factors. *Computers in Human Behavior*, 80, 441-459.
- Wing, J. M. (2006). Computational thinking. *Communications of the Acm*, 49(3), 33-35.
- Zhao, W., & Shute, V. J. (2019). Can playing a video game foster computational thinking skills? *Computers & Education*, 141, 103633.

# Programming Socio-scientific Games: A Computational Thinking Approach to Real-world Problems

Marianthi GRIZIOTI<sup>1\*</sup>, Chronis KYNIGOS<sup>2,3</sup>

<sup>1,2</sup>Educational Technology Lab, National and Kapodistrian University of Athens, Greece

<sup>3</sup>Linneus University, Sweden

mgriziot@eds.uoa.gr, kynigos@eds.uoa.gr

## ABSTRACT

This paper discusses how computational thinking could be studied beyond computer science education, as a means for dealing with real-world multidisciplinary problems. It suggests the approach of students playing and modifying simulation games, that represent socio-scientific issues, with the use of integrated computational affordances, including coding, data editing and map design. It further presents the initial results of an empirical study, concerning the computational practices developed by junior-high school students while modified a simulation game in an authoring tool called ChoiCo.

## KEYWORDS

block-based programming, simulation games, game modding, socio-scientific issues

## 1. INTRODUCTION

Central role to the 21<sup>st</sup>-century education has the cultivation of globally sensitised citizens who can recognise and deal with complex, ill-structured issues affecting our world and societies (UN, 2018). These involve the so-called Socio-Scientific Issues (SSI); open-ended controversial problems and dilemmas with multiple variables and factors are not easy to be described with formal mathematical models (Sadler, 2002). Global warming, sustainable lifestyles, urban development, ethical science are some examples. However, understanding and dealing with such issues requires system-based multidisciplinary approaches that differ significantly from the silo, linear problem-solving of school activities. To address this challenge, we suggest utilising computational thinking beyond the limits of computer science as a tool for realising authentic multidisciplinary problems. In this context, we studied children computational strategies while playing and modifying simulation games that dealt with SSI in an online environment called ChoiCo (Choices with Consequences) <http://etl.ppp.uoa.gr/choico/>.

## 2. THEORETICAL FRAMEWORK

### 2.1. Computational thinking and game modding for Socio-Scientific Issues

Several studies have shown that new coding technologies, such as programmable simulations and game design can enable children to develop computational thinking skills and understand concepts that were not accessible before (Kynigos & Grzioti, 2018). Computational thinking refers to the ability of efficiently applying practices, (e.g. pattern recognition, abstraction, automation) and concepts, (e.g. variables,

iteration, functions) from computer science for solving different kinds of problems computationally (Wing, 2009, Grover & Pea, 2018). However, so far, it has been studied mainly through well-structured programming tasks in computer science education. With the need to integrate SSIs in education, it seems that it is a good time to study computational thinking in other STEAM contexts, utilizing it as a vehicle for understanding complex behaviors of real-world phenomena. To achieve this integration, we suggest the approach of modding digital simulation games with coding and other high-level computational tools. The term modding refers to the process of modifying elements of a fully functional game to create a variation of it, usually called "mod" (Kynigos & Yiannoutsou, 2018). Modding differs from designing a game from scratch, in that the "modder" makes focused modifications to game elements, aiming to integrate his/her ideas into the gameplay experience. In that sense, coding is used as a means of self-expression on the game's structural ideas. However, most educational game design environments, such as Scratch and Alice, focus on procedural or object-oriented programming without supporting the development of complex system-based simulation games, especially from inexperienced students. On the other hand, modelling tools, such as NetLogo, even though they can produce very accurate simulations of dynamic phenomena, they usually require advanced technological and scientific knowledge to intervene with the model or create one from scratch. Therefore, to develop game modding activities on SSI simulation games we used ChoiCo authoring tool developed by the Educational Technology Lab (NKUA).

## 3. THE CHOICO AUTHORIZING TOOL

ChoiCo is an online authoring tool that allows the play and design of "choice-driven simulation" games representing socio-scientific issues through a system of choices and consequences. The player revolves in map-like areas making choices with positive or negative consequences to a set of game fields that should not cross certain "red lines" (Fig. 1). The user can modify all game elements with three computational affordances: a) a map editor for editing the game interface and choices, b) an interactive database for the game choices, consequences and fields, and c) block-based programming for game rules. With ChoiCo's integrated affordances, children can progressively access and modify the game expressing their personal views on the gameplay values.

## 4. EMPIRICAL STUDY

To investigate the computational thinking strategies that children develop and use when dealing with real-world



issues through ChoiCo games, we organised a 15-hours empirical study in a junior-high school with 8 students who worked in 4 groups. The study was divided into 8 sessions. For the study, we created a ChoiCo game called "CT Chef" that dealt with the sustainable development of a restaurant. The player is a restaurant owner who has to make choices (e.g. food dishes, supplies and other actions) to keep his/her restaurant open by balancing food quality, profit and regular customers (Fig. 1). The choices affect five game fields: money, healthy food, customers per day, animal product supplies, vegetable supplies, and no of Ch.



**Fig 1: The game CT Chef in ChoiCo environment**

In the first two sessions, students played the game and discussed possible improvements. In the next sessions, they collaborated to create a game mod. We collected a set of qualitative data including interviews, audio and screen recordings of each group, student games and researcher diaries. The data were analysed qualitatively, using the "critical incident" as the analysis unit and a coding schema that was enhanced with new codes during the analysis. We aimed to identify patterns relevant to a) the development of computational strategies by students to make sense of the SSI b) uses of ChoiCo affordances significant to the learning process.

## 5. INITIAL RESULTS

All teams created 2 variations of the original game, resulting in 8 game mods. The analysis has revealed that students developed and applied computational strategies to and make decisions and deal with the simulated socio-scientific issue. We briefly present three of the main categories.

### 1. Searching for patterns into the simulated phenomenon.

A computational strategy developed by all teams was recognising choices patterns either in the gameplay, the database or the code. We identified 3 different strategies of pattern recognition relevant to the game SSI: a) search for patterns on the choices made as players to achieve a higher game score, b) identify patterns in the game data and classify the choices to categories according to these patterns. c) create new patterns on the choices' data using the database and block-based programming. For pattern recognition students combined computational concepts with societal and scientific game axioms, such as balanced diet and restaurant profit. For instance, group2 created a data

pattern that for every 3 choices that increase Health a little bit, there should be one choice that decreases it.

### 2. Controlling the system behaviour with automation.

A second strategy developed by all groups was the use of automation to control and intervene in the game flow. This strategy involved complex conditional structures, Boolean calculations and command sequencing, practices that are usually difficult for young children to understand. Students realised these computational ideas by connecting them with the game context.

**3. Prediction and handling of events.** Students during modding aimed to create a balanced game in terms of realism, fun and difficulty. To achieve that, they applied computational practices that involved: a) the prediction of fields values for hypothetical gameplay scenarios. This allowed them to think of different approaches to the represented issue and to get familiar with the concept of variation and balance in a complex system. b) The creation of internal rules for handling events that may emerge during play. For instance, "If the player selects a particular choice, change the background".

## 6. CONCLUSION

The presented study's initial results showed that with the appropriate affordances, children may develop and apply computational thinking strategies to make sense of complex multidisciplinary problems and dilemmas such as balanced diet, food habits, and business management. It seems that game modding of open-ended simulation games could provide a meaningful context for applying programming and computational thinking in STEAM fields and real-world problems. Further research and analysis on this topic could contribute to the development of an integrated computational thinking approach where it is applied in multidisciplinary problems and even in social studies and humanities.

## 7. REFERENCES

- Grover S., & Pea. R. (2018). Computational Thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*. 19.
- Kynigos C., & Grizioti M. (2018). Programming Approaches to Computational Thinking: Integrating Turtle Geometry, Dynamic Manipulation and 3D Space. *Informatics in Education*, 17 (2), 321-340.
- Kynigos,C.,&Yiannoutsou,N.(2018). Children challenging the design of half-baked games: Expressing values through the process of game modding. *International Journal of Child-Computer Interaction*, 17, 16–27.
- Salder T. (2011). Socio-scientific issues-based education: What we know about science education in the context of SSI. *Socio-scientific Issues in the Classroom*. p. 355-369. Springer, Dordrecht,
- United Nations. (2018). *Sustainable Development Goals*.
- Wing. J.M. (2006). Computational thinking. *Communications of the ACM* 49 (3)

# **Computational Thinking and Unplugged Activities in K-12**

# Research on the Design of Unplugged Computer Science Teaching Activities in Elementary School—Taking the Fruit Delivery Game Course as an Example

Bingqing YANG

Faculty of Education, Beijing Normal University, China  
yangbq@mail.bnu.edu.cn

## ABSTRACT

In 21st century, Computational Thinking is a fundamental skill for every person, and CT skills will be added into the international PISA test in 2021. There is an international trend that many countries have begun to focus on computational thinking education. In China, we regard Computational Thinking as one of the core literacies of the Information Technology Curriculum. The practice of computational thinking education in elementary school is important, but the knowledge in this field is so abstract that it's difficult for elementary students to understand. But unplugged computer science activities could simplify complex computer science concepts into operable teaching activities. In the unplugged environment, students could learn knowledge and develop their CT skills through interesting games and activities. Therefore, this study used LACID theory to design unplugged computer science ability activity for elementary school students. We take the Fruit Delivery Game Course as an example, and 30 students participated in the course. The study found that 1) the LACID theory could provide effective guidance for teacher to design Unplugged Computer Science Activities at the elementary level; 2) students generally had good learning experiences, and they were highly satisfied with the unplugged course. 3) "difficulty stratification of activity design" and "summarizing knowledge points by subjects" were two effective strategies.

## KEYWORDS

computational thinking, Unplugged Computer Science Activity, elementary school students

## 1. INTRODUCTION

Computational thinking is a fundamental skill for everyone, not just computer scientists. The concept of Computational Thinking was first proposed by Professor Jeannette M. Wing in 2006, she thought that Computational Thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science (Wing, 2006). Researchers are increasingly focusing on computational thinking, and computational thinking is attracting the attention of all disciplines including science and humanities (Bundy, 2007). Recently, the OECD proposed that computer science and computational thinking could cultivate students' problem-solving, creative and collaborative skills, and pointed out that computational thinking would be added into The Program for International Student Assessment (PISA) in 2021 (OECD, 2018).

Wing emphasized that if we want to ensure a universal and solid foundation of understanding and prepare everyone for

CT skills, then this kind of learning was best done from the early stages of childhood (Wing, 2008). With the international trend of computational thinking education, schools in China has paid attention to the education of CT skills. But, in the K-12 stage of computational thinking education research, we seemed to pay more attentions to students in grades 6-8. As there were only 20% studies focused on grades 3-5 students (Yu & Wang, 2020). Therefore, the challenge we faced is to find a suitable approach for the teaching of computational thinking in elementary schools, not just teach students to program.

## 2. LITERATURE REVIEW

The core idea of Unplugged Computer Science activities is to pay attention to children, especially young children. We expect that young students could have the chance to experience the thinking path of scientists through playing unplugged activities. Use hands-on activities to cultivate their abstract thinking, decomposition, algorithm, and problem-solving abilities. For elementary school students, they would accomplish learning tasks through collaboration and interaction with their peers in unplugged learning. It might be better to develop students' computational thinking and problem-solving skills through unplugged teaching.

The concept of Unplugged Computer Science was proposed by Professor Tim Bell from the University of New Zealand with two teachers Ina H. Witten and Mike Fellows. The concept of CS Unplugged was brought forward based on their practical teaching experience. According to them, CS Unplugged is a collection of free teaching material that teaches computer science through engaging games and puzzles that use cards, string, crayons and lots of running around (Bell, T, 1998; Bell, Alexander, Freeman, & Grimley, 2009). With the popularity of the CS Unplugged project worldwide, teaching practice courses with cultural characteristics have been continuously added.

Unplugged is a special type of CT education. Results showed that the CT skills of the students who participated in the unplugged class significantly increased after unplugged teaching (Brackmann C P, et.al, 2017). The main characteristic of the Unplugged activities are: No computers, Games, Kinesthetic, Student directed, Easy implementation, Growing body of ideas, Sense of story (Nishida T, et al, 2009).

Because it is feasible and can be promoted in every elementary school, it's very important to bring CS Unplugged in primary education. Firstly, not all schools are adequately equipped with computers in China. Unplugged activities are extremely practical when computer equipment is insufficient. Secondly, unplugged form is very friendly to some elementary school students who don't like programming, especially girls. By participating in

unplugged activities, they might be more willing to learn programming. Thirdly, although programming is a popular way to training students' CT skills, but it might lead to students' misunderstandings of computer science. In contrast, unplugged activities might prevent this happening. Fourthly, it is more appropriate to use unplugged form in the short-term teaching of elementary schools. Fifthly, with the understanding the basic concepts of computer science, we could make important decisions related to computer's security and reliability in daily life.

The purpose of this research is to provide Unplugged CS teaching to the elementary school students in China. The question is "how to design the CS Unplugged teaching activities, and looking for effective strategies in teaching implementation".

### 3. FRUIT TRANSFER GAME COURSE DISIGN BASED ON THE LACID THEORY

The Learning-Activity-Centered Instructional Design (LACID) theory provides an instructional design model. LACID theory is learning activity-centered, and CS unplugged is all about activity. According to the theory, the main steps are: 1) trying to design the instructional plan, 2) knowledge modeling, 3) clarifying the idea of activities segmentation, 4) redesigning the instructional plan (Yang, K. C, 2005).

The study takes the Fruit Transfer Game Course as an example. The main concept is "network topology", which is an important concept in computer science.

There are many kinds of network topology, and each form has its advantages and disadvantages. In real life, it is necessary to use a certain network topology to establish the connection between computers. By learning the concept, students would understand the basic principle of computer information transmission

After the 4 steps of design, the teaching process was determined to be divided into 3 activities. Activity 1: "introduction the different types of internet information transmission", Activity 2: "fruit transfer activity (including Circular/Liner fruit delivery activity)", and Activity 3: "discussion and summary of the course knowledge".

### 4. IMPLEMENTATION AND FINDINGS

There were 30 students participated in the course. We had questionnaire and interview after class. In the questionnaire, students were asked to rate the course and provide suggestions from their perspectives. In interviews, they were asked about their learning experience and future learning tendencies.

Through the analysis of questionnaire and interview data, we found that most students were very satisfied with the unplugged course. As their learning experience, students thought that they had a very pleasant learning experience. As for their attitude towards future learning, unplugged learning had positive influence on students' learning preferences and attitudes. Almost all students who were interviewed indicated that they would be willing to

participate in this unplugged course in the future. They also hoped there were similar elective courses in school. In addition, we found two effective strategies discovered during the implementation: "difficulty stratification of the activity design"; "summarizing knowledge points by subjects".

### 5. RESEARCH SUMMARY

It makes sense for students to experience more than one form of learning in computational thinking classroom. We found that the LACID theory could provide effective guidance for teacher to design Unplugged Computer Science Activities at the elementary level. It is worth mentioned that unplugged activities are no longer limited to informal instructional situations, such as homes, science museums, and extracurricular educational institutions. This study shows that unplugged computer science activity could be carried out as formal curriculum in schools in the future.

### 6. REFERENCES

- Bell, T., Witten, I. H., & Fellows, M. (1998). *Computer Science Unplugged: Off-line activities and games for all ages*. New Zealand: Computer Science Unplugged.
- Bell, T., Alexander, J., Freeman, I., & Grimley, M. (2009). Computer science unplugged: School students doing real computing without computers. *The New Zealand Journal of Applied Computing and Information Technology*, 13(1), 20-29.
- Bundy, A. (2007). Computational thinking is pervasive. *Journal of Scientific and Practical Computing*, 1(2), 67-69.
- Brackmann C P, Román-González M, Robles G, et al. Development of computational thinking skills through unplugged activities in primary school[C]//Proceedings of the 12th Workshop on Primary and Secondary Computing Education. 2017: 65-72.
- Nishida T, Kanemune S, Idosaka Y, et al. A CS unplugged design pattern[J]. *ACM SIGCSE Bulletin*, 2009, 41(1): 231-235.
- OECD. (2018). *PISA 2021 ICT framework*. Organization of Economic Co-operation and Development. Retrieved June 25, 2020, from <https://www.oecd.org/pisa/sitedocument/PISA-2021-ICT-framework.pdf>
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717-3725.
- Yang, K. C. (2005) *A practical guide for instructional design centered on learning activities*. Beijing: Publishing House of Electronics Industry.
- Yu, X. H., & Wang M. (2020) How Far is the Cultivation of Computational Thinking in K-12: From the Perspective of Computational Thinking Assessment. *Open Education Research*, 26(01):60-71.

# **Computational Thinking and Subject Learning and Teaching in K-12**

# A Hybrid Approach to Teaching Computational Thinking at a K-1 and K-2 Level

Damien Constantine ROMPAPAS<sup>1</sup>, Steven YOON<sup>2</sup>, Jonothan CHAN<sup>3</sup>

<sup>1</sup>Brewed Engagement Extended Reality Labs, South Australia, Australia

<sup>2,3</sup> Jules Ventures, Singapore

hyperlethalvector92@gmail.com, kc@jules.sg, jon@jules.sg

## ABSTRACT

Computational Thinking (CT) has been described as taking an approach to solving problems, designing systems and understanding human behavior that draws on concepts fundamental to computing. It is the ability to integrate human creativity and insight with concepts derived from Computer Science. We argue that it is best to learn the fundamentals of CT at a young age, when the mind is most malleable, instead of much later when these concepts are taught as part of Computer Science courses. However, challenges arise not only when trying to teach these complex concepts to young children, but also when applying these teachings through kindergarten environments. We present a definition of the basic fundamental CT concepts and then describe a unique hybrid approach of offline and online activities to teach these fundamentals to students at the kindergarten (K1 and K2) level (children aged 4-6 years old). Finally, we validate this approach with a pilot class to determine its learning effectiveness.

## KEYWORDS

E-Learning, Child Education, Computational Thinking, Blended Learning, Gamification

## 1. INTRODUCTION

The “4C’S” – critical thinking, creativity, collaboration and communication have already been recognized as core 21st Century skills to be embedded into school curricula. As technology such as A.I., machine learning and robotics advance rapidly; our children are faced with the prospect that over 80% of future job needs will be disrupted. The need to understand how to use computational tools and to be able to problem-solve is becoming a fundamental competency. “Computational Thinking” is the “5th C” of 21st century skills and is being embedded as part of core curricula in education systems across the world. Computational Thinking (CT) has first been described by Papert Et. Al and Wing Et. Al as taking an approach to solving problems, designing systems and understanding human behavior that uses concepts fundamental to computing. It is the ability to integrate human creativity and insight with concepts derived from Computer Science. We can list previously defined CT skills from outside sources, such as (Barr Et Al 2011), into a general diagram to highlight the four most fundamental of CT skills. These CT skills are described as follows:

**Algorithmic Thinking:** Getting to a solution through the clear definition of the instructions that need to be followed.

**Decomposition:** Also known as factoring, is to break down a complex problem or system into parts that are easier to conceive, understand, program and maintain.

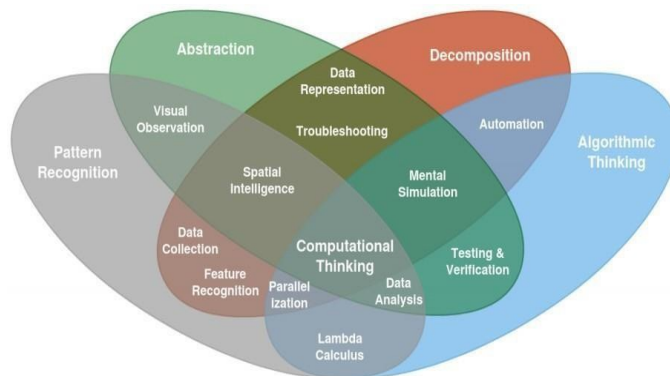


Figure 1. Categorization of previously defined CT elements. Although not comprehensive all listed topics lie under one of the listed categories and are a core part of kindergarten curriculum.

**Abstraction:** To generalize several complex solutions or definitions based on similarities or common rules. Then apply these generalizations to an alternative context.

**Pattern Recognition:** The process of classifying input data into objects or classes based on key features, and infer new solutions based on previously classified data.

We assert that these skills should be taught at an early age, when the child is most malleable (Samuelson and Carlson Et al 2008). There are two major challenges that must be addressed when teaching to this audience. One of the most difficult challenges is how to approach teaching these skills to children given that at the K1 and K2 level, their language and motor skills are still developing. The second challenge lies in providing a digital teaching medium which can be accepted. This is primarily due to resistance to the use of teaching through a digital platform (Turbill Et. Al 2001) even though it is an effective medium for teaching concepts that are hard to understand (Lieberman Et al 2009). In this paper we present a methodology for teaching these fundamental CT skills using a hybrid of online and offline activities through a tablet computer and physical practice / worksheets. We discuss the design of the online animated videos which teach the high-level concept of the basic CT skills which is then augmented through teacher interactions. We also discuss the design of digital games to facilitate simulated practice of these CT skills, and their translation to real-world offline activities within the class. We evaluate the effectiveness of this methodology through a pilot study in which a short implementation of this design is used. Overall, the core contributions of this work are:

- The first formal derivation and definition of the fundamental CT skills.

- The design of a hybrid approach of online and offline activities to teach the fundamental CT skills applicable to K1 & K2 groups.
- An empirical method of evaluating the student CT skills taught using this approach.

We believe that through this hybrid design, children can learn the concepts of CT and apply these problem-solving skills early on in their lives and continue developing these skills to significantly improve their future academic progress and daily life activities.

## 2. RELATED WORK

We assert it is imperative these fundamental CT skills are taught at an early age. To devise a valid approach, a careful analysis of previous frameworks for teaching CT, and methods for engaging children must be conducted. In this section we discuss three key avenues of related work;

(i) CT in education; (ii) engaging children using digital media as a teaching platform; and (iii) the use of simulation as a method to practice CT skills.

### 2.1 Computational Thinking in Education

The idea of teaching CT is not new. In the 1960's, Alan Perlis was one of the first who argued for the need for college students of all disciplines to learn programming and the "theory of computation" (Guzdal 2008, Perlis 1962). Teaching CT shouldn't be limited to college courses as introducing these CT concepts can be applied as a tool to improve the skills taught in K-12, and key problem-solving skills used outside of school (Barr Et. Al, 2011). Similarly, we also derive how the basic subjects of CT supplement the basic components of the general K1 and K-2 curriculum (Table 1). This is not the only instance of applying CT in an educational environment (Kazimoglu Et al 2012). Here the authors approach learning CT through digital game mediums. The benefit of this approach is that it allows students to learn the application of CT in pre-programmed simulation environment. Although this approach has shown to be effective, the games and interfaces used are aimed towards older audiences, likely making them too complex for younger children to adopt and use. This makes it difficult to directly apply this approach without making it more child friendly. Although not implemented, (Falkner Et. Al 2015) discusses how and when CT should be taught. However, their questionnaire suggests that teachers at that level only consider CT as a useful subject in Information Technology and Mathematics subjects. Because classes are designed to teach children as young as 6 years old (in K-2 grades) coding as a supplemental enrichment class, we assert the fundamentals of CT must be taught as an additional core subject instead of an enrichment class to maximize the impact of the benefits. To the best of our knowledge, our teaching method is the first that can be applied to allow teaching fundamental CT concepts to children at a K1-K2 level which can be accepted by kindergartens.

### 2.2 Using Digital Media to Teach Children

Media as a platform for teaching is not a new concept, in fact it was (Meir Et Al 1969) in the late 1960's who explored how educational media, would contribute to the

early years of childhood. Although this is only exploring physical art media it supports later investigations by (Burns Et. Al 2004), which highlighted that video can be used as an interactive teaching medium, provided that it is carefully designed and integrated with online in-class materials. Additionally, (Lieberman et. Al 2009) investigates the effectiveness of digital media as a teaching platform for younger children (aged 3 to 6), showing that digitally assisted media can greatly assist in explaining high level concepts in a way that children can understand. These studies sparked the creation and usage of video games and media for entertainment and education (also known as edutainment). Such mediums in teaching environments have highlighted increased attention during use and retention of information afterwards when engaging with edutainment media at an educational capacity (Ritterfeld 2006). Examples for such edutainment tools are: mathematics (Elliot Et. Al, 1997), Creativity and Learning (Montemayor Et. Al, 2004), and Reading and Literacy (Verdugo Et. Al, 2004) (Teaching English to children with English as a second language). Our work extends this by utilizing animated video which introduces and teaches difficult high-level CT concepts to children in a way that can be understood, engaging and interactive.

### 2.3 State of CT Teaching in K-12

In the UK, the "Barefoot Computing" approach using traditional paper and pen has been adopted since 2014, with trained teachers teaching CT in primary schools. In recent years, CT are being taught using new tools (Sung Et. Al 2016) in hardware such as Arduino and educational robots and coding software such as Scratch and Scratch Junior. However, limitations of these tools are as follows (i) high cost of hardware; (ii) unable to teach the full CT concepts; and (iii) require significant investment in trained teachers. All these factors limit how CT can be effectively delivered and deployed at scale in kindergartens. The right use of mobile devices can enhance the learning experience of students as well as strengthen teacher- development programs. Our work differs by applying specifically-designed software content on a mobile platform (Grover Et. Al 2013).

### 2.4 Use of Gamification and Simulation to Practice CT Concepts

As the core of kindergarten education is learnt through play, we strongly encourage the use of digital simulation environments, which in turn are transformed into video games, the process of gamification. Gamification allows for stress free, engaging and entertaining online practice of CT concepts. This in turn will relieve anxiety that can be experienced when applying the high level concepts to real world contexts. Examples of such simulation environments are shown by (Kazimoglu Et. Al 2012, Montemayor Et. Al 2004). However, these games are designed with older target audiences in mind. Our work crucially differs from related work in two ways. First: multiple games that simulate separate CT concepts are used in our digital application; Second, our user interface and experience is designed and implemented with simplicity in mind, allowing younger children to fully enjoy the experience whilst practicing the fundamental CT concepts



A)	Pattern Recognition	Abstraction	Decomposition	Algorithmic Thinking
Pearly Whirly	-	-	-	++
Manta Match Mania	-	+	++	-
Crabby Patty	++	+	-	-
Tumble Trouble	+	-	-	++
Chomp Chomp	++	++	-	-







Figure 2: (a) How each game weighs against the four CT skills. Each + denotes a stronger relationship, each - denotes a weaker relationship. (b), (c), (d), (e), (f) each show in-game screenshots of Manta Match Mania, Tumble Trouble, Pearly Whirly, Crabby Patty and Chomp Chomp respectively.

### 3. DESIGN OF A HYBRID APPROACH FOR K1 & K2

This section describes the design of a unique approach for teaching the complex CT fundamentals in a way that can be understood by younger children. We also discuss how teachers facilitate the additional activities and how they can evaluate and report the students progression in the curriculum.

#### 3.1 Design Requirements

From looking at current kindergarten curriculum as well as general feedback from acting kindergarten principals and teachers, we summarize that the curriculum design requires the following:

- Children should learn through play and exploration  
Children should be encouraged to learn even if the concept is complex
- Children should be exposed to digital medium whilst applying concepts to real world scenarios, limiting their screen time
- Curriculum should be intuitive for teachers to understand and teach, even if they are not proficient in the subject being taught
- Curriculum should be designed so that teachers are only required to supplement the lessons, and can be done with little to no pre-requisite knowledge of the subject
- Teachers should be able to evaluate the progress of the class and/or an individual student

#### 3.2 Teaching Through Animations

As the starting sequence in scaffolding, children would watch a pre-scripted video animation when they are first introduced to a new complex CT concept. The animation features “Doodle” as the primary teacher cum online character who will engage the children; complementing the “offline” kindergarten teacher whose role is to re-enforce learning. This allows teachers with limited CT proficiency to confidently teach these complex concepts. The animations are done in the same spirit of educational children TV shows, utilizing pauses between questions as well as humorous gags to keep the attention of the children and allow them to actively engage. The animations are ordered to first introduce each CT skill, provide examples on what this skill entails, then expand and show how the skill is applied to real-world situations.

#### 3.3 CT Practice Through Games

The online practice is provided via the digital application which is run on an android tablet device. This application features the child avatar known as the ‘Buddy’ who builds a relationship with the child and game story as the Buddy helps them in small ways (Such as giving hints on how to complete difficult levels). This further enhances the engagement whilst relieving the anxiety of the educational factor being displayed to the children. The application contains six games which incorporate one or more CT elements in the gameplay (See Table 4(a) for CT relations). By transposing CT exercises via gamification, we are able to allow kids a safe virtual environment to practice CT skills. The 5 Games which are included in the School of Fish application are:

**Pearly Whirly:** This game instructs kids to pre-program the ‘Sally Submarine’ to navigate through a maze and collect each of the pearls. The kids pre-program a series of either a ‘left’ or ‘right’ command. Upon execution the submarine will continuously move forward whilst making either a left or right turn at each junction based on the next command in a sequence. The level is completed when Sally is able to navigate the maze and collect all of the pearls in one sequence of inputs.

**Manta Match Mania:** This game runs in the same spirit of a tangram puzzle. Players utilize the ‘junk’ puzzle pieces on the right and arrange them so that they cover the requested ‘junk part’ on the left. This needs to be done within a given time frame otherwise the player loses one of 3 lives and retries the puzzle. Each time a ‘junk part’ is successfully constructed the player earns some ‘pig coins’ and continues to the next puzzle. The game is completed when enough pig coins are collected.

**Crabby Patty:** Players are presented with a 3x2 or 4x2 array of crabs who will pose to form a pattern, with one of the crabs being hidden under a bucket. The aim is to select one of four solutions which they think matches the hidden crab. This is repeated until all the puzzles are solved, with incorrect answers removing one ‘life’. The game ends when all puzzles are solved, or all lives are lost.

**Tumble Trouble:** Colored critters fill the screen, and the player tries to clear the critters by drawing lines to match 4 or more in a row. This game adds two twists; first they must clear a specific number of critters from a limited



supply, second a special 'clam' critter requires surrounding critters to be cleared several times before the clam is cleared. The game is complete either when these two goals are met, or there are no more possible moves.

**Chomp Chomp:** A supplemental game. Players are presented with their buddy requesting a particular kind of food, and a 5x5 grid of randomized food from 5 particular types. The objective of the player is to 'feed' the buddy by filling his 'hunger gauge'. They do this by swapping food around to match 3 of the same type of food which fills this gauge. The game ends when the hunger bar is filled.

### 3.4 Integration of Offline Activities

The final sequence of the scaffolding journey where offline activities is used to reinforce the skill acquisition process by getting children to apply the CT skills learnt through the online games to real-world teacher-led play activities. This is implemented with the toys and equipment the kindergartens already have in classrooms to perform activities which practice CT and problem-solving as play activities, so as not to discourage kids from interacting and allows the kids to enjoy the learning experience. Teachers help the kids follow the instructions given and are instructed to allow the kids to figure out the solutions themselves. Some examples of these activities include but are not limited to:

- Making various animals with building blocks
- Recognizing patterns from the surrounding environments
- Planning the steps of what the child will do during the day
- Breaking a big jigsaw puzzle into smaller parts then use abstraction to group them, making the puzzle easier to solve
- Breaking a large math equation into smaller parts

### 3.5 Evaluating and Grading student Performance

Teachers require a means of grading and evaluating the progress of a student through the curriculum. A method of grading is provided via a dashboard application, which allows teachers to mark attendance to modules and track the child progress. This progress is empirically evaluated in two ways, The CT competency index and the puzzle quiz delivered at the start, mid and end of the curriculum. We define the CT competency index as an empirical point system and allocate points across three main topics;

- Curriculum modules: for which a child is awarded points upon completion of the given module
- Animations: for which a child is awarded points upon watching one of the Doodle animated lessons.
- Online Activities: Each of the core CT Games described in Section 3.3 have 100 levels. Each of these levels can be completed with a rating from 1 to 3 stars. 3 stars are given if the best approach/solution to the level was used. The total of all earned stars for each game contribute to points in the CT skill category which that game practices.

The curriculum modules only comprise 6 score for two reasons; One is that the animations and online activities are usually a subset of the curriculum, hence a big part of the score is redistributed into the animated episodes (where the concepts are taught) and the online game activities (where they are practiced and reinforced). The second reason is that the delivery of these classes cannot be monitored, making the marking of these modules a subjective judgment from the teacher (which they do by marking the student as attended) and therefore cannot be empirically measured.

The final raw CT score is calculated as follows:

$$\text{RawCTSubjectScore} = \text{GameContributions} + \text{ModuleContributions} + \text{AnimationContributions}$$

Where:

$$\text{GameContributions} = \frac{\sum_{i=1}^4 (\text{CTGameScore}[i] \times \text{Multiplier}[i])}{4}$$

For each of the four games that contribute to the final raw score. We do not directly show the raw scores to the teacher, instead we show a graphical comparison of either a child's score compared to the rest of the class or a child's score against the rest of enlisted users via a percentage comparison. This allows the teacher to highlight that a child may be weaker in a particular skill and can suggest ways that that child can improve to the parent when reporting progress. We also use a variation of Raven's progressive matrices (Burke 1985) to test their ability to systematically decompose patterns, selecting the correct missing piece. This variation does not calculate IQ, but only the raw correct answers as a grading metric. This test is taken before the start, halfway during, and after the curriculum is complete.

## 4. PILOT STUDY: EVALUATING THE TEACHING EFFECTIVENESS

This pilot study aims to validate how effectiveness of the methodology and curriculum described in this paper in teaching the fundamental CT skills. For this we developed a 12-hour variation of our curriculum, containing the introduction to each CT skill, some online interactive practice in class via the tablet device, and some offline activities. Additional worksheets are given out to be completed outside of class hours either with teacher or parental supervision. Before the classes begun, the children were given the introduction to the course where they learn how to use a tablet. They are then asked to complete the puzzle test described in Section 3.5. This test is given a second time after the completion of the course. Our hypothesis stated that:

**H1** Children improve the amount of correct answers given in the puzzle test after taking the classes

**H2** Any improvement is independent from whether the children have previously used a tablet device before.

### 4.1 Participants

Two classes of mixed K1 and K2 students aged 4-6 years old were selected from volunteer kindergartens. The first class C1 has had no previous exposure to tablet devices whilst C2 already uses some tablets as part of their curriculum. Before forms were distributed by the teachers to the child's parents, only children whose parents have

completed and signed the form were allowed to participate. C1 contained 15 students comprised of 6 female and 9 males, C2 comprised of 10 students 5 male, 4 female. Making 25 students total. Later 2 males from C1 and 2 females and 1 male from C2 were removed from the experiment results due to absence from some of the classes.

#### 4.2 Task and Procedure

A 12-hour implementation of the designed described in Section 3 is used in this experiment. In this shortened version the modules which introduce each of the fundamental CT concepts as well as one online and one offline activity which practices the respective skill is used. The children first learn how to use a tablet and conduct the first puzzle test (described in Section 3.5) on the first day, then the classes run on days 2 - 5, The second puzzle test is then executed on day 6. These classes are taught by a researcher whilst a kindergarten teacher is present at all times to supervise and facilitate as needed. Please see the additional materials for this 6-day curriculum.

#### 4.3 Variables

**Dependent Variables:** Empirically we looked at the amount of correct answers given in the test. Each answer is collected as a binary outcome. Observations of how the students partake in the classes and engage with the content are made and general feedback from the supervising kindergarten teachers and principals are collected through interviews.

**Independent Variables:**

**Class** { C1, C2 } between-subjects

We measured the scores between the two classes to see if having previous experience with the tablet device causes any effect on the effectiveness of the curriculum. C1 has had no previous interaction with tablet devices while C2 has.

**TestTakenAt** { Pre, Post } within-subjects

We measured the scores of the puzzle test before and after the 10-hour course was taken to see if there is an improvement.

#### 4.4 Results

The results of this pilot study are described in three ways; The directly measured variables, the observations made during class participation and the feedback given by the supervising teachers & principals

**Measured Results:** As this was a between subjects study, for each of the measured dependent variables described in Section 4.3 we analyse each the measured Dependent Variables using a two-way ANOVA test against the Independent Variables. H1 stated that after the classes the children will have more correct answers. Figure 3 shows at what rate a child answered each question correctly before and after the classes, from this we can say this improvement is applied to questions which previously had a lower percentage of being correct. The results from the two-way repeated measures ANOVA test further support **H1**, showing a significant interaction effect between the pre and post-test scenario for both C1 & C2 on the amount of correct answers in the puzzle test with confidence level  $p < 0.05$  ( $\sim 0.003$ ). **H2** states that any improvement in test

results is independent from whether the children have previously used a tablet device before. Both two-way ANOVA results highlighted in Figure 10 and 11 show that there is no significant interaction ( $p > 0.05$ ,  $\sim 0.379$ ) between the two classes on the amount of correct answers in the Pre and Post-test environments, therefore **H2** is accepted.

**Observed Results:** The children were actively engaging with the classes; they answered the questions that were queried by Doodle during the animated episodes. They would answer questions asked by the experiment conductor

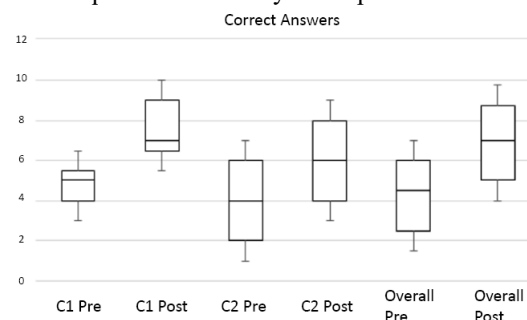


Figure 3. Results of the pilot study. We can see some improvement of the amount of correct answers and that this improvement is similar in both classrooms. The large variance does suggest that the sample size might be too small.

as well as the supervising teacher. The children at first had difficulty engaging with the online activities but after a small amount of practice were able to complete the given activities. An interesting observation was made during the execution of the puzzle tests. The students took longer and were systematically solving the questions in the post test environment.

**Feedback Results:** The teachers and principals were briefly interviewed before and after the classes and posttest were conducted. Overall principals were positive about the unique style of how the classes were executed. At first they rejected the idea of tablet computers being used in class but after watching how the kids actively engaged during tablet play they later retracted their rejection. They were concerned that some training (although minimal) in the use of the game and dashboard applications might be required in order for such a curriculum to be effective

#### 4.5 Discussion

Our measured results support both **H1** and **H2**. We recognize that the sample size is too small for a within subjects experiment. This was unavoidable as kindergarten classes typically only contain 5-15 students per supervising teacher. Even with this small size the results were significant. We observed that during the posttest the children took a more systematic approach to solving the puzzles. This raises the question as to whether a child exposed to this teaching method will take a different approach to solving problems due to a changed mindset and can be investigated in future studies. We also observed that the children engaged very well with the Doodle

character as he taught the concepts in the animated videos and enjoyed the online activities in the tablet. The feedback from the teachers additionally state that the children enjoyed practicing these skills outside of the classroom as well as through the interactive offline activities.

## 5. CONCLUSION & FUTURE WORK

This work represents a first step into a method of teaching CT to a K1 & K2 audience, and opens several new venues for future work. Although the experiment described in this paper validated the hypothesis that our unique hybrid design of offline and online activities is effective in teaching a subject as complex as CT to a K1 & K2 audience, we acknowledge that these results are only preliminary and are an estimate due to a small sample size. Still, these results are significant and suggest that future work involve the full implementation of a curriculum which utilizes this hybrid approach be completed. Furthermore, a repeated experiment using this full implementation with a larger sample size will lead to the same conclusions. The results also suggested after the children were exposed to this new problem-solving methodology, they took a new approach to solving the puzzle test. This raises the question on whether we need new test methods to further evaluate each of the fundamental CT skills individually rather than as a whole. Additional future psychological studies can possibly reveal on how a child's problem-solving mindset changes after being exposed to a curriculum which teaches CT methodology.

## 6. ADDITIONAL MATERIALS

Additional materials can be found online here: <http://bit.ly/3s0UY84>

These materials include the shortened curriculum used for the study, a table summarizing each game's contribution to the overall CT score, and sample questions used as part of the study's evaluation. Sample Doodle episodes can be found here: <http://bit.ly/3vkdBGp>

## 7. REFERENCES

- Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. *Acm Inroads*, 2(1), 48-54.
- Burke, H. R. (1985). Raven's Progressive Matrices (1938): More on norms, reliability, and validity. *Journal of Clinical Psychology*, 41(2), 231-235.
- Burns\*, C., & Myhill, D. (2004). Interactive or inactive? A consideration of the nature of interaction in whole class teaching. *Cambridge journal of education*, 34(1), 35-49.
- Elliott, A., & Hall, N. (1997). The impact of self-regulatory teaching strategies on "at-risk" preschoolers' mathematical learning in a computer-mediated environment. *Journal of Computing in Childhood Education*, 8.
- Falkner, K., Vivian, R., & Falkner, N. (2015, January). Teaching computational thinking in k-6: The cser digital technologies mooc. In *Proceedings of the 17th Australasian computing education conference (ace)* (Vol. 30).
- Grover, S., & Pea, R. (2013). Computational thinking in K-12: A review of the state of the field. *Educational researcher*, 42(1), 38-43.
- Guzdial, M. (2008). Education Paving the way for computational thinking. *Communications of the ACM*, 51(8), 25-27.
- Kazimoglu, C., Kiernan, M., Bacon, L., & MacKinnon, L. (2012). Learning programming at the computational thinking level via digital game-play. *Procedia Computer Science*, 9, 522-531.
- Lieberman, D. A., Bates, C. H., & So, J. (2009). Young children's learning with digital media. *Computers in the Schools*, 26(4), 271-283.
- Meierhenry, W. C., & Stepp, R. E. (1969). Media and early childhood education. *The Phi Delta Kappan*, 50(7), 409-411.
- Montemayor, J., Druin, A., Chipman, G., Farber, A., & Guha, M. L. (2004). Tools for children to create physical interactive storyrooms. *Computers in Entertainment (CIE)*, 2(1), 12-12.
- Papert, S. A. (2020). *Mindstorms: Children, computers, and powerful ideas*. Basic books.
- Perlis, A. (1962). The computer in the university. *Computers and the World of the Future*, 180219.
- Ramírez Verdugo, D., & Alonso Belmonte, I. (2007). Using digital stories to improve listening comprehension with Spanish young learners of English. *Language Learning & Technology*, 11(1), 87-101.
- Ritterfeld, U., & Weber, R. (2006). Video games for entertainment and education. *Playing video games: Motives, responses, and consequences*, 399-413.
- Samuelsson, I. P., & Carlsson, M. A. (2008). The playing learning child: Towards a pedagogy of early childhood. *Scandinavian journal of educational research*, 52(6), 623-641.
- Sung, Y. T., Chang, K. E., & Liu, T. C. (2016). The effects of integrating mobile devices with teaching and learning on students' learning performance: A meta-analysis and research synthesis. *Computers & Education*, 94, 252-275.
- Turbill, J. (2001). A researcher goes to school: Using technology in the kindergarten literacy curriculum. *Journal of Early Childhood Literacy*, 1(3), 255-279.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.

# Using the Beginners Computational Thinking Test to Measure Development on Computational Concepts Among Preschoolers

María ZAPATA-CÁCERES<sup>1\*</sup>, Nardie FANCHAMPS<sup>2,3</sup>

<sup>1</sup>Universidad Rey Juan Carlos, Spain

<sup>2</sup>Fontys University of Applied Science, Netherlands

<sup>3</sup>Zuyd University of Applied Science, Netherlands

maria.zapata@urjc.es, nardie.fanchamps@fontys.nl

## ABSTRACT

The implementation of programming in primary education is in the forefront of attention in many countries. The application of programmable robots offers many opportunities to learn the basic concepts of programming. Learning and understanding these underlying concepts is not only reserved for students of five years and older but can also be learned at a younger age. Until now, making a development on Computational Thinking (CT) objectively measurable among preschoolers was not possible since no validated instrument was available for this purpose. Furthermore, it is unclear which capabilities of CT are achieved at each age and which are not reachable. To establish which CT skills are of interest to students and within the reach of each age group and therefore, teachable, this study has been carried out. To assess CT, the Beginners Computational Thinking test (BCTt) was used, along with direct observation and interviews. Results show the suitability of the BCTt among 5 years-old students and, partially among 4 years-old students. When applying two types of programmable robots a significant increase in the development of CT was observed. A development of specific complex programming concepts can also be demonstrated. In addition to the skills shown, it also appears that children are highly motivated to learn programming at a very young age.

## KEYWORDS

Computational Thinking, preschoolers, primary education, programmable robots, assessment

## 1. INTRODUCTION

Currently, many countries have focused on learning Computational Thinking (CT) increasingly at earlier stages such as early childhood education (Bocconi et al., 2016). However, there is still scarce knowledge available as to whether very young children can learn CT-skills in an understandable and applicable way (Hunsaker, 2018). Many questions remain, such as: can preschoolers understand and functionally apply underlying programming concepts, which addressed computational concepts can young children learn at all, and how can a development in CT among young children be made measurable even when they are not able to read?

Computational Thinking was redefined as the ability to solve challenging problems using skills derived from the world of computer science (Wing, 2006). Since then, several definitions are found in the literature, nevertheless, they can be summarized as “the mental skills and practices for: a) designing computations that get computers to do jobs for us, and b) explaining and interpreting the world as a complex of

information processes.” (Denning & Tedre, 2019). The CT components that arise more often in literature are abstraction, decomposition, algorithms, and debugging (Shute, Sun, & Asbell-Clarke, 2017).

Constructivism in education (Vygotsky & Cole, 1978) and the developmental theory (Piaget, 1972), establish dynamic and collaborative learning where the learner actively creates knowledge. Based on these leaning theories and in reaction to technological innovations, various theoretical frameworks have emerged. Papert addressed CT primarily as the relationship between programming and thinking skills, and its possible further application to multiple disciplines (Papert, 1993). Furthermore, Collaborative learning promotes critical thinking skills much more effectively than do individualistic environments (Johnson & Johnson, 2008).

The CT 3d framework (Brennan, Resnick, & MIT Media Lab, 2012) divides CT into three facets: a) computational concepts (the concepts applied in programming), b) computational practices (the problem-solving practices arising while programming), and c) computational perspectives (the views that programmers hold about themselves, their relationships to others, and the technological world around them). The 3d Framework is adopted in this article as is considered to provide a wide coverage of CT for young students. It is expected from learners to be able to understand CT concepts and develop CT practices and perspectives with increasing maturity depending on their age (Kong, 2016).

Based on these theories and since using computer technology and programming at school is a way of learning these skills from a very early age (Lye & Koh, 2014), programmable robots are ideally suited for this purpose, because these environments can make the output of a created computer program concrete and tangible (Fanchamps, Slangen, Hennissen, & Specht, 2019). Moreover, collaborative peer-based environments enhance reflection and improves CT learning experiences (Buitrago Flórez et al., 2017). In our research project, we are therefore interested in whether a development on CT, through a better understanding of underlying programming concepts, using collaborative robotics environments is also feasible for preschoolers.

Previous research indicates that preschoolers are able to program simple robotics projects in a playful and engaging way, enhancing their computer thinking skills, while they work collaboratively (Bers, Flannery, Kazakoff, & Sullivan, 2014), as they not only learn computer concepts but solve problems by breaking them down into steps; they use abstraction and algorithms to translate the problem into a

codable language; and they apply debugging, as they must modify their solution if it is not correct. To use Blue-bot seems to be adequate to introduce robotics to preschoolers (Alvarez, Bellegarde, Flahaut, & Lafouge, 2018); similarly, there is evidence of the use of Cubetto at schools as a suitable robotic environment for preschoolers for a first programming experience (Sáez Fernández, Viera López, & Pérez Marín, 2018).

With regard to the assessment of CT, several instruments have been developed over the last two decades, but most of them are aimed at middle or high school students and are based on interviews or project analysis based on block programming, such as Fairy assessment in Alice (Werner, Denner, Campe, & Kawamoto, 2012). To make this development for preschoolers measurable, we are using the Beginners Computational Thinking test (BCTt) (Zapata-Cáceres, Martín-Barroso, & Román-González, 2020), which has been developed specifically for use among young children and has proven to be reliable for the assessment of CT focusing on 3d framework computational concepts, as we demand in this research. However, it is recommended to be used in parallel with other assessment tools as a system of assessments (Grover, Pea, & Cooper, 2015), therefore, qualitative data is also collected as part of our research to address other 3d framework CT dimensions.

Considering the research that has already been carried out, we are interested in the following main research question: Is the BCTt usable among preschoolers and what interpretation can be made regarding the measured results?

To answer this main research question, the following sub-questions have been formulated: (1) To what extent can the BCTt be used among preschoolers?, (2) Are preschoolers capable of understanding underlying programming concepts?, and (3) To which addressed computational concept can a development among preschoolers be measured?

The research has been carried out under the following hypothesis: (1) Applying collaborative peer-based programming environments that provide the opportunity to understand programming concepts at an early age, motivates young children to learn how to program, (2) preschoolers, who learn to program making use of Blue-bot or Cubetto, show a measurable understanding of programming concepts, and (3) programming with Blue-bot or Cubetto contributes to the development of CT-skills among preschoolers.

Our elaboration describes the method used, the participants involved, the case study designed, and the results obtained from the data analysis. Finally, conclusions, implications and suggestions for further research are presented.

## 2. METHOD

To investigate our research questions, we used a pre-test and post-test design. This includes a) pre-assessment of CT-skills among preschoolers, b) a robotics-intervention using two different types of programming environments, and c) a post-test among preschoolers assessing CT-skills.

In addition to the data obtained by applying the BCTt, we also collected qualitative data via direct observation and

interviews, asking preschoolers various questions about the experience at the end of the programming sessions.

### 2.1. Participants

This study was conducted among preschoolers in the age category 4 and 5 years old of a K-12 School in The Netherlands. Preschoolers were divided randomly into pairs and assigned to an experimental group to conduct programming sessions with either Blue-bot (n=10) or Cubetto (n=10). The control group (n=36) did not participate in the programming sessions.

### 2.2. Materials

In the two treatment conditions we used Blue-bot and Cubetto as programming environments. Two different types of physical robots are to be programmed. The programming itself is performed by applying physical command cards, each containing a specific programming command (e.g. forwards, backwards, left, right, etc.). To compose a program these command cards must be sequenced in the correct order in a command reader, which creates a programming string. This created program can then be sent to either the Blue-bot or Cubetto robot via a Bluetooth connection. In both environments the following concepts were addressed: “sequence”, “loop-simple”, “loop-nested” and “conditional if-then”.

To measure an effect on preschoolers CT-skills, the validated BCTt was used as a pre-test and post-test. The BCTt consists of 25 questions in which preschoolers must link programming sequences to different visual situations, e.g. see Figure 1, so 3d framework computational concepts and, partially, computational practices are assessed. The specific concepts addressed in the BCTt are sequences, loop-simple, loop-nested, conditional if-then, conditional if-then-else, and conditional while.

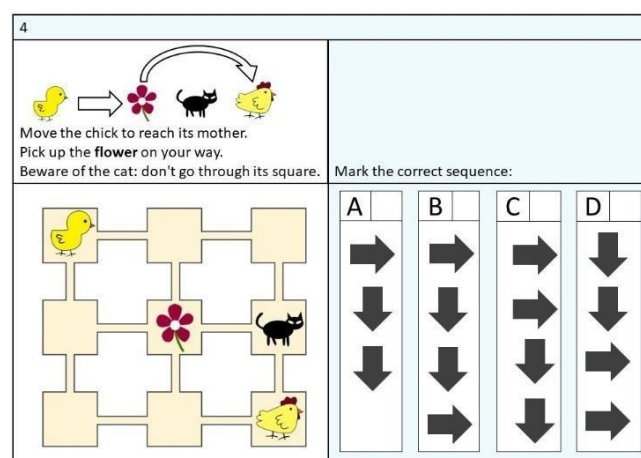


Figure 1. BCTt question example (question number 4).

To determine the reliability of the BCTt for the specific research sample, the overall Cronbach's alpha was calculated ( $\alpha = 0.911$ ). The designers of the BCTt indicate a value of  $\alpha = 0.824$  for Cronbach's alpha. From this we can conclude that we amply meet the requirements for internal reliability. In addition, qualitative data has been collected to address other CT-skills: direct observation during the programming sessions; and interviews at the end of the sessions in which students were asked orally about their

feelings when interacting with the environments and with their peers, and other questions related to what they were supposed to learn and the characteristics of the environments.

### 2.3. Procedure

To enable young students who have had no previous contact with computer concepts and with low or no reading and writing skills to take the test, the BCTt includes very few texts just to support the symbols used for the statements, yet the translation to Dutch language of these texts has been calibrated by a research group of 8 education experts.

Prior to the programming sessions, the BCTt was administered a pre-test to the whole sample ( $N=56$ ). Because preschoolers of this age category cannot yet read, in accordance with the accompanying protocol, each question is read out loud two times. At the same time, preschoolers looked at the corresponding image and determined which programming sequence they think is correct. Subsequently, preschoolers were divided ad-randomly over the two treatment conditions and the control group, obtaining equally balanced groups.

For five weeks, one group then received five programming sessions of 30 minutes each using Blue-bot. In the same conditions, the other group received five programming sessions using Cubetto. During the closing session, preschoolers which used Blue-bot had to solve a programming problem by applying the “repeat” loop; similarly, preschoolers which used Cubetto had to solve a programming problem by applying the “if-then” function. After completion of the programming sessions, the BCTt was again administered as a post-test in all groups in the same conditions in order to be able to identify differences in the CT development. In addition, qualitative data was collected via direct observation while interacting with the programming environment and with interviews conducted after the programming experience.

## 3. RESULTS AND DATA-ANALYSIS

In order to be able to answer the main research question by means of the formulated sub-questions and hypotheses to be investigated, both qualitative (inventory) and quantitative data (SPSS) were analyzed.

### 3.1. Qualitative data

As the BCTt assesses 3d framework computational concepts and, partially, computational practices, but ignores computational perspectives, qualitative data is collected, through direct observation and with interviews at the end of the programming experience, in order to broaden the CT assessment.

Students were highly motivated and enthusiastic throughout the entire programming experience, showing strong self-regulation and deep attention over long periods of time. The group using Cubetto were interested on using Blue-bot at a later date and vice versa. Direct observation shows that students understand the pictogram-based questions and are able to make the link between the visual definition of the problem and the solutions to be found represented by combinations of directional arrows from which the correct

answer should be chosen. No differences were observed in terms of gender, approach, motivation, or skills, but they were between 4 and 5 years-old students, as there was a limit on what 4 years-old students were able to understand.

With regard to CT components, students could think of a task to be executed by the robot, make an abstraction dividing the problem into smaller parts and translate it into an algorithm. Similarly, their persistence in the search of the answer was remarkable, since they were able to change their minds and correct their code errors, using the debugging process. Furthermore, the peer-based collaborative environment was a strong support for reflection, as thinking aloud and finding the solution together with a peer was key to solving the problems.

Further observations show that both age categories understand the underlying principles of sequencing in order to be able to compile a program to be executed via the concatenation of the directional arrows as commands. For 4-year-old students the limit seems to be in questions that include loops. Students 5 years-old can presumably understand and answer the questions related to loops as well as those related to conditionals. Considering the results of the first research question on to what extent the BCTt can be used among preschoolers, it becomes clear that this instrument is applicable among 5 years-old students and 4 years old students to some extent. This was later confirmed by the quantitative data results.

Regardless of the programming environment used, the oral interviews conducted at the end of the programming experience also show very high motivation in the students, as the most common feelings they expressed when talking about the experience were those of happiness and joy, and that to solve problems make them feel smart. All the students like to use either cards or the Tablet to program. 90% of preschoolers who used Blue-bot thought they fully understood the concept of 'repetition', compared to 70% of those who used Cubetto. Although waiting their turn makes 65% of students feel "impatient" or "unhappy" and 20% find it difficult to ask their peers for help, the collaborative methodology was well rated by the 100%, who felt “happy”, “glad” or “good” about working with a partner. All the students felt that they have learned something new and happened to know their right and left better.

### 3.2. Beginners Computational Thinking Test (BCTt)

To examine the hypotheses formulated in this research and to determine whether, and if so, which of the two programming environments used, both Blue-bot or Cubetto lead to significant differences with respect to the control group, and/or whether significant differences may occur in a comparison between the two programming environments, a variance analysis (*Anova*) with Levene's test was performed. Subsequently, post-hoc tests were performed to demonstrate possible significant effects and to confirm or reject hypothesis. Eta squared ( $\eta^2$ ) was calculated to reveal the magnitude of the effects. All statistical analyses assume a significance level of 5% ( $p \leq 0.05$ ). The results concerning the second hypothesis whether preschoolers, who learn to program using Blue-bot or Cubetto, show a measurable



understanding of underlying programming concepts are shown by a development on the averages (Table 1).

**Table 1. Differences by Computational Concept Addressed**

Blue-Bot	Pre-assessment		Post-assessment	
	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>
Total (25)	0.27	.21	0.42	.27
Sequence	0.36	.23	0.72	.22
Loop simple	0.36	.28	0.54	.25
Loop nested	0.11	.16	0.23	.30
Conditional if-then	0.15	.24	0.25	.35
Conditional if-then-else	0.00	.00	0.25	.35
Conditional while	0.60	.74	0.33	.19
Cubetto	Pre-assessment		Post-assessment	
	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>
Total (25)	0.30	.21	0.48	.27
Sequence	0.44	.19	0.87	.15
Loop simple	0.42	.33	0.64	.34
Loop nested	0.16	.23	0.21	.27
Conditional if-then	0.10	.21	0.25	.35
Conditional if-then-else	0.05	.16	0.30	.42
Conditional while	0.50	.85	0.37	.48
Control group	Pre-assessment		Post-assessment	
	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>
Total (25)	0.32	.20	0.27	.19
Sequence	0.48	.18	0.55	.28
Loop simple	0.43	.33	0.33	.31
Loop nested	0.15	.23	0.13	.19
Conditional if-then	0.11	.27	0.10	.20
Conditional if-then-else	0.10	.23	0.10	.23
Conditional while	0.50	.74	0.10	.19

*Note.* Total = number of questions correct BCTt questionnaire; *M* = average; *SD* = standard deviation.

Students who have programmed making use of Blue-bot or Cubetto show a development on programming concepts in a direct comparison with the control group. Table 1 shows that pupils who applied Blue-bot or Cubetto successfully solved more computational thinking issues and developed more understanding of the programming concepts “sequence”, “loops” (simple, nested) and “conditionals” (if-then, if-then-else). A development on “conditionals-while” could not be demonstrated.

The answer to the third hypothesis, whether programming making use of Blue-bot or Cubetto contributes to a development of CT-skills among preschoolers, can be deduced from the data presented in Table 2. The data show that in both treatment conditions, in contrast to the control group, a significant development can be measured for the

total number of computational thinking issues solved (Total) and for the computational concepts addressed “sequence”, “loop-simple”, and “conditional-while”. A further examination of the data by applying post-hoc tests reveals that 1) for the “total number of computational thinking issues solved” both Blue-bot and Cubetto cause the significant effect, 2) for “sequence” Cubetto causes the significant difference, 3) for “loop-simple” Cubetto causes the significant difference, and 4) for “conditional-while” Cubetto causes the significant difference. Despite a strong development on the averages, as shown in Table 1, no significant increase can be demonstrated for the computational concepts addressed “loop-nested”, “conditional if-then” and “conditional if-then-else”.

**Table 2. Analysis of Variance (Anova)**

	Quantity						
	<i>SS</i>	<i>df</i>	<i>MS</i>	<i>F</i>	<i>p</i>	<i>SD</i>	$\eta^2$
Total (25)*	.485	2,53	.243	.491	.011	.24	.156
Sequence*	.858	2,53	.429	.653	.003	.28	.198
Loop-simple*	.934	2,53	.467	4.99	.010	.33	.158
Loop-nested	.097	2,53	.049	.923	.404	.23	.034
Conditional if-then	.300	2,53	.150	2.17	.124	.27	.076
Conditional if-then-else	.419	2,53	.209	2.40	.101	.30	.083
Conditional while*	.797	2,53	.399	4.09	.022	.33	.134

*Note.* Quantity = measured value; Total = number of questions correct BCTt-questionnaire; Computational concept addressed = sequence, loop simple, loop nested, conditional if-then, conditional if-then-else, conditional while; *SS* = sum of squares; *DF* = degrees of freedom; *MS* = mean square; *F* = f-value; *p* = significance level; *SD* = standard deviation;  $\eta^2$  = Eta squared; \*significant effect measured.

To determine if preschoolers aged 4 and 5 years old are capable of understanding the BCTt underlying programming concepts, a comparison between both ages was performed. Results on Table 3 show that the limit for 4-year-old preschoolers is the understanding of the concepts “sequence” and “loop-simple”. On the other concepts (“loop-nested”, “conditional if-then”, “conditional if-then-else”, “conditional while”) no further development is measurable. This in contrast to the 5-year old preschoolers who show a development on all concepts, except for the “conditionals while” concept.

**Table 3. Age Difference by Computational Concept**

Age 4 years	Pre-assessment		Post-assessment	
	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>
Total (25)	0.17	.06	0.19	.10
Sequence	0.37	.16	0.53	.26
Loop simple	0.24	.18	0.28	.26
Loop nested	0.00	.00	0.00	.02
Conditional if-then	0.00	.00	0.00	.00

Conditional if-then-else	0.00	.00	0.00	.00
Conditional while	0.00	.00	0.00	.00
Age 5 years	Pre-assessment		Post-assessment	
	<i>M</i>	<i>SD</i>	<i>M</i>	<i>SD</i>
Total (25)	0.52	.14	0.56	.21
Sequence	0.59	.17	0.80	.24
Loop simple	0.69	.30	0.64	.31
Loop nested	0.38	.18	0.42	.17
Conditional if-then	0.30	.33	0.39	.31
Conditional if-then-else	0.18	.29	0.41	.37
Conditional while	1.32	.65	0.48	.37

Note. Total = number of questions correct BCTt-questionnaire; *M* = average; *SD* = standard deviation.

Furthermore, Table 4 presents a t-test analysis to assess whether there are significant differences between preschoolers aged 4 years and 5 years old concerning the understanding of computational concepts addressed in the BCTt. From these findings, it can be noted that 5-year old preschoolers score significantly better than 4-year-old preschoolers on all the concepts present in the BCTt.

Table 4. T-test analysis comparing 4 and 5 years old

	Quantity				
	<i>t</i>	<i>df</i>	<i>p</i>	<i>CI</i>	<i>d</i>
Total (25)*	-7.69	27.29	0.000	-0.47- -0.27	2.24
Sequence*	-3.85	47.29	0.000	-0.40- -0.12	1.04
Loop-simple*	-4.44	38.62	0.000	-0.52- -0.19	1.24
Loop-nested*	-11.26	21.56	0.000	-0.49- -0.34	3.38
Conditional if-then*	-5.92	21.00	0.000	-0.52- -0.25	1.79
Conditional if-then-else*	-5.24	21.00	0.000	-0.57- -0.25	1.57
Conditional while*	-6.20	21.00	0.000	-0.65- -0.32	1.87

Note. Total = number of questions correct BCTt-questionnaire; Computational concept addressed = sequence, loop simple, loop nested, conditional if-then, conditional if-then-else, conditional while; *t* = *t*-value; *df* = degrees of freedom; *p* = significance level; *CI* = confidence interval; *d* = Cohen's *d* effect size; \*significant effect measured

#### 4. CONCLUSIONS

Preschoolers showed understanding of the underlying programming concepts and a significant overall improvement in regard to CT-skills with mayor size effects compared to those of the control group. In addition, preschoolers were highly motivated when working in pairs when using robotic programming environments. Therefore, it could be concluded that collaborative peer-based environments are appropriate and enhance the learning of

programming concepts at an early age, thus, it is advisable that CT-skills start to be taught at least at the age of 4.

Both Blue-bot and Cubetto were proved suitable for preschoolers and caused a significant overall improvement in CT-skills. Cubetto causes the most significant improvements in specific computational concepts, our hypothesis is that its layout allows children to abstract (one of the computational practices of the 3D framework) more easily from the proposed code, since a) command cards are displayed as a path, and b) the command cards are in the shape of the symbol used. On the contrary in Blue-bot a) the sequences are displayed from top to bottom, and b) the symbols are drawn inside de command cards. Although Cubetto seems more suitable for preschooler, we suggest using different programming environments with the same sample of students, so that they can benefit from them in a cross-curricular way, as both contribute to the development of CT-skills among preschoolers.

However, the concepts that can be taught to each age group are different. From our results we can deduce that 5 years-old students are able to understand all computational concepts addressed in the BCTt, and show a significant improvement, after the programming sessions, in each of them (except in the "conditional while"), including the "conditional if-then-else", even though this concept was not practiced in any of the programming sessions. The difference between the results of the two age groups is very significant, as 4 years-old students show understanding and improvement on only two of the six concepts addressed in the BCTt ("sequence" and "loop-simple"). Since the "conditional if-then" was practiced in the programming sessions and no understanding nor improvement was shown, it can be concluded that "conditional if-then" and, therefore, "conditional if-then-else" might not be reachable concepts for 4 years-old students. However, although they did not show understanding or improvement in the "conditional-while", since this concept was not addressed in the programming environments, it cannot be stated with certainty that it is not within the reach of 4-year-olds.

The BCTt shows a very high reliability as an instrument to assess CT-skills for 4 ( $\alpha = 0.802$ ) and 5 years-old students ( $\alpha = 0.889$ ) and, in combination with qualitative data, provides an adequate CT assessment for these age groups. Both 4- and 5-year-old students can complete the BCTt, understand the pictograms and are able to interpret the programming sequences posed as possible solutions. However, four of the six computational concepts addressed in the BCTt seem not reachable for 4 years-old students, as they were not able to answer the questions related to them nor show any improvement in the post-test in regard to those concepts. For this reason, we suggest a new version of the BCTt should be made targeted to 4 years-old students and/or younger students. Furthermore, the research should be replicated with larger samples, more programming sessions (with a duration according to the limited attention span of very young children) and in other countries to confirm our findings.



## 5. ACKNOWLEDGMENTS

The authors would like to thank primary school De Hovenier Montfort, the Netherlands, for their cooperation.

## 6. COMPLIANCE WITH ETHICAL STANDARDS

The Ethical research board (cETO) of the Open University of the Netherlands has assessed the proposed research and concluded that this research is in line with the rules and regulations and the ethical codes for research in Human Subjects (reference: U2019/01324/SVW)

## 7. REFERENCES

- Alvarez, J., Bellegarde, K., Flahaut, J., & Lafouge, T. (2018). *Blue bot project experiment*. HAL CCSD.
- Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers and Education*, 72, 145-157. doi: 10.1016/j.compedu.2013.10.020
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A., Engelhardt, K., Kampylis, P., et al. (2016). Developing computational thinking in compulsory education. *European Commission, JRC Science for Policy Report*, 68.
- Brennan, K., Resnick, M., & MIT Media Lab. (2012). New frameworks for studying and assessing the development of computational thinking. *American Educational Research Association Meeting, Vancouver, BC, Canada.*
- Buitrago Flórez, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a generation's way of thinking: Teaching computational thinking through programming. *Review of Educational Research*, 87(4), 834. doi:10.3102/0034654317710096
- Denning, P. J., & Tedre, M. (2019). *Computational thinking*. Cambridge, MA; London: The MIT Press.
- Fanchamps, N., Slangen, L., Hennissen, P., & Specht, M. (2019). The influence of SRA-programming on algorithmic thinking and self-efficacy using Lego robotics in two types of instruction. *International Journal of Technology and Design Education*, doi:10.1007/s10798-019-09559-9
- Grover, S., Pea, R., & Cooper, S. (2015). (2015). Systems of assessments for deeper learning of computational thinking in K-12. Paper presented at *the Proceedings of the 2015 Annual Meeting of the American Educational Research Association*, pp. 15-20.
- Hunsaker, E. (2018). *Computational thinking*. in A. otenbreit-leftwich & R. kimmons, *The K-12 Educational Technology Handbook*. EdTech Books.
- Johnson, R. T., & Johnson, D. W. (2008). Active learning: Cooperation in the classroom. *The Annual Report of Educational Psychology in Japan*, 47, 29-30.
- Kong, S. (2016). A framework of curriculum design for computational thinking development in K-12 education. *Journal of Computers in Education*, 3(4), 377-394.
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51-61. doi:10.1016/j.chb.2014.09.012
- Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas* Basic Books, Inc.
- Piaget, J. (1972). *Psychology and epistemology: Towards a theory of knowledge*. University of California.
- Sáez Fernández, C., Viera López, G., & Pérez Marín, D. (2018). Propuesta metodológica de la enseñanza de la programación en Educación Infantil con Cubetto. *IE Comunicaciones: Revista Iberoamericana de Informática Educativa*, (28), 1-8.
- Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142-158. doi://doi.org/10.1016/j.edurev.2017.09.003
- Vygotsky, L. S., & Cole, M. (1978). *Mind in society*. Cambridge, Mass. [u.a.]: Harvard Univ. Press.
- Werner, L., Denner, J., Campe, S., & Kawamoto, D. The fairy performance assessment. pp. 215-220.
- Wing, J. M. (2006). Computational thinking test. *CACM Viewpoint*, 33-35. Retrieved from <http://www.cs.cmu.edu/~wing/>
- Zapata-Cáceres, M., Martín-Barroso, E., & Román-González, M. (2020). Computational thinking test for beginners: Design and content validation. Paper presented at *the Proceedings of the 2020 IEEE Global Engineering Education Conference (EDUCON)*, pp. 1905-1914.

# Storytelling through Programming in Scratch: Interdisciplinary Integration in the Elementary English Language Arts Classroom

Emrah PEKTAŞ<sup>1\*</sup>, Florence R. SULLIVAN<sup>2</sup>

<sup>1,2</sup> College of Education

<sup>1,2</sup> University of Massachusetts Amherst, USA

epektas@educ.umass.edu, fsullivan@educ.umass.edu

## ABSTRACT

The focus of this paper is to investigate how elementary students learned computer science concepts through storytelling in Scratch. To serve this purpose, we conducted artifact interviews with 4th graders who were engaged with a computer science (CS) integrated module in their English language arts (ELA) class. Students created stories in Scratch with a focus on character traits. The constructionist design of the Scratch tool supports student learning through tinkering, the creation of meaningful artifacts, and through the theatrical metaphor that underlies interface design. This paper explores how two 4th graders demonstrated their CS/CT and ELA knowledge through the design of a Scratch artifact and how Scratch facilitated this interdisciplinary learning. While there have been studies in middle school and in after-school contexts that focus on digital storytelling and writing, there are few papers that examine interdisciplinary integration in the formal school context at the elementary level.

## KEYWORDS

Interdisciplinary Integration, Artifact Interview, Scratch

## 1. INTRODUCTION

The CS for All movement is very important for reaching all children to learn CS/CT skills in the US. We report on one such initiative in Western Massachusetts in a mid-size, ethnically diverse city: Springfield. The CS for All Springfield project focuses on supporting elementary school teachers, across 33 schools, to introduce computer science across grades K-5 through integrating Computer Science/Computational Thinking (CS/CT) into subject areas, as outlined by the state's Digital Literacy and Computer Science (DLCS) standards. This project involves over 150 teachers and will serve over 11,000 students.

## 2. RELATED WORK

### 2.1. Computational Thinking

Wing (2006) defines computational thinking (CT) as "solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (p. 33). Brennan and Resnick (2012) define computational thinking with three dimensions: computational concepts such as conditionals, computational practices such as decomposing problems or remixing others' or your own work, and computational perspectives such as expressing, which is defined as using computation for self-expression. In their study, Brennan and Resnick considered

the Scratch online community as inspiring for Scratchers to reuse and remix. They state that the Scratch online community supports Scratchers in reusing and remixing, "by helping them find ideas and code to build upon, enabling them to potentially create things much more complex than they could have created on their own" (p. 8). As well as the Scratch online community, Scratch itself provides tools for users to remix such as creatively remixing two existing characters through costumes in Scratch.

### 2.2. Interdisciplinary Integration

Many elementary schools in the US are not able to offer CS as a stand-alone topic due to the demands of the curriculum. Therefore, if CS is going to be taught in the US, it will be taught through interdisciplinary integration. The CS for All community needs research that delves into how to integrate CS/CT across the curriculum in meaningful ways, such that students learn the content in both areas. Our project takes such an interdisciplinary approach. For the purposes of this paper, we are focusing on integration of CS/CT ideas into the English Language Arts (ELA) curriculum using the Scratch program. Resnick et al. (2009) designed Scratch based on three constructionist design principles: (1) to make it more tinkerable, (2) more personally meaningful, and (3) more social than other programming environments.

Scratch is an excellent tool for use in ELA due to the theatrical metaphor that underlies the user interface of the Scratch system and permits the development of interactive stories (Resnick et al., 2009). Since Scratch provides a "stage" upon which interactive animations and stories can be displayed, ELA teachers are able to approach important topics in reading and writing fiction, for example, the narrative story arc, the location of the story and character elements. Scratch animations unfold temporally, which allows for the narrative to evolve over time. Meanwhile, the background feature in Scratch allows the location of the story to change as the narrative evolves, and the sprite element (including provided characters, costumes, and the "say" blocks) allow students to create characters with specific traits, and these traits can also change and/or evolve with the narrative arc of the story. Elementary teachers can take advantage of these elements to integrate CS/CT into their ELA instruction. Moreover, Maloney et al. (2008) argue: "...the design of the Scratch blocks simplifies the mechanics of programming by eliminating syntax errors, providing feedback about placement of command blocks, and giving immediate feedback for experiments (p. 371)." These combined design elements - ease of use of the tool, the stage metaphor, and the ability for students to tinker and create meaningful programs - makes Scratch an ideal tool

for integrating CS content with ELA in the elementary classroom.

### **2.3. Using Scratch for ELA integration**

Others have conducted research on using Scratch in the ELA classroom. Burke and Kafai (2010; 2012) have investigated the use of Scratch in teaching writing at the middle school level. Their results indicate that programming in Scratch can assist children in developing their storytelling and creative writing skills. They state that “writing to program can also serve as programming to write, in which a child learns the importance of sequence, structure, and clarity of expression—three aspects characteristic of effective coding and good storytelling alike.” (2010, p. 348)

Fields et al., (2014) examined students' collaborative creation of interactive stories using Scratch, in which students received feedback on their stories. Their findings suggest that online collaborative creative storytelling and constructive feedback have the potential to generate both more complex story designs and code development. This co-evolution of coding skills and writing skills is a key element of integration. Meanwhile, Smith and Burrow (2016) present two anecdotal case studies conducted with their own young children as they observed them using Scratch, Jr. These teacher educators recognized the utility of Scratch for assisting in story development through their children's tinkering with design elements of the system. They then use this knowledge to help construct Scratch integration lessons for their pre-service teacher education students.

The work presented in this study extends the current research and specifically focuses on a fourth-grade ELA assignment that uses Scratch to teach students about both character traits and algorithmic development. In this study, we examine the following research question: How do 4th graders learn CS/CT concepts integrated into ELA through Scratch and an interdisciplinary integrated module?

## **3. METHODS**

### **3.1. Context of the Study**

This study took place in the context of CS for All Springfield, a large, four-year study of the iterative design and development of integrated CS/CT modules across the elementary curriculum in the Springfield Public School (SPS) district. Students in the Springfield Public Schools are 18.9% Black, 66.6% Latinx, 10.2% White, and 4.3% Asian, Native American, non-Hispanic, and multi-race students. Eighty-three percent of district students are considered high needs, and 76.7% are economically disadvantaged. The manifold goals of the larger study include an understanding of teacher professional development needs regarding CS/CT integration, barriers to such integration, and assessment of student learning. The study reported here is one aspect of the latter research goal.

### **3.2. Setting and Participants**

Participants in this study were drawn from three different classrooms involved in year two of the four-year long study. All artifact interview participants were selected by the teacher. The interviews were collected with a select

group of fourth grade students who completed ELA/Scratch projects. A total of twelve children were interviewed for this study. Seven children were interviewed individually, and two groups of students were also interviewed (one group of three and one group of two). Each interview lasted 10 to 15 minutes. For the purposes of this paper, we are focusing on two individual interviews only. Those two interviews were collected from a classroom where CS/CT was integrated into ELA. Other interviews were collected from classrooms where CS/CT was integrated into either math or social studies. The goal of our analysis is to understand how Scratch can support ELA learning in the upper elementary grades. Therefore, it was important for us to focus on the two interviews in the ELA classroom where students were successful with the curriculum. These two interviews are representative of the strength of the integrated approach for using Scratch to teach ELA. In this way, these interviews are examples of what is possible. Later work will be looking across all data. These two interviews were conducted with two 10-year-old boys (pseudonymously known as Martin and Kyle) who worked together on their ELA/Scratch project, but were interviewed separately. The size of the class where these interviews were conducted was 18 students (11 boys and 7 girls). Nine students were Black while seven were Latinx and two were White Americans. Martin was a mixed race student (Latinx and Black), Kyle was a White (European-American) student. As for their programming experience, the classroom teacher reported that they did not have much experience in coding prior to the CS for all lessons. She added that some may have had a few classes last year in code.org with the computer teacher but other than that this was their first time ever using Scratch.

### **3.3. Curriculum Design/Tools**

Two 4th grade teachers worked together to adopt a pre-existing curriculum to integrate CS/CT components across ELA. This dyad developed a six-lesson module to pilot in their classroom. The unit was designed based on “character traits” in 4th grade ELA. Character traits is an English Language Arts (ELA) unit taught in the district based on Massachusetts ELA standards (2017). In the module, 4th graders were taught to identify character traits and what behaviors a character trait is associated with, to describe their own character traits, to develop a story and manage sequencing the events, to use Scratch and to create a short Scratch animation (story) based on a few character traits.

### **3.4. Data Collection**

The data were collected using the artifact interview method. By artifact we mean a completed or almost-completed work in a programming tool such as Scratch that students created while engaged in the integrated unit. The data consists of both their statements during the interviews and the actual Scratch program created. Ginsburg (1997) argues that clinical interview is a powerful technique for gaining insight into a child's way of thinking. Ginsburg writes that during a clinical interview the interviewer asks open-ended questions such as “how did you do it?” or “why” and does “an immediate interpretation of the subject's response” and “on-the-spot hypothesis making and testing” (p. 34). The artifact

interview method evolved out of the clinical interview method. That is, an artifact interview is about interviewing a child's conceptual understanding of a topic. In this approach, the interviewer engages in conversation with the Scratcher about their computational products and practices, using work samples to guide the conversation. It is similar to and newer than clinical interviewing, and focuses on an artifact that a child has created. Students' laptop screens and voices were recorded during their artifact interviews. During artifact interviews, students were asked open-ended questions such as "Why did you use these particular blocks and sprites?" and "What steps did you follow to create your project?" All the interviews were conducted at the end of the module implementations. The researchers acted as participant observers in the classroom during module implementation. They worked to build a good rapport with the students during this time and the interviews were done in a conversational mood so that students engaged in the interview. Additionally, field notes and photographs of the module implementation were gathered.

### 3.5. Data Analysis

After transcribing the artifact interviews, we analyzed not only the participants' statements on their Scratch artifacts during interviews but also the Scratch programs themselves. During the module implementation, the participants engaged in both Massachusetts' DLCS standards (2016) such as "3-5.CT.b4: Individually and collaboratively create tests and modify a program in a graphical environment (e.g., block-based visual programming language)", which is coding/programming in Scratch in this case, and Massachusetts' ELA standards (2017) such as "W 4.3: Write narratives to develop real or imagined experiences or events using effective technique, descriptive details, and clear event sequences", which is writing a story based on a few character traits in this case. In our data analysis, we focused on how separate standards of the two disciplines intersect in the participants' Scratch stories.

## 4. RESULTS

The entire class worked in groups of three or four to write a story based on the character traits they picked at the beginning of the lessons and to create a short Scratch story based on their written story. Martin and Kyle were two members of a group of four. During the lessons, the group came up with a story called *Zombie Apocalypse* with three parts, beginning, middle, and end. The story was framed around a few characters traits such as brave (Martin's choice), daring (Kyle's choice), and fearless. Based on these character traits, using a graphic organizer and a story map, the group in the leadership of Kyle wrote a story around the following idea: carrying Zombies in it, a meteor called the Nebula hits the Earth, and then Zombies in the Nebula come out, destroy the Earth, and zombify all human beings. However, only three people and one goblin, students themselves, remain alive and the three people fight against zombies, kill them all, and save the world. Based on this story, Martin and Kyle were responsible for creating the beginning of the story in Scratch, so the projects of the two

students were similar in terms of graphical content, and each of them separately designed the short Scratch story up to the point where zombies spread all over the world, which represented the beginning of the story.

### 4.1. Martin's project

Briefly, the first scene of Martin's Scratch project was that the Earth and the Nebula appear in outer space, and the Nebula slowly approaches (gliding) the Earth and crashes into it. After that, the backdrop switches to another one where there is fire all over the place in a city on the Earth because of the crash. This second scene is to represent the idea that the entire Earth was being destroyed by zombies. After this scene, the backdrop switches to another background in which there are three zombies that were released from the Nebula after the crash. These scenes are shown in order in Figure 1 below.

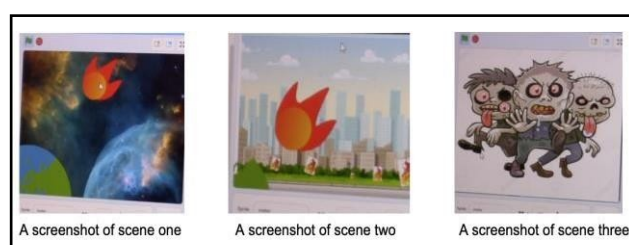


Figure 1 (Screenshots of scenes of Martin's Scratch story)

#### 4.1.1. Remixing for self-expression

Martin was a Scratcher who could effectively utilize the tools that the Scratch environment provides such as remixing different characters in costumes according to the purpose of his project such as representing the character trait "brave" he picked. Figure 2 below is a screenshot of Martin's character development on Scratch:

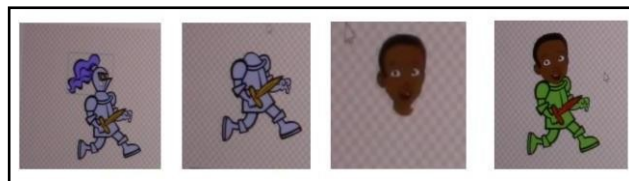


Figure 2 (Screenshots of evolution of Martin's character)

As seen in Figure 2 above, Martin remixed the body of a knight sprite and the head of a black person sprite in Scratch library. When Martin was asked how he created it, he was simultaneously telling and showing the interviewer on his laptop screen how he made it:

When I go to Devin, I took off. Let's go to people [in Scratch library]. Took off his [a black person sprite's] face [head]. [...] go here and then right then I choose the knight. [...] I got rid of the head [of the knight]. [...] Delete that, then get rid of the body [of the person sprite]. That, delete. [...] Copied this [the person's head], then go here [knight body without head], paste [the head on the knight body costume]. Then [...] got rid of the neck [of the person sprite]. [...] And put the head on the top of the- [knight body]" (Interview 103019)

As a mixed-race person of Latinx and Black descent, and choosing “brave” as his character trait for this project, Martin not only represented himself as a Black person in the story but also remixed it with a knight that represents bravery. After Martin described and showed how he remixed two characters on Scratch to create a new character, the researchers asked “why did you create that sprite?” Martin replied as follows:

[...] my character should be like just free with [a] shirt on [be]cause the zombies can easily get them. I want him to have protection. Then if the zombie[s] start climbing on their back and they're almost about to get him, he can take it off and run away. (Interview 103019).

Martin thought that he cannot fight against zombies as a normal person with his regular clothes, so he chose to be a knight whose costumes protect against zombies and this part of the narrative needed to be shown visually and created computationally. Martin was able to make all this happen with the tools that Scratch provides for its users such as characters, backdrops, and blocks for action.

#### 4.2. Kyle's project

In his project, just like Martin's, Kyle's short Scratch story starts with a scene in outer space, as seen in Figure 3, where the Nebula approaches the Earth and hits it. And then, all zombies in the Nebula spread across all corners of the world. Right after this scene, the backdrop changes to another backdrop in which a few zombies were placed on different coordinates on the background and the buildings in that area were ruined. This background was to reflect the idea of zombies being released from the Nebula and of destroying the Earth.



Figure 3 (Screenshots of scenes of Kyle's Scratch story)

##### 4.2.1. Programming for narrative coherence

Not knowing that the glide block existed on Scratch, Kyle did create a script that functions as a glide block in Scratch as shown in Figure 4 below.

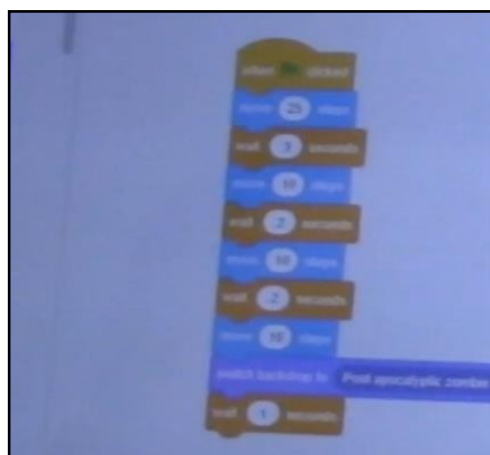


Figure 4 (A screenshot of Kyle's glide block)

When we asked Kyle how he created that script, he responded:

All right, so this is the meteor. And, when I click this [green flag], which means go, it'll move 25 steps. In this case it would be 25 steps closer to the earth. [...] And it goes 25 steps toward the earth, and then it waits three tenths of a second, and then moves 10 steps and waits two tenths of a second, and then goes another 10 steps, two tenths a second, and then goes another 10 steps. So, [it] looks like it's actually flying through, it's called the galaxy, towards earth. (Interview 103019)

According to the story, Kyle needed to animate an action where the meteor approaches the Earth before hitting it. Not knowing that the glide block existed in Scratch, Kyle programmed a script that functioned as a glide block in Scratch. This accomplished his goal of narrative coherence, such that viewers could see the meteor *flying* towards Earth. In this case, Kyle's programming activity was guided by the narrative of the meteor moving from outer space to the earth.

## 5. DISCUSSION

Martin and Kyle's individual work in Scratch demonstrates the powerful way in which the Scratch program can be used to support ELA lessons in the elementary classroom. The students were given the opportunity to write a story in Scratch that met the following State of Massachusetts (2017) writing standard for fourth grade: “Write narratives to develop real or imagined experiences or events using effective technique, descriptive details, and clear event sequences.” The sequencing of events maps very well to the temporal nature of the Scratch interface as demonstrated by the change in background. Both Martin and Kyle changed the background in the “beginning” of the story to demonstrate plot movement. Moreover, Scratch supported student imagination as the group (following Kyle's lead) created a “Zombie” story - an aspect of popular culture as demonstrated in Zombie video games such as: “Zombie Apocalypse.” Both students were able to develop descriptive details and support the presentation of the details by engaging in computational practices and they did so by using

computational means. For example, Martin engaged in the activity of remixing in order to create a character that could successfully fight against the zombies. Interestingly, in this case, Martin elected to modify the knight to have the head of a Black boy. Arguably, Martin was placing himself in the story. This is an important constructionist design element of Scratch - Martin was able to create a more meaningful narrative, by placing himself in the story. This type of imagination is also important for success in writing and interpreting narratives, as one is able to personally connect to a story (Eagen, 1992).

Meanwhile, Kyle demonstrated a keen understanding of the need for the story to unfold in a visually meaningful way, and since he was not aware of the glide block in Scratch (which would have allowed his zombie filled meteor to visually move across the screen) he created his own glide block. This is an especially important point regarding interdisciplinarity and Scratch. The narrative is that the meteor moved through space and collided with Earth. In order for that narrative to be communicated, Kyle needed to show the meteor moving smoothly across the screen over time. To solve this problem, Kyle created the code with imperceptibly short time variables (three-tenths of a second for every 25 steps). In writing this code, Kyle both learned how to program Scratch with some level of precision, and also served the narrative by creating the visual effect of the meteor streaking through space. Effectively, Kyle was able to serve the narrative while learning to code.

## 6. CONCLUSION

In this study, we used an interdisciplinary integration approach, with CS/CT concepts being integrated into ELA, to examine how two 4th graders expanded their ELA knowledge and their CS/CT knowledge through Scratch while engaged in the lesson. Scratch is an environment that allows for this integration because it was designed in a way that supports the creation of a story. Scratchers can create narrative elements through the tools that Scratch provides such as rich, visual graphics like sprites / characters, backdrops / setting, and action that can be created through blocks / programming. These narrative elements can be easily tinkered within Scratch to write a story as seen in our study. Scratch is a powerful tool for interdisciplinary integration, especially when children are provided with the opportunity to collaboratively engage in narrative, fictional writing assignments such as the one featured in this classroom.

**7. ACKNOWLEDGEMENTS:** The work in this paper has been funded by a grant from the National Science Foundation, DRL - 1837086. Any opinions expressed in this paper are those of the authors and do not necessarily represent those of the National Science Foundation.

## 8. REFERENCES

- Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada* (Vol. 1, p. 25).
- Burke, Q., & Kafai, Y. B. (2010, June). Programming & storytelling: opportunities for learning about coding & composition. In *Proceedings of the 9th international conference on interaction design and children* (pp. 348-351).
- Burke, Q., & Kafai, Y. B. (2012, February). The writers' workshop for youth programmers: digital storytelling with scratch in middle school classrooms. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 433-438).
- Eagen, K. (1992). *Imagination in teaching and learning: The middle school years*. The University of Chicago Press.
- Fields, D. A., Kafai, Y. B., Strommer, A., Wolf, E., & Seiner, B. (2014). Interactive storytelling for promoting creative expression in media and coding in youth online collaboratives in Scratch. *Proceedings of constructionism*, 19-23.
- Ginsburg, H. (1997). *Entering the child's mind: The clinical interview in psychological research and practice*. Cambridge University Press.
- Maloney, J. H., Peppler, K., Kafai, Y., Resnick, M., & Rusk, N. (2008, March). Programming by choice: urban youth learning programming with scratch. In *Proceedings of the 39th SIGCSE technical symposium on Computer science education* (pp. 367-371).
- Massachusetts Department of Elementary and Secondary Education. (2017). English Language Arts and Literacy Framework. Retrieved from <https://www.doe.mass.edu/frameworks/ela/2017-06.pdf>
- Massachusetts Department of Elementary and Secondary Education. (2016.). Digital Literacy Computer Science Framework. Retrieved from <https://www.doe.mass.edu/frameworks/current.html>
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, 52(11), 60-67.
- Smith, S., & Burrow, L. E. (2016). Programming multimedia stories in Scratch to integrate computational thinking and writing with elementary students. *Journal of Mathematics Education*, 9(2), 119-131.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.



# Students' Learning of Computational Thinking in Schools with Different Curriculum Approaches Including Individual Student Characteristics

Amelie LABUSCH<sup>1\*</sup>, Birgit EICKELMANN<sup>2</sup>

<sup>1,2</sup>Paderborn University, Germany

amelie.labusch@upb.de, birgit.eickelmann@upb.de

## ABSTRACT

Although computational thinking (CT) has emerged as an important 21st century key competence (Voogt, Fisser, Good, Mishra & Yadav, 2015; Wing, 2006), it becomes apparent that there are great differences (Bocconi, Chiocciariello, Dettori, Ferrari & Engelhardt, 2016). Selecting four countries (Denmark, Finland, Germany, and the USA) with different approaches of curricular anchoring, linear regression analyses with relevant variables were conducted based on the data from the IEA International Computer and Information Literacy Study 2018 (Fraillon, Ainley, Schulz, Friedman & Daniel Duckworth, 2019).

The social background of the students, the extent to which different computational thinking-related skills are learned at school, studying computer science (CS) or a similar subject, and the students' gender were included in these analyses. The results first indicated that in all countries there was a close link between social background and students' competences in computational thinking as well as between the extent to which computational thinking-related skills were learned at school and students' competences in computational thinking. Second, there were also differences in competence with regard to studying computer science in Germany, Denmark, and Finland and gender-specific differences in favor of boys in Germany, Denmark, and the USA. Third, it became apparent that the results offer individual points of improvement for each educational system – regardless of which approach of curricular anchoring they follow.

## KEYWORDS

Computational thinking competences, IEA-ICILS 2018, School curriculum approaches, Individual characteristics

## 1. INTRODUCTION

In times of progressive digitalization, increasingly sophisticated technologization based on algorithmic structure, and the associated changes in all areas of life, the question arises as to what competences children and young people must acquire to successfully participate in society and be prepared for an adequate working life. Since school holds a key role in the acquisition of students' competences and addressing the issue of relevant competences in the field of digitalization and information technology in education, the key competence computational thinking (CT) emerges (Labusch & Eickelmann, 2020). In a general overview of the different approaches in various educational systems three different approaches to the curricular anchoring of computational thinking can be identified (Eickelmann, 2019): (1) computational thinking as a cross-

curricular competence, (2) computational thinking as part of computer science, and (3) computational thinking as an individual subject or learning area. For the later analyses, four countries, participating in the international option computational thinking in the International Computer and Information Literacy Study 2018 (ICILS 2018), were selected that could be classified under the different approaches at the time of study's data collection in 2018.

In Finland, 'algorithmic thinking' has been anchored in mathematics since 2014. The revision of the core curriculum around algorithmic thinking and programming has already been completed in 2014, implementation started in 2016 with a two-year implementation phase. Finland has included computational thinking in the national curriculum as a cross-curricular competence (first approach) that is anchored across disciplines (Bocconi, Chiocciariello & Earp, 2018).

In Germany, where the development of school curricula is guided at the federal state level, the integration of computational thinking varies from state to state. However, schools rely on the long tradition of computer science teaching (second approach) as an optional subject (Bocconi, Chiocciariello, Dettori, Ferrari & Engelhardt, 2016).

Denmark has been piloting the integration of computational thinking in model schools since the summer of 2018, both as a separate subject (third approach) and as part of a subject integration approach (first approach) (Undervisningsministeriet, 2018). The cross-curricular topic 'IT and Media' in K0 to K9 is integrated in all subjects and includes elements of computational thinking such as problem-solving and logical thinking (Bocconi, Chiocciariello & Earp, 2018).

In the USA, school curricula and policies vary regionally (all three approaches). Some companies are working with the US government to develop new computer science standards, and many states have issued new guidelines for curricula (Hsu, Irie & Ching, 2019).

When considering the extent to which individual aspects of the acquisition of a competence, school assessment studies examining other areas of competence have shown the importance of using individual characteristics as explanations. Thus, for students' computer and information literacy, a close link between social background and competences could be shown for all participating countries in ICILS 2013 and 2018 (Eickelmann et al., 2019; Fraillon, Ainley, Schulz, Friedman & Duckworth, 2019).

Moreover, it is relevant to what extent students have learned computational thinking-related skills at school. In recent years, many partial competences of computational



thinking have been discussed (e.g. Bauer, Butler & Popovic, 2015; Lye & Koh, 2014). What they all have in common – roughly summarized – is the partial competence of decomposition or analysis of data or problems. It is also concerned with the conception or simulation of solutions and the representation of processes.

Since computer science lessons were not offered nationwide in 2018, it is necessary to look at whether students have participated in computer science lessons in the respective current school year. Another aspect is the gender of students, where differences have been shown to be crucial for computational thinking itself (e.g. Román- González, Pérez- González & Jiménez-Fernandez, 2017). These considerations result in the following research question:

To what extent can differences in students' competences in computational thinking be explained by their social background, their school learning of computational thinking-related skills, studying the subject computer science, as well as their gender in four countries with different curricular anchoring of computational thinking?

## 2. METHODS

### 2.1. Study and Data

The following analyses are based on data from the International Computer and Information Literacy Study 2018 (ICILS 2018). The competences in computational thinking (CT) are defined in the framework of ICILS 2018 as "an individual's ability to recognize aspects of real-world problems which are appropriate for computational formulation and to evaluate and develop algorithmic solutions to those problems so that the solutions could be operationalized with a computer" (Fraillon, Ainley, Schulz, Duckworth & Friedman, 2019, p. 27). The construct of these competences formed the basis for the development of the computer-based student tests. The students (international average age of 14.4 years) worked on two computational thinking test modules of 25 minutes each. In addition, questionnaires for students, teachers, school principals, and ICT coordinators were used to determine

the framework conditions. (Eickelmann et al., 2019; Fraillon, Ainley, Schulz, Friedman & Duckworth, 2019).

### 2.2. Analyses

To analyze the data, addressing the research question, linear regression analyses were carried out for the selected countries (Finland, N=2,546 students; Denmark, N=2,404 students; Germany, N=3,655 students; USA, N=6,790 students). Four regression models were calculated for each of the four countries.

In the first model, indicators of social background – cultural capital (model I) and HISEI (model II) – were drawn upon. Following this approach, the eighth graders in ICILS 2018 were asked how many books they had at home (without magazines, newspapers, comics, and textbooks). The later analysis refers to the distinction between a maximum of 100 books (low cultural capital) available and more than 100 books (high cultural capital) available at home. To describe the socio-economic status of a student

family, analyses refer to the highest occupational status of parents (HISEI). A low HISEI score (below 40 points) is available, for example, for postmen and women, train conductors and hairdressers. A medium HISEI value (40 to 59 points) is found, for example, for police officers, nurses, social workers, and administrative staff. A high HISEI score (60 or more points) is given, for example, to teachers, journalists, and lawyers.

In the second model, three items were selected from a scale for the extent of school-based learning of computational thinking skills. The selection was based on theories and research, a high affinity to the computational thinking tests, and - determined with a preliminary analysis - the power of variance explanation. The items 'to break a complex process into smaller parts', 'to use simulations to help understand or solve real world problems', and 'to make flow diagrams to show the different parts of a process' were considered. A distinction was made between 'at least to a moderate extent' as a reference category and 'to a small extent or not at all'.

In the third model, the studying of the subject computer science (CS) or a similar subject is used. In the main survey, the students were asked whether they had studied computing, computer science, information technology, informatics or similar in the respective current school year.

In the fourth model, the gender of the students was used to explain the variance (options 'female' and 'male').

## 3. RESULTS

The following four tables show the corresponding regression models for students in Finland, Germany, Denmark, and the USA.

Table 1. Regression Model I Explaining Differences in Students' CT Competences by Social Background.

	Finland		Germany		Denmark		USA	
	b	(SE)	b	(SE)	b	(SE)	b	(SE)
cultural capital <sup>A</sup>	27.6*	(4.2)	48.6*	(5.5)	25.5*	(3.9)	44.4*	(3.2)
medium HISEI value	20.0*	(5.2)	30.2*	(5.9)	18.8*	(4.9)	22.5*	(3.2)
high HISEI value	44.9*	(5.4)	51.2*	(8.0)	30.8*	(5.7)	48.0*	(3.9)
constant	481.7		443.5		498.9		466.8	
R <sup>2</sup>	.07		.13		.05		.09	

b - regression weight (unstandardized).  
dependent variable: students' computational thinking.  
\* significant coefficient (p < .05).

<sup>A</sup> 0 - maximum of 100 books; 1 - more than 100 books.

IEA: International Computer and Information Literacy Study 2018

© ICILS 2018

It turns out that the correlation between computational thinking competences and cultural capital was significant in all four countries. While it was 25.5 points in Denmark and 27.6 points in Finland, it was 44.4 points in the USA and 48.6 points in Germany. For the medium HISEI value in model II, there were values between 18.8 points (Denmark) and 30.2 points (Germany), which were all significant. For the high HISEI value, there were significant values between 30.8 points (Denmark) and 51.2 points (Germany). The explanation of variance amounted to between 5 percent (Denmark) and 13 percent (Germany). This also reveals that for Germany, for example, 13 percent of the variance in the competences in computational

thinking could be explained solely by the students' social background, without further examining other factors.

**Table 2.** Regression Model II Explaining Differences in Students' CT Competences by Social Background and School Learning of CT-related Skills.

	Finland		Germany		Denmark		USA	
	b	(SE)	b	(SE)	b	(SE)	b	(SE)
cultural capital <sup>A</sup>	23.8*	(4.4)	50.8*	(5.6)	23.3*	(3.8)	42.4*	(3.2)
medium HISEI value	17.5*	(4.9)	22.6*	(6.3)	18.8*	(5.1)	20.7*	(3.2)
high HISEI value	40.4*	(5.3)	41.9*	(8.0)	28.5*	(5.6)	45.5*	(4.2)
to break a complex process into smaller parts <sup>B</sup>	19.2*	(4.5)	-4.6	(5.7)	12.7*	(4.4)	13.0*	(4.0)
to use simulations to help understand or solve real world problems <sup>B</sup>	-23.5*	(6.4)	-37.9*	(5.5)	-33.4*	(3.8)	-20.4*	(3.7)
to make flow diagrams to show the different parts of a process <sup>B</sup>	-22.3*	(6.0)	-12.5*	(5.9)	-9.4	(5.1)	-11.4*	(3.5)
constant	493.2		470.0		513.4		484.3	
R <sup>2</sup>	.11		.18		.11		.11	

b - regression weight (unstandardized).

dependent variable: students' computational thinking.

\* significant coefficient (p < .05).

<sup>A</sup> 0 - maximum of 100 books; 1 - more than 100 books.

<sup>B</sup> 0 - to a small extent or not at all; 1 - at least to a moderate extent.

IEA: International Computer and Information Literacy Study 2018 © ICILS 2018

The extent to which the ability to break a complex process into smaller parts was learned in school was either not in any (as in Germany) or in a positive with the competences in computational thinking (see table 2). However, the correlation between the extent of learning the ability to use simulations to help understand or solve real world problems and the competences in computational thinking was negative in all four countries (USA: -20.4, Finland: -23.5, Denmark: -33.4, Germany: -37.9).

Regarding the extent to which students learned to make flow diagrams to show the different parts of a process in school, either no correlation (Denmark) or a negative correlation (USA: -11.4, Germany: -12.5, Finland: -22.3) to the competences in computational thinking was evident in these countries. From model II to model III, the explanation of variance increased again – in Denmark, Finland, and the USA to 11 percent each and in Germany even to 18 percent.

**Table 3.** Regression Model III Explaining Differences in Students' CT Competences by Social Background, School Learning of CT-related Skills, and Studying CS.

	Finland		Germany		Denmark		USA	
	b	(SE)	b	(SE)	b	(SE)	b	(SE)
cultural capital <sup>A</sup>	23.8*	(4.4)	50.8*	(5.6)	23.3*	(3.8)	42.4*	(3.2)
medium HISEI value	17.5*	(4.9)	22.6*	(6.3)	18.8*	(5.1)	20.7*	(3.2)
high HISEI value	40.4*	(5.3)	41.9*	(8.0)	28.5*	(5.6)	45.5*	(4.2)
to break a complex process into smaller parts <sup>B</sup>	19.2*	(4.5)	-4.6	(5.7)	12.7*	(4.4)	13.0*	(4.0)
to use simulations to help understand or solve real world problems <sup>B</sup>	-23.5*	(6.4)	-37.9*	(5.5)	-33.4*	(3.8)	-20.4*	(3.7)
to make flow diagrams to show the different parts of a process <sup>B</sup>	-22.3*	(6.0)	-12.5*	(5.9)	-9.4	(5.1)	-11.4*	(3.5)
constant	493.2		471.0		513.4		484.3	
R <sup>2</sup>	.11		.18		.11		.11	

b - regression weight (unstandardized).

dependent variable: students' computational thinking.

\* significant coefficient (p < .05).

<sup>A</sup> 0 - maximum of 100 books; 1 - more than 100 books.

<sup>B</sup> 0 - to a small extent or not at all; 1 - at least to a moderate extent.

IEA: International Computer and Information Literacy Study 2018 © ICILS 2018

Model III included whether the students had studied computer science or a similar subject in the corresponding current school year (time of measurement in 2018). While there was no correlation in the USA, Finland (17.3 points), and Germany (20.6 points) showed a positive correlation with the level of competence in computational thinking. In Denmark, by contrast, there was a negative correlation (-26.2), which implies that those students who studied computer science or similar subjects scored on average 26.2 points less than those who did not. In Germany, all coefficients of social background increased from model III to model IV. The explanation of variance remained at 11 percent in the USA, rose to 12 percent in Denmark and Finland, and to 19 percent in Germany.

**Table 4.** Regression Model IV Explaining Differences in Students' CT Competences by Social Background, School Learning of CT-related Skills, Studying Computer Science, and Gender.

	Finland		Germany		Denmark		USA	
	b	(SE)	b	(SE)	b	(SE)	b	(SE)
cultural capital <sup>A</sup>	24.5*	(4.5)	53.3*	(5.7)	23.0*	(3.8)	43.4*	(3.2)
medium HISEI value	18.3*	(4.9)	23.4*	(6.3)	16.7*	(4.8)	20.7*	(3.3)
high HISEI value	41.2*	(5.4)	41.1*	(8.2)	26.6*	(5.4)	44.2*	(4.2)
to break a complex process into smaller parts <sup>B</sup>	18.9*	(4.5)	-6.8	(5.6)	12.2*	(4.5)	13.1*	(4.0)
to use simulations to help understand or solve real world problems <sup>B</sup>	-23.6*	(6.3)	-39.3*	(5.6)	-31.9*	(3.7)	-19.6*	(3.7)
to make flow diagrams to show the different parts of a process <sup>B</sup>	-23.3*	(5.9)	-14.0*	(5.9)	-8.8	(5.2)	-11.1*	(3.5)
studying computer science related subjects in current school year <sup>C</sup>	17.5*	(5.2)	18.6*	(6.2)	-27.1*	(7.8)	-7.4	(4.0)
gender <sup>D</sup>	0.6	(3.7)	-16.2*	(5.4)	-7.2*	(3.3)	-17.2*	(3.6)
constant	487.4		472.6		521.9		496.1	
R <sup>2</sup>	.12		.20		.12		.11	

b - regression weight (unstandardized).

dependent variable: students' computational thinking.

\* significant coefficient (p < .05).

<sup>A</sup> 0 - maximum of 100 books; 1 - more than 100 books.

<sup>B</sup> 0 - to a small extent or not at all; 1 - at least to a moderate extent.

<sup>C</sup> 0 - no; 1 - yes.

<sup>D</sup> 0 - male; 1 - female.

IEA: International Computer and Information Literacy Study 2018 © ICILS 2018

In Model IV, the students' gender was considered. This showed that even when social background, school-based acquisition of computational thinking-related skills, and studying computer science or a similar subject were included, gender differences were found for Denmark (7.2 points), Germany (16.2 points), and the USA (17.2 points). In all three cases, the boys scored significantly higher than the girls. In Denmark, Finland, and the USA, the explanation of variance persisted and was in the overall model 11 percent in the USA and 12 percent in Denmark and Finland. In Germany, it rose to 20 percent. In Germany, Denmark, and the USA, a small increase in the cultural capital coefficient was also observed, while in the USA, the medium HISEI value increased minimally.

## 4. DISCUSSION

The overall review reveals that especially students' social background can explain differences in computational thinking competences. As it has already been the case for other competence domains, a close link between the social

background and the educational success of the students could be established for the competence area of computational thinking. At this point, all countries should consider the strategic and conceptual development of core curricula to overcome the high socially caused educational disparities that have been identified.

Moreover, the results proved that there are computational thinking-related skills that are conducive to learning and those that seem to be counterproductive. It should be noted here that the study is limited to reveal whether and to what extent the individual skills were learned at school, while it didn't examine how skills were learned. It would be useful to carry out in-depth analyses, possibly with qualitative design, to see exactly in what form, for instance, simulations and flowcharts were used.

Overall, however, there is a tendency towards making teaching more productive so that students can achieve higher levels of competence in computational thinking.

The differences in competence between girls and boys are also remarkable, especially when other variables are controlled. Here, it is important to foster girls and get them more enthusiastic about computational thinking, and if necessary, to teach them in a gender-sensitive way. In parallel, boys are to be further fostered to make the best possible use of their potential.

In answer to the question as to what differences exist between the countries with regard to the approach of anchoring computational thinking in the curriculum, it should be emphasized that the results should not be used as a basis for concluding that one approach is better than the other. In this context, it should be emphasized that the four considered countries are all very highly developed and have advanced educational systems. This is another reason why in-depth analysis that include aspects of the education system would be necessary.

## 5. REFERENCES

- Bauer, A., Butler, E. & Popovic, Z. (2015). Approaches for teaching computational thinking strategies in an educational game: A position paper, In *Blocks and beyond workshop*. IEEE, Atlanta, Georgia, USA, pp. 121–123.
- Bocconi, S., Chiocciariello, A., Dettori, G., Ferrari, A. & Engelhardt, K. (2016). *Developing computational thinking in compulsory education – Implications for policy and practice*. Publications Office of the European Union.
- Bocconi, S., Chiocciariello, A. & Earp, J. (2018). *The Nordic approach to introducing CT and programming in compulsory education*. Report prepared for the Nordic@BETT2018 Steering Group.
- Eickelmann, B. (2019). Measuring secondary school students' competence in computational thinking in ICILS 2018 – Challenges, concepts and potential implications for school systems around the world. In S.C. Kong & H. Abelson (Eds.), *Computational Thinking Education* (p. 53–64). Singapore: Springer.
- Eickelmann, B., Bos, W., Gerick, J., Goldhammer, F., Schaumburg, H., Schwippert, K. et al. (Eds.) (2019). *ICILS 2018 #Deutschland. Computer- und informations-bezogene Kompetenzen von Schülerinnen und Schülern im zweiten internationalen Vergleich und Kompetenzen im Bereich Computational Thinking*. [ICILS 2018 #Germany – Students' computer and information literacy in second international comparison and competences in computational thinking]. Münster, Germany: Waxmann.
- Fraillon, J., Ainley, J., Schulz, W., Duckworth, D. & Friedman, T. (2019). *IEA International Computer and Information Literacy Study 2018: Assessment framework*. International Association for the Evaluation of Educational Achievement (IEA).
- Fraillon, J., Ainley, J., Schulz, W., Friedman, T. & Duckworth, D. (2019). *Preparing for life in a digital world: IEA International Computer and Information Literacy Study 2018 International Report*. International Association for the Evaluation of Educational Achievement (IEA).
- Hsu, Y.-C., Irie, N. R. & Ching, Y.-H. (2019). Computational Thinking Educational Policy Initiatives (CTEPI) across the globe. *TechTrends*, 63(3), 260–270.
- Labusch, A. & Eickelmann, B. (2020). Computational Thinking Competences in Countries from Three Different Continents in the Mirror of Students' Characteristics and School Learning. In S.C. Kong, H.U. Hoppe, T.C. Hsu, R.H. Huang, B.C. Kuo, K.Y. Li et al. (Eds.), *Proceedings of International Conference on Computational Thinking Education 2020* (pp. 2–7). Hong Kong: The Education University of Hong Kong.
- Lye, S. Y. & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51–61.
- Román-González, M., Pérez-González, J.-C. & Jiménez-Fernandez, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the computational thinking test. *Computers in Human Behavior*, 72, 678–691.
- Undervisningsministeriet (2018). *Computational tankegang [Computational thinking]*. Retrieved: <https://www.emu.dk/grundskole/teknologiforstaelse>
- Voogt, J., Fisser, P., Good, J., Mishra, P. & Yadav, A (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, 20(4), 715–728.
- Wing, J. M., (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.

# A Standard Decomposition Process to Inform the Development of Game-Based Learning Environments Focused on Computational Thinking

Elizabeth L. ADAMS<sup>1\*</sup>, Ching-Yu TSENG<sup>2\*</sup>, Paul FOSTER<sup>3\*</sup>, Vinson LUO<sup>4\*</sup>,  
Leanne R. KETTERLIN-GELLER<sup>5\*</sup>, Eric C. LARSON<sup>6\*</sup>, and Corey CLARK<sup>7\*</sup>

<sup>1,2,3,4,5,6,7</sup> Southern Methodist University, Dallas, TX

eladams@smu.edu, etseng@smu.edu, pdfoster@smu.edu, vluo@smu.edu, lkgteller@smu.edu, eclarson@smu.edu,  
coreyc@smu.edu

## ABSTRACT

This study describes a standard decomposition process, which is designed to decompose content standards into observable components that might illustrate computational thinking skills. These components will be integrated into an online game-based learning environment as evidence of learning (EoL) and mastery (EoM). Focusing on three computer science standards, we describe how the standard decomposition process was used to generate standard decomposition tables. We show samples of the content of these decomposition tables and describe how these tables evolved based on educator feedback.

## KEYWORDS

Computational thinking, Design-based implementation research, Game-based learning, Middle grades

## 1. INTRODUCTION

The definition of computational thinking (CT) has evolved over the last several decades. In early work, Papert (1972) generated the term CT to describe children's learning during programming experiences. More recently, Wing (2006) broadened the definition of CT to include students' thought processes. Jansen et al. (2018) concluded that CT provides people with a method to restructure complex real-world problems into systematic and well-structured problems and supports people in designing solutions that can be manipulated by machines or humans. Grover and Pea (2013) further built on this perspective, stating "CT's essence is thinking like a computer scientist when confronted with a problem" (p. 39). Similarly, Aho (2012) considered that CT assists people in representing the solutions for solving complex problems as computational steps and algorithms. In this study, we adopt the CT definition as: a thought process (including a set of thinking skills) that occurs when students are confronted with a problem that can be formulated into steps and the solution can be executed by humans or machines.

Most CT research focuses on programming-based environments. For example, Kazimoglu et al. (2012) had students design a program to control a robot. Brennan & Resnick (2012) used Scratch (a visual programming language) to develop CT skills, and Basawapatna et al. (2011) designed CT games. Many tools are available for educators to teach students how to code and write programming languages. In our study, we extend this work by defining CT skills more broadly and encouraging students to practice and make connections between CT skills.

We use an online game-based learning environment to provide middle grades students with unique learning opportunities focused on CT. Game-based learning offers unique affordances for "stealth" learning (Sharp, 2012). For example, when playing games, students experience a state of flow (Csikszentmihalyi et al., 2014), which contributes to immersive learning experiences while playing. CT education researchers are working to extract and quantify these learning experiences to understand if and what students are learning during immersive gameplay (e.g., Grover et al., 2015; Grover et al., 2017).

Immersive game-based learning environments are innovative, covert ways to assess students' learning. The assessment information gathered within game-based learning environments could support teachers in tailoring student learning experiences based on students' needs. In this study, we use the terms **Evidence of Learning (EoL)** and **Evidence of Mastery (EoM)** to describe observable behaviors to show students are progressing toward mastery (i.e., EoL) or show evidence of mastery (i.e., EoM). In our study, game developers will use this information to design learning experiences and integrate them into an existing commercial game. The most salient evidence of students' learning will be extracted and communicated to teachers to inform differentiated instruction focused on CT skills.

## 2. CURRENT PROJECT PURPOSE

This study is part of a larger interdisciplinary project designed to develop a game-based learning environment within the existing Minecraft mod "Lumber Jack Tycoon." The learning environment will be developed for middle grades students, designed around focus CSTA computer science standards with an emphasis on CT. Teachers will receive information about their students' progress toward mastering learning standards through integration between the game, a data collection cloud infrastructure, and a learning management system called Canvas.

We use design-based implementation research (DBIR) to guide the development of the game-based learning environment (Confrey, 2019; Fishman, et al., 2007; Penuel et al., 2011). As such, we rely heavily on co-development with educators who work directly with students who the game will ultimately serve. We formed an Educator Advisory Panel (EAP), which included middle grades educators with an interest in computer science and CT. The five EAP educators represented six middle schools across four public school districts in the southern United States. Three educators identified as teachers, one identified as an instructional coach, and one identified as an instructional technology specialist.

Working with five EAP members, we identified middle grades CSTA computer science standards to focus on within the game (subsequently referred to as the focus standards). These standards were: (a) high priority for teachers' instruction, (b) tended to be difficult to teach, (c) may be taught efficiently in Minecraft, and (d) were relevant to CT (Tseng et al., 2020). The selected standards were grouped thematically into four groups including: (a) data and analysis, (b) problem decomposition, (c) teamwork and organization, and (d) equity and impact. For the purpose of this paper, we target the focus standards for data and analysis. We selected this group of standards given the strong connections to STEM disciplines and CT.

As part of the larger project, we developed a process for decomposing the game and standards separately and then integrating those decompositions to create the game-based learning environment. For the purpose of this study, we describe the standard decomposition process that was developed to unpack or decompose content standards into components that illustrate CT skills. We refer to this process as the standard decomposition process throughout this paper. Our research question is: *Using the standard decomposition process and incorporating educator feedback, what are the evidences of learning and mastery for three middle grades CSTA focus standards relating to the data and analysis thematic group (2-DA-07, 2-DA-08, and 2-DA-09)?*

### 3. METHOD

Guided by DBIR, we partnered with educators to decompose the three focus standards. A primary goal of this work was to create standard decomposition tables that could be used to inform assessment development within the game-based learning environment. The standard decomposition process included seven phases, which began in August 2020 and are ongoing. In this section, we describe the seven phases (3.1 - 3.7) that comprise the standard decomposition process.

#### 3.1 Identify Existing Curricula Related to the Focus Standards.

We developed a repository of curricular resources related to middle grades computer science and CT. These curricular resources were identified through a web search, as well in consultation with our EAP and other researchers engaged in this work. These resources included well-developed data and analysis units with learning activities that were focused on conceptual understanding, rather than programming or coding.

#### 3.2 Review Curricular Resources.

Two researchers separately reviewed the curricular resources to decompose each standard into:

1. **Steps** related to each standard, suggesting an order for the cognitive processes that students might engage in related to the overall standard
2. The **importance or objectives (OI)** for each step within the standard decomposition
3. The **pre-knowledge, skills, and abilities (pre-KSAs)** that students would need to develop as evidence of

learning or evidence that they are progressing toward mastery within each step of the standard decomposition (e.g., necessary pre-requisite knowledge related to each standard)

4. The **knowledge, skills, and abilities (KSAs)** that students would need to develop as evidence of mastery within each step of the standard decomposition

#### 3.3 Reconcile Differences.

Researchers met to collaboratively discuss standard decomposition tables and combine their separate tables into one standard decomposition table including steps related to each standard, each with corresponding OIs, pre-KSAs, and KSAs.

#### 3.4 Gather Educator Feedback on the Steps, OIs, Pre-KSAs, and KSAs.

We met virtually with five EAP members to discuss the focus standards and the extent to which the steps, OIs, pre-KSAs, and KSAs reflected their expectation of what their students should know and be able to perform related to the focus standard. For 2-DA-08, we drafted example **evidence of learning (EoL)** corresponding to the pre-KSAs and **evidences of mastery (EoM)** corresponding to the KSAs, which reflected observable behaviors that students demonstrate in the classroom related to each standard. During the meeting we also encouraged the five educators to provide EoL and EoM related to 2-DA-07 and 2-DA-09. Following the meeting, we solicited additional feedback on the standard decomposition tables using Google Documents. Two of five educators participated in the additional opportunity to provide feedback.

#### 3.5 Integrate Feedback from Educators and Generate EoL and EoM based on Existing Curricula and Educator Feedback.

Following the virtual meeting with educators, we systematically reviewed the meeting transcript and researcher notes to refine the content of the standard decomposition tables based on educator feedback. In addition, we generated EoL and EoM for 2-DA-07 and 2-DA-09 based on educator feedback and the review of curricular resources.

#### 3.6 Confer with Educators and Gather Educator Feedback on the EoL and EoM.

We invited educators to provide feedback asynchronously on the complete standard decomposition tables using an online platform called Google Jamboard. One of the purposes of this review was to ensure that we accurately captured educator feedback in our revisions. A second purpose was for educators to provide feedback on the EoL and EoM for 2-DA-07 and 2-DA-09. Two of five educators participated in this opportunity.

#### 3.7 Integrate Feedback from Educators.

We systematically reviewed the educator comments related to the updated standard decomposition tables and refined the language in the standard decomposition tables based on educators' feedback.

## 4. RESULTS

In this section, we summarize the EoLs and EoMs for the focus standards and summarize changes that we made based on educators' feedback. These tables directly relate to this study's research question, which focuses on identifying EoL and EoM. Tables 1 through 3 include sample EoL and EoM statements from the full standard decomposition tables. The contents of these tables identify a sample of behaviors that students demonstrate to show EoL or EoM with an emphasis on CT related to the focus standards, informed by a review of existing curricula and feedback from five educators.

Table 1 includes a sample of EoL and EoM for 2-DA-07: Represent Data using Multiple Encoding Schemes. We identified three steps within this standard including (1) access data, (2) clean data, and (3) create and apply encoding rules.

Table 2 includes a sample of the EoL and EoM for 2-DA-08: Collect Data using Computational Tools and Transform the Data to Make it More Useful and Reliable. We identified four steps within this standard including (1) collect data, (2) clean data, (3) organize data, and (4) explain data.

Table 3 includes a sample of EoL and EoM for 2-DA-09: Refine Computational Models based on the Data [Students] have Generated. We identified two steps within this standard including (1) review model output, and (2) refine the model.

*Table 1. Sample of Standard Decomposition Table for “2-DA-07: Represent Data using Multiple Encoding Schemes”.*

Steps	EoL	EoM
Access Data	Manipulate data using computing devices to aid human processing	Identify the type of data (e.g., numeric, categorical)
		Explain why different types of data are valuable
Clean Data	Filter variables to identify which data are necessary	Recognize patterns within a column or row of data
	List possible encoding methods	Evaluate different encoding methods used
Create and Apply Encoding Rules	Describe the necessary features of an encoding system	Compare encoding methods with other students' work
	Choose the best way to encode information based on how it will be used	Resolve conflicts when using encoding rules

*Table 2. Sample of Standard Decomposition for “2-DA-08: Collect Data using Computational Tools and Transform the Data to Make it More Useful and Reliable”.*

Steps	EoL	EoM
Collect Data	Identify examples of data and non-data	Identify and record relevant data
Clean Data	Make decisions about how to handle missing data	Compare cleaning strategies with other students
Organize Data	Identify different systems for representing data	Employ an effective data organization system with team members
Explain Data	Evaluate different organizational systems	Explain how data were identified, collected, and stored in a way that connects to solving a problem

*Table 3. Sample of Standard Decomposition Table for “2-DA-09: Refine Computational Models based on the Data [Students] have Generated”.*

Steps	EoL	EoM
Review Model Output	Extend encoding schemes to rules of models	Describe how data generated by the model help solve a problem
Refine the Model	Identify opportunities to improve the model	Create an improved model (i.e., more accurate, efficient, simpler, and/or intuitive)

The sample content from Tables 1 through 3 reflects the types of behaviors that students would be expected to display in the classroom related to each of the focus standards, with an emphasis on CT skills.

Because this study's research question specifies the incorporation of five educators' feedback across iterations of the EoL and EoM, we share general findings related to how the standard decomposition tables evolved based on educator feedback. In the initial synchronous feedback session, the educators registered concern about students' lack of familiarity with computers. Further, the educators emphasized the need for scaffolding. Based on educator comments on specific statements, we made a number of revisions and additions. Following the first feedback session, the number of statements for 2-DA-08 increased two-fold and many of the previous statements were clarified based on educator feedback. Time constraints meant only eight suggestions were received on 2-DA-07

and none on 2-DA-09. During the follow up asynchronous feedback opportunity, two educators identified having students do things multiple ways, the use of peers for sharing and review, and the use of manipulatives as positives. Although there was a similar number of changes suggested on specific items in the second round of feedback, most of the comments were on clarifying the language of the standards and making the verbs as observable as possible.

## 5. DISCUSSION

This paper describes a standard decomposition process designed to inform the development of an online game-based learning environment in Minecraft. The process described in this paper explicates student behaviors that build from progressing toward mastery (i.e., EoL) to mastery (i.e., EoM). As such, the types of behaviors or cognitive processes that students are expected to do are articulated. The standard decomposition process defined student behaviors connected to the standards that emphasize CT skills. This information subsequently informs what students will actually be expected to do within the gaming experience.

The standard decomposition tables are a contribution to the field of education focused on computer science and CT because they build on existing assessment work in CT (Grover et al., 2015; Grover et al., 2017). The standard decomposition tables were co-developed with five educators within a DBIR framework. The phases described in the methods of this paper outline a process for gathering and integrating educator feedback systematically. Due to space limitations, this paper includes a sample of the standard decomposition tables.

Minecraft allows us to build a community-based gaming environment that facilitates an understanding of CT. Our next step is to integrate the learning standard decompositions with the game element decompositions. This integrative step will result in the development of learning experiences within Minecraft. This game development process is highly scalable for others interested in doing similar game development work.

## 6. REFERENCES

- Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal*, 55(7), 832-835.
- Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., & Marshall, K. S. (2011). *Recognizing computational thinking patterns*. Paper presented at SIGCSE, Dallas, TX.
- Breanna, K., & Resnick, M. (2012). *New frameworks for studying and assessing the development of computational thinking*. Paper presented at the American Educational Research Association, Vancouver, Canada.
- Csikszentmihalyi, M., Abuhamdeh, S., & Nakamura, J. (2014). *Flow and the foundations of positive psychology*. Springer.
- Jansen, M., Kohen-vacs, D., Otero, N., & Milrad, M. (2018, June). A complementary view for better understanding the term computational thinking. *Proceedings of the International Conference on Computational Thinking Education 2018*. Hong Kong: The Education University of Hong Kong.
- Confrey, J. (2019). Leading a design-based research team using agile methodologies to build learner-centered software. In K. R. Leatham (Ed.) *Designing, Conducting and Publishing Quality Research in Mathematics Education* (pp. 123-142). Springer.
- Fishman, B. J., Penuel, W. R., Allen, A., Cheng, B. H., & Sabelli, N. (2007). Design-based implementation research: An emerging model for transforming the relationship of research and practice. *National Society for the Study of Education*, 112(2), 136-156.
- Grover, S., & Pea, R. (2013). Computational thinking in K—12: A review of the state the field. *Educational Researcher*, 42(1), 38–43.
- Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer Science Education*, 25(2), 199-237.
- Grover S., Basu, S., Bienkowski, M., Eagle, M., Diana, N., and Stamper, J. (2017). A framework for using hypothesis-driven approaches to support data-driven learning analytics in measuring computational thinking in block-based programming environments. *ACM Transactions on Computing Education*, 17(3) 457-468.
- Kazimoglu, C., Kiernan, M., Bacon, L., & Mackinnon, L. (2012). A serious game for developing computational thinking and learning introductory computer programming. *Social and Behavioral Sciences*, 47, 1991-1999.
- Papert, S. (1972). Teaching children thinking. *Programming Learning and Educational Technology*, 9(5), 245-255.
- Penuel, W. R., Fishman, B. J., Cheng, B. H., & Sabelli, N. (2011). Organizing research and development at the intersection of learning, implementation, and design. *Educational Researcher*, 40(1), 331-337.
- Tseng, C., Ketterlin-Geller, L. R., Clark, C. & Larson, E. (2020). *STEM+C educator advisory panel summer 2020* (20-18). Dallas, TX: Research in Mathematics Education, Southern Methodist University.
- Sharp, L. A. (2012). Stealth learning: Unexpected learning opportunities through games. *Journal of Instructional Technology*, 1, 41-48.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.



# **Computational Thinking and Teacher Development**

# Different Paths, Same Direction: How Teachers Learn Computational Thinking in STEM Practices through Professional Development

Sally WU<sup>1</sup>, Amanda PEEL<sup>2\*</sup>, Connor BAIN<sup>3</sup>, Michael HORN<sup>4</sup>, Uri WILENSKY<sup>5</sup>

<sup>1, 2, 3, 4, 5</sup>Northwestern University, USA

sally.wu@northwestern.edu, amanda.peel@northwestern.edu, connorbain2015@u.northwestern.edu, michael-horn@northwestern.edu, uri@northwestern.edu

## ABSTRACT

One approach to expanding computational thinking (CT) in K-12 education is for mathematics and science teachers to integrate CT into their curriculum. However, teachers must first engage with computational practices themselves and gain confidence in their ability to teach CT to their students. To this end, we developed a four-week professional development for 11 science and mathematics high school teachers. We engaged teachers in four CT-STEM practices focused on data, modeling, algorithms, and programming. Then, each teacher co-designed a computationally enhanced curriculum for their classroom in collaboration with a member of our research team. Data from pre-post surveys showed an overall increase in teachers' reported confidence in teaching CT-STEM practices after the professional development. However, teachers' change in confidence varied across the four practices and across individual teachers. The variance aligns with the variance in teachers' responses on what they learned. Different teachers reported learning a variety of knowledge and skills, whether about CT itself, specific CT tools, or how to integrate CT into a particular curricular topic. These findings suggest that engaging teachers in co-design of computational-enhanced STEM curriculum may cultivate multiple pathways that help teachers integrate CT into K-12 classrooms.

## KEYWORDS

computational thinking, STEM education, CT integration, teacher professional development, curriculum design

## 1. INTRODUCTION

Computational thinking (CT) has been recently emphasized in K-12 education, particularly in mathematics and science learning (Barr & Stephenson, 2011; Grover & Pea, 2013). However, the adoption of CT has been hindered by the difficulties teachers often face when trying to integrate CT into their curriculum. CT is relatively new and many teachers are not equipped with the skills and tools to integrate it effectively into their curriculum (Aljowaed, & Alebaikan, 2018; Yadav, Gretter, Hambrusch, & Sands, 2016). Further, teachers are unfamiliar with how to teach CT practices, particularly where they intersect with their content areas (Ketelhut, Mills, Hestness, Cabrera, Plane, & McGinnis, 2020; Wu, Looi, Liu, & How, 2018).

Researchers have identified key CT-STEM practices that reflect authentic STEM practices used in modern science (Weintrop et al., 2016). These CT-STEM practices aim to help students develop science and mathematics content understanding by engaging students in computational

inquiry. The CT practices in mathematics and science classrooms are organized into four strands: data practices, modeling and simulation practices, computational problem-solving practices, and systems thinking practices. Our team has expanded these practice categories to include algorithms and programming since these are key CT practices identified by others (e.g., Brennan & Resnick, 2012; Grover, 2017; Peel, Dabholkar, Wu, Horn & Wilensky, in press; Selby & Woollard, 2013; Tang, Yin, Lin, Hada, & Zhai, 2020). In this paper, we focus on modeling and simulation (using, modifying, and creating computational models) and data practices (collecting, visualizing, and analyzing data), as well as algorithms and programming.

While the CT-STEM practices are present throughout mathematics and science content, they can vary in their use across subject areas. As such, teachers and curriculum designers typically choose to focus on one or two central CT-STEM practices in a unit. For example, a physics teacher may design a unit that focuses on computational modeling by having students use a model to understand a phenomenon, collect and analyze data from the model, and explore the model's algorithm and how that program runs the model. Designing curricula with such practices requires software and knowledge of that software that allows teachers to build a complex model, collect data, and allow students to code.

The integration of CT into science and math classes requires both curriculum designers and teachers to reimagine classroom practices and to learn how to incorporate computational methods and tools (Ball & Forzani, 2009; Windschitl et al., 2012). Teachers require professional development, resources, and support to learn about CT, how to use the computational tools, and how to teach CT-STEM practices. While recent approaches to supporting teachers with CT integration have begun to emerge, this area of professional development and research is in early stages (Ketelhut et al., 2020; Yadav, Mayfield, Zhou, Hambrusch, & Korb, 2014; Yadav, Zhou, Mayfield, Hambrusch, & Korb, 2011).

We have begun to address this gap through professional development that focuses on developing computationally enriched STEM units with teachers. We position teachers as active co-designers in modifying their existing STEM curricula to include computational tools and practices. Co-designing with teachers foregrounds their views on curriculum alignment with teaching practices and expectations for student learning (Allen & Penuel, 2015; Coburn, 2005; Penuel, Riel, Krause, & Frank, 2009). Co-

design, as we have defined it, engages teachers in constructionist learning in that teachers learn through the construction of new CT-integrated units (Kelter et al., 2020a). When teachers actively design and create lessons and computational tools, they learn about CT and its integration in the classrooms (Peel et al., 2020b). However, it is unclear how teacher knowledge develops regarding specific CT-STEM practices. Prior work has shown variation in teacher outcomes from co-design and professional development experiences, given their different goals and prior experiences with CT (Kelter et al., 2020a; Naimipour, Guzdial, & Shreiner, 2020; Svihla, Reeve, Sagy, & Kali 2015). We expand this work by exploring how individual teachers differ in their learning of CT-STEM practices. Specifically, we investigate: (1) Did teachers develop confidence in teaching each CT-STEM practice through a four-week professional development? and (2) What did they learn about CT?

**Table 1. Teacher Pseudonyms and CT Background.**

Teacher	Background / Experience with CT
Beth	3rd year freshman biology teacher, has PhD in microbiology, uses some CT practices, e.g., “asked students to write step by step procedures to design experiments and I have asked about the rules something needs to follow to work properly (like a ribosome incorporating new amino acids from a set of instructions on mRNA)”
Betsy	17th year inclusion biology and chemistry teacher, little CT experience, “used [CT] sparingly when teaching Chemistry”
Carrie	11th year honors chemistry teacher, participated in CTSI 2019 and two prior PDs with CT-STEM team, uses CT to teach specific content, e.g., “Asking students to come up with the ‘rules’ for gas particle movement prior to having them work through a NetTango on the same subject.”
Chelsea	9th year chemistry and physics teacher, implemented units developed by Carrie during CTSI2019
Emma	11th year environmental science and biology teacher, participated in CTSI 2019 and two prior CT-STEM PDs, engages students in CT in various ways, e.g., “Use of computational models (simulations, sage modeler), strong focus on data collection, analysis, visualization. Discussions of how scientists look at real world problems, using tools that have been developed to look at phenomena or problems we are studying”
Evan	15th year AP environmental sciences teacher, little experience with CT: “I have done small chunks of CT work through the years, but never intentionally set out to instill a CT nature into my curriculum or science pedagogy.”
Matt	8th year AP Statistics and geometry teacher, participated in CTSI 2019 and “use other simulation tools for my AP Statistics class on a regular basis.”
Martin	15th year mathematics, no prior experience with CT, implemented unit developed by Matt during CTSI2019
Marshall	10th year mathematics, computer sciences, and social sciences teacher, has experience teaching programming and algorithms
Paul	31st year AP physics teacher, some prior experience with code, used CT to teach specific concepts, e.g., “Have students write code for laws, i.e., Snell's Law, etc.”
Parvez	23rd year freshman physics, some prior experience with Java, implemented a short CT-STEM curriculum three years ago, use computational models, specifically “controlled PhET simulations a lot with worksheets with directions and critical thinking questions”

## 2. METHOD

CT-STEM Summer Institute (CTSI), a four-week professional development workshop that positioned teachers and researchers as co-designers of curricula. In 2020, 11 high school science or mathematics teachers from four U.S. public schools with varying experience with CT

participated (See Table 1). Note that pseudonyms are given to align with teachers' subject area: biology, chemistry, environmental sciences, mathematics, and physics. All teachers received up to \$4000 U.S. dollars for participation and were asked to create a CT-STEM curriculum for their classroom that would be implemented in the following school year. The teachers, seven graduate students, and one post-doctoral researcher were assigned to co-design teams by subject area.

Due to the COVID-19 pandemic, CTSI was held online. Table 2 shows an overview of activities held during the four-week professional development. For discussions, workshops, and co-design meetings, teachers and researchers met on Zoom for synchronous discussion or instructional activities. Otherwise, they communicated asynchronously via emails and Slack and worked on materials asynchronously.

**Table 2. Overview of Professional Development Activities over four weeks, Organized by Day.**

Week	Monday	Tuesday	Wednesday	Thursday	Friday
1	Introductions  Intro to CT Lesson	CT-STEM units  CT-STEM Practices	Intro to programming  Computational tools, Part 1	Computational tools, Part 2  Unit planning	Intro to co-design teams  Reflection
2-4	Co-design  Review units	Co-design	Co-design  CT-STEM Workshop	Co-design  Cross-Team Conference	Co-design  Reflection  <i>Mini-Expo (Week 3)</i> <i>Expo (Week 4)</i>

To introduce teachers to computational practices and tools, the first week of CTSI (4.5 days) consisted of workshops led by the researchers. Sessions introduced teachers to CT-STEM practices through lessons designed for students. Lessons demonstrated how computational tools can engage students in CT-STEM practices while learning disciplinary content. For example, the Intro to CT lesson (<https://ct-stem.northwestern.edu/curriculum/preview/495/page/0/>) first asked teachers to use, modify, and debug a series of computational models that simulate how fire spreads through a forest (Wilensky, 1997) using *NetLogo*, a multi-agent programmable modeling environment (Wilensky, 1999). Next, teachers collected and analyzed ‘density vs. percent burned’ data using *CODAP* (Common Online Data Analysis Platform, 2020), a web-based data analysis environment. Then, they posed research questions about other variables that may affect the spread of fire and discussed how scientists use such computational models. On each student page, teachers identified what CT-STEM practices students would engage in and how they would support students in these activities. Their ideas were discussed within a small breakout room and with the whole group to ensure that everyone was on the same page, similar to how a teacher may engage with students online.

In addition to *NetLogo* and *CODAP*, teachers engaged with *NetTango*, a blocks-based programming interface for exploring *NetLogo* Web models (Horn, Baker, & Wilensky, 2020) in a lesson focused on basic programming (<https://ct-stem.northwestern.edu/curriculum/preview/1505/page/0/>)

and a lesson on ecology and predator-prey dynamics using blocks to create wolf-moose interactions (<https://ct-stem.northwestern.edu/curriculum/preview/353/page/0/>).

Teachers completed each lesson as students and discussed CT pedagogy for using these tools. Additional CT tools such as Python and SageModeler (2020) were introduced to specific teachers interested in using them.

Because the goal of CTSI is for teachers to design a CT-integrated curriculum for their classroom, the last three weeks of CTSI provided co-design time for teams of teachers and researchers to work on computational models and curricular units. Each subject-area team included at least two teachers, one researcher, and one undergraduate research assistant. Teams varied in how they communicated and co-designed curricula. Some teams met every day to check in and work together on models, curricula, and/or student activities via video conference. Others worked asynchronously using online tools (primarily Slack and Google Drive) and met when activities were ready or when someone on the team needed help. Regardless, all teams reviewed each other's work and gave feedback on materials at least once a week, typically on Monday or Tuesday.

To foster community across teams, all teachers and researchers participated in weekly Wednesday workshops on relevant topics (e.g., CT pedagogy) as well as Friday reflection sessions. Teachers also received additional feedback on their units in various formats. Every Thursday, each teacher was paired with another teacher outside of their team to discuss their units (Cross-Team Conferences). Further, in Week 3, they discussed their unit with CT professionals in their subject area (Mini-Expo). Finally, at the end of CTSI, the teachers showcased their co-designed CT-STEM curriculum to CT professionals, colleagues, friends, and family in an Expo open to the community, using videos about their units and discussions with those who attended: [https://padlet.com/sally\\_wu/CTSIExpo](https://padlet.com/sally_wu/CTSIExpo).

## 2.1. Data Sources

To assess changes in teacher confidence after CTSI (RQ1), we conducted 36-item *pre/post surveys* that asked teachers to rate on a 5-point Likert Scale (1 = Strongly Disagree, 5 = Strongly Agree) their confidence in teaching each of four CT-STEM practices: data, simulation and modeling, algorithm, programming. For example, data practices items include "I am confident in my ability to identify computational data practices in an educational STEM activity." and "I am confident in my ability to answer student questions regarding computational data activities." Teachers also responded to an open-ended question that asked: "What did you learn from CTSI?"

## 3. RESULTS

We use survey responses to assess whether teachers developed confidence in each of the four CT practices (RQ1) and what teachers learned about CT (RQ2) after the four-week professional development.

### 3.1. Overall change in teachers' reported confidence

As shown in Figure 1, our teachers, on average, felt some confidence with all four CT-STEM practices prior to our

summer institute. After the summer institute, they, on average, reported even higher confidence in their ability to teach all four CT-STEM practices, particularly in modeling and algorithms. We ran an asymptotic Wilcoxon-Pratt Signed-Rank test to analyze the pre-post changes in teacher confidence across each of the four practices. Two of the practices, modeling and algorithms, were statistically significant at the 10% level with p-values of 0.026 and 0.081 respectively. The changes in data practices and programming practices were not significant at generally accepted significance levels with p-values of 0.130 and 0.197. We did not expect statistical significance because of our relatively small sample size and with some participants who showed zero differences (partially due to a ceiling effect).

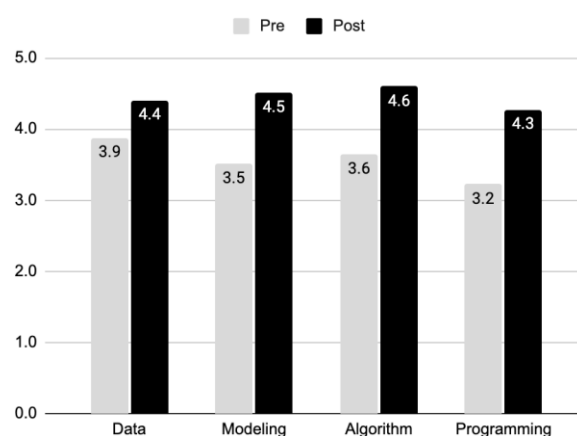


Figure 1. Average Reported Confidence in Four CT-STEM Practices.

To better understand the overall change, we examined each teachers' pre and post confidence rating for each of the four practices (Table 3). Some teachers rated their confidence as extremely high (max = 5) at both the pre and post surveys, such as Emma on data and modeling practices and Marshall on algorithms and programming practices, resulting in no pre-post differences. The pre-survey ratings show some variation in teachers' prior confidence in data, modeling, and algorithm practices (range = 2.0 to 5.0), as well as large variation in teachers' prior confidence in programming (range = 0.0 to 5.0). At the post-survey, teachers reported generally high confidence in all practices (range = 3.3-5.0).

Table 3. Teachers' Reported Confidence Rating in Each of the Four CT-STEM Practices.

Teacher	Data		Modeling		Algorithm		Programming	
	Pre	Post	Pre	Post	Pre	Post	Pre	Post
Beth	3.0	5.0	2.6	4.9	2.6	5.0	0.0	5.0
Betsy	2.8	3.9	3.7	3.8	2.9	4.8	2.8	3.8
Carrie	4.3	4.0	4.4	4.9	4.4	4.2	3.4	3.3
Chelsea	4.0	3.9	4.0	3.6	4.0	4.0	3.8	4.0
Emma	5.0	5.0	5.0	5.0	4.9	4.7	4.7	4.3
Evan	3.0	4.8	2.0	5.0	2.0	5.0	1.0	5.0
Matt	4.0	5.0	3.0	5.0	3.0	5.0	3.0	5.0
Martin	3.4	4.1	2.6	3.8	3.7	4.6	3.7	3.4
Marshall	4.7	5.0	3.1	5.0	5.0	5.0	5.0	5.0
Paul	4.2	4.0	4.4	4.3	4.3	4.1	4.0	3.9
Parvez	4.2	3.8	3.9	4.3	3.3	4.2	4.1	4.2

Several teachers reported greater confidence in all practices after CTSI, particularly Beth, Evan, and Matt. Some teachers reported greater confidence in some specific practices but not others. For example, Betsy reported only a slight difference in confidence in regard to modeling and overall increased confidence in all other practices. Martin Marshall, and Parvez, who teach math or physics, also fall into this category, reporting similar pre-post difference in data and programming practices and general increases in modeling and algorithm practices. Finally, a set of teachers reported similar confidence in all four practices before and after CTSI, specifically Carrie, Chelsea, Emma, and Paul.

### 3.2. What teachers learned

We explore what teachers learned (RQ2) by analyzing their responses to “What did you learn from CTSI?” We use the qualitative responses to understand the variance in individual teachers’ reported change in confidence across teachers who reported greater confidence in all practices after CTSI (Beth, Evan, Matt), greater confidence in some specific practices but not others (Betsy, Martin, Marshall, Parvez), and similar confidence before and after CTSI (Carrie, Chelsea, Emma, Paul).

Teachers who generally showed a general increase in confidence across all four practices (Beth, Matt, Evan) mentioned learning about a variety of related topics that helped them co-design their curriculum, including CT, specific tools (e.g., NetLogo), their subject-area content, collaboration, and pedagogy:

*I learned how to use NetLogo, NetTango, and CODAP, how to integrate such models into a content-heavy, nuanced unit, and a lot about collaboration. (Beth)*

*OMG- I have learned to embrace co-design, learned to work at odd hours, Slack the heck out of my co-design mates and learn the basics of coding and manipulating code. I have also learned to value the feedback my mates have given me, their patience and to learn to take feedback positively for a growth-mindset. I have also learned that my science pedagogy is in need of a redshift, or rather a new lens to look at science through- that of CT. I am grateful to be energized by all the possibilities and potential accomplishments this will translate into for my students. (Evan)*

*Increased my understanding of computational thinking, computational modeling, incorporating CT practices in a mathematics classroom, and my ability to develop and adapt NetLogo & CODAP models to fit my needs in a statistics classroom. (Matt)*

All three teachers discussed how learning about CT and specific tools helped them address their teaching goals. Beth and Matt mentioned learning about computational models and integrating them into their classroom. Further, Evan described how he grew as a designer and teacher, including learning to value his team’s feedback and gaining “a new lens to look at science” through CT.

The teachers who showed increased confidence in some of the practices (Betsy, Martin, Marshall, Parvez) also reported gaining specific skills and CT knowledge that support students in their classrooms:

*I learned a lot about programming. I learned that I can still learn. I learned what computational thinking is and how to apply it in the classroom. (Betsy)*

*I learned what the heck CT means, I learned how to incorporate CT into my lessons, I learned a little bit of programming in NetLogo and how to use CODAP, I brushed up slightly on my Python skills, and I deepened my understanding of my own content (specifically sampling distributions). (Martin)*

*I got a lot of technical skills in Python, CODAP, and NetLogo. I can build (but better modify!) agent-based simulations! I can make quick data visualizations with CODAP! I can make MUCH BETTER data visualizations with Python! I also learned some nuanced ideas about how to better incorporate CT into a scaffolded unit, and I think the segue into the more advanced concepts is done much more smoothly than I initially planned.” (Marshall)*

*Learned to code in NetLogo and make custom designed lessons. Also learned to use NetTango block modeling for students. (Parvez)*

These teachers all mention learning to code/program. Most of them have some prior experience and thus were able to gain specific tools and skills to build activities for their students. This may explain why they did not necessarily gain confidence in teaching particular CT-STEM practices, but instead, they came away with new ideas for how to use tools with their students, such as “MUCH BETTER data visualizations with Python” and “nuanced ideas about how to better incorporate CT into a scaffolded unit” (Marshall).

The set of teachers that showed little change in reported confidence (Carrie, Chelsea, Emma, Paul) also mentioned learning about CT, content, and curriculum design. However, compared to other groups, they mention more specific strategies for teaching their subject area and curricular topic:

*I was introduced to Sage Modeler, and I also learned additional details about working with the [curriculum editor] interface. (Carrie)*

*CTSI was helpful this year to separate various aspects of computational thinking. (Chelsea)*

*I was able to develop a new unit around infectious diseases, this led to a lot of content knowledge about particular diseases, as well as CT knowledge of how to model and think about these diseases. In working with my team I was able to break down specific knowledge points for kids to figure out and develop models to help them do that. (Emma)*

*I have learned a lot about coding. Also, the coding forced me to think about physics- concepts and equations- such that I could write correct codes to model phenomena. (Paul)*

These teachers reported learning different things, from CT itself (Chelsea), to specific tools (Carrie), and ways to think about teaching their content through models (Emma), or coding (Paul). Particularly for Emma and Paul, it may be that designing CT-STEM activities helped them realize what they did not yet know about their topic and “forced” them to think deeper about the nature of their content from the CT perspective.



## 4. DISCUSSION

Given that CT integration is difficult for teachers, our work begins to shed light on how and what teachers can learn about CT through professional development focused on co-design of CT-integrated curriculum. Our teachers, on average, felt some confidence with all four CT-STEM practices prior to our summer institute, which is not surprising given that they showed interest in CT and wanted to participate in CTSI over the summer. Yet, after the professional development, our teachers, on average, reported higher confidence in teaching CT-STEM practices, particularly in modeling and algorithm practices. This gives us encouragement that the professional development may be a good approach to help teachers engage in and learn about CT-STEM practices, even if they are already confident in some of those practices.

Our findings also show that teachers' confidence did not change in the same way across the four CT-STEM practices. Some seemed to gain confidence in all practices, while other teachers only gained confidence in a few practices, and still others showed no change after our professional development. These differences were expected, given that a few teachers were already extremely confident in teaching CT-STEM practices prior to the professional development, and thus it did not affect their confidence with specific practices. For them, the experience may have been an opportunity to learn about specific software and tools that they can use to design curricula or engage students in their classroom, rather than an opportunity to learn about CT-STEM practices. Teachers' responses to what they learned from the summer institute suggests that they learned many related skills and ideas for designing and implementing CT-STEM curriculum. Some learned how to program and code using particular tools and strategies to address their teaching goals. Others learned about CT itself, how it may be "a new lens to look at science" (Evan), and how to integrate it into existing content in their curriculum. Some teachers described changes in how they view content and teaching, which may affect their pedagogy, which will be analyzed in a future paper on how teachers implemented their units.

Even though our study only involved 11 teachers, it revealed much variation in what teachers gained through our professional development, which researchers and educators should take into account when designing and assessing such programs. Our findings align with prior work that shows variation in teacher outcomes from co-design experiences and a need for multiple sources of data to capture teachers' pathways (Kelter et al., 2020; Naimipour et al., 2020; Peel et al., 2021; Svihla et al., 2015). Although quantitative measures are important to ensure that teachers gain the prerequisite skills and knowledge to develop quality curricula and engage their students in CT, expanding qualitative analyses or designing alternative measures to capture teachers' learning and teaching of CT will help illuminate the various ways in which they grow in their pedagogy. Further, additional measures will help reveal what aspects of co-design and professional development experiences are essential to

ensure teachers' growth builds on their divergent needs and prior experiences with CT.

Given the variance in teachers' goals and experience with CT, we designed our professional development to be adaptable so that teachers can gain the knowledge, skills, and insights required to learn about and integrate CT into their classroom. In our professional development, the co-design sessions were particularly flexible for teachers so that it supports constructionist design (Kelter et al., 2020). Each co-design session foregrounds teachers' goals and thus was shaped by the different types of support and levels of engagement with CT needed to help each teacher achieve their goals. Teachers may be restructuring how they introduce content, brainstorming what features to include in an CT activity, writing student questions, or programming computational models with the support of their co-designers. The variation in the co-design process allows for differentiated support based on teachers' prior experiences and teaching goals for their curriculum. Hence, regardless of their approach, goals, and needs during the professional development, all teachers moved towards the same destination: They all reported learning about some aspect of CT, produced a CT-STEM curriculum, and felt confident in their ability to teach CT-STEM practices to their students alongside content in mathematics and science classrooms. This work shows promise for professional development focused on constructionist design as a way to engage teachers in CT education. We advocate for additional work that empowers teachers as designers of CT curriculum and identifies additional pathways for teachers who may or may not have prior confidence and experience in CT. Such work will help us build additional professional development opportunities that effectively engage teachers in integrating and teaching CT in STEM classrooms.

## 5. REFERENCES

- Allen, C. D., & Penuel, W. R. (2015). Studying teachers' sensemaking to investigate teachers' responses to professional development focused on new standards. *Journal of Teacher Education*, 66(2), 136-149.
- Aljowaed, M., & Alebaikan, R. A. (2018). Training Needs for Computer Teachers to Use and Teach Computational Thinking Skills. *International Journal for Research in Education*, 42(3), 237-284.
- Ball, D., & Forzani, F. (2009). The work of teaching and the challenge for teacher education. *Journal of teacher education*, 60(5), 497-511.
- Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community? *Acm Inroads*, 2(1), 48-54.
- Coburn, C. E. (2005). Shaping teacher sensemaking: School leaders and the enactment of reading policy. *Educational policy*, 19(3), 476-509.
- Common Online Data Analysis Platform [Computer software]. (2020). Concord, MA: The Concord Consortium.

- Grover, S. (2017). Assessing algorithmic and computational thinking in K-12: Lessons from a middle school classroom. In *Emerging research, practice, and policy on computational thinking* (pp. 269-288): Springer.
- Grover, S., & Pea, R. (2013). Computational Thinking in K-12 A Review of the State of the Field. *Educational researcher*, 42(1), 38-43.
- Horn, M., Baker, J. & Wilensky, U. (2020). NetTango Web [Computer Software]. Evanston, IL. Center for Connected Learning and Computer Based Modeling, Northwestern University. <http://ccl.northwestern.edu/nettangoweb/>.
- Kelter, J. Z., Peel, A., Bain, C., Anton, G., Dabholkar, S., Aslan, Ü., Horn, M., & Wilensky, U. (2020). Seeds of (r)Evolution: Constructionist Co-Design with High School Science Teachers. In B. Tangney, J. R. Byrne, & C. Girvan (Eds.), *Proceedings of the 2020 Constructionism Conference*, Dublin, Ireland, May 26— May 29, 2020 (p. 497-505).
- Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2020). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, 29(1), 174-188.
- Naimipour, B., Guzdial, M., & Shreiner, T. (2020, October). Engaging Pre-Service Teachers in Front-End Design: Developing Technology for a Social Studies Classroom. In *2020 IEEE Frontiers in Education Conference (FIE)* (pp. 1-9). IEEE.
- Peel, A., Dabholkar, S., Anton, G., Wu, S., Wilensky, U., & Horn, M. (2020). A Case Study of Teacher Professional Growth Through Co-design and Implementation of Computationally Enriched Biology Units. In Gresalfi, M. and Horn, I. S. (Eds.), *The Interdisciplinarity of the Learning Sciences, 14th International Conference of the Learning Sciences (ICLS) 2020*, Volume 4 (pp. 1950-1957). Nashville, Tennessee: International Society of the Learning Sciences.
- Peel, A., Dabholkar, S., Wu, S., Horn, M.S., Wilensky, U. (in press). An Evolving Definition of Computational Thinking in Science and Mathematics Classrooms. *Proceeding of the 2021 International Conference of Computational Thinking Education and STEM (CTE-STEM)*.
- Peel, A., Kelter, J., Wilensky, U., Horn, M. (2021). Designing Professional Learning Experiences to Support Teachers' Computational Thinking Learning and Confidence. Presented at the Annual Meeting of the National Association of Research in Science Teaching (NARST) 2021.
- Penuel, W. R., Riel, M., Krause, A., & Frank, K. A. (2009). Analyzing teachers' professional interactions in a school as social capital: A social network approach. *Teachers college record*, 111(1), 124-163.
- SageModeler [Computer software]. (2020). Concord, MA: The Concord Consortium and the CREATE for STEM Institute at Michigan State University.
- Selby, C., & Woollard, J. (2013). *Computational thinking: the developing definition*. Retrieved from <https://eprints.soton.ac.uk/356481/>
- Svihla, V., Reeve, R., Sagy, O., & Kali, Y. (2015). A fingerprint pattern of supports for teachers' designing of technology-enhanced learning. *Instructional science*, 43(2), 283-307.
- Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, 148, 103798.
- Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology*, 25(1), 127-147.
- Wilensky, U. (1997). *NetLogo Fire model*. Retrieved December 1, 2019, from <http://ccl.northwestern.edu/netlogo/models/Fire>
- Wilensky, U. (1999). *NetLogo*. Retrieved December 1, 2019, from <http://ccl.northwestern.edu/netlogo/>
- Windschitl, M., Thompson, J., Braaten, M., & Stroupe, D. (2012). Proposing a Core Set of Instructional Practices and Tools for Teachers of Science. *Science Education*, 96(5), 878-903.
- Wu, L., Looi, C.-K., Liu, L., & How, M.-L. (2018). Understanding and Developing In-Service Teachers' Perceptions towards Teaching in Computational Thinking: Two Studies. *Proceedings of the 26th International Conference on Computers in Education*, Philippines: Asia-Pacific Society for Computers in Education.
- Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: understanding teacher experiences and challenges. *Computer Science Education*, 26(4), 235-254.
- Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, 14(1), 5.
- Yadav, A., Zhou, N., Mayfield, C., Hambrusch, S., & Korb, J. T. (2011). *Introducing computational thinking in education courses*. Presented at the Proceedings of the 42nd ACM technical symposium on Computer Science Education.



# An Experience of Conducting Online Teacher Development for Computational Thinking Teaching in a Primary School Context

Siu-Cheung KONG

Department of Mathematics and Information Technology  
The Education University of Hong Kong, Hong Kong  
sckong@eduhk.hk

## ABSTRACT

During the COVID-19 pandemic, we conducted an online-only teacher development course (TDC) on computational thinking (CT) teaching in a primary school context. Twelve in-service primary school teachers participated in the course, which consisted of thirteen 3-hour lessons. Analysis of the participants' CT concepts test results showed that they successfully developed a good understanding of CT. They also significantly improved in all four content knowledge-related dimensions of technological pedagogical content knowledge (TPACK) of programming for CT development. Participants' evaluation of teaching survey reflected that they agreed the course was of high quality. These positive results implied that TDCs on CT teaching can be conducted online successfully. This study indicated the importance of providing a sustained TDCs for teachers to gain sufficient CT knowledge and pedagogies of CT teaching. For successful online teaching, it was essential to maintain the teachers' engagement in class by adjusting the teaching pace to ensure all of them can follow the tasks and assigning sufficient tutors to render timely assistance when support was needed in completing the programming tasks.

## KEYWORDS

computational thinking, online learning and teaching, primary school, programming, teacher development

## 1. INTRODUCTION

Wing (2006) argued that computational thinking (CT) is a thinking process by which one formulates problems and finds solutions by drawing on the fundamental concepts of computer science. To help children become creative problem solvers, the integration of CT into K-12 education through programming is an important initiative (Hsu, Chang, & Hung, 2018). However, there is a limited number of teachers with CS background (Yadav, Gretter, Hambrusch, & Sands, 2016) and a lack of specific pedagogies for teaching CT through programming (Menekse, 2015). Thus, it is critical to conduct professional development for teachers to equip them with sufficient CT knowledge and related pedagogies. Few empirical studies were found on effective TDCs in CT in relation to programming (Menekse, 2015; Kong, Lai, & Sun, 2020). Also, the organization of online-only TDCs on CT teaching was a new and emerging area that needed to be explored. This study aimed to report an experience of conducting an online TDC on CT teaching and the evaluation results of the teachers' learning progress in CT concepts, and technological pedagogical content

knowledge (TPACK) of programming for CT development.

## 2. BACKGROUND

### 2.1. CT in Relation to Programming in a Primary School Context

Programming is regarded as an effective method for developing students' CT (Kong & Abelson, 2019). Block-based programming environments such as Scratch and App Inventor provide a pleasant learning experience for young students because their visual programming languages are favorable for children to learn (Lye & Koh, 2014); as a result, these environments can stimulate students' interest in programming (Weintrop & Wilensky, 2017). Brennan and Resnick (2012) proposed a CT framework after observing young students' behavior while programming with Scratch. This framework consisted of three components: CT concepts, CT practices, and CT perspectives. *CT concepts* refers to the concepts applied in programming. *CT practices* refers to the problem-solving practices used in programming. *CT perspectives* refers to how programmers see themselves, their relationships with others, and the digital world. This framework provided a concrete direction for CT development in relation to programming among young learners. The TDC in this study adopted these three components to develop teachers' competencies in delivering CT lessons in primary school.

### 2.2. Importance of Teacher Development in CT

Two major challenges to incorporating CT education in primary schools were found in this study, which may be overcome by conducting effective TDCs. First, there were only a small portion of teachers with a CS background (Yadav et al., 2016). Some of them were not familiar with block-based programming environments (Hubbard, 2018), which would make it difficult for them to teach CT through programming. Second, sustained professional development was inadequate. Bower et al. (2017) found that most TDCs are short in duration and put emphasis only on teaching how to program rather than on incorporating related pedagogies (Menekse, 2015). Consequently, teachers could only make use of some general pedagogical strategies that were not specific to CT development when teaching (Bower & Falkner, 2015). Kong et al. (2020) suggested that primary school teachers' CT knowledge and their TPACK had significantly improved after completing two 39-hour courses. Therefore, it is critical to provide sustained TDCs for teachers to equip them with ample knowledge and pedagogies of teaching CT through programming.

### 2.3. Factors of Conducting Effective TDCs on CT

There are two main factors leading to an effective TDC on CT development. First, the course design requires a sustained period of learning instead of a one-off lesson (Garet, Porter, Desimone, Birman, & Yoon, 2001). A sustained course provides a better training opportunity for teachers since they have adequate time to acquire the content knowledge. For CT development, Kong et al. (2020) suggested that teachers need to learn how to teach CT and apply what they had learned by producing group projects. In addition, teachers' active engagement has proved to be a critical factor of a successful development course (Darling-Hammond, Hyler, & Gardner, 2017). Kong et al. (2020) proposed a pedagogy (i.e., to play, to think, and to code) to teach CT, which highly required the participants' engagement in class. They got ample opportunities to play the App at first. Then they had to think about how to produce the App. Finally, they needed to compose the programs by themselves. Teachers need active participation and engagement rather than passively receiving the knowledge (Darling-Hammond et al., 2017).

## 3. METHODOLOGY

### 3.1. Design and Structure of the Course

The design principle of this online TDC was to fulfil the important factors for successful teacher development including a sustained period of learning and teachers' active engagement. The TDC lasted for 5 weeks, and it provided both theory-based CT content knowledge and hands-on experiences of programming pedagogies. The course was conducted entirely online on Zoom. It consisted of thirteen 3-hour lessons and was divided into three parts.

The first part consisted of seven lessons. Six sample CT units were shown to demonstrate how to use App Inventor and pedagogical content knowledge (i.e., to play, to think, and to code) to teach CT. At the beginning of each lesson, the teachers played the apps. Then, they were guided to think about the components and logical flow of the apps. The teachers tried to deconstruct programming tasks into smaller steps so that they could learn how to turn abstract concepts into an algorithm to solve a problem. Finally, the teachers followed the student guides to code and test the app. If they were not able to finish coding the app in class, they could use the student guides, which demonstrated how to drag and drop the blocks step by step, to continue their learning after class. After teaching each unit, we had reflection on CT concepts, practices, and perspectives development with the teachers.

In the second part, the teachers had a lesson in which they observed an online CT lesson conducted by a local primary school. This lesson aimed to enhance their pedagogical understanding of how to deliver CT in relation to programming in a primary school context. During the observation, teachers could learn by reflecting on the successful and those less successful experience and incorporating the remarkable part into his/her own teaching. We consolidated this learning by holding a reflection section in the next lesson for the teachers to share their observation.

The third part of the course consisted of five lessons. The teachers worked in groups to produce a portfolio of artifacts, including a mobile app made in App Inventor for teaching primary school students, student worksheets, and a unit of CT teaching scheme with pedagogical design, to be presented in the last lesson of the course. They were given sufficient time to discuss their CT content knowledge and the pedagogical design of their teaching. This offered a chance for them to put their CT and pedagogical content knowledge into practice. Figure 1 shows the design and structure of this TDC.

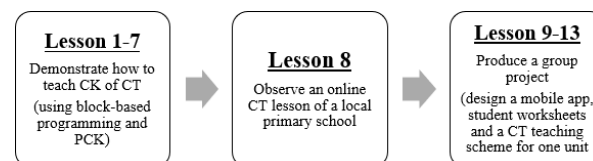


Figure 1. Design and Structure of the TDC

### 3.2. Participants and Procedures

The TDC was conducted in April and May 2020 with 39 lesson hours and 3 hours per lesson. This 5-week program was conducted entirely online on Zoom due to the pandemic. To ensure that the TDC maintain as high quality as face-to-face meeting, we had to provide support to teachers online when they encountered difficulties in handling the programming hands-on activities to increase their engagement. We slowed down the programming process to make sure that all teachers went through these tasks. We divided a programming task into several sub-tasks. We needed to ensure that they finished the sub-task before going to the next sub-task so that they could complete activities with successful experience. Also, we asked them to work in small groups with tutor supports when they were conducting the programming tasks and when testing and debugging the programs. The teachers needed to cooperate in completing the programming tasks when working in groups. The tutors would give immediate help to the teachers when all group members got stuck in the programming activities. In face-to-face meetings, we have one to two tutors in a classroom, but we assigned additional tutors for each group in this online course to provide support to the participants.

A total of 12 in-service primary school teachers attended the TDC. Eight (66%) were male and four (33%) were female. Most of the participants taught IT (66%) at their schools, whereas the others taught subjects such as mathematics, Chinese, and visual arts. Their average teaching experience was 15.8 years. To evaluate their learning progress, we asked them to complete (1) a CT concepts test and (2) a TPACK questionnaire concerning programming for CT development at the beginning of the course. They also completed two post-tests in the second last lesson. At the end of the course, participants were asked to completed the evaluation of teaching survey. All teachers participated in the tests and surveys online.

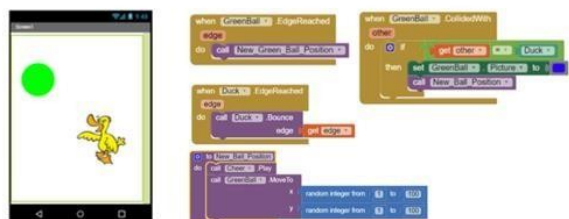
### 3.3. Measures

#### 3.3.1. CT Concepts Test

A multiple-choice CT concepts test was adopted to evaluate participants' learning progress (Kong et al., 2020). The pre- and post-tests were conducted in the first and

second last lessons. The test consisted of 25 items, which required teachers to analyze the outcome of executing the scripts in block-based programming environments. The test assessed progress in four categories of CT concepts, including repetition (3 items), conditionals (8 items), data (8 items), and procedures (6 items). Figure 2 shows a sample question about procedure.

Frank designed an app using App Inventor. Below are the Designer and Blocks for the app.



(a) What will happen when 'Green ball' touches the edges, using the above blocks?

- ☐ 'Green ball' will rebound
- ☐ 'Green ball' will change color
- ☐ 'Green ball' will be moved to another random place
- ☐ 'Green ball' will disappear
- ☐ 'Green ball' will reappear at the bottom left corner

(b) What will happen when 'Green ball' touches the 'Duck'?

- ☐ The cheer sound will be played
- ☐ The 'Duck' will change to blue color
- ☐ The 'Duck' will be moved to another random place
- ☐ 'Green ball' will rebound
- ☐ None of the above

Figure 2. A Sample Item for Assessing CT Concepts

### 3.3.2. TPACK of Programming for CT Development Questionnaire

Seven dimensions of TPACK were delineated by Mishra and Koehler (2006). Since the foundation of CT development was the content knowledge (CK) of programming for CT development, this study focused on evaluating the teachers' TPACK development in four CK-related dimensions (i.e. content knowledge [CK], technological content knowledge [TCK], pedagogical content knowledge [PCK], and technological pedagogical content knowledge [TPACK]). In the CT context, CK refers to the CK of CT concepts, practices, and perspectives in a programming context; TCK refers to knowledge of the use of programming features in a block-based programming environment to teach CT; PCK refers to the knowledge of teaching CT (e.g. unplugged activities) without using technologies; and TPACK refers to the knowledge of using technologies and pedagogies for teaching CT in relation to programming in a context of an exemplary use (Kong et al., 2020).

The questionnaire contained 29 items, which was modified from a TPACK instrument developed by Kong et al. (2020). The sample item of CK was "I have sufficient knowledge about programming." The sample item of PCK was "Without using technology, I can help my students to understand the content knowledge of programming through various ways." The sample item of TCK was "I can choose appropriate tools in App Inventor to teach students how to program." The sample item of TPACK was "I can teach lessons that appropriately combine the content of programming, technologies, and teaching approaches." Each item of this instrument was anchored from 1 (strongly disagree) to 5 (strongly agree).

## 4. RESULTS AND DISCUSSION

### 4.1. Results of the CT Concepts Test

Significant improvement was found in the results of the CT concepts test as a whole ( $t(11)=3.36$ ,  $p<.01$ ) after the completion of the course, as shown in Table 1. Among the individual concepts, the teachers showed the greatest improvement in their understanding of the concept of procedures by the end of the course ( $t(11)=4.71$ ,  $p<.01$ ).

Table 1. Pre- and Post-test Results of CT Concepts Test.

	Pre-test		Post-test		t-value
	Mean	SD	Mean	SD	
Repetition	91.67	15.08	94.44	12.97	.43
Conditionals	80.21	24.11	90.63	13.19	1.97
Data	69.79	24.11	80.21	12.45	1.45
Procedures	56.94	28.83	86.11	18.58	4.71**
Total (25 items)	72.67	19.36	86.67	8.58	3.36**

Note. \*\*  $p<.01$

### 4.2. Results of the TPACK of Programming Questionnaire

Significant improvements were found across all CK-related dimensions in TPACK after the completion of the course, as shown in Table 2. Of these, teachers' TCK showed the greatest improvement, which indicated that they had gained much more confidence in using the tools in the App Inventor block-based programming environments to prospectively develop their students' CT.

Table 2. Pre- and Post-test Results of the TPACK of Programming Questionnaire.

	Pre-test		Post-test		t-value
	Mean <sup>a</sup>	SD	Mean <sup>a</sup>	SD	
CK	3.02	.81	4.04	.57	4.62**
PCK	3.19	.73	3.98	.41	3.93**
TCK	2.21	1.16	4.08	.60	5.65***
TPACK	3.00	1.02	4.00	.47	4.38**

Note. \*\*\*  $p<.001$ , \*\*  $p<.01$ . <sup>a</sup>1 = Strongly Disagree; 2 = Disagree; 3 = Neutral; 4 = Agree; 5 = Strongly Agree.

### 4.3. Results of the Participants' Evaluation of Teaching Survey

At the end of the course, all 12 participants responded to the evaluation of teaching survey. Table 3 shows that all teachers agreed that they experienced the thinking process during the lessons and they considered that the teaching was of high quality.

Table 3. Participants' Evaluation of Teaching Survey Results

Items	Mean	SD
1. Delivering the course in an organized way.	3.75	.62
2. Inspiring students to think and learn.	3.83	.39
3. The overall teaching was of high quality.	3.75	.45

Note. 1 = Strongly Disagree; 2 = Disagree; 3 = Agree; 4 = Strongly Agree.

Table 4 shows that the most useful aspect of this course was understanding the importance of "to think" rather than rushing to code in CT learning. Tutor support was also essential in online learning. Two teachers appreciated that the teaching team installed all the apps to the tablet and allowed them to borrow and bring home before the course. They suggested that it was a good practice to upload the learning materials to the online platform so that they could

prepare prior to the lesson. They appreciated using these apps to play around and think along with the teacher and learning with their peers before starting to code in class.

Table 4. Participants' Responses of the Evaluation of Teaching Survey

Responses	Number of teachers
1. Understand the importance of the thinking process of CT	4
2. Support of the whole teaching team	3
3. Well preparation and arrangement before the class	2

## 5. IMPLICATIONS AND CONCLUSION

This study reports a successful experience of conducting an online TDC on CT teaching. The results showed that teachers' CT concepts, and TPACK of programming for CT development improved considerably after the course. The teachers' positive experience and responses provided two practical implications for conducting TDCs on CT in the future.

First, the design of the course required a sustained period of learning with the provision of theory-based CT CK and hands-on practice of programming pedagogies. It was important to offer sufficient examples to illustrate the teaching of CT with a focus on CT concepts, practices, and perspectives, as teachers needed to nurture their students' CT in these three dimensions. Second, we need to enhance the teachers' engagement in online course by adjusting the teaching pace and assigning sufficient tutors to provide support (Bao, 2020). In online teaching, we suggest adjusting the teaching pace to facilitate teachers' successful programming experience rather than rushing to complete all tasks designed in the unit. Since more lesson time is used to finish the programming tasks, sometimes we need to give up the more difficult part of the tasks. We also suggest breaking down the programming tasks into sufficient sub-tasks to make sure all participants can follow and complete the sub-tasks one by one. Apart from adjusting the teaching pace and content, allocating sufficient tutors to render timely support to the online class is essential to enhance the participants' engagement. While the instructors may not remotely view each participant's progress during the online lesson, the tutors can provide immediate suggestion to participants in small groups during the programming activities, in particular testing and debugging. The greatest limitation of this study was its small sample size. We shall conduct similar evaluations in scenarios with more teachers to investigate the effectiveness of TDCs in online teaching mode.

## 6. REFERENCES

- Bao, W. (2020). COVID-19 and online teaching in higher education: A case study of Peking University. *Human Behavior and Emerging Technologies*, 2(2), 113-115.
- Bower, M., & Falkner, K. (2015). Computational thinking, the notional machine, pre-service teachers, and research opportunities. In *Proceedings of the 17th Australasian Computing Education Conference* (pp. 37-46). Sydney: Australian Computer Society.
- Bower, M., Wood, L. N., Lai, J. W. M., Howe, C., Lister, R., Mason, R. ... Veal, J. (2017). Improving the computational thinking pedagogical capabilities of school teachers. *Australian Journal of Teacher Education*, 42(3), 53-72.
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association* (pp.1-25). Vancouver, Canada.
- Darling-Hammond, L., Hyler, M. E., & Gardner, M. (2017). *Effective teacher professional development*. Palo Alto, CA: Learning Policy Institute.
- Garet, M. S., Porter, A. C., Desimone, L., Birman, B. F., & Yoon, K. S. (2001). What makes professional development effective? Results from a national sample of teachers. *American Educational Research Journal*, 38(4), 915-945.
- Hsu, T., Chang, S., & Hung, Y. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, 126, 296-310.
- Hubbard, A. (2018). Pedagogical content knowledge in computing education: A review of the research literature. *Computer Science Education* 28(2), 117-135
- Kong, S. C., & Abelson, H. (Eds.) (2019). *Computational Thinking Education*. Singapore: SpringerOpen.
- Kong, S. C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices and pedagogy. *Computers & Education*, 151, 103872.
- Lye, S., & Koh, J. H. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, 41, 51-61.
- Menekse, M. (2015). Computer science teacher professional development in the United States: A review of studies published between 2004 and 2014. *Computer Science Education*, 25(4), 325-350.
- Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record*, 108(6), 1017-1054.
- Weintrop, D., & Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Transactions on Computing Education*, 18(1), 1-25.
- Wing, J. M. (2006). Computational thinking. *Communication of the ACM*, 49(3), 33-35.
- Yadav, A., Gretter, S., Hambrusch, S. & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education*, 26(4), 235-254.

# **Computational Thinking and STEM/STEAM Education**

# ARTEC Logic Puzzle: The Role of Computational Thinking with Extension to Extended Logic

Chung-Oi KOK  
WOW Educational International, Singapore  
cokuan@googlemail.com

## ABSTRACT

STEM education requires learners to utilise computational thinking process to solve complex problems. One such STEM activity is Logic Puzzle that was developed by Artec, a Japanese Research and Development education Company that established STEM education products such as robotics and logic puzzles for preschool children in Japan. Logic Puzzles are learning tools for children to strengthen numeracy skills as children learn to solve puzzle problems that relate to spatial relations of direction, position and distance, processes of ordering and patterning, matching, sorting, and comparing, counting, and applying simple measurements. These processes employ computational thinking which associates with four thinking steps, namely, decomposition, pattern recognition, abstraction, and algorithm. However, while applying computational thinking processes to solve problems in logic puzzle activity, extended logic a thinking process that enables the mind to have multiple interpretations is also applied. This paper discusses how STEM: logic puzzle activity enables learners to apply computational thinking with the four thinking steps along with extended logic to solve problems in logic puzzle.

## KEYWORDS

Artec logic puzzle, computational thinking, extended logic

## 1. INTRODUCTION

Logic Puzzle is related to STEM. Logic Puzzle consists of 12 different themes across 48 lessons. The author shall draw on her own classroom experience of using the theme on balance game to elaborate how children learned to solve logic puzzle problems that relate to STEM. Children learn about law of physics concerning distribution of weight to balance 2 numbers of 3-dimensional (D) blocks. Children are required to count the numbers of blocks to assemble 3-D blocks based on 2-D pictures in the logic puzzle workbook. Children apply engineering skillsets to balance the blocks on top of each other without falling off. Consequently, technology is included as science and mathematics are involved to balance the 3-D blocks. The process of solving logic puzzle problems enables children to apply computational thinking in 4 thinking steps, decomposition, pattern recognition, abstraction, and algorithm (Wing, 2008, 2011), (Selby, Cynthia & Woollard, John, 2013), (German, 2019), (Charoula et al. 2016). The following sections discuss how learners apply the 4 thinking steps along with extended logic to solve logic puzzle 25 on balance game.

## 2. COMPUTATIONAL THINKING, EXTENDED LOGIC AND LOGIC PUZZLE 25



Figure 1. Logic Puzzle Box



Figure 2. Logic Puzzle Workbook, Balance Game, Challenge 1.2

The logic Puzzle box contains different types of Artec blocks and parts for learners to solve logic puzzle problems in the logic puzzle workbook.

### 2.1 Algorithm

Algorithm is primarily a guide or manual or set of instructions to work on a particular piece of work. (<https://techterms.com/definition/algorithm>). Learners have to follow the balance game instructions by building 1 set of green and yellow 3-D blocks from the given 2-D pictures in the workbook of challenge 1.2. Next, they have to position the green 3-D block on the given green rectangle and balance the 3-D yellow block on the latter without falling off. This also applies to the 3-D yellow block on the yellow rectangle.

### 2.2 Abstraction

The 3 phases of abstraction are singling out object from a situation, symbolising singled out object as a concept and arranging the singled-out object to connect to a system (Winter, 2014). The first phase is for learners to single out the relevant coloured Artec blocks to build 1 set of green and yellow 3-D blocks from the puzzle box. The second phase is to learn from the instructor concerning the purpose of putting 2 numbers of 3-D blocks on top of one another to symbolise the concept of balancing. The third phase is for learners to arrange 2 numbers of 3-D blocks to be put on top of one another to fulfill the concept of balance based on the rule or instructions of logic puzzle 25.



## 2.3 Decomposition

The process of first phase of abstraction relates to decomposition in breaking down the matter (Donze & Wong, 2018) and segregation (Yamaguchi, 2017). Learners are required to break down the 2-D pictures from the workbook to single out the quantity and types of Artec blocks by segregating the required ones from the puzzle box to build the 3-D blocks.

## 2.4 Pattern Recognition

The third phase of abstraction relates to pattern recognition as learners have to find and recognise a pattern or sequence from 2 unassociated items (Ripley & Taylor, 1987), (Baron, 2006). Learners have to find a sequence to put two different 3-D blocks on top of one another without falling off to fulfil the concept of balancing, which relates to the 3<sup>rd</sup> phase of abstract as discussed in 2.2.

## 2.5 Extended Logic

Recognising the pattern of balancing both 3-D blocks in a vertical position, learners also apply extended logic thinking process that enables them to interpret in multiple perspectives (Wiseman, 2004) and (Kok, 2011). Learners also realised that there were more than 1 solutions to solve puzzle 25. Learners are motivated to think beyond one solution to solve the problem. In other words, the puzzle activity enables learners to extend beyond 1 solution to another, thus offering learners to be flexible in thinking too. The entire process to solve Logic Puzzle in Balancing Game fosters computational thinking development.

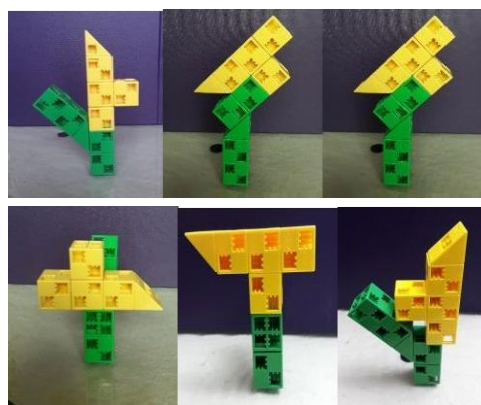


Figure 3.

## 3. CONCLUSION

The discussion shows that logic puzzle enables learners to apply 4 thinking domains of algorithm, abstraction, decomposition, pattern recognition and extended logic. Each thinking domain is applied interconnectedly to solve puzzle problem. For example, in the first phase of abstraction, learners have to break down the 2-D pictures to single out the relevant Artec blocks and parts to form 3-D blocks, thus relating to decomposition. The third phase of abstraction relates to the process of finding a pattern or sequence to put both 3-D blocks on top of one another to fulfil the concept of balancing. Learners also realise that there is more than 1 solution to solve the puzzle problem because they are able to “stretch” or “extend” the answer into multiple interpretations that relates to application of extended logic. Therefore, logic puzzle designed by Artec company, a STEM activity enables learners to apply

computational thinking skills along with extended logic to solve problems.

## 4. REFERENCE

- Baron, R. (2006). *Opportunity Recognition as Pattern Recognition: How Entrepreneurs "Connect the Dots" to Identify New Business Opportunities*. Retrieved December 17, 2020, from <http://www.jstor.org/stable/4166221>
- C, O. Kok. (2011). *The Rationale for Visual Arts Education in Singapore: Analysis of Policies and Opinions*, Durham University, UK. Retrieved December 17, 2020, from <http://etheses.dur.ac.uk/845/>
- Donze, J., & Wong, S. (2018). *Where did the leaves go? Investigating decomposition through an inquiry-based project*. Science Scope. Retrieved November 15, 2020, from <http://doi:10.2307/26611842>
- German, S. (2019). *Computational thinking*. Science Scope, 42(9), 36-39. Retrieved March 3, 2021 from <http://doi:10.2307/26899029>
- Charoula, A., Joke, V., Andrew, F., Mary Webb, M, Cox, Joyce, Smith, & Jason, Z. (2016). A K-6 Computational Thinking Curriculum Framework: Implications for Teacher Knowledge. *Journal of Educational Technology & Society*, 19(3), 47-57. Retrieved March 8, 2021, from <http://www.jstor.org/stable/jeductechsoci.19.3.47>
- Ripley, B., & Taylor, C. (1987). *Pattern Recognition*. Retrieved November 22, 2020, from <http://www.jstor.org/stable/43420690>
- Selby, Cynthia & Woollard, John (2013) *Computational thinking: the developing definition*. University of Southampton (E-prints) 6pp. Retrieved March 3, 2021 <https://eprints.soton.ac.uk/356481/>
- Winter, R. (2014). *James and Dewey on Abstraction. The Pluralist*. Retrieved November 22, 2020 <https://doi:10.5406/pluralist.9.2.0001>
- Wing, J. (2008). *Computational thinking and Thinking about computing*. Philosophical transactions: Mathematical, physical and Engineering sciences, 366(1881), 3717-3725. Retrieved November 13, 2020, from <http://www.jstor.org/stable/25197357>
- Wing, J. (2011). *Research notebook: Computational Thinking--What and Why?* Retrieved March 3, 2021, from <https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why>
- Wiseman, B. (2008). *Qualia thinking the senses*. Retrieved September 21, 2010, from <https://www.swetswise.com.ezphost.dur.ac.uk/eAccess/vjewToc.do?titleID=425276&yevoID=2348073>
- Yamaguchi, K. (2017). *Decomposition Analysis of Segregation*. Retrieved November 22, 2020, from <http://www.jstor.org/stable/26429068>



# **Computational Thinking and Data Science**

# Infusing Computational Thinking into the Accounting Practice Course

Tao WU<sup>1\*</sup>, Maiga CHANG<sup>2</sup>

<sup>1</sup> Zhujiang college of South China Agricultural University, Guangzhou, China

<sup>2</sup> School of Computing and Information Systems, Athabasca University, Alberta, Canada  
scauzjcwutao@gmail.com, maiga.chang@gmail.com

## ABSTRACT

In the digital age, the demand for digital talents in our commercial society has greatly increased. Digital talents mainly refer to the general names of professionals in various industries who can perform data analysis and forecast trends on the basis of the building data models. Taking Accounting Practice course as an example, this paper expounds the teaching model of integrating computational thinking into non-stem subjects. The main strategy adopted in this paper is to design different practical tasks according to three teaching difficulties. Based on the concept of constructivism, students can use different tools at different stages and find an efficient problem-solving model

## KEYWORDS

Computational Thinking (CT), Constructivism, Accounting, curriculum design, framework

## 1. INTRODUCTION

In the era of data, enterprises are eager to make decisions, arrange inventory, advertise and deliver related consumer products by collecting and using data. Therefore, the demand for undergraduates with data analysis skills is increasing rapidly. To meet the needs of the business community, the Association of Advanced Business Schools (AACSB) takes data analysis as an essential skill into accounting practice and theory courses, and they have developed the A7 certification standard with independent AACSB certification. Accounting is a major that uses data analysis most in business disciplines, and undergraduates need to obtain more training in data analysis skills. However, it is not easy to liberate students from the complicated regulations and become masters of digital resources.

Wing (2006, 2008) defined computational thinking as a general thinking to solve problems, which was developed by others (National Research Council 2010). The accounting courses aim to develop students' skills and enable them to understand how to use data to formulate and solve business problems. The injection of computational thinking provides accounting professionals with the opportunity to use technology to analyze data and solve the data-analysis problems.

## 2. COMPUTATIONAL THINKING FRAMEWORK OF FINANCIAL ACCOUNTING

In this paper, our goal is to provide the CT in a practical framework and procedures for implementing computational thinking in accounting majors.

Based on the characteristics of accounting and the two dimensions of computational thinking, the research team proposes a theoretical framework for integrating computational thinking into accounting courses, as shown in Figure 1.

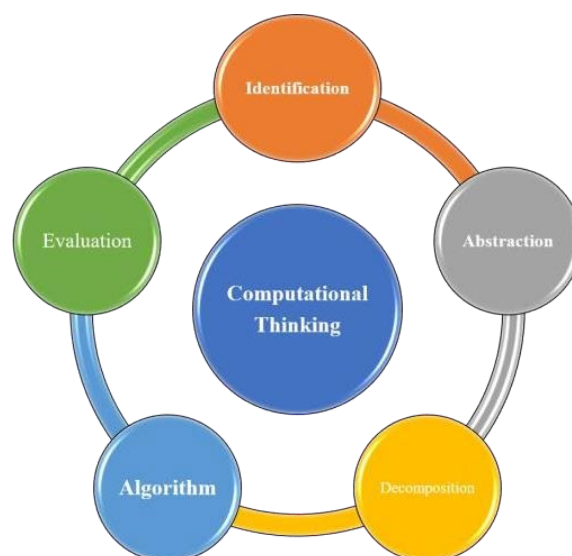


Figure 1. CT framework for integrating computational thinking with Accounting work

The proposed framework in accounting course has five components: Identification, Abstraction, Decomposition, Algorithm, as well as Evaluation (see Figure 1).

## 3. EXAMPLE OF CT IN ACCOUNTING COURSE

In this section, we provide an example from our framework. Here, we have carefully selected the professional course “financial statement analysis”, which has the closest relationship with data analysis skills in the accounting major as an example.

At first, instructor divides courses into three levels (see Table 1 below) according to the difficulty of using tools and course contents.

Table 1. CT classify learning Content in three difficult level

	Basic (Easy)	Mastery(Medium)	Advance(Difficult)
Identification	Identify subject, data, formula expression	Identify various indicators and can infer the relationship between each other	Identify the correct data in a fuzzy data set
Thinking process	Single-step reasoning	Multi-step reasoning	Critical thinking and multi-level reasoning
Tool (software)	Bookkeeping software	Mind-map; Excel; Tableau	Excel; Open-source software
Algorithm	Basic formula	Weighted index processing	Complex Model

We take the chapter "Application of DuPont Analysis" as an example to briefly summarize the CT application project in Zhujiang College of South China Agricultural University.

Chapter: Applications of DuPont Analysis

Objective: Based on the concept of constructivism, using the model of computational thinking to solidify students' problem-solving path, to understand and master related concepts.

Difficult level: Medium

Methods: Group cooperation/individual completion of case analysis

Tools: Mind -manager; Excel (software)

Task: Provide complete financial data for five years and incomplete data for the sixth year of an enterprise. After mastering the index decomposition of DuPont analysis, students are required to predict the ROE index of the sixth year with 5- year data.

Assessment: The instructor rates their answers based on the criteria listed in Table 2.

*Table 2.* Evaluation Norms based on the elements of computational thinking

Norm	A+	A	B	C
Accuracy	Completely correct	Completely correct	Partially correct	Few correct
Abstract	The formula expression completely corrects.	The formula expression completely corrects.	Can't complete all formulas	Can't understand all formulas
Algorithm	Build Model and verify right	No Model, Calculate right	Flaws in the calculation process	Flaws in multi-step calculations, but simple calculations OK

## 4. DISCUSSION AND CONCLUSION

Our aim is to integrate computational thinking into practical courses of accounting major by providing a thinking framework. Applying computational thinking to practical courses and course evaluation through instructional design can encourage students to master the ability of using technical tools to solve practical problems, and enable students to have a thinking path to solve problems. The essence of the problem can only be discovered in the plight of nowhere to go. Through reasoning the characteristics of the problems in the thinking process, students finally mastered the technical tools and solved the problems. Several elements of the framework require teachers to set the difficulty levels according to task content and students' technical ability in curriculum design.

On the other hand, adding the 3A element of computational thinking to the grading index of students' homework will help to cultivate students' skills and application of learning CT in these three aspects. There are still many issues to be explored in CT application teaching of non-STEM disciplines, such as different students' preferences in the use of technology tools, and how to reconcile the differences in learning time when different students master the use of tools.

## 5. REFERENCES

- Alles, M. G. (2015). Drivers of the use and facilitators and obstacles of the evolution of big data by the audit profession. *Accounting Horizons*, 29(2), 439-449
- American Institute of Certified Public Accountants (AICPA) (2018). AICPA pre-certification core competency framework. Available: <https://www.aicpa.org/interestareas/accountingeducation/resources/corecompetency.html>
- AACSB International (2013). Eligibility procedures and accreditation standards for accounting accreditation. Available: <http://www.aacsb.edu/accreditation/standards/2013-accounting>
- National Research Council (2011). Report of a Workshop of Pedagogical Aspects of Computational Thinking. Washington, DC: The National Academies Press. Available: <https://doi.org/10.17226/13170>.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35. <https://doi.org/10.1145/1118178.1118215>
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal*.

# VizBlocks: A Data Visualization Literacy Education Tool

Travis Jia Yea CHING<sup>1\*</sup>, Bimlesh WADHWA<sup>2\*</sup>

<sup>1,2</sup>National University of Singapore, Singapore  
travisching007@gmail.com, bimlesh@nus.edu.sg

## ABSTRACT

In the conversation of computational thinking as a vital ingredient of STEM, the role of data literacy education has been overlooked. Data literacy is fundamental to computational thinking, yet research on tools for data literacy is still in its infancy. This paper explores a way to promote data literacy education through a new platform called VizBlocks. It proposes that having an information repository of data literacy resources complemented by a visual programming tool, will enable K-12 children to creatively learn data visualization.

## KEYWORDS

Data Visualization Literacy, Visualization in Education, Visualization with Children, Visual Programming Paradigm

## 1. INTRODUCTION

In recent years, the argument for adding computational thinking (CT) to every child's analytical ability as a vital ingredient of STEM learning sparked by Jeannette Wing has rallied educators, education researchers, and policy makers. An examination of the current state of discourse on computational thinking in K-12 education shows that with broadly agreed on definitions of CT in K-12 education, focus has been shifted to investigating ways to promote and assess the development of CT (Grover, & Pea, 2013). In the U.S., the AI4K12 Initiative is a developing guideline on artificial intelligence (AI) education for K-12 students. In Touretzky and Gardner-McCune's recent work, they explore the key insights that K-12 students can gain into the big ideas of AI, and how the learning of AI may influence other aspects of their educational experience (Touretzky & Gardner-McCune, 2022).

Despite the conversation on promoting CT education broadening into the sub-branches of computer science, conversation on how data literacy education plays an important role in promoting CT has been overlooked. One of the core CT skills is representing data through models. This however cannot be achieved without a firm grasp of data literacy. An examination gauging the ability to interpret data visualizations indicates that the public has a low level of data literacy (Börner et al., 2015). This reflects that most people are unable to effectively comprehend valuable information using data visualizations which helps in learning, problem solving and making informed decisions. Education pertaining to data literacy is thus essential to be conducted in conjunction with CT education.

## 2. VISUAL PROGRAMMING PARADIGM

The visual programming paradigm which encodes source code as graphical elements lends very well for encoding key components (e.g., axis and labels) and data points that construct a data visualization as graphical elements. These

graphical elements function like Lego blocks, allowing users to write a multitude of programs or construct various types of visualizations, simply by arranging them in a way that "fits". This bottom-up approach deconstructs the thought process behind the concept being taught. It not only greatly reduces the learning curve, but it also reinforces knowledge on the individual parts that make up a program or visualization in this case.

In addition, Scratch by MIT's success in adopting the visual programming paradigm reveals an approach to educating data visualization literacy that addresses the areas of improvement (AOI) found in existing tools:

Table 1. Advantages of visual programming paradigm

Advantages	Existing Tool's AOI
A more powerful form of free-flow visualization	Construct-A-Vis (Bishop et al., 2019).
Efficient and accessible collaborative learning feature	Construct-A-Vis, C'est La Vis (Alper et al., 2017).
Encourage critical analysis of any variety of visualizations	C'est La Vis, Diagram Safari (Gäbler et al., 2019).

## 3. VIZBLOCKS

### 3.1. Objectives

The core objectives of VizBlocks are:

1. Build a tool based on a visual block-based paradigm to enable K-12 children to learn data visualization literacy creatively
2. Build an information repository of data visualization literacy resources

To allow children to learn data visualization literacy skills creatively, they can use a visual programming extension of Scratch called Vizblocks. By extending the successful model of Scratch, the goal is to allow children to creatively learn data visualization literacy skills whilst strongly enforcing knowledge on the individual parts that make up a data visualization.

The information repository would serve as a one-stop platform for elementary school teachers and students to access materials used for teaching and learning of data visualization literacy, access the Vizblocks tool as well as contribute to the resources. This in turn, builds a community of shared learning.

### 3.2. Vizblocks Tool

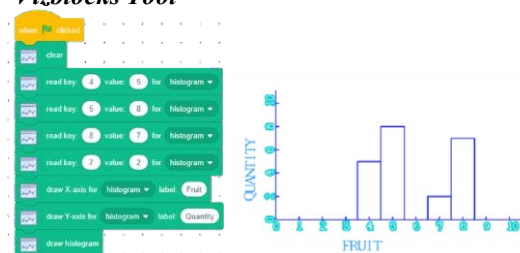


Figure 1. Vizblocks extension blocks to draw histogram.

Vizblocks currently supports the creation of 8 types of visualizations:

1. Dot Plot
2. Pictograph
3. Bar Chart
4. Pie Chart
5. Histogram
6. Line Chart
7. Scatter Plot
8. Heatmap

The choice for these 8 types of data visualizations was made by studying Pre-K-12 Guidelines for Assessment and Instruction in Statistics Education II (Bargagliotti et al., 2020).

Vizblocks is built with little assumption of children's prior knowledge of Scratch. Most data visualizations can be built in a drag and drop manner without programming knowledge. However, since Vizblocks is built on Scratch, children can make use of existing Scratch blocks to read in data programmatically instead of using multiple similar blocks for the same purpose. An added benefit of learning with Vizblocks is that children might be keen to explore computational thinking to ease visualization creation.

### 3.3. Vizblocks Information Repository

The information repository alongside the VizBlocks extension has been built and is currently deployed at <https://vizblocks.comp.nus.edu.sg>.

Users can access the Vizblocks tool through the website, by simply clicking the “new project” button or on existing projects. They can create, read, update, and delete projects on the cloud.

The Vizblocks website also supports a “Studio” feature. From an educator's point of view, a studio functions as a classroom where folders can be organized as submission boxes. It is also a place where teachers and students can communicate; From a student's point of view, a studio can be a collection of similar projects, serving to organize projects for ease of access. It can also be a place for like-minded students to gather and learn from each other.

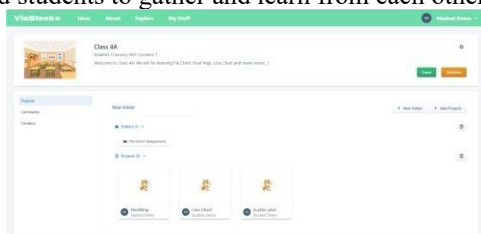


Figure 2. Studio on the VizBlocks website.

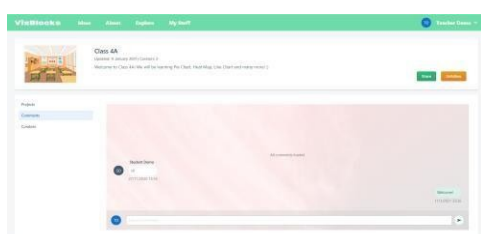


Figure 3. Users communication in Studio.

In addition, educators can download lesson plans with detailed step-by-step guides on the website. There is also an assessment test functionality to help educators gauge their students' performance.



Figure 4. Pie Chart Pre-Assessment Test on VizBlocks

## 4. CONCLUSION

For K-12 children and educators who need to receive or give education on data visualization literacy, VizBlocks is both an information repository and visual programming tool that allows creative learning of data visualization literacy through a visual block-based paradigm, easy access to relevant materials and a community of shared learning. Unlike existing tools such as C'est La Vis, Construct-A-Vis and Diagram Safari, VizBlocks is a more powerful free-form visualization tool. Its bottom-up approach not only has a low barrier of entry but also reinforces knowledge on the core concepts of visualizations thereby equipping children with the skill to critically analyze any variety of visualizations. Additionally, it has an extensive support for collaborative learning that is not constrained by physical proximity and additional hardware.

## 5. REFERENCES

- Alper, B., Riche, N. H., Chevalier, F., Boy, J., & Sezgin, M. (2017). *Visualization Literacy at Elementary School*. *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. doi:10.1145/3025453.3025877
- Bishop, F., Zagermann, J., Pfeil, U., Sanderson, G., Reiterer, H., & Hinrichs, U. (2019). *Construct-A-Vis: Exploring the Free-Form Visualization Processes of Children*. *IEEE Transactions on Visualization and Computer Graphics*, 1-1. doi:10.1109/tvcg.2019.2934804
- Börner, K., Maltese, A., Balliet, R. N., & Heimlich, J. (2015). *Investigating aspects of data visualization literacy using 20 information visualizations and 273 science museum visitors*. *Information Visualization*, 15(3), 198-213. doi:10.1177/1473871615594652
- Gäbler, J., Winkler, C., Lengyel, N., Aigner, W., Stoiber, C., Wallner, G., & Kriglstein, S. (2019). *Diagram Safari: A Visualization Literacy Game for Young Children*. *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts - CHI PLAY '19 Extended Abstracts*. doi:10.1145/3341215.3356283
- Grover, Shuchi & Pea, Roy. (2013). *Computational Thinking in K-12 A Review of the State of the Field*. *Educational Researcher*. 42. 38-43. 10.3102/0013189X12463051.
- Touretzky, D. S., & Gardner-McCune, C. (2022). *Chapter IX: Artificial Intelligence Thinking in K-12*.

# **Computational Thinking Development in Higher Education**



# Making the Thinking Results of Programming Visible and Traceable with a Multi-layer Board Game

YungYu ZHUANG<sup>1\*</sup>, Andito SAPUTRO<sup>2</sup>, Mahesh LIYANAWATTA<sup>3</sup>, Jen-Hang WANG<sup>4</sup>, Su-Hang YANG<sup>5</sup>,  
Gwo-Dong CHEN<sup>6</sup>

<sup>1,2,3,4,6</sup>National Central University, Taiwan

<sup>5</sup>Chien Hsin University of Science and Technology, Taiwan

yungyu@ncu.edu.tw, saputroandito180314@gmail.com, mahesh.saitm@gmail.com, harry@cl.ncu.edu.tw,  
yoko@uch.edu.tw, gwodong@gmail.com

## ABSTRACT

Learning programming is never easy since not only the knowledge but also the strategies to use the knowledge are necessary for programming. Although board games have been recognized as a promising approach to teaching computational thinking and programming, they are usually limited to turn-based design and lack the training of thinking a plan. On the other hand, learning with mini- languages and visual programming basically needs the use of computers and thus requires the ability to operate computers. We implemented the stored program concept and combined it with the idea of making thinking visible in a multi-layer board game to help to learn programming. Learners' thinking results of programming can be reflected on this new kind of board game and synchronized with problems along with solutions. We conducted an experiment on the learning performance improvement by comparing it with a well-designed board game for learning computational thinking, and the results showed the effectiveness of using such a multi-layer board game.

## KEYWORDS

programming, computational thinking, board games, make thinking visible, teaching and learning strategies

## 1. INTRODUCTION

Programming skill might not be an ability that is directly related to one's professional, but it is generally agreed that it is useful in everyone's career. Over the past decades, programming is moving into many of the domains previously dominated by writing (Vee, 2013). Without programming, as Soloway (1993) claimed, we will have cut off half the power of computational medium. Programming is further conceptualized as computational thinking (CT), which refers to a universally applicable attitude and skill set for everyone (Wing, 2006). The study conducted by Hsu et al. (2018) showed that CT had gained the attention of scholars and educators, and the subject of programming constitutes the biggest proportion of CT research papers.

Programming is the ability to make digital technology do whatever, and some call this skill human-machine interaction (Prensky, 2008). A recent study conducted by Siegmund et al. (2020) even showed an interesting result that a clear left-lateral activation during program comprehension. Programming empowerment was defined as a person's perceived autonomy and competence to use CT effectively (Kong et al., 2018). Papert (1972) also

argued that providing children with access to computers can give them the power to invent. Interacting with digital media is the ability to read while being able to create our own games, animations, or simulations is the ability to write (Resnick et al., 2009). However, learning programming is never easy since not only knowledge but also the way knowledge is used or applied, i.e., strategies, are necessary for programming (Davies, 1993). Robin et al. (2003) recommended focusing on the combination and use of new language features besides the learning of those features, and the result of a survey conducted by Lahtinen et al. (2005) also supports this argument. The variability in program design shows the interaction with programmers' knowledge (Rist, 1990); this makes programming hard to learn and master. D tienne and Soloway (1990) identified different strategies involved in program understanding. Brooks (1983) noted at least three distinct sources of differences in the ability of program comprehension: programming knowledge, domain knowledge, and comprehension strategies.

The approaches to teaching novices programming include board games and educational programming environments. When we reviewed these approaches, we made three observations. First, invisible programming thinking makes it difficult to learn. Programming is a process of thinking abstraction and composition, and such thinking is invisible. Second, directly learning with computers is difficult to novices, even though many mini-languages and visual environments are given. We argue that it is difficult to trace the execution of programs and, to human brains, running programs on computers is too fast! Novice programmers need to slow down the execution and see the states. Third, the lack of teaching how to think a whole plan in existing programming board games. Programming is thinking a whole plan to deal with all situations in advance rather than making every decision individually. The devised plan actually results in a stored program that can be loaded and modified for executing again; it is the basics of computers. These observations led us to develop a board game that can visualize learners' thinking results of programming on top of problems.

## 2. RELATED WORK

### 2.1. Board games for CT and programming

Board games are cost-effective instructional materials that can be integrated into a set of game-based learning strategies (Santos, 2019), and the game was one of the main pedagogies in CT research (Tang et al., 2020). Many



board games have been developed to improve students' CT and programming achievement with or without the help of technology. Kuo and Hsu (2020) utilized a board game named Robot City in their study and proved that the game could deepen students' higher-level thinking and motivate students to learn. Wu et al. (2018) designed a CT board game, namely Interstellar Explorer, to develop players' logical thinking, problem-solving ability, imagination, and creativity. Yen & Liao (2019) showed that the use of board games as teaching material for programming courses could significantly improve the learning outcomes of field-independent learners. These research results show that board games are a promising approach to cultivate learners' knowledge of CT and programming. To teachers, the cost-effective feature makes it easy and simple to adopt board games in classrooms. To learners, board games are realistic and tangible. If computer concepts can be properly transformed into the rules and elements of board games, learners may get the computing ideas without computers.

## 2.2. Mini-languages and visual programming

Mini-languages and visual programming are other approaches to teaching CT and programming. For the purpose of education, giving learners a small set of language elements, natural-language-like syntax, or visual interface for writing code can avoid the difficulty in learning a practical programming language in the industry. LOGO and the use of Turtle graphics is a famous example (Papert, 1980). With a small syntax and simple semantics, even a very young one can have a grip on programming (Brusilovsky et al., 1997). Programming is also regarded as a direct approach to foster CT (Lye & Koh, 2014). There are also many visual programming environments designed for education, including Scratch, App Inventor, and Alice. This approach greatly lowers the threshold of learning programming. Chang (2014) demonstrated the effects of using the two visual programming environments and explored the relationships among learning engagement, learning anxiety, and learning playfulness. However, this approach directly relies on computers, and they might be the next learning materials after learners got the idea through board games.

## 2.3. Make thinking visible

Making thinking visible is to have a window into learners' thinking by some sort of organizing structure such as thinking routines (Perkins, 2003; Tishman & Palmer, 2005; Ritchhart et al., 2011). It is developed by Project Zero, an educational research group at Harvard University. Making thinking visible is to know what students understand and how they are understanding. One of the ways to make thinking visible is to surface the many opportunities for thinking during subject matter learning, and thinking routines are helpful tools in the process. Each thinking routine is made up of a series of steps helping learners to think. By operating such a pattern, we can scaffold learners' thinking and make that thinking visible.

Programming can be made visible if we properly design a thinking module and thinking routines for it. Like other thinking, programming thinking is also pretty much

invisible. Our observation is that invisible thinking is the reason why programming is difficult for novices. The thinking happens under the hood, within our mind-brain, and this makes it difficult for experts to teach novices. It is more difficult when we write and run programs on computers since computers run so fast and learners can only see the results and trace them on code. We based our research on the idea of making thinking visible to concretely design the system along with thinking routines.

## 2.4. Stored program and problem solving

The ability to store programs in computers makes it possible to not only execute but also modify programs (Aspray, 1990). The stored program concept is based on the universal Turing machine and included in the von Neumann architecture, which is employed by almost every computer in the past 70 years.

The process of programming can be regarded as thinking a whole plan based on the stored program concept. The plan is to deal with all situations in advance rather than making every decision individually, and we need to revise programs again and again. Shneiderman (1980) mentioned the importance of planning in computer programming based on the four stages in problem-solving given by Polya (1957): understanding the problem, devising a plan, carrying out the plan, and looking back. Bishop-Clark (1992) examined the problem solving of a novice programmer writing a first draft program and suggested instructors should consider emphasizing the planning stage. Unfortunately, so far as we know, existing board games for teaching CT and programming encourage learners to think every small step since these existing board games are basically turn-based. Awareness of runtime is very important as well. Some conditions cannot be determined until we really execute programs. The usage of branches (if-else) is meaningful only if we don't know what the conditions exactly are.

# 3. OUR METHODOLOGY AND THE MULTI-LAYER BOARD GAME DESIGN

In order to help novices learn programming, we developed a thinking module to make the thinking results of programming visible and traceable and designed a board game with this thinking module.

Since programming is a process of thinking, it is quite invisible. When a programming problem is given, programmers need to think about how to use their knowledge with strategies to write their solution. The problem is naturally given in the form of words, i.e., a description of the problem, while the solution is a piece of code in terms of decomposition: sequences, branches, and loops (Dahl et al., 1972). This means there is a gap between programming problems and their solutions, and programming itself is to cross the gap. Programming is invisible and difficult so that learning programming takes time and novices tend to drop at the beginning. In order to increase learners' development at the beginning to prevent novices from dropping, we developed a thinking module to bridge the gap between the words of problems and the code of solutions, as shown in Figure 1. Thinking from

word-level to code-level is invisible, and making thinking visible can make it much easier to learn (Perkins, 2003).

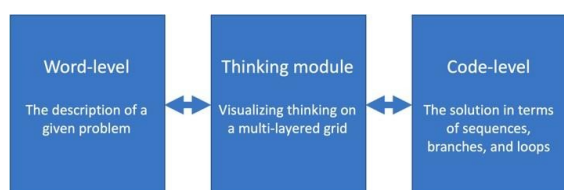


Figure 1. The thinking module bridges the gap between the words of a given problem and the code of its solution.

### 3.1. The thinking module

The design of our thinking module is shown in Figure 2, which is a multi-layered grid that visualizes the thinking of programming. Our methodology is making problems, solutions, and thinking results of programming visible by the multi-layered grid along with thinking routines. In this thinking module, problems can be visualized on top of the grid with conditions and constraints, and the solution to a problem is a route on the grid. The grid at the bottom is a printed matter, which corresponds to the story map in many board games. The problem layer represents obstacles on the grid, i.e., rocks and lakes in other map-based strategy board games. The transparent layer is a thin clear plastic sheet, like the transparency (slide) used for projectors before, where we can draw and erase. With the transparent layer, learners can draw their thinking results on top of the problem, i.e., overlapping the thinking layer with the problem layer, as shown in Figure 3. This makes learners' thinking results visible and traceable. Learners can repeatedly draw, observe, and modify the route to visualize their thinking results. Furthermore, learners can synchronize their thinking results with the problem to find out the error or even synchronize with a given solution. This multi-layered grid can make problems, solutions, and learners' thinking results visible.

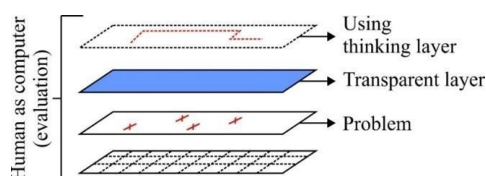


Figure 2. The design of our thinking module.

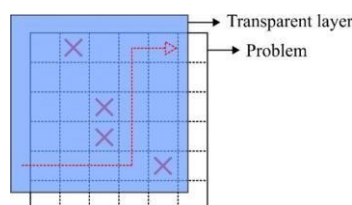


Figure 3. Learners can draw their thinking results of programming on the transparent layer (the red dotted line).

### 3.2. The board game design

We designed a multi-layer board game based on this thinking module. As shown in Figure 4, there are a grid board, many obstacles, and a set of command cards in the board game. The obstacles are placed on the grid to represent the problem, and learners need to list command cards in order to move from the Start at the left-bottom to the Goal at the right-top. The rule is similar to many map-

based strategy board games---giving commands to control the movement. The no-entry sign means the places that cannot be crossed. The player can use command pieces like Forward, Turn-left, Turn-right, If-then, and Do-while to conduct the movement. For If-then and Do-while command pieces, a flag is used as the timing to change or stop the action. For example, using Do-while along with a flag and Forward means going forward until encountering the specified flag. Before putting these command pieces together, learners can draw the route by pen and try to figure out the commands to use. In fact, the used commands form a program written in a mini-language, and the drawing on the transparent layer is the program execution. This design asks learners to think like a computer and visualize the execution. The problem shown in Figure 4 is a little hard since it needs the use of sequence, branch, and loop to solve. Figure 5 shows another problem where If-then and Forward are given along with many animal pieces and food pieces. The player can only move on the places without a no-entry sign and give specific foods to different animals according to the rules, for example giving meat to lions. In Figure 5, a learner is drawing the route by pen and listing the commands to use.

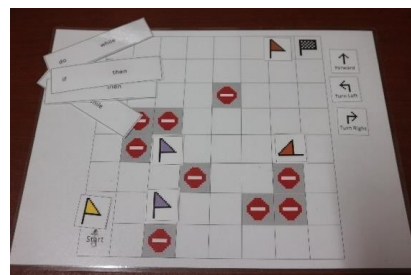


Figure 4. A draft version of our multi-layer board game.



Figure 5. A learner is drawing the route and listing the commands to use.

Unlike the design in existing board games, a whole plan needs to be set up first for a given problem rather than considering a solution for the current situation every time. This design follows the stages in problem-solving analyzed by Polya (1957). It helps learners to understand the problem with visualized elements on the problem layer and encourages learners to devise a plan. After the plan is devised, carrying it out on the thinking layer step by step and looking back to check the result. Our design helps learners to think like what computers actually do. By overlapping the thinking layer with the problem layer and the solution layer, it is easier to figure out where the error is; learners' programs are visible and traceable. The thinking layer can also be kept as a learning portfolio or compared with others' thinking results. Furthermore, the thinking layer is a stored program, which can be overlapped with the problem layer next time to execute

again. The design itself is analog and unplugged, but it emulates computer behavior. For those who need guidance, and additional transparency can be inserted as the scaffolding layer between the thinking layer and the problem layer to help learners write down parts of the solution gradually.

Although we follow the way of traversing upon the grid in existing CT board games, there are three main differences from them, as shown in Figure 6. First, learners can directly draw and erase on top of the problem to trace the movement. By observing the drawn route, learners can understand where the error is and how to fix it. Second, unlike other turn-based board games, it requires learners to think and draw a whole plan. The drawn route is a complete plan to be carried out. Third, our design trains learners to simulate the execution as computers. The route drawn on the transparent layer represents the stored program concept, and learners can simulate the execution slowly to understand what computers do.

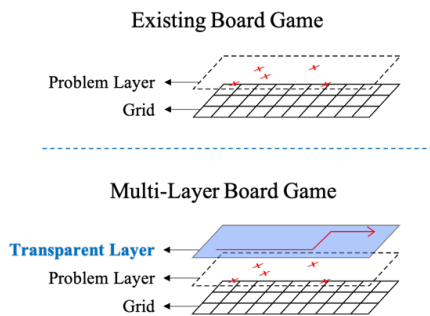


Figure 6. The transparent layer make programming thinking visible and traceable.

### 3.3. The learning design

This design is based on cognitive constructivism. Instead of teaching how to program directly, our board game lets students improve their understanding through experiences. The use of familiar and concrete models increases novices' understanding of computers and programming (Mayer, 1981). Transforming programs to the routes on a grid is such a concrete model. We followed the idea of making thinking visible (Ritchhart et al., 2011) and concretely applied it to our design. Thinking routines for programming are given:

“Which commands should we use and compose?” “What is the result of executing this single command?” “How to modify this command for fixing the error?”

Teachers can regularly use these thinking routines to develop student thinking of programming. Our design makes it possible to synchronize thinking results with the problem and compare thinking results with the solution. To students, programming is a process of decomposition. It is decomposing a program into smaller parts in the form of sequence, branch, and loop. The design of our thinking module makes every decomposition visible and traceable. On the grid, sequences are mapped to a block as in other board games, i.e., every sequence makes a step forward, and branches are transformed to matching different conditions, and loops are represented with matching the sentinel. Note that the flags in Figure 4 are sentinels in

loops and the question marks in Figure 5 denote runtime conditions. Teachers ask students to put flags to specify when the loop should be stopped and write branches in advance to handle different possible conditions.

What we want to teach is to make a whole plan in advance for a given problem and allowing to trace step by step instead of thinking every small step. What learners thought can be drawn and stored; is to teach the stored program concept. We hide some conditions to ask learners to use branches meaningfully; it is to teach being aware of runtime.

## 4. EXPERIMENT RESULTS AND DATA ANALYSIS

To know the learning performance improvement on programming using our multi-layer board game, we conducted an experiment by recruiting 60 participants, randomly divided into two groups to learn with different board games, and compared the learning achievement of programming between two groups. All subjects are the students from two universities in Taiwan, and the learning materials used in the control group and experiment group are the Robot City board game and our multi-layer board game, respectively. In the Robot City board game, players can collect resource pieces for the given tasks and control their avatars' movement (robots) by command cards. Note that Robot City is a multiplayer board game, while the current version of our board game is to encourage players to discuss and think of the movement together. We chose Robot City for comparison since it is a famous map-based strategy board game in Taiwan, and the elements in our board game are very close to it, except our board game has a transparent layer and non-turn-based design.

### 4.1. Measuring tools and experimental procedure

To understand the learning performance improvement, we designed eight questions for pre-test and post-test: 3 for sequence logic, 3 for branch logic, and 2 for loop logic. In the preparation of pre-test and post-test, Carnegie Mellon University's definition of CT and Oracle reference for programming are used as references.

The experimental procedure is shown in Figure 7. In our experiment, we exclude any subjects who already learned programming. In the beginning, we gave a pre-test to know how they understand CT and programming. Then we gave an introduction to the learning material, i.e., how to play the Robot City board game and our multi-layer board game for the control group and the experiment group, respectively. Then we gave subjects 60 minutes to play and conducted a post-test after that; we assume it is enough to get a brief understanding of the given board game. Due to the limitations on the number of players in a board game, we arranged 2~4 subjects in a game for both groups. Note that for the case of the experiment group, we further divided the game into three parts: sequence, branch, and loop. In other words, subjects in the experiment group are asked to solve the three kinds of problems in order during the 60 mins. On the other hand, subjects in the control group followed the rules in Robot City to freely play and learn the three kinds of logic.

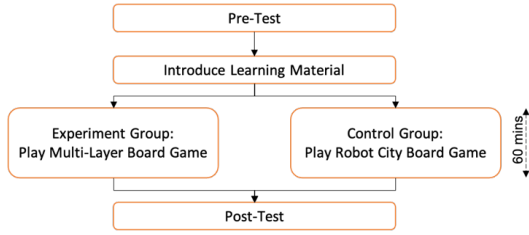


Figure 7. The experimental procedure.

## 4.2. Experiment results and discussion

After we collected and analyzed the data, we found that the data are not normally distributed. In both pre-test and post-test, the skewness and kurtosis values of the two groups are mostly more than  $\pm 1$ . A possible cause might be the background of the individual subject. Unlike the

students in a normal class who are usually selected based on some rules or entrance exams, we simply recruited subjects from several departments in the universities. Another possible cause might be the design of the test sheet, which might not make the variations in the value broadly meet the normal distribution criteria. As shown in Table 1, the Shapiro-Wilk significance values for each test

are mostly smaller than .05, which means the data are not normally distributed.

Table 1. Tests of normality.

		Kolmogrov-Smirnov			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
Total	PreTest	.177	57	.000	.944	57	.010
	PostTest	.121	57	.000	.946	57	.013
Sequence	PreTest	.299	57	.000	.763	57	.000
	PostTest	.445	57	.000	.561	57	.000
Branch	PreTest	.274	57	.000	.877	57	.000
	PostTest	.263	57	.000	.827	57	.000
Loop	PreTest	.279	57	.000	.800	57	.000
	PostTest	.186	57	.000	.881	57	.000

Table 2 shows the mean values of pre-test and post-test for the two groups, where both Robot City and the multi-layer board game improved the learning in all three kinds of logic. However, we found the multi-layer board game has succeeded in providing a significant increase in the aspects of branch logic and loop logic. Table 3 shows the results of the Mann-Whitney U test, which can be used to analyze the data that we cannot assume normality in both groups. In Table 3, there is a significant difference between the experiment group and the control group in the post-test result regarding the aspects of branch logic and loop logic. According to the information in Table 2 and Table 3, we can know that the subjects learned the concept of branch and loop better with the multi-layer board game. The use of a transparent layer might help the reflection of thinking results, and carrying out a plan as executing a stored program might give a better understanding of program execution. For the branch logic, learners may also know how to use if-else better since we hide several runtime conditions and ask learners to think a whole plan in advance. As to the loop logic, we use sentinels instead of counters to help learners to understand how to construct loop logic.

Group		Total		Sequence		Branch		Loop	
		PreTest	PostTest	PreTest	PostTest	PreTest	PostTest	PreTest	PostTest
Experiment	Mean	24.7667	34.6000	12.9333	14.2000	7.9333	12.5333	3.9000	7.8667
	N	30	30	30	30	30	30	30	30
	Std. Deviation	4.03163	4.85372	1.99885	1.34933	2.25806	2.81294	1.64736	2.67470
Control	Mean	25.8148	29.3704	13.8519	14.2963	8.7778	10.1852	3.1852	4.8889
	N	27	27	27	27	27	27	27	27
	Std. Deviation	3.38591	4.12499	1.97497	1.70553	1.84669	2.20205	1.77671	1.78311

Table 3. Mann-Whitney U test of the two groups.

		Group	N	Mean	Rank	U	p
Total	PreTest	Experiment	30	27.77	833.00		
		Control	27	30.37	820.00	368.000	.550
	PostTest	Experiment	30	37.17	1115.00		
		Control	27	19.93	538.00	160.000	.000*
Sequence	PreTest	Experiment	30	25.43	763.00		
		Control	27	32.96	890.00	298.000	.064
	PostTest	Experiment	30	27.90	837.00		
		Control	27	30.22	816.00	372.000	.484
Branch	PreTest	Experiment	30	27.22	816.50		
		Control	27	30.98	836.50	351.500	.352
	PostTest	Experiment	30	35.58	1067.50		
		Control	27	21.69	585.50	207.500	.001*
Loop	PreTest	Experiment	30	32.23	967.00		
		Control	27	25.41	686.00	308.000	.095
	PostTest	Experiment	30	36.75	1102.50		
		Control	27	20.39	550.50	172.500	.000*

\* $p < 0.05$

Table 2. Mean values of test results of the two groups.

The experiment in this study also shown the effectiveness of learning with CT board games such as Robot City. Besides, most subjects also mentioned that they had more fun in playing Robot City and the multi-layer board game focuses more on learning. The current version of our board game cannot attract children and needs learners to discuss initiatively for controlling the same avatar.

## 5. CONCLUSIONS

We followed the idea of making thinking visible to develop a thinking module for learning programming and designed a multi-layer board game with a transparent layer. Such a new kind of board game is based on the stored program concept and asks learners to think of a whole plan. The experiment results showed that such a multi-layer board game could indeed make an effective contribution to the learning performance improvement on programming. This paper reports our first step toward a board game for learning programming. We plan to fuse more computer concepts in the multi-layer board game, and making the game more fun is included in our future work as well.

## 6. REFERENCES

- Aspray, W. (1990). The stored program concept. *IEEE Spectrum*, 27(9), 51.
- Bishop-Clark, C. (1992). Protocol analysis of a novice programmer. *ACM SIGCSE Bulletin*, 24(3), 14-18.
- Brooks, R. (1983). Towards a theory of the comprehension of computer programs. *International journal of man-machine studies*, 18(6), 543-554.
- Brusilovsky, P., Calabrese, E., Hvorecky, J., Kouchnirenko, A., & Miller, P. (1997). Mini-languages: a way to learn programming principles. *Education and information technologies*, 2(1), 65-83.
- Chang, C. K. (2014). Effects of using Alice and Scratch in an introductory programming course for corrective instruction. *Journal of Educational Computing Research*, 51(2), 185-204.
- Dahl, O. J., Dijkstra, E. W., & Hoare, C. A. R. (Eds.). (1972). *Structured programming*. Academic Press Ltd..
- Davies, S. P. (1993). Models and theories of programming strategy. *International Journal of Man-Machine Studies*, 39(2), 237-267.
- Détienne, F., & Soloway, E. (1990). An empirically-derived control structure for the process of program understanding. *International Journal of Man-Machine Studies*, 33(3), 323-342.
- Hsu, T. C., Chang, S. C., & Hung, Y. T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, 126, 296-310.
- Kong, S. C., Chiu, M. M., & Lai, M. (2018). A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education*, 127, 178-189.
- Kuo, W. C., & Hsu, T. C. (2020). Learning Computational Thinking Without a Computer: How Computational Participation Happens in a Computational Thinking Board Game. *The Asia-Pacific Education Researcher*, 29(1), 67-83.
- Lahtinen, E., Ala-Mutka, K., & Järvinen, H. M. (2005). A study of the difficulties of novice programmers. *Acm sigcse bulletin*, 37(3), 14-18.
- Looi, C. K., How, M. L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education*, 28(3), 255-279.
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior*, 41, 51-61.
- Mayer, R. E. (1981). The psychology of how novices learn computer programming. *ACM Computing Surveys (CSUR)*, 13(1), 121-141.
- Papert, S. (1972). Teaching children thinking. *Programmed Learning and Educational Technology*, 9(5), 245-255.
- Papert, S. (1980). *Mindstorms; Children, Computers and Powerful Ideas*. New York: Basic Book.
- Perkins, D. (2003). *Making thinking visible. New horizons for learning*.
- Polya, G. (2004). *How to solve it: A new aspect of mathematical method (Vol. 85)*. Princeton university press.
- Prensky, M. (2008). *Programming is the new literacy*. Edutopia magazine.
- Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. (2009). *Scratch: programming for all*. *Communications of the ACM*, 52(11), 60-67.
- Ritchhart, R., Church, M., & Morrison, K. (2011). *Making thinking visible: How to promote engagement, understanding, and independence for all learners*. John Wiley & Sons.
- Rist, R. S. (1990). Variability in program design: the interaction of process with knowledge. *International Journal of Man-Machine Studies*, 33(3), 305-322.
- Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, 13(2), 137-172.
- Santos A. (2019) Board Games as Part of Effective Game-Based Learning Strategies. In: Spector M., Lockee B., Childress M. (eds) *Learning, Design, and Technology*. Springer, Cham
- Shneiderman, B. *Software psychology: human factors in computer and information systems*. 1980. Winthrop Inc, Cambridge MA.
- Siegmund, J., Peitek, N., Brechmann, A., Parnin, C., & Apel, S. (2020). Studying programming in the neuroage: just a crazy idea?. *Communications of the ACM*, 63(6), 30-34.
- Soloway, E. (1993). Should we teach students to program?. *Communications of the ACM*, 36(10), 21-24.
- Tang, K. Y., Chou, T. L., & Tsai, C. C. (2020). A content analysis of computational thinking research: An international publication trends and research typology. *The Asia-Pacific Education Researcher*, 29(1), 9-19.
- Tishman, S., & Palmer, P. (2005). Visible thinking. *Leadership compass*, 2(4), 1-3.
- Vee, A. (2013). Understanding computer programming as a literacy. *Literacy in Composition Studies*, 1(2), 42-64.
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
- Wu, S.-Y., Fang, J.-C., & Lian, S.-M. (2018). Design a Computational Thinking Board Game Based on Programming Element. *International Conference on Computational Thinking Education 2018*, Hong Kong.
- Yen, J. C., & Liao, W. C. (2019). Effects of Cognitive Styles on Computational Thinking and Gaming Behavior in an Educational Board Game. *International Journal of Learning Technologies and Learning Environments*, 2(2), 1-10



# A Framework for Integrating Computational and Design Thinking Processes

Riccardo CHIANELLA<sup>1\*</sup>, Diego REITANO<sup>2</sup>, Ettore MORDENTI<sup>3</sup>, George BARITSCH<sup>4</sup>

<sup>1,2,3,4</sup> School of Design, Politecnico di Milano, Italy

riccardo.chianella@asp-poli.it, diego.reitano@asp-poli.it, etttore.mordenti@mail.polimi.it,  
georgeoscar.baritsch@mail.polimi.it

## ABSTRACT

Due to the rapid change brought by new emerging technologies, computational thinking (CT) has become a fundamental skill. Contrarily to the large number of studies focused on introducing CT in STEM subjects, we direct our research towards a broader context, that of design. Given the importance of CT concept acquisition in terms of future design thinking education, this paper presents a qualitative study at the intersection of teaching design thinking and CT. We develop an innovative framework to integrate the two processes in design courses and we explore its potential and limitations with design lecturers who could potentially introduce the framework in their teaching practice. Moreover, we reflect on what needs to change for CT education to be successfully implemented in design schools across the world. This study refers to the example of Italy which, similarly to other countries, could constructively improve its design teaching with CT to secure its large design industry for the future.

## KEYWORDS

computational thinking, design thinking, design university

## 1. INTRODUCTION

### 1.1. Context

Technological progress directly impacts the emergence of new skills required by workers. The integration of these skills should be at the center of attention for those educational institutions preparing students entering the job market with relevant courses and subjects. In particular, the fourth industrial revolution gave life to fast-moving technological trajectories enabling new forms of creation based on the development of augmented, ubiquitous and embedded technologies, where computation sits at the core of the design production (Schwab, 2017). Therefore, we argue that computational thinking (CT) should be integrated into design to support its rapid evolution in the technology era. By teaching designers how computers think and integrating it within their practice, they can better cope with emerging technologies. CT prepares students to become better problem solvers and critical thinkers (DeSchryver & Yadav, 2015).

Existing research successfully explored possible ways of introducing computational thinking concepts in university non-STEM subjects. For example, Basawapatna et al. (2011) applied the CT process to game design. However, existing research in the design field still considered CT only as a hard skill, merely linked to coding or 3D modeling. Given the lack of studies considering CT as an integral to the design process, we identify a research gap in the field of CT for design education. This work aims to fulfill this gap by proposing a framework implementing CT into design thinking which could be applied to a broad variety of design classes.

This paper is organized as follows: first, we present the affinities between design thinking and CT. Subsequently, we introduce and define a proposed framework combining the two processes. Through a series of interviews, we test how the framework could be implemented into real-life design studios and workshops. The resulting findings will lead into a discussion which aims to identify its positive aspects and limitations. We finally describe the further research that has to be developed in order to better integrate CT in design thinking.

### 1.2. Affinities between CT and design

To ensure coherence throughout the article, we adopt Wing's definition considering CT as "an approach to solving problems, designing systems and understanding human behavior that draws on concepts fundamental to computing" (Wing, 2008). According to Wing, learning CT concepts is now seen as a practice for leading students to develop more transversal skills which do not just include programming. As reported by Soleimani (2019), computation should be considered as a thinking process, as "it is about effectively structuring information and developing logics". Tabesh (2017) proposed a four-stage model of the computational thinking process: decomposition, pattern recognition, abstraction, algorithm design. Following these premises, we argue that the implementation of CT in design education should integrate the processes of design, rather than the tools.

In one of his writings, Denning analyzed the potential in combining CT and design thinking. He stated that "If the two kinds of thinking were blended together, some significant advances in software design and development would surely follow" (Denning, 2013). Moreover, Shute, Sun & Asbell-Clarke defended that CT could help designers go beyond the limits of design thinking, which is still too tied on "product specifications and the requirements imposed by both the human and the environment" (Shute, Sun, Asbell-Clarke, 2017). These statements lay the foundation to our proposal.

The way in which design is taught among most universities around the world is by giving value to the development of a design process. This plays a crucial role in guiding designers across projects, whether they are designing objects, clothing, interfaces or interiors. Similarly to CT, the design process cannot be simplified into a problem-solving activity (Goel & Piroli, 1992), yet it is still based on an iterative and step-by-step sequence of actions (Lawson, 2006). Many design processes have been created, each one with a specific focus, content, structure or graphical notation (Bobbe, Krzywinski & Woelfel, 2016). Despite that, all processes show many similarities (Eckert & Clarkson, 2005).

For this study, we focused on the widely-known design thinking process developed by Stanford d.school (Plattner et al., 2009). This process integrates most of the existing ones;

it is taught in many design universities and has been promoted by numerous companies from the design field, including Apple, IDEO and SAP (Efeoglu et al., 2013).

### 1.3. Framework

Our proposed framework associates the four stages of CT (decomposition, pattern recognition, abstraction and algorithm creation) defined by Tabesh (2017) to the five stages of Design Thinking (empathize, define, ideate, prototype and test) described by Plattner et al. (2009).

Previous research has proven that the best way to teach students about CT is by associating its basic principles to already-known practices within their subject (Lu & Fletcher's, 2009). Moreover, the framework is shown in a circular ring exemplifying it as a process that never ends. The proposed framework is visualized at *Fig. 1*, followed by a description of how stages are linked to each other.

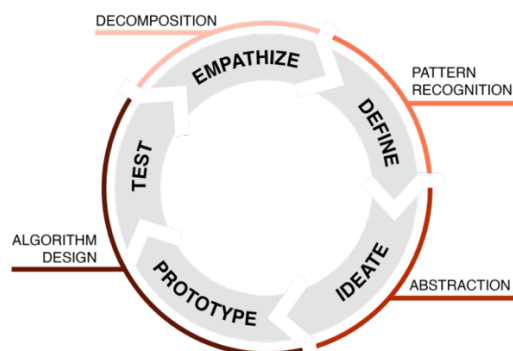


Figure 1. "Circular Framework For Computational And Design Thinking Processes". Design Thinking process (inner circle) and CT process (outer circle).

- **Empathize:** it is the stage in which designers come to understand people who experience a certain need or problem through ethnographic (or desk) research and observation. Through **decomposition** (CT), designers deconstruct a problem in many parts.
- **Define:** designers analyze deconstructed information and use the CT principle of **pattern recognition** to formulate insights: non-obvious, actionable statements that show a deep understanding of the investigated problem or need. They lead to a design challenge.
- **Ideate:** the phase in which the challenge is taken on and multiple solutions are generated to address it. A degree of openness to outer influences is required in order to produce innovative ideas. **Abstraction** is the CT principle that lets designers expand the solution space into other contexts and ultimately find a wider range of solutions.
- **Prototype and Test:** through prototyping designers produce artifacts that represent the solution in its current status. By testing the solution, designers evaluate it and identify areas of improvement. Subsequently, the whole process is iterated until the final solution is implemented, following a process of **algorithm design**.

In order for students to fully take advantage of the framework, they must put it into practice in their design studios. Following Volcz (2018), we suggest alternating a

series of 4 to 5 theoretical lectures, one for each stage of the framework, with hands-on practice, which can be achieved in design studios or workshops. Directions on the specific deliverables should be defined according to the specific type of project. We will do it by applying it to three exemplary design classes.

## 2. METHODOLOGY

To further analyze how the framework could be put into practice in a real design course, interviews were conducted with three prominent lectures of the School of Design of Politecnico di Milano. We kept our research qualitative rather than quantitative to collect in-depth insights tied to the specific needs of each design course taught by each interviewee. These interviews were semi-structured and consisted of a set of priority-defined open-ended questions. They took place via an institutional teleconferencing platform and lasted about one hour each. Interviews began by questioning interviewees' previous knowledge and CT understanding in relation to Wing's views. After that, lecturers were shown the framework we developed. The discussion was structured by analyzing each phase individually, asking the interviewees to comment on its consistency and applicability in the didactic context, concentrating on their research area. In regard to the latter comment we asked interviewees to also provide real examples in order to make the *modus operandi* more understandable.

Unsure on their level of CT understanding, we shared an introduction to our study with the participants prior to our interview. We ensured that all our participants knew that we were referring to Wing's definition of CT and we gave them the chance to ask for clarifications. Therefore, this section was a chance for making sure that participants understood the purpose of our study, thus ensuring valuable feedback. After our introduction, interviewees focused on the framework. More specifically, they shared their general impressions on how the CT process could enrich their students' learning outcomes. Then they went through each individual step of the framework and theorized some possible hands-on assignments based on the current courses that they taught. Finally, they shared eventual perplexities or improvements to the framework. Participants were asked to think out loud, allowing us to follow their reasoning and ensure that their suggestions were reliable and logical.

Collected data was analyzed through affinity diagramming to organize what could seem unstructured or dissimilar qualitative data (Hartson & Pyla, 2012). Contextual inquiry data containing quotes from the interviews was fragmented into post-its. Then the post-its were clustered according to their similarities. Finally, clusters were atomized into concise insights.

## 3. FINDINGS

The table below (*Table 1*) reports the results of the interviews regarding some practical ways in which the Circular Framework for Computational and Design Thinking Processes (*Fig. 1*) could be implemented in a design course.



Table 1: Application of the framework to some design courses

	Course in: Knitwear Design	Course in: Shapes and Algorithms for Generative Design	Course in: Methods and Instruments for Design
<b>Empathize (Decomposition)</b>	Study how fast animals run. Clusterize findings based on movement, anatomy, species, etc.	Study a particular natural phenomenon by breaking it down into its constituent elements. <i>e.g.</i> Analyze waves in liquids.	Study the structure and behavior of resistant and light materials. Break down findings into clusters.
<b>Define (Pattern Recognition)</b>	Within clusters, find a pattern that provides the key to solving the design challenge. <i>e.g.</i> the shape of the paw or texture of the skin.	Pick inspiring behaviors and find the pattern that makes them similar. <i>e.g.</i> When objects fall in water, they create ripples.	Identify cross-cluster patterns that provide a solution to a certain problem. <i>e.g.</i> Which structure is the lightest and most rigid one?
<b>Ideate (Abstraction)</b>	Abstract the findings and create a concept. <i>e.g.</i> The texture of the skin inspires a new material for a shoe.	Abstract that pattern and create a code that resembles it. <i>e.g.</i> An input for a 2D visual effect that resembles ripples of water.	Abstract findings and integrate the structure into an existing object. <i>e.g.</i> The structure can substitute plastic parts in safety helmets.
<b>Prototype and Test (Algorithm Design)</b>	Make prototypes and incrementally improve the required features. Test and reiterate. <i>e.g.</i> When tested, does this correlate with improvements in running?	Complete code. Run it and test it. Improve it and iterate the process till the desired effect is achieved.	Write a code that recreates the chosen 3D structure (CAD modeling, mathematical strength tests). 3D-print the structure and test it in real life scenarios.

## 4. DISCUSSION

### 4.1. Positive aspects of the framework

As reported by the interviewed lecturers, the mathematical and programming skills of design students in Politecnico di Milano are perceived as relatively low when compared to the European standards. Participants shared the common belief that this was due to the students' low interest in these subjects and a lack of depth being offered in these areas. It was thus important to create an accessible framework for students without an advanced knowledge in math or coding.

*"By looking at the background our students have, this [framework] is the only way in which design students could ever understand it: by comparing it to their reality." (Lecturer in Methods and Instruments for Design)*

Design lecturers referring to their previous experiences generated another powerful finding: by integrating CT in the design process, students learn how to be more versatile. For example, a participant reported the example of a shoe design project, where students incorporated computational tools in the "empathize" stage. He mentioned the act of generating new insights by studying the aerodynamics of distinct solid shapes instead of referring to existing solutions. By quoting the lecturer:

*"Once implemented in design processes, students who know computational thinking will be able to ask the right questions about their projects. They will learn how to go beyond the study of existing products by abstracting the modalities with which a product is used." (Lecturer in Knitting Design)*

Moreover, lecturers perceived it as being particularly suitable for advanced design courses and as a research tool where students who are already familiar with the design process can confidently change their workflow and apply CT to design innovative projects.

*"To get the full potential out of this framework, I would rather introduce it in the Master's course I'm teaching, rather than the Bachelor's one." (Lecturer in Shapes and Algorithms for Generative Design)*

*"This framework is more linked to our research fields, as it allows a deeper understanding of the topic." (Lecturer in Methods and Instruments for Design)*

### 4.2. Limitations of the framework

During the interviews, we came across different definitions of CT introduced in the examples given to us by the lectures,

causing time to be spent to ensure a common understanding. Additional efforts must be put into establishing a shared understanding, prior to CT being introduced into design classrooms within an institution.

Another limitation recognized by the interviewees regarding the proposed framework concerns the fact that our association of CT and design thinking works only at an introductory level and cannot be applied to learn topics too far from design. The idea of using our framework solely in an introductory level is also partially due to a lack of skills of students regarding computation. According to our interviewees a greater knowledge of hard sciences would be needed to be able to fully understand the CT process. However, the introduction of hard science subjects could be a deterrent for students to enroll:

*"Students who come to design school do not like math. This implies that if we add more math courses, less students will apply, and we will lose funds. Many students coming from high schools are frightened by these subjects." (Lecturer in Methods and Instruments for Design)*

Our interviews supported that for most projects an overlap can be seen between CT and design thinking. However, there seem to be some design areas in which CT should *not* be integrated. This is for areas with a strong sensorial component (e.g. fashion design), where a project's success heavily relies on factors that cannot be abstracted, like the sensorial perception of a material to the users. There is a gap between the minimum level of abstraction required by CT and the sensorial qualities of certain design contexts.

*"...An important role in the design process is the presence of errors, which can often generate an interesting finding. I believe that by applying CT, some projects would be error-free, and thereby become less innovative." (Lecturer in Knitting Design)*

Finally, according to two of the interviewees, the collaboration of design lecturers and computer science lecturers is preferable to develop an effective program to introduce CT as an integration to design thinking. This level of multidisciplinary collaboration has yet to be achieved, though through the interviews it was found to be feasible.

## 5. CONCLUSIONS

This study contributed to the creation of a new method to introduce CT to design students. This interdisciplinary approach was finalized with the creation of a methodical

framework merging computational and design thinking. Although this was designed to be implemented in those design universities where students lack mathematical and programming skills, our study focused on the School of Design of Politecnico di Milano. Here, the design process has never been associated with CT before. Our work can hence be considered as an example to address CT in other design schools around the world.

After analyzing the opportunity from a theoretical perspective, we developed the ‘Circular Framework for Computational and Design Thinking Processes’ with the purpose of integrating CT in the design thinking process. This framework does not see CT as a compulsory skill for designers. However, by introducing it alongside design thinking in traditional design methods, it can help shape designers who are more aware of technological power and are more versatile. The framework was further developed by running some qualitative research with lecturers in the School of Design. Design lecturers were asked to further improve the new method by sharing their expertise and applying it to the courses they were currently teaching.

The findings from our research justified how to introduce CT to design students and shared how it could be used to design innovative projects. However, the model will still suffer when put into some design teachings as one cannot consider the individuality of each project or course. Moreover, lecturers expressed their wish for establishing new collaborations between design and computer science experts to introduce the topic more properly. The framework and definition of CT was discussed though views still contradicted in small areas, exemplifying why design scholars must agree on a single CT definition for this framework to be utilized.

Now that the framework has been defined, we strongly believe that the first step towards a more in-depth version is to test it within a design studio. Recognizing the weight that a student’s perception has on reliability and applicability of our framework, further development should include a qualitative collection of students’ feedback on the “Circular Framework For Computational And Design Thinking Processes”. Moreover, we should include some quantitative research method, for instance by assessing the level of CT skills of students prior to and following introduction to the framework. Finally, given that our research was conducted in Politecnico di Milano, we would like to draw attention on other design schools in other countries to further research on how this framework could be implemented in their curriculum. A global discussion on the topic would bring up new limitations and advantages, hence improving the framework on a world-wide level. The concept stemming from our work is thereby an attempt to stimulate a deeper reflection on the intrinsic relationship between design education and CT. This could be expanded even more by exploring how design thinking practices can be applied to the design of computational solutions.

## 6. REFERENCES

Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., & Marshall, K. S. (2011, March). Recognizing

computational thinking patterns. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, 245-250.

Bobbe, T., Krzywinski, J., & Woelfel, C. (2016). A comparison of design process models from academic theory and professional practice. In *DS 84: Proceedings of the DESIGN 2016 14th International Design Conference*, 1205-1214.

Denning, P. J. (2013). Design thinking. *Communications of the ACM*, 56(12), 29-31.

DeSchryver, M. D., & Yadav, A. (2015). Creative and computational thinking in the context of new literacies: Working with teachers to scaffold complex technology-mediated approaches to teaching and learning. *Journal of Technology and Teacher Education*, 23(3), 411-431.

Eckert, C. M.; Clarkson, P. J. (2005): The reality of design. In P. J. Clarkson, C. M. Eckert (Eds.): *Design Process Improvement A review of current practice*. London, 1–29.

Efeoglu, A., Möller, C., Sérié, M., & Boer, H. (2013). Design thinking: characteristics and promises. In *Proceedings of 14th International CINet Conference on Business Development and Co-creation* (pp. 241-256).

Goel, V., Piroli, P. (1992): The structure of design problem spaces. In *Cognitive Science* 16 (3), pp. 395–429

Hartson, R., & Pyla, P. S. (2012). *The UX Book: Process and guidelines for ensuring a quality user experience*. Elsevier.

Lawson, B. (2006). *How designers think: The design process demystified*. Routledge.

Lu, J.J., & Fletscher, G.H.L. (2009). Thinking About Computational Thinking. *ACM SIGCSE Bulletin*, 41(1), 260-264.

Plattner, H., Meinel, C., & Weinberg, U. (2009). *Design-thinking. Landsberg am Lech: Mi-Fachverlag*.

Schwab, K. (2017). *The fourth industrial revolution. Currency*.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, 22, 142-158.

Soleimani, A. (2019). Computational Design Thinking and Thinking Design Computing. In *2019 Reynolds Symposium: Education by Design. Portland, Oregon, October 18-19, 2019*.

Tabesh, Y. (2017). Computational thinking: A 21st century skill. *Olympiads in Informatics*, 11(2), 65-70.

Volcz, I. (2018). The Use of Computational Thinking to Advance Learning in the Pre-university Subject of Digital Literacies. In *Proceedings of the International Conference on Computational Thinking Education 2018. Hong Kong: The Education University of Hong Kong*. 68-71

Wing, J. (2008). Computational thinking and thinking about computing. *Philosophical Transactions Of The Royal Society A: Mathematical, Physical And Engineering Sciences*, 366(1881), 3717-3725.

# The Effects of an AR Programming Game on Students' Different Prior Computational Thinking Skills

Huai-Hsuan HUANG<sup>1\*</sup>, Vandit SHARMA<sup>2</sup>, Kaushal Kumar BHAGAT<sup>3</sup>, Wen-Min HSIEH<sup>4</sup>, Nian-Shing CHEN<sup>5</sup>

<sup>1,4,5</sup> National Yunlin University of Science and Technology, Taiwan

<sup>2,3</sup> Indian Institute of Technology Kharagpur, India

b10441011@gmail.yuntech.edu.tw, vanditsharma02@gmail.com, kkntnu@hotmail.com,  
wmhsieh@gmail.yuntech.edu.tw, nianshing@gmail.com

## ABSTRACT

Cultivating students' computational thinking (CT) skills is challenging and often entails introducing them to basic programming concepts. These concepts are quite abstract and difficult to visualize. Our study aims to address this gap through an AR-based programming game. We evaluated our game on 27 participants, ranging from freshmen to seniors and having different prior CT skills. The results show that our AR programming game significantly improved students' CT skills, especially for students with lower prior CT levels. Some pedagogical implications and design guidelines for teachers and game developers to improve AR programming games are also discussed.

## KEYWORDS

computational thinking, AR-Game, programming skills, tangible object, embodied cognition

## 1. INTRODUCTION

CT can be defined as a set of fundamental skills that employ computer science concepts towards "solving problems, designing systems, and understanding human behavior" (Wing, 2006, p.33; Kong, Lai & Sun, 2020). The importance of developing CT skills was emphasized by Wing (2006), who mentioned that these skills should be developed not only by students majoring in computer science but by everyone. Students with well-developed CT skills can perform better in various domains of expertise (Hooshyar, Malva, Yang, Pedaste, Wang & Lim, 2021). Therefore, from early to higher studies, educators should put a large amount of emphasis on developing students' CT skills.

Research has shown that CT can be developed through programming or coding (Zhang & Nouri, 2019; Wei, Lin, Meng, Tan, Kong & Kinshuk, 2020). However, coding is often challenging for beginners since it requires the knowledge of abstract concepts such as simulation, algorithm, abstraction and so on (Lye & Koh, 2014). Therefore, many CT instructors have started to use block-based programming tools, essentially turning complex programming languages into simplified blocks (Zhang & Nouri, 2019; Zhao & Shute, 2019) that are easy to teach. Scratch<sup>1</sup> and Code.org<sup>2</sup> are two examples of such tools created specifically to improve the programming skills of beginners. These websites simplify abstract programming

languages and gamify the learning process, thus building students' confidence and motivation in learning how to code (Czerkawski & Lyman, 2015). Several researchers have investigated the effectiveness of these web and game-based programming tools. Results from a systematic review study (Hu, Chen & Su, 2020) that collected 29 related empirical studies indicated that these programming websites significantly improve learners' coding skills. Theodoropoulos and Lepouras (2020) also reviewed 44 studies to confirm the effectiveness of digital programming games to develop CT. Apart from the abovementioned games that are played on computers, numerous block-based programming games, such as Light-Bot<sup>3</sup>, RoboLogic<sup>4</sup>, and Sprite Box<sup>5</sup>, running on portable devices, i.e., smartphones and tablets, have sprouted in recent years (Lindberg, Laine & Haaranen, 2018). These games allow learners to develop CT anytime and anywhere. Despite several websites and mobile applications being available for educational use, many programming beginners still feel that coding is abstract and frustrating (Dohn, 2019).

Providing programming learners an immersive coding environment could be a solution to this problem. Augmented Reality (AR), featuring its integration of virtuality and reality, can augment the physical world with interactive virtual additions and immerse users in an authentic scenario (Rauschnabel, Rossmann & tom Dieck, 2017). Many past studies found that AR can be implemented in education, especially for subjects having abstract concepts. For example, research works that applied AR in mathematics (Cai, Liu, Shen, Liu, Li & Shen, 2018) and geometric (Gecu-Parmaksiz & Delialioglu, 2019) courses showed significant improvements in students' learning performance because AR can (1) visualize abstract knowledge and (2) engage them in hands-on learning activities. By virtue of these two advantages, we applied AR to develop a programming game for students to enhance their CT skills. To examine the effectiveness of our AR-Game, CodAR (Sharma, Talukdar & Bhagat, 2019), we evaluated it with undergraduate students. Based on this evaluation, we aim to answer the following research question through this study: What is the impact of playing the AR-Game on CT skills for students with different prior CT levels?

<sup>3</sup> <https://lightbot.com/> <sup>4</sup> <https://apps.apple.com/us/app/robologic-lite/id300893278>

<sup>5</sup> <https://spritebox.com/hour.html>

<sup>1</sup> <https://scratch.mit.edu/>

<sup>2</sup> <https://code.org/>

## 2. METHOD

### 2.1. Participants

A total of 31 university students ranging from freshmen to seniors participated in this study. The students were from different disciplines (e.g., engineering, management, design, humanities) and had registered in a general education course, The Introduction of Smart Systems and Mobile Phone Apps, at a university in Taiwan. Only one-fourth of students majored in computer engineering and routinely practiced writing programs. Other students had a basic knowledge of CT skills due to previous instruction this course gave. Note that four students' data could not be collected completely; thus, their data were excluded.

### 2.2. Procedure

The study took about two classes of the general education course in the middle of the semester. Figure 1 presents the procedure of the research design. The students first took a 30-minute pre-test assessing their prior CT level. The first author, who has played the game several times during its development, served as the instructor. The instructor first gave a brief introduction to the game. Students learned how to sign in to the game, arrange the play cards, scan the AR markers on the play cards, and then they started to play the game. During the game-play, the instructor provided prompt assistance but did not guide participants to achieve the game's goals. After around 30 minutes, students had to stop playing the game and then take the post-test for 30 minutes.

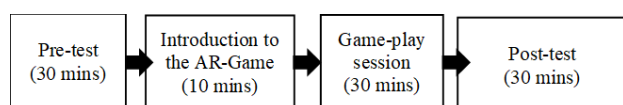


Figure 1. The Procedure of Research Design

### 2.3. The AR-Game

Empirical evidence has shown that embodied strategies are especially effective for learning (Macedonia, 2019). To leverage the benefits of embodied cognition, marker-based AR was adopted to design this game. According to previous research (Gecu-Parmaksiz & Delialioglu, 2019), applying AR in education can help students visualize abstract knowledge and also engage them in real-time instruction during learning. Therefore, this study used an AR Programming mobile game to teach CT.

Two physical objects were required to play the AR-Game: a smartphone/tablet and a set of playing cards. There were six types of playing cards used in this game, out of which one was used to generate the world for a level in AR, and the rest acted as programming blocks. In the game, students were expected to place the playing cards in a logical sequence and use devices to scan and visualize the playing cards required for each level (See Figure 2).

To complete the game, students went through 12 levels of programming. The coding scenario involved helping the game character, a bunny, collect all carrots placed on platforms arranged differently for each level. The player's primary objective was to guide the bunny to collect all

the carrots by properly arranging the programming blocks specified for that level.

Whenever a player would place the playing cards and view them through the handheld smartphone/tablet, the entire game would come to life by the virtual superimposed elements appearing over the playing cards (See Figure 3).



Figure 2. Student's Manipulation of the Game

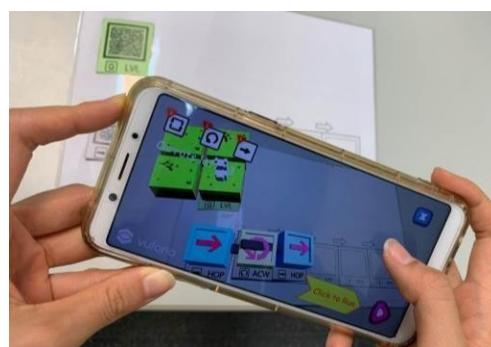


Figure 3. The Presentation of the Game

These virtual superimposed elements enabled players to visualize their code corresponding to the chosen sequence of the playing cards by watching the bunny in action. Players could find problems with their solution through this programming presentation visualized in real-time. Since virtual superimposed items can be viewed from different perspectives using AR, such a three-dimensional programming environment could also help students having an inferior sense of direction to overcome such difficulties. Additionally, various in-game animations kept the students engaged and motivated towards completing the game while learning CT during the process.

### 2.4. Data Collection

The instruments used in this study included a pre-test and a post-test, which were both identical. To examine the effectiveness of the developed game, these tests were designed based on a commercial mobile application, LightBot<sup>3</sup>, adapting it into a pen-and-paper test. The validity of the assessment was checked through audit trail and literature review (Gouws, Bradshaw & Wentworth, 2013). The assessment included five ordering-sequence questions. Each question contained an image that presented a problem statement to the test takers. The solution to each problem statement was a sequence of commands that guided the main character in LightBot, a robot, to reach its destination.

## 2.5. Data Analysis

The pre and post-test scores were determined together by the first author and a sophomore student majoring in Computer Science and Information Engineering at the university. The scores were consistently discussed before being determined.

The students' prior CT level as determined by their pre-test scores was used to identify the top 40% of the class as the higher prior CT level (HPCT) group ( $M = 8.9$ ,  $SD = 1.87$ ), and students at the bottom 40% of the class as the lower prior CT level (LPCT) group ( $M = 3.7$ ,  $SD = 1.00$ ). The scores of these two groups were separated by an interval of six. To differentiate these two target CT level groups clearly, seven students were excluded from the analysis. There were ten students each in both the HPCT and LPCT groups. Finally, to examine the impact on students' CT skills after playing the game, a paired sample t-test was conducted to measure the CT gains made by HPCT and LPCT groups.

## 3. RESULT

The results from analyzing the pre-test and post-test scores for both HPCT and LPCT groups are presented in this section. Note that seven students' learning gains were excluded from this two-group analysis since they had an average prior CT level.

### 3.1. Improvements in CT skills

As shown in Table 1, LPCT students ( $t = 5.511$ ,  $p = .00037$ ) made a significant progress after the playing the game.

*Table 1.* Results for paired sample t-test to compare pre-test and post-test scores for the LPCT and HPCT Groups

Groups	Mean	SD	t	Sig. (2-tailed)
<b>HPCT Group (N=10)</b>				
Pre-test	8.9	1.868	1.167	.2729
Post-test	9.4	2.009		
<b>LPCT Group (N=10)</b>				
Pre-test	3.7	1.004	5.511	.0004*
Post-test	7.3	2.051		

Note: \* $P < .001$

## 4. DISCUSSION

Our quantitative results showed that the lower prior CT level group improved more than its counterpart, suggesting that this game is especially useful for students with lower CT skills. This is consistent with the empirical study conducted by Hooshyar et al. (2021), who compared students' CT gains from two different learning approaches- conventional technology-enhanced learning and game-based learning. Similar to us, they also separated students in the game-based learning group into two sub-groups based on their prior CT level. They found that the students in the game-based learning group outperformed the group under conventional instruction. Moreover, students in the lower prior CT sub-group benefited more from the intervention. McLaren et al. (2017) also obtained similar results from their

comparative research, explaining this phenomenon using the "expertise reversal effect" (Kalyuga, 2007), where they suggested that, more or less, some groups benefit more than others under such instructional techniques. We further examined why the low prior CT group benefitted more from this game. Two key areas where our game was able to address difficulties typically faced by such students were- (1) sequencing program scripts (Spohrer, 1989) and (2) understanding the computational process (Du Boulay, 1986). Overcoming these difficulties requires scaffolding techniques such as making predictions, planning, and monitoring the coding process (Basu, Biswas, Sengupta, Dickes, Kinnebrew & Clark, 2016). In practice, our AR-Game served as a useful scaffolding tool, helping students with lower prior CT skills overcome the two abovementioned difficulties by visualizing the coding process, enabling them to evaluate their programs and solve the problems (Wong & Cheung, 2020).

We can identify two potential limitations that exist with this study. First, since we have chosen participants from a CT-related course as our data collection resource, this possibly resulted in a selection bias. Collecting data from a broader student population would help us obtain more consistent results in the future. The second limitation is concerned with the fact that study duration might not be long enough to have significant effects on students' CT skills. However, since students with different prior CT skills were subjected to the same amount of intervention, we believe that our study nevertheless provides reliable results. Future longitudinal research in this direction should keep these limitations in mind.

## 5. CONCLUSION

In response to the pressing need to equip learners with CT skills, this study presented an AR-based programming game to engage students in a tangible and authentic coding environment for CT development. The findings show that students having lower prior CT skills are more likely to benefit from the game. They can take advantage of AR by predicting and debugging their coding through the visual programming experience.

Pedagogical implications for teachers who teach programming can be drawn based on the results of this study. First, the developed game can be used by programming beginners who face problems in understanding concepts and lack the required CT skills. Second, to design a curriculum that caters to students with different CT levels, students' prior knowledge should be taken into consideration. Therefore, it is suggested that teachers group students based on their prior CT skills to maximize learning outcomes.

For game developers to improve the gaming experience for everyone, serious games should have a pre-test section to identify players' prior knowledge, e.g., CT level in our case. This information can then be used to adapt the game according to the students' skills. For instance, developers can increase complexity for more experienced students by matching them to a higher game level. Such practices will help maintain student interest in the game as well as maximize their learning outcomes.



## 6. REFERENCES

- Basu, S., Biswas, G., Sengupta, P., Dickes, A., Kinnebrew, J. S., & Clark, D. (2016). Identifying middle school students' challenges in computational thinking-based science learning. *Research and Practice in Technology Enhanced Learning*, 11(1). <https://doi.org/10.1186/s41039-016-0036-2>
- Cai, S., Liu, E., Shen, Y., Liu, C., Li, S., & Shen, Y. (2020). Probability learning in mathematics using augmented reality: impact on student's learning gains and attitudes. *Interactive Learning Environments*, 28(5), 560–573. <https://doi.org/10.1080/10494820.2019.1696839>
- Czerkawski, B. C., & Lyman, E. W. (2015). Exploring Issues About Computational Thinking in Higher Education. *TechTrends*, 59(2). <https://doi.org/10.1007/s11528-015-0840-3>
- Dohn, N. B. (2020). Students' interest in Scratch coding in lower secondary mathematics. *British Journal of Educational Technology*, 51(1), 71–83. <https://doi.org/10.1111/bjet.12759>
- Du Boulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research*, 2(1), 57-73.
- Gecu-Parmaksiz, Z., & Delialioglu, O. (2019). Augmented reality-based virtual manipulatives versus physical manipulatives for teaching geometric shapes to preschool children. *British Journal of Educational Technology*, 50(6), 3376–3390. <https://doi.org/10.1111/bjet.12740>
- Gouws, L. A., Bradshaw, K., & Wentworth, P. (2013). Computational thinking in educational activities: an evaluation of the educational game light-bot. In *Proceedings of the 18th ACM conference on Innovation and technology in computer science education* (pp. 10-15).
- Hooshyar, D., Malva, L., Yang, Y., Pedaste, M., Wang, M., & Lim, H. (2021). An adaptive educational computer game: Effects on students' knowledge and learning attitude in computational thinking. *Computers in Human Behavior*, 114, 106575.
- Hu, Y., Chen, C. H., & Su, C. Y. (2020). Exploring the Effectiveness and Moderators of Block-Based Visual Programming on Student Learning: A Meta-Analysis. *Journal of Educational Computing Research*. <https://doi.org/10.1177/0735633120945935>
- Kalyuga, S. (2007). Expertise reversal effect and its implications for learner-tailored instruction. *Educational psychology review*, 19(4), 509-539.
- Kong, S. C., Lai, M., & Sun, D. (2020). Teacher development in computational thinking: Design and learning outcomes of programming concepts, practices and pedagogy. *Computers and Education*, 151. <https://doi.org/10.1016/j.compedu.2020.103872>
- Lindberg, R. S. N., Laine, T. H., & Haaranen, L. (2018). Gamifying programming education in K-12: A review of programming curricula in seven countries and programming games. *British Journal of Educational Technology*. <https://doi.org/10.1111/bjet.12685>
- Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12?. *Computers in Human Behavior*, 41, 51-61.
- Macedonia, M. (2019). Embodied Learning: Why at School the Mind Needs the Body. *Frontiers in Psychology*, 10(October), 1–8. <https://doi.org/10.3389/fpsyg.2019.02098>
- McLaren, B. M., Adams, D. M., Mayer, R. E., & Forlizzi, J. (2017). A computer-based game that promotes mathematics learning more than a conventional approach. *International Journal of Game-Based Learning (IJGBL)*, 7(1), 36-56.
- Rauschnabel, P. A., Rossmann, A., & tom Dieck, M. C. (2017). An adoption framework for mobile augmented reality games: The case of Pokémon Go. *Computers in Human Behavior*, 76, 276–286. <https://doi.org/10.1016/j.chb.2017.07.030>
- Sharma, Vandit & Talukdar, Jeevankur & Bhagat, Kaushal. (2019). CodAR: An Augmented Reality Based Game to Teach Programming. In *Proceedings of the 27th International Conference on Computers in Education* (pp. 600-602).
- Spohrer, J. C., & Soloway, E. (1989). Simulating Student Programmers. In *IJCAI* (Vol. 89, pp. 543-549).
- Theodoropoulos, A., & Lepouras, G. (2020). *Digital Game-Based Learning and Computational Thinking in P-12 Education* (pp. 159–183). <https://doi.org/10.4018/978-1-7998-4576-8.ch007>
- Wei, X., Lin, L., Meng, N., Tan, W., Kong, S.-C., & Kinshuk. (2020). The Effectiveness of Partial Pair Programming on Elementary School Students' Computational Thinking Skills and Self-Efficacy. *Computers & Education*, 104023. <https://doi.org/10.1016/j.compedu.2020.104023>
- Wing, J. M. (2006). Computational thinking. In *Communications of the ACM* (Vol. 49, Issue 3). <https://doi.org/10.1145/1118178.1118215>
- Wong, G. K. W., & Cheung, H. Y. (2020). Exploring children's perceptions of developing twenty-first century skills through computational thinking and programming. *Interactive Learning Environments*, 28(4), 438–450. <https://doi.org/10.1080/10494820.2018.1534245>
- Zhang, L. C., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers and Education*, 141(June), 103607. <https://doi.org/10.1016/j.compedu.2019.103607>
- Zhao, W., & Shute, V. J. (2019). Can playing a video game foster computational thinking skills? *Computers and Education*, 141(January), 103633. <https://doi.org/10.1016/j.compedu.2019.103633>

# A Systematic Review of Distributed Pair Programming Based on the Team Effectiveness Model

Fan XU<sup>1\*</sup>, Ana-Paula CORREIA<sup>2</sup>  
<sup>1,2</sup>The Ohio State University, United States  
xu.3849@osu.edu, correia.12@osu.edu

## ABSTRACT

Distributed Pair Programming attracts increasing attention as an approach to improve students' collaborative problem-solving skills. A systematic review of the empirical studies published in the last decade was conducted, and 23 articles were reviewed in the analysis. The results show the educational contexts and the subjects of the selected studies, as well as the mainstream programming language and the popular integrated development environments in DPP-based learning activities. The extracted interventions were classified based on the Team Effectiveness Model and we found that individual factors and team environment attracted major investigations, whereas insufficient exploration was on the task structure, pair efforts, and team dynamics in the collaboration process in DPP-related practices to enhance students' computational thinking.

## KEYWORDS

Distributed Pair Programming, Systematic Review, Computational Thinking, STEM Education

## 1. INTRODUCTION

Programming has been integrated into school education in many countries and regions (Falkner et al., 2014). A growing number of researchers address the importance of collaboration and teamwork in the learning of computational thinking (Al-Jarrah & Pontelli, 2014). The Pair Programming (PP) technique, referring that two programmers sit side by side in front of only one set of computer devices and work collaboratively on the same design, algorithm, code, or test (Beck, 2000), has also been widely used in education for more than a decade (Salleh et al., 2014).

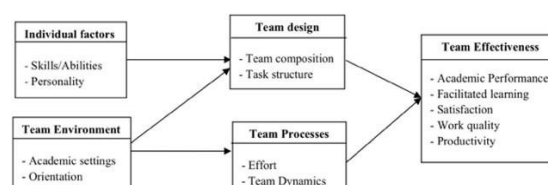
In light of the ever-evolving trends towards global collaboration, distance education, and the current teleworking new normal during the pandemic, programmers commonly work in a geographically distributed manner. The Distributed Pair Programming (DPP) is increasingly needed, which means two programmers work collaboratively on the same project from distributed locations under the support of tools that allows screen sharing and communication (Baheti et al., 2002). However, previous research showed major investigations into collocated PP but limited exploration on the impact of the geographical distribution (Hanks et al., 2011). To explore what interventions have been used and their effectiveness in supporting DPP, a systematic review of empirical studies was conducted in this study.

## 2. FRAMEWORK

Faja (2011) indicated that limited attention has been paid to theories that have the potential to reveal factors important to the successful adoption of pair programming. The author

constructed the team effectiveness model as a conceptual framework to understand PP as a team learning activity. In this study, this model will be adopted to review the DPP-related studies and the corresponding team effectiveness.

Figure 1. Team Effectiveness Model in Pair Programming.



## 3. METHODS

A systematic literature review was conducted in this study to address our targeted research problems, following Kitchenham and Charters' guidelines (2007). The overarching research purpose of this review is to identify and understand the factors that influence the effectiveness of the implementation of the DPP approach. To be specific, the following research questions will be addressed:

**RQ1.** What are the contexts where the DPP approach were implemented in the existing empirical studies?

**RQ2.** What are the mainstream programming language and Integrated Development Environments (IDE) in DPP activities?

**RQ3.** What factors were investigated to facilitate DPP? Does each intervention have positive effectiveness?

The primary data were collected by searching for related articles published in the last decade across eight databases (ACM Digital Library; IEEE Xplore; ISI Web of Science; Science Direct; ERIC; Education Research Complete; Academic Search Complete; and Education Full Text) with the search string ("*virtual pair programming*" OR "*remote pair programming*" OR "*distributed pair programming*") AND (*experiment* OR *measurement* OR *evaluation* OR *assessment*) AND (*effective* OR *efficient* OR *successful*) AND (*empirical research* OR *empirical study* OR *data* OR *sample* OR *participants*).

Upon completion of the primary search, the identification of relevant literature continued with the secondary search. All of the reviewed articles and references in the articles identified from the primary sources were reviewed based on the exclusion criteria: 1) Articles in other types other than peer-reviewed articles; 2) Articles in languages other than English; 3) Articles without available full texts; 4) Articles with no supporting empirical evidence on DPP or alternative terms. Finally, 23 studies were qualified for the synthesis.



## 4. PRELIMINARY RESULTS

The results show that the majority of studies (n=16) conducted formal experiments with controllable variables. And there were four case studies, and the remaining three used surveys to achieve their research objectives. Besides one case study that analyzed professional programmers' programming processes and two experiments that recruited programmers working in the software development industry as or partially as the research subjects, most of the studies were conducted in educational contexts: 17 with undergraduate students, two with graduate students and two with both undergraduates and graduates. And by analyzing the contexts where the reviewed studies were implemented, we found that 13 studies in total were designed and developed as part of the computer science courses like Object-oriented Programming. Whereas the rest ten were conducted as independent experiments in the laboratory context.

Regarding the programming language, the absolute majority of the studies explored DPP as a technique to learn Java (n=16), while there is respectively one article that mentioned SQL, Python, R, and visual programming. It is also shown that SCEPPSys (n=8) and Eclipse (n=4) are the mainstream IDE to support the application of DPP.

Of the identified 23 studies, six focused on the impact of the DPP approach. It has been suggested that DPP, under favorable conditions, is beneficial for students to perform better in exams, to produce codes of higher quality, and to gain a learning experience with higher enjoyment. However, Zacharis (2010) indicated that more effort was needed from pairs for completing the same programming tasks compared with individuals. Therefore, to fulfill its advantages as a learning approach, suitable interventions provided by the instructor are critical.

Figure 2. Influencing Factors and the Effectiveness.

Factors	No. of study	Pass rate	Performance	Program quality	Compatibility	Completion time	Productivity	Feel good factor	Satisfaction	Engagement	Student contribution	Task contribution	Interaction	Collaboration	Collaboration difficulty	Learning perception	Knowledge acquisition	Level of understanding
<b>Individual factors</b>																		
Actual skill level	3																	
Perceived skill level	3																	
Skill difference	1																	
Programming experience	2																	
Time management ability	1																	
Personality	1																	
Compatibility	2																	
Gender	2																	
Self-esteem	1																	
Confident level	2																	
"Feel good" factor	2																	
<b>Team Environment</b>																		
IDE/Tool	5																	
Scripted Roles	3																	
Automatically intervention	1																	
Sharing selection	1																	

Note: Green=Positive effect, Yellow=No effect, Blue=Mix

The rest 17 of the studies explored the influencing factors which possibly have a bearing on the effectiveness of DPP. These factors, as well as their corresponding effects on different aspects of the outcomes, were presented in Figure 2. The analysis result indicates that the individual factors that directly related to participants attracted major

investigation. And efforts have also been made on the impact of the team environment to the impact of DPP, such as the IDE and the scripted-role orientation.

## 5. CONCLUSION

The results show that most studies were conducted in an undergraduate educational context with computer science as the subject. It is also indicated that Java is the mainstream programming language in DPP-based learning activities, and the popular integrated development environments include SCEPPSys and Eclipse. The analysis of interventions in the selected studies found that individual factors and team environment attracted major investigations, and most of the investigated factors have a positive effect on DPP. Whereas limited attention has been paid to the DPP task structure, pair efforts, and team dynamics in the collaboration process. Future work should explore how these factors affect DPP and how to use them in learning practices to enhance students' computational thinking.

## 6. REFERENCES

- Al-Jarrah, A., & Pontelli, E. (2014, October). " AliCe-ViLlagaE" Alice as a Collaborative Virtual Learning Environment. In *2014 IEEE Frontiers in Education Conference (FIE) Proceedings* (pp. 1-9). IEEE.
- Beck, K. (2000). *Extreme programming explained: embrace change*. Addison-Wesley Professional.
- Baheti, P., Gehringer, E., & Stotts, D. (2002, August). Exploring the efficacy of distributed pair programming. In *Conference on Extreme Programming and Agile Methods* (pp. 208-220). Springer, Berlin, Heidelberg.
- Faja, S. (2011). Pair programming as a team based learning activity: a review of research. *Issues in information systems*, 12(2), 207-216.
- Falkner, K., Vivian, R., & Falkner, N. (2014, January). The Australian digital technologies curriculum: challenge and opportunity. In *Proceedings of the Sixteenth Australasian Computing Education Conference-Volume 148* (pp. 3-12).
- Hanks, B., Fitzgerald, S., McCauley, R., Murphy, L., & Zander, C. (2011). Pair programming in education: a literature review. *Computer Science Education*, 21(2), 135-173.
- Kitchenham, B. A., & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering technical report. *Software Engineering Group, EBSE Technical Report, Keele University and Department of Computer Science University of Durham*, 2.
- Organization for Economic Cooperation and Development. (2013). *PISA 2015: Draft collaborative problem solving framework*.
- Salleh, N., Mendes, E., & Grundy, J. (2014). Investigating the effects of personality traits on pair programming in a higher education setting through a family of experiments. *Empirical Software Engineering*, 19(3), 714-752.
- Zacharis, N. Z. (2010). Measuring the effects of virtual pair programming in an introductory programming java course. *IEEE Transactions on Education*, 54(1), 168-170.

# **Computational Thinking and Special Education Needs**

# Proposal for the Production of Virtual Reality Environments in Elementary Education with a Constructivist Approach

José E. GUZMÁN-MENDOZA<sup>1</sup>, Héctor CARDONA-REYES<sup>2\*</sup>, M. Lorena BARBA-GONZÁLEZ<sup>3</sup>,  
Klinge O. VILLALBA-CONDORI<sup>4</sup>, Dennis ARIAS-CHAVEZ<sup>5</sup>, M. Luisa Fernanda RÁBAGO-GONZÁLEZ<sup>6</sup>

<sup>1, 3</sup> Center for Research in Mathematics, Quantum: Knowledge City, Zacatecas, Mexico.

<sup>2</sup> CONACY Research fellow, CIMAT Zacatecas, Mexico.

<sup>4</sup> Universidad Católica de Santa María, Perú

<sup>5</sup> Universidad Continental, Arequipa, Perú

<sup>6</sup> Technological University of Guadalajara

mitc.eder@gmail.com, hector.cardona@cimat.mx, maria.barba@cimat.mx, kvillalba@ucsm.edu.pe,

darias@continental.edu.pe, fernanda.lfrg21@gmail.com

## ABSTRACT

Currently, education continues to search for new strategies that contribute to an improvement in the learning of children in primary education and one of the ways is to use emerging technologies. In this sense, virtual reality environments allow the child to build this knowledge in an immersive way and through various forms of interaction to be more intuitive with the environment allowing to reach knowledge. In this work, virtual reality environments are proposed as a support for basic education in regular children and children with learning difficulties through the constructivism approach. The elements involved in the design of the software and the multidisciplinary work for the creation of these virtual reality environments are presented. A case study is presented where a virtual reality environment is used by teachers of a basic education institution in Mexico as part of their strategies to strengthen attention and hyperactivity activities and cognitive stimulation in children and to give continuity to the activities under the confinement measures established as a consequence of the pandemic.

## KEYWORDS

Virtual reality, constructivism, elementary education, special needs.

## 1. INTRODUCTION

Elementary education over the years has adapted to the technological changes and situations of a globalized environment. Today the world is experiencing one of the most complex scenarios in terms of health restrictions due to the COVID-19 pandemic (Daniel, 2020, Burgess and Sievertsen, 2020). These effects have caused the learning theories implemented in elementary education to undergo adaptations to guarantee the continuity of teaching in schools. It is also important to consider that in classrooms there is a wide diversity of students who require personalized attention and learning strategies should be directed to best meet their needs. To this end, the use of technology has played a very important role in this process of educational adaptation (Dean, 2002), allowing students to develop their forms of organization, structures, and use of diverse resources to incorporate them into their virtual classes and build better learning.

Constructivism is a paradigm oriented to allow the student to carry out his learning process in a dynamic way by discovering his environment through the interaction

between the objects and their related environment, allowing him to solve problems using past experiences (Tuncel and Bahtiyar, 2015), at all times the teacher becomes a guide, facilitator, and motivator (Shantz, 1995). There are several types of tools that offer different forms of interaction, one of which is virtual reality environments that allow simulating elements of the real world using 3D representations. Learning-oriented virtual reality environments allow the user to interact with a reduced abstract representation of the environment and the user can create some short constructions within this environment which behave according to a set of concepts under which it has been modeled (Requena, 2008). This work proposes virtual reality environments as a complement to the learning process in elementary school children through scenarios that promote cognitive development, based on the constructivist model so that the student builds knowledge using a virtual reality environment. This work consists of five sections, Section 2 presents proposals available in the literature suggested in technology use in education. In section 3, the constructivist approach is presented through the incorporation of virtual reality environments to support the educational process. Section 4 presents a case study in which the implementation of a virtual reality environment with elementary education children is described. Finally, a section of conclusions and future work are presented.

## 2. RELATED WORKS

Throughout the decades, various researchers in the field of education have tried to define models for integrating ICTs effectively into educational processes. One of the emerging technologies that have become more relevant today is Virtual Reality (VR).

From a pedagogical point of view, researchers and experts are constantly modifying the learning theories used to develop VR scenarios. However, several studies interconnecting VR with constructivist theory have demonstrated the potential of this pairing to create educational environments where students effectively learn representations of concepts, which maximizes learning. One such study that demonstrates the effectiveness of using VR and constructivism to develop meaningful learning is the work of (Collins Jonny, 2018) where they use VR to create an immersive environment using HTC Vive technology to create an interactive system where students can interact with figures in 4 dimensions, such as

the hypercube. With the experiment, they demonstrated that the students who participated in the study acquired the expected knowledge on the proposed topics.

In this sense, it is observed that constructivism allows managing how a student can learn new knowledge. In the case of the work of (Piovensan Melchoiri Peruzza, 2004), they use a constructivist model to structure their contents in a modular way. In their work, the authors present a didactic tool called "ConstruiRV", which is characterized as a distributed virtual reality system applied to the educational environment. ConstruiRV is used in the classroom as a pedagogical resource for teachers, and since it is designed under the constructivist model, it allows students to learn the concepts of a given discipline through virtual experiences lived in the virtual environment, making learning much more lasting, in addition to cooperating in the learning of other students within the network, even if they are physically distant. Another technology similar to ConstruiRV is the Anatomy Builder VR, which is a virtual reality system that was developed by (Hwaryoung Seo Jinsi, 2017) and is intended to support the teaching of anatomy. The authors propose as the backbone of the project to seek an alternative constructivist pedagogical model for learning canine anatomy. This study demonstrated how a constructivist approach can support the teaching of anatomy using VR technology in an active, experiential way.

### 3. VIRTUAL REALITY ENVIRONMENTS WITH A CONSTRUCTIVIST APPROACH.

When we talk about constructivism in education it is inevitable to think about the construction of knowledge, "Science does not discover ready-made realities, but builds, creates and invents scenarios: in this way it tries to make sense of what happens in the world, in society, in people". (Segal, 1994). In this context, schools are there, building educational improvements. It is important to reflect on the conception of the teaching-learning process since this conception guides the methodology chosen to carry it out. From constructivism, this process can be thought of as a dialectical interaction between the teacher's knowledge and the student's knowledge, which enter into discussion, opposition, and dialogue, leading to a productive and meaningful synthesis (Granja, 2015).

One of the technologies that have had great relevance in recent years is virtual reality, commonly known for its use in video games and used to create simulations in the industry (Bell and Fogler, 1995). Virtual reality offers a high degree of immersion to the user and a variety of interaction possibilities when performing a task. In this context, virtual reality environments oriented to the educational environment help the student to abstract objects and cognitive processes that are difficult to visualize or imagine and concepts that are difficult to represent or explain verbally (Sanchez et al., 2000). They also offer the advantage of addressing areas where traditional methods have little or no presence (Bell and Fogler, 1995). According to (Requena, 2008), classroom teaching limits the interaction between teachers, students, and learning materials due to the short time allocated and

the wait between sessions for the teacher to evaluate and provide feedback on the student's results. In this sense, virtual reality environments can be a tool for students to implement their ideas and learning by obtaining feedback in a short time and can even be carried out from their homes. In this way, virtual reality environments help to encourage rapid interaction and real-time feedback, keep students active by performing activities on their own or collaboratively with other students and finally allow teachers to have tools to measure student performance and provide feedback.

## 4. CASE STUDY

This section presents the implementation of the Attention-VR virtual environment in elementary school students from two institutions in Mexico. The objective of this Virtual Reality environment is to support the development of their cognitive processes during the COVID-19 health contingency, which currently continues to limit face-to-face classroom activities and is restricted to online and blended learning activities with small groups of students, as shown in Figure 1.



Figure 1. Elementary school teachers working in small groups with children with learning disabilities in an educational institution in Mexico.

### 4.1. User Profile

The first elementary school has a population of 102 children in 6 grades. Of the total population, 92 children were identified as regular, i.e., they do not present any disability or learning problem. Of the remaining 10 children, 8 have been identified with learning problems related to Attention Deficit and Hyperactivity Disorder (ADHD) and 2 with learning problems related to Asperger's disease. In the second elementary school, a total of 14 children are reported, of which one has learning problems related to Autism Spectrum Disorder (ASD) and the rest have learning problems related to Intellectual Disability (ID). Figure 1 shows the work in the classroom under sanitary restrictions, with limited groups of students and with safety protections such as masks and face masks.

### 4.2. Technological Platform

The platform selected for Attention-VR is based on Google Cardboard (Powell et al., 2016, Pierce, 2015) and the Unity3D video game engine (Parisi, 2015) was used



for its development. This platform allows the creation of low-cost and accessible virtual reality experiences since a Cardboard-based application can be installed on most smartphones allowing parents and teachers to have access from anywhere. Figure 2 shows the implementation of Attention-VR, which consists of a low-cost virtual reality viewer, a generic Bluetooth controller, and a smartphone.



Figure 2. Elementary school teachers working in small groups with children with learning disabilities in an educational institution in Mexico.

#### 4.3. Attention-VR

The goal of Attention-VR is that the child can discover and interact with 3D objects in an immersive environment and make decisions to solve problems designed by teachers and educational experts. The instructional design of Attention-VR is focused on the development of the areas of Location, Attention, motivation, structure, following instructions, motivation, and feedback, among others. As can be seen in Table 1.

Table 1. Proposed instructional design for the Attention-VR virtual reality environment.

Area	Instruction
Location	The child is placed in time and space in a specific way. The child identifies specific important objects within the immersive environment and discards those that represent a distraction. At each moment he/she obtains feedback to accomplish the task.
Attention and feedback	
Organization and sequence	Small steps to accomplish within the virtual environment are indicated to accomplish a goal.
Motivation and structuring	Instructions are presented through elements such as audio, animations, and avatars so that the child can continue with the task.
Balance between demand and motivation	That the child is able to stay motivated in the game and at the same time is required to take a final step to achieve it. Without exceeding the demand since the child may lose motivation completely if he/she feels frustrated.
Reward and satisfaction	That the child feels satisfaction for having achieved the goal, leaving aside distractions, and reaching the happy ending, the final reward is represented in the form of the visual and auditory stimulus.

Attention-VR consists of a virtual map (see Figure 3) containing two levels. The first level consists of recognizing the virtual environment presented and collecting a series of objects, always a virtual assistant provides feedback and instructions for the child to solve the task. The second level consists of finding a treasure by avoiding obstacles, such as enemies, static distracting objects, etc. As in the first level, the child can resort to the help of avatars that always help the child to find the treasure and escape to the pirate ship. Attention-VR also has the possibility of new levels can be designed and

incorporated with new activities according to the educational needs of children. These levels allow the child to explore the environment, apply his or her judgment and knowledge for deduction to the situations presented within the virtual environment and formulate strategies for the achievement of the tasks presented within each level, seeking to build learning by having a meaningful experience.



Figure 3. Complete map with available levels of Attention-VR virtual reality environment.

Each level has 3D virtual elements that at all times assist and feedback to the child in every situation presented within the virtual environment. Among the elements, we can find (see Figure 4). *Helper avatars*. Some avatars can be consulted and provide instructions regarding the activity to be performed, offer feedback information, remind the user of the steps to follow to accomplish the task, etc. *Distracting objects*. These are 3D objects that can be animated or static, such as enemies, dynamic objects that move through the scenario but do not have any functionality related to the task to be performed. *Rewards*. These are animated or static 3D objects that can refer to the partial or total achievement of a task and *Feedback elements*. These are multimedia elements within the virtual environment such as audio that transmits indications about the task, motivational phrases during the execution time, clues, and key tips for the correct resolution of the task.



Figure 4. Interactive user support elements within the Attention-VR virtual reality environment.

#### 4.4. Results

The implementation of Attention-VR was carried out on 25 children from two primary education institutions of different school grades, of which 8 children have learning disabilities associated with ADHD. With the help of parents and teachers, Attention-VR was installed on the smartphones of each of them and they were provided with a virtual reality viewer based on Google Cardboard.

Throughout, the children were supervised by an adult. None of the children experienced any discomfort such as dizziness or vomiting (LaViola Jr, 2000, Kolasinski, 1995). The first preliminary results show that the minimum duration to complete the tasks was 6 minutes and the maximum was 15 minutes. In the end, some of the opinions obtained by the children indicated some suggestions, such as increasing the difficulty, no monsters, an airplane, horses. And positive comments in which the children liked that they were congratulated every time they finished the indicated tasks, that the system talked to them at all times, among others.

## 5. CONCLUSIONS AND FUTURE WORKS

Virtual reality is a technology that is increasingly positioned as an alternative to support education by facilitating teachers and students to improve the teaching-learning processes. A constructivist approach allows that through virtual reality environments students can carry out their learning process interactively and dynamically in which knowledge from past experiences can be applied to solve new problems and new meaningful experiences. And in the case of teachers, they can count on tools for the generation of new educational content and mechanisms for student follow-up. The importance of the constructivist model is that the student takes charge under the guidance of his teacher, building collaboratively and nurturing meaningful learning. As future work, we are working on the definition of new levels to be incorporated in Attention-VR based on the design and needs of teachers and education experts. We are also working on the design and incorporation of evaluation instruments to measure the user's experience, perception, and performance within the virtual environment.

## 6. ACKNOWLEDGMENTS

The authors would like to thank CONACYT for the support provided for this research, the "José María Morelos y Pavón Institute", Ojuelos de Jalisco, Mexico, and the "Regular Education Support Services Unit No. 19" (USAER-19), Aguascalientes, Mexico, for their collaboration and for providing the personnel and resources to carry out the case study, and all the people who participated in the communities of Vaquerías and Matancillas, Jalisco, Mexico. This work is dedicated to the memory of the student Gerardo Ortiz Aguiñaga (1995- 2020) student of CIMAT Zacatecas Mexico and to his passion and dedication for the development of this research, for which we honor his memory.

## 7. REFERENCES

- Bell, J. T., & Fogler, H. S. (1995, June). The investigation and application of virtual reality as an educational tool. In *Proceedings of the American Society for Engineering Education Annual Conference* (pp. 1718-1728).
- Burgess, S., & Sievertsen, H. H. (2020). Schools, skills, and learning: The impact of COVID-19 on education.
- Collins, J., Regenbrecht, H., & Langlotz, T. Back to the future: Constructivism learning in virtual reality. *IEEE International Symposium on Mixed and Augmented Reality Adjunct*, 2018.
- Daniel, J. (2020). Education and the COVID-19 pandemic. *Prospects*, 49(1), 91-96.
- Dean, A. (2002). Telelearning: Invention, Innovation Implications: Towards a Manifesto. *Australasian Journal of Educational Technology*, 5(2), 1-12.
- Granja, D. O. (2015). El constructivismo como teoría y método de enseñanza. *Sophia*, (19), 93-110.
- Kolasinski, E. M. (1995). Simulator sickness in virtual environments (Vol. 1027). US Army Research Institute for the Behavioral and Social Sciences.
- LaViola Jr, J. J. (2000). A discussion of cybersickness in virtual environments. *ACM Sigchi Bulletin*, 32(1), 47-56.
- Melchiori Peruzza, A. P. P., & Zuffo, M. K. (2004, June). ConstruiRV: constructing knowledge using the virtual reality. In *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry* (pp. 180-183).
- Parisi, T. (2015). Learning virtual reality: developing immersive experiences and applications for desktop, web, and mobile. "O'Reilly Media, Inc."
- Pierce, D. (2015). Google cardboard is vr's gateway drug. *Wired*. Retrieved, 17.
- Powell, W., Powell, V., Brown, P., Cook, M., & Uddin, J. (2016, March). Getting around in google cardboard—exploring navigation preferences with low-cost mobile VR. In *2016 IEEE 2nd Workshop on Everyday Virtual Reality (WEVR)* (pp. 5-8). IEEE.
- Pulgar, J. L. (2005). Evaluación del aprendizaje no formal. *Recursos prácticos para el profesorado*.
- Requena, S. H. (2008). El modelo constructivista con las nuevas tecnologías: aplicado en el proceso de aprendizaje. *RUSC. Universities and Knowledge Society Journal*, 5(2), 26-35.
- Sanchez, A., Barreiro, J. M., & Maojo, V. (2000). Design of virtual reality systems for education: A cognitive approach. *Education and information technologies*, 5(4), 345-362.
- Segal, L. (1994). *Sonar la realizad: El constructivismo de Heinz Von Foerster*. Paidós.
- Seo, J. H., Smith, B. M., Cook, M., Malone, E., Pine, M., Leal, S., ... & Suh, J. (2017). Anatomy builder vr: Applying a constructive learning method in the virtual reality canine skeletal system. *IEEE Virtual Reality (VR)*, 18(22):399-400, 2017.
- Shantz, D. (1995). Teacher education: Teaching innovation or providing an apprenticeship?. *Education*, 115(3), 339-344.
- Tuncel, İ., & Bahtiyar, A. (2015). A case study on constructivist learning environment in content knowledge courses in science teaching. *Procedia-Social and Behavioral Sciences*, 174, 3178-3185.



# **Computational Thinking and Evaluation**

# A Preliminary, Systematic Review of Teaching and Learning Computational Thinking in Early Childhood Education

Anika SAXENA<sup>1\*</sup>, Gary WONG<sup>2</sup>

<sup>1,2</sup> Faculty of Education, The University of Hong Kong, Hong Kong.  
anikareena@gmail.com, wongkwg@hku.hk

## ABSTRACT

Computational thinking (CT) and its implementation in the K-12 curriculum have recently become important topics in education and research worldwide. Due to the burgeoning interest in CT in education, there has been a marked increase in empirical research in this area. Many researchers suggest that CT should be introduced and fostered early in education, as it is a precursor of academic success. However, there is little evidence from research that sums up empirical research findings to give further teaching and learning directions specific to early childhood education (ECE). Following the pre-analysis, 32 articles were selected and included in the study. Content analysis was applied to determine and evaluate the shared codes and themes related to the findings. The results demonstrate that ECE practitioners should consider incorporating CT concepts with core subject areas following an integrated teaching and learning approach in ECE, using various developmentally appropriate pedagogical practices.

## KEYWORDS

computational thinking, systematic review, early childhood education, teaching/learning strategies

## 1. INTRODUCTION

In early childhood education (ECE), computational thinking (CT) refers to developing behavioural attributes and skills related to patterning, sequencing, planning, and processing (Brennan and Resnick, 2012; Berland and Wilensky, 2015; Wilson and Moffat, 2010). Numerous studies have been published to propose teaching and learning of CT in the early years.

The published literature contains a broad range of definitions for CT. For example, Papert (1980) first described computational thinking as “how children develop procedural thinking through computer programming”. Jeannette Wing (2006) claimed that CT is a fundamental skill for everyone and should be added to every child's ability. Kazimoglu, Kiernan, Bacon, and Mackinnon, (2012), define CT as an approach to problem-solving, systems design and understanding human behaviours based on computer-based concepts, and consider CT to be a skill that requires the use of computer systems to solve problems in any discipline.

In contrast, some researchers advocated that unplugged activities can be used to develop CT skills, leading to developing similar skill sets without using computers. Children can apply their concrete skills more abstractly in later stages, with computational concepts (Rial-Fernández & Santacruz-Valencia, 2019; Saxena et al., 2020). In early years education, CT skills should be employed in play,

discovery, and creative activities, during which children can practice their abilities to plan, sequence, and logically connect their ideas (Rehmat, Ehsan and Cardella, 2020). By enhancing their creations, children also review and make authentic improvements, known as debugging, in computational aspects and core to CT (Lavigne et al., 2020). CT also involves evaluating problems, constructing ideas, and designing projects (Bers et al., 2014) and (Komis et al., 2016). The overall goal of implementing CT skills in the curriculum is to develop thinking skills and their potential application in various fields. Hence, CT is not just limited to coding, robotics, and mathematical processing (Bers et al., 2002).

Although many discussions and initiatives have been taken in recent years, there is a lack of concrete, research-based evidence or guidelines regarding exposure and age-appropriate inclusion of CT in ECE. There is a lack of empirical evidence demonstrating how early educators should align these early learners' skills to succeed. With the expansion of research in this area, it is necessary to synthesize scientific evidence from quality, published studies to enhance our understanding of developmentally appropriate CT practice in early years education and to plug the potential gaps in this research area.

### 1.1. Computational thinking in early childhood

Computational thinking is becoming an essential skill in the 21st century. Bers (2019) claimed CT as a new literacy skill in ECE classrooms, stating that it should be taught as another vital literacy area. Many researchers addressed the need and claimed that the rationale for supporting the introduction of computational thinking in the early years should not be focused on creating a future workforce, but the future citizenry. Due to the lack of a standard consensus definition of CT; it is widely understood as a logical and algorithmic way of thinking for problem-solving. Papert (1987) describes CT as the combination of critical thinking and computing; potentially incorporating and enhancing skills for problem-solving, communication, collaboration, creativity, and computation.

On the other hand, ECE practitioners find it challenging to create hands-on opportunities to develop in their ECE learning environment. Early childhood is a critical time during which young children play, grow, and explore the world around them. This systematic review will address the needs of practitioners. It will investigate the feasibility, efficiency, and potential pedagogical approaches for developmentally appropriate CT development for preschool children. Moreover, we aim to study the impact of programming activities on particular CT skills.

ECE educators' increasing interest and needs call for a systematic review of the previous studies that can serve as a "pathfinder" for future research. Few representative literature reviews in ECE have been found. Çetin, M., & Demircan, H. Ö. (2020) presented a literature review to support computing education and its integration through STEM in the early years. Manches & Plowman (2017) reported the need to consider the conceptual thinking that underlies computational thinking, specifically in the early childhood education sector. Other researchers (Isnaini, Budiyo, and Widiastuti, 2019) and (Umam, Budiyo, & Rahmawati (2019) presented a literature review and identified the potential contribution of robotics as an educational tool. Several systematic reviews, such as Xia, L., & Zhong, B. (2018) presented systematic reviews focusing on either K-12 education or specific learning tools only. Thus, there is a need to conduct a systematic review of review papers following the framework from Kitchenham et al. (2009) to ensure quality and content that is specific to CT in ECE, including children from birth to eight years old. This systematic literature review will allow us to address some of CT's critical issues in ECE and lead us to answer the following research questions.

RQ1: How have people studied CT in ECE?

RQ2: What is the computational concept, perspectives and tools to implement CT in ECE?

## 2. Methodology

### 2.1. Research Design

A systematic review methodology was employed in this study. Following the guidelines set out by Kitchenham et al. (2009), a systematic review was initiated to evaluate research literature, using systematic and rigorous methods. Lam and Kennedy, (2005), identified systematic review methodology as the most robust approach that provides a mechanism to analyse evidence-based research among a range of published studies. This current, systematic literature review is focused on studies of computational thinking in early childhood educational settings published in the last decade. Only peer-reviewed articles (rather than project descriptions, analyses of programs, guidelines for practice, and reports or conference papers), are included in this review.

### 2.2. Procedures

This systematic review adhered to the guidelines laid out by Kitchenham et al. (2009), and was performed initially following the search string of keywords ("computational thinking" OR "robotics" OR "coding" OR "programming") AND ("Early Childhood Education" OR "preschool" OR "Kindergarten" OR "young learners") for peer-reviewed studies and carried out in the ScienceDirect, ERIC, SCOPUS, ACM, Springer Link, IEEE Xplore databases. These databases were searched without any constraints on the publication date. The search resulted in 229 studies (33 webs of science, five in ScienceDirect, nine in ERIC, 52 in SCOPUS, five in IEEE Xplore, 21 in SpringerLink, and 104 in ACM) on November 15th, 2020. Duplicates, inaccessible studies, and publications not written in the English language were excluded from this collection.

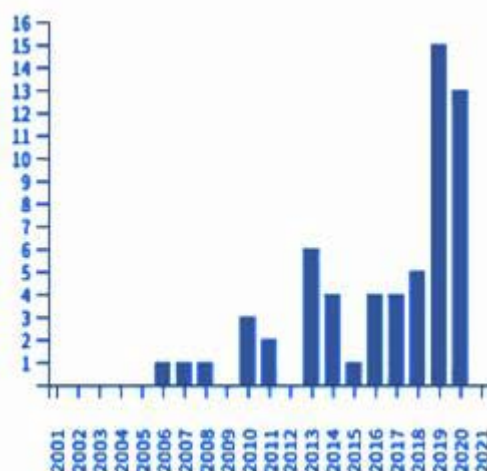


Figure 1. Distribution of articles about CT in ECE by year

Figure 1 shows the distribution of articles about CT by year, particularly in the ECE sector. It is evident that research has developed its strength over the years and offers a recent increase in publications.

Following the procedures mentioned in figure 2. The researchers examined crucial points of the study and their relationships with each other.



Figure 2. Data collection and analysis process

The remaining studies' abstracts were screened, and both empirical and nonempirical studies were included if they addressed the inclusion criteria detailed in Table 1 below.

Table 1: Inclusion Criteria (IC)

IC1	Paper reports of CT studies in ECE. (RQ1)
IC2	Paper aimed at the Computational Concept, Perspectives, and Tools to implement CT in ECE. (RQ2)
IC3	Paper aimed at focusing on teaching and learning of CT in ECE. (RQ2)

After analysing the titles and abstracts of these papers concerning the above inclusion criteria, 32 studies were selected as relevant to the current research.

### 2.3. Data Analysis

The data collected were analysed using content analysis. The next step was to identify commonalities and differences amongst all the studies, based on research questions. The standard codes and themes related to the findings were determined and evaluated during the analysis. Then, the categories were revised based on the consensus among the

researchers. Tables were created for themes, including the frequency values, and later converted into charts for visualisation.

### 3. FINDINGS

This section details the analysis of selected papers (32) from different perspectives, such as sample groups, duration of CT intervention, pedagogical connection, and teaching and learning of CT in ECE. The research studies are summarised and synthesised into four different categories based on research questions.

#### 3.1. How have people studied CT in ECE?

##### 3.1.1. Sample Size

The selected papers' sample size shows that most of the studies have been done with either 20-40 participants, or with over 80 participants, as shown in figure 3. This indicates that with growing awareness of CT in ECE, several studies are now focused on interventions with a broader range of participants to see the results more precisely, leading to more appropriate, age-specific intervention.

Figure 3. Sample size of

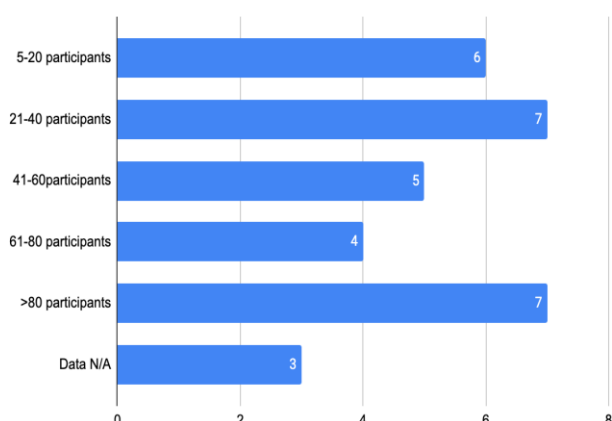


Figure 3. Sample size of empirical studies

##### 3.1.2. Sample groups within early childhood

Amongst 32 papers, 30 described age or learning level of participants. Few studies have focused on kindergarten to grade 2 students. The frequency of the sample level or age groups is shown in figure 4.

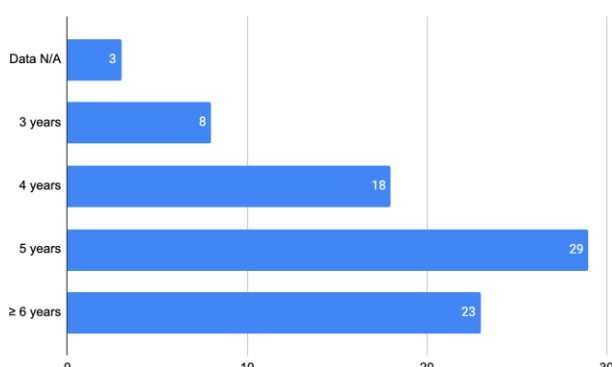


Figure 4. Age-specific sample groups of selected studies

The largest sample group is 5-6 year olds, and only a few studies were conducted with 3-5 year old children. However,

few researchers included the duration of their studies, as seen in figure 5.

##### 3.1.3. Duration of empirical studies

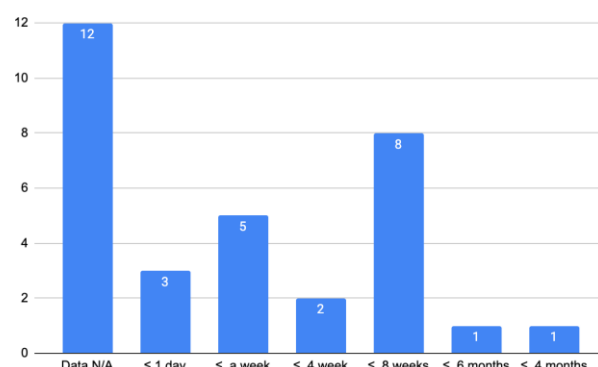


Figure 5. Duration of empirical studies.

Twelve selected studies failed to clearly and explicitly explain the duration intervention. The other studies suggested that most studies are short term, being conducted for less than four weeks, either in summer camps or during after school programs. The number of studies undertaken in regular classrooms remains limited.

##### 3.1.4. Research methods used

The most popular research design amongst 32 research papers is the non-experimental design, followed by quasi-experimental and experimental design (Figure 6). The nonexperimental research design tends to be the closest to real life situations. For an experimental design to be classified as a true experimental design, participant groups need to be selected through random assignment. If chosen through random sampling, this is considered to be quasi-experimental. (Trochim & Donnelly, 2006)

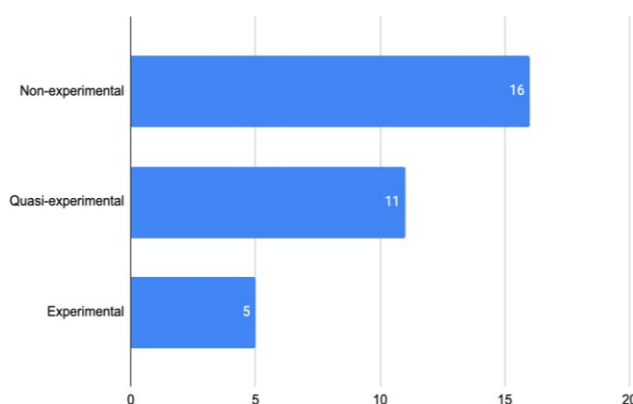


Figure 6. Research methods used in empirical studies.

##### 3.1.5. Data gathering techniques used in these studies.

The most frequently used data-gathering techniques amongst these studies are the learning assessment in pre-tests and post-tests, and classroom observations. Some studies adopted classroom observations and gathered the quality of teaching and learning during experiments, and few studies used it as a tool to obtain more comprehensive qualitative analysis. Students' surveys in an age-appropriate manner, e.g. choosing smiles, have been used. Teacher and parent interviews were conducted in a few studies. Videos,

photographs, and artefacts have also been used for analysis in some studies (see figure 7).

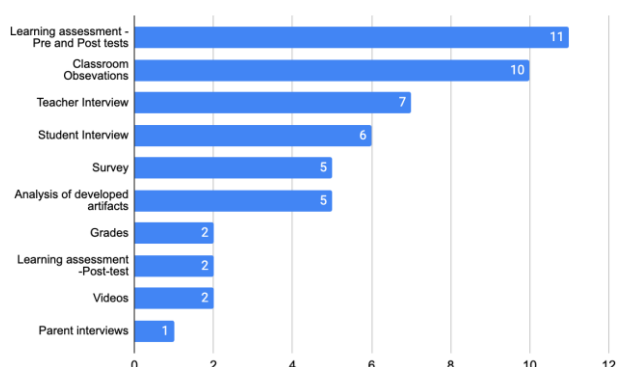


Figure 7. Data gathering techniques used in these studies.

The next section of this paper will analyse data related to conceptual understanding, and perspectives with the available tools used in these empirical studies.

### 3.2. What are the computational concepts, practices, perspectives, and tools needed to implement CT in ECE?

The selected studies have been examined following CT skills in Brennan and Resnick's (2012) framework. CT concepts, and perspectives are described in empirical studies. Most of the studies labelled automation or coding as the emerging concept (see figure 8). However, sequencing, problem-solving, control of flow, decomposition, and debugging are also used in some studies.

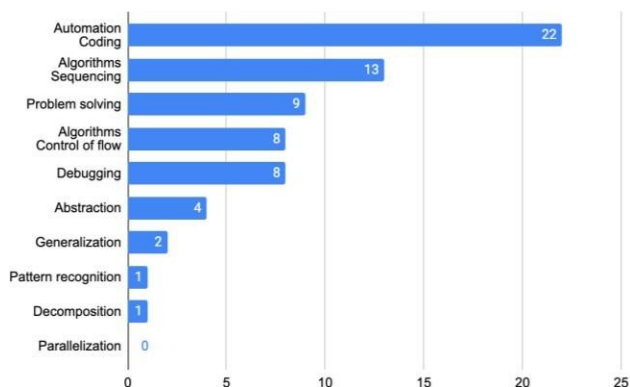


Figure 8. CT Computational concepts used in these studies.

The linkage of the CT conceptual understanding of the competence and skill developed in each area will be explored in the discussion section. It has been observed that in the majority of studies there was regular use of unplugged activities to support plugged experience. It is clearly stated in the chart below (figure 9) that ScratchJr, KIBO and Bee-Bots are the most dominant plugged tools in the ECE sector. In contrast, it is clear from figure 10 that other methods and plugged tools have been used to support young learners' transition from concrete to abstract learning. Tangible computing, CHERP Blocks from TangibleK, and many other regular preschool activities related to sequencing, music and movement, puzzles, matching games, picture story, card games and even making bracelets to learn to sequence, and debugging have been discussed. As Brennan and Resnick (2012) mentioned in their research, concepts

are more comfortable identifying and accessing than practices and perspectives.

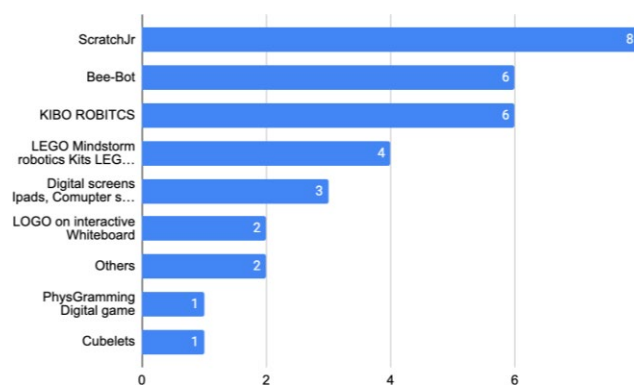


Figure 9. CT Computational practices (Plugged)

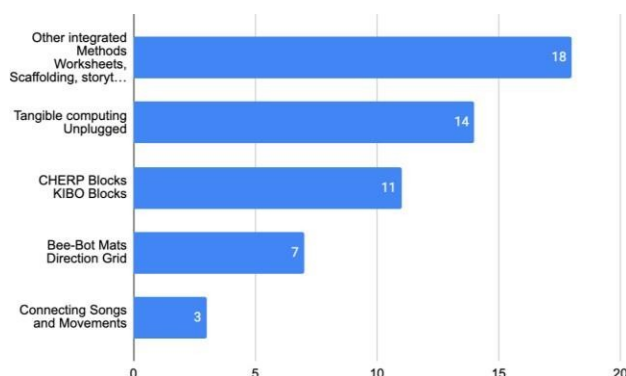


Figure 10. CT Computational practices (Unplugged)

### 3.3. Computational perspectives used in these studies.

CT perspectives refer to the evolving student' understanding of their relationship to others and the technological world. These perspectives include but are not limited to expressing, questioning, and connecting. Studies indicated that students developed a personal interest when they were engaged in design and engineering. Several studies showed that students also learned to collaborate with peers and develop creative thinking skills.

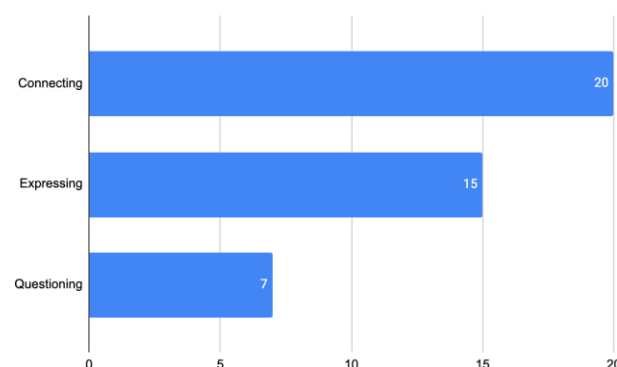


Figure 11. CT perspectives described in these studies.

## 4. DISCUSSION

### 4.1 What are the implications for teaching and learning indicated by these empirical studies?

In this section, we generalise the viewpoints of the 32 papers based on RQ.3 and make a comparison with existing studies. Further research perspectives will be discussed.

In this review, we analyse scholarly articles on CT for teaching and learning in ECE. By doing so, we would like to investigate the impact of those studies. The themes generated regarding teaching and learning will be discussed under three subcategories.

### 4.2 A constructivist view of teaching and learning.

The vital aspect mentioned in several selected empirical research studies is the concept of learning-by-doing following Papert's (1980) idea that children should be allowed to work with tangible objects to promote their computational thinking, and defined as constructionism. Currently, the CT learning process is widely used at various levels of education, starting from preschool (Bers 2002; Cejka et al. 2006; Kazakoff and Bers 2012, 2014; García-Valcárcel-Muñoz-Repiso and Caballero-González Y. A. 2019), and in multiple fields and various dimensions: science, technology, engineering and mathematics (STEM) (Sullivan et al. 2016), engineering (Bers, M. U.; González-González and Armas-Torres 2019), and other branches of STEM (Sullivan, A., & Bers, M. U. (2018).

Uurlings, Coppens and Borghans (2019) described the relationship between the development of computational thinking concepts and carefully selected and purposefully developed educators. The educator's role is to provide opportunities for young learners for hands-on learning experiences. Bers, (2018), in her recent research paper, mentioned that developing CT competencies through hands-on activities will have a positive impact on the child. Many opportunities should be given for students to ask questions, define problems, develop and use models, plan and carry out investigations, analyze and interpret data using mathematical and algorithmic thinking., construction of applications and designing solutions, engaging in their own learning process, communicating and collaborating with peers.

During the first two years of preschool education, it is necessary to work with simple tools González et al. (2017). Solutions for early-stage learning are needed, and teachers have to use a playful approach to open up the world of math, science, and language skills. It is required to foster the love for discovery and investigation in young students and to develop the social and emotional skills to be prepared for a lifetime of successful learning.

In general, the proposed educational activities will emphasise the importance of having pre-determined goals, and they will stimulate logic and analysis capacity. The requested continuous learning by doing.

### 4.3 An integrated approach to teach CT in ECE.

Early years curricula all globally follow an integrated approach for teaching and learning. Why should CT be taught as a stand-alone subject area? The second implication projected out through this review of published papers is that

we need to teach CT skills by taking the approach to integrate this teaching with regular art, music, mathematics, and science instruction. This allows students to develop a deeper understanding of the core subject area curriculum while also facilitating the development of students' CT practices and skills (Bers 2002; Cejka et al. 2006; Kazakoff and Bers 2012, 2014; García-Valcárcel-Muñoz-Repiso and Caballero-González Y. A. 2019, Rehmat et al. (2020). Researchers and early years frontline practitioners should support the view that 'play' is an essential medium for learning in early years education and that it is part of a system that contributes to embrace a cross-curriculum, integrated approach that recognises the physical, cognitive, linguistic, and social and emotional aspects of learning (Plowman & Stephen, 2005).

### 4.4 Pedagogical approaches to teach CT in ECE.

The selected empirical research also suggests that ECE children should be exposed to CT concepts following various pedagogical approaches. Kazakoff, Sullivan and Bers (2013); Lee and Junoh (2019; Portelance, Strawhacker and Bers (2016) noted in their studies that children should initially be given unplugged activities that do not involve computers or computer programming and, for example, be given a practical example of tasks with algorithm designs, including a detailed step-by-step instruction set for solving a problem or completing a task. The order of experiences should move from (1) unplugged, (2) tinkering, (3) making, and (4) remixing, providing developmentally appropriate tasks for young children Rial-Fernández and Santacruz-Valencia (2019). Roussou and Rangoussi (2019), suggested that ECE classroom CT should be incorporated in a playful way, suitable for the development of children, leading to notable enhancement of the CT skills.

## 5. CONCLUSION

The reviewed literature added a new dimension in teaching and learning CT in ECE towards a developmentally appropriate integrated approach. It is crucial that we consider 'play' to be an essential element of teaching young children, focus on teaching CT conceptual understanding from early on, and that we choose the right tools for the job.

Suggestions for future studies can be divided into three groups. First, there is a need to understand developmentally appropriate practice in early years education through research in the cognitive domain. Secondly, more practical examples from evidence-based research should inform this field of study and fill it with practical implementation plans to enhance the teaching and learning of CT in ECE. Thirdly, there is a need to develop authentic CT assessment tools to access CT skills development in ECE. When children learn to code, they should have prepared with a programmer to code and create an algorithm.

## 6. REFERENCES

- Berland, M., & Wilensky, U. (2015). Comparing virtual and physical robotics environments for supporting complex systems and computational thinking. *Journal of Science Education and Technology*, 24(5), 628-647.
- Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering:



- Exploration of an early childhood robotics curriculum. *Computers & Education*, 72, 145-157.
- Bers, M. U., Ponte, I., Juelich, C., Viera, A., & Schenker, J. (2002). Teachers as designers: Integrating robotics in early childhood education. *Information technology in childhood education annual*, 2002(1), 123-145.
- Bers, M. U., Flannery, L., Kazakoff, E. R., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education*, 72, 145-157.
- Bers, Marina U., Carina González-González, and M<sup>a</sup> Belén Armas-Torres. "Coding as a playground: Promoting positive learning experiences in childhood classrooms." *Computers & Education* 138 (2019): 130-145.
- Bers, M. U. (2019). Coding as another language: a pedagogical approach for teaching computer science in early childhood. *Journal of Computers in Education*, 6(4), 499-528.
- Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada* (Vol. 1, p. 25).
- Cejka, E., Rogers, C., & Portsmore, M. (2006). Kindergarten robotics: Using robotics to motivate math, science, and engineering literacy in elementary school. *International Journal of Engineering Education*, 22(4), 711.
- Çetin, M., & Demircan, H. Ö. (2020). Empowering technology and engineering for STEM education through programming robots: a systematic literature review. *Early Child Development and Care*, 190(9), 1323-1335.
- García-Valcárcel-Muñoz-Repiso, A., & Caballero-González, Y. A. (2019). Robotics to develop computational thinking in early Childhood Education. *Comunicar. Media Education Research Journal*, 27(1).
- González, Y. A. C., & Muñoz-Repiso, A. G. V. (2017, November). Educational robotics for the formation of programming skills and computational thinking in childish. In *2017 International Symposium on Computers in Education (SIIE)* (pp. 1-5). IEEE.
- Isnaini, R., Budiyo, C., & Widiastuti, I. (2019, December). Robotics-based learning to support computational thinking skills in early childhood. In *AIP Conference Proceedings* (Vol. 2194, No. 1, p. 020044). AIP Publishing LLC.
- Kazakoff, E., & Bers, M. (2012). Programming in a robotics context in the kindergarten classroom: The impact on sequencing skills. *Journal of Educational Multimedia and Hypermedia*, 21(4), 371-391.
- Kazimoglu, C., Kiernan, M., Bacon, L., & Mackinnon, L. (2012). A serious game for developing computational thinking and learning introductory computer programming. *Procedia-Social and Behavioral Sciences*, 47, 1991-1999.
- Kitchenham, B., Brereton, O. P., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering—a systematic literature review. *Information and software technology*, 51(1), 7-15.
- Komis, V., Romero, M., & Misirli, A. (2016, November). A scenario-based approach for designing educational robotics activities for co-creative problem solving. In *International Conference EduRobotics 2016* (pp. 158-169). Springer, Cham.
- Lam, R. W., & Kennedy, S. H. (2005). Using meta-analysis to evaluate evidence: practical tips and traps. *The Canadian Journal of Psychiatry*, 50(3), 167-174.
- Lavigne, H. J., Lewis-Presser, A., & Rosenfeld, D. (2020). An exploratory approach for investigating the integration of computational thinking and mathematics for preschool children. *Journal of Digital Learning in Teacher Education*, 36(1), 63-77.
- Lee, J., & Junoh, J. (2019). Implementing Unplugged Coding Activities in Early Childhood Classrooms. *Early Childhood Education Journal*, 47(6), 709-716.
- Manches, A., & Plowman, L. (2017). Computing education in children's early years: A call for debate. *British Journal of Educational Technology*, 48(1), 191-201.
- Papert, S. (1980). "Mindstorms" Children. *Computers and powerful ideas*.
- Papert, S. (1987). Computer criticism vs. technocentric thinking. *Educational Researcher* (Vol. 16, No. 1) January/February 1987.
- Portelance, D. J., Strawhacker, A. L., & Bers, M. U. (2016). Constructing the ScratchJr programming language in the early childhood classroom. *International Journal of Technology and Design Education*, 26(4), 489-504.
- Rehmat, A. P., Ehsan, H., & Cardella, M. E. (2020). Instructional strategies to promote computational thinking for young learners. *Journal of Digital Learning in Teacher Education*, 36(1), 46-62.
- Rial-Fernández, B., & Santacruz-Valencia, L. P. (2019). The Teaching of Programming is not the Future but the Present. In *2019 International Symposium on Computers in Education (SIIE)* (pp. 1-6). IEEE.
- Roussou, E., & Rangoussi, M. (2019, April). On the use of robotics for the development of computational thinking in kindergarten: educational intervention and evaluation. In *International Conference on Robotics and Education RiE 2017* (pp. 31-44). Springer, Cham.
- Saxena, A., Lo, C. K., Hew, K. F., & Wong, G. K. W. (2020). Designing unplugged and plugged activities to cultivate computational thinking: An exploratory study in early childhood education. *Asia-Pacific Education Researcher*, 29(1), 55-66.
- Sullivan, A., & Bers, M. U. (2016). Robotics in the early childhood classroom: learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade. *International Journal of Technology and Design Education*, 26(1), 3-20.
- Trochim, W. M., & Donnelly, J. P. (2010). Research methods knowledge base. 2006. Internet WWW page, at

- URL:< [http://www. socialresearchmethods. net/kb/](http://www.socialresearchmethods.net/kb/)>(version current as of October 20, 2006).
- Umam, M. U. K., Budiyanto, C., & Rahmawati, A. (2019, December). Literature review of robotics learning devices to facilitate the development of computational thinking in early childhood. In *AIP Conference Proceedings* (Vol. 2194, No. 1, p. 020133). AIP Publishing LLC.
- Urlings, C. C., Coppens, K. M., & Borghans, L. (2019). Measurement of Executive Functioning Using a Playful Robot in Kindergarten. *Computers in the Schools*, 36(4), 255-273.
- Wilson, A., & Moffat, D. C. (2010, September). Evaluating Scratch to Introduce Younger Schoolchildren to Programming. In *PPIG* (Vol. 1, No. 1, pp. 1-12).
- Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35.
- Xia, L., & Zhong, B. (2018). A systematic review on teaching and learning robotics content knowledge in K-12. *Computers & Education*, 127, 267

ORCID Anika Saxena <https://orcid.org/0000-0002-8527-6238>

ORCID Gary Ka Wai Wong <https://orcid.org/0000-0003-1269-0734>

# **Computational Thinking and Non-formal Learning**

# Bringing Physical Computing to an Underserved Community in an Informal Learning Space

Chin-Lee KER<sup>1\*</sup>, Bimlesh WADHWA<sup>2</sup>, Peter, Sen-Kee SEOW<sup>3</sup>, Chee-Kit LOOI<sup>4</sup>

<sup>2</sup> National University of Singapore, Singapore

<sup>1, 3, 4</sup> National Institute of Education, Nanyang Technological University, Singapore

chinlee.ker@nie.edu.sg, bimlesh@nus.edu.sg, peter.seow@nie.edu.sg, cheekit.looi@nie.edu.sg

## ABSTRACT

This study investigates how underserved children in the community develop Computational Thinking skills through learning physical computing with the support from older tutor volunteers. The children learned to construct physical computing projects by learning to code the micro:bit, and using various input sensors and controlling output devices. We observed the students and their interaction with the mentors to understand how they develop their Computational Thinking skills as they construct the projects. From our findings, learning with tutors can provide the support in developing Computational Thinking skills in the children.

## KEYWORDS

Computational Thinking, Physical Computing, Computing Education, Out-of-school Learning

## 1. INTRODUCTION

The wide-spread availability of devices like the micro:bit has brought physical computing into the mainstream of computing education. In Singapore, the micro:bit has gained popularity in schools buoyed by the Ministry of Education (MOE) initiative, for all primary school students to have 10 hours of coding in school. An important imperative of such program is to develop students' Computational Thinking (CT) skills as they participate in such activities. Students participate in coding activities which are run by external trainers engaged by the schools. From our previous study (Seow, Wadhwa, Lim & Looi, 2020), we found that such training programs may not help students to develop CT skills as students do not explicitly engage in using cognitive skills such as abstraction, or practice such algorithmic or system building.

In this study, we investigate how students develop CT skills as they participate in activities in an informal context outside schools. They learn to code the micro:bit and build computing projects supported by mentors during one hour sessions spanning over 8 weekends.

In our study, we were guided by the following questions:

1. How do we design a physical computing program for out-of-school context?
2. What are the roles of mentors in developing CT skills for students?
3. What CT skills do students develop as they participate in computing activities outside school?

## 2. RELATED WORK

### 2.1. Physical Computing and Computational Thinking

Learning physical computing is an emerging approach to learn computing. It teaches students about coding and CT through hands-on activities with sensors using small

computing boards like the micro:bit (Rogers et al., 2017). The project-approach to physical computing, an often-used pedagogy in schools, serves as an open-ended exploratory approach to examine the CT competencies that students should learn. We observed that among many other factors that inhibit the development of CT skills, the inherent complexity of problem and solution space could overwhelm students. Additionally, the cognitive load in designing and developing their solutions could also hinder them in the development of CT skills.

Papert (1972) described CT as a mental skill a child can develop from practicing programming. Wing (2008) catalysed a 'CT for all' movement. However, CT definition has been debated, and it is often argued if CT makes better problem solvers or if practice of coding can help develop CT skills, with claims that everyone can benefit by CT not yet being fully substantiated by studies (Nardelli, 2019). In this study, we have adopted the CT definitions proposed by Digital Promise as it succinctly describes cognitive processes and computation practices (Digital Promise, 2020).

### 2.2. Learning Computing in School

In 2020, Singapore has made it mandatory for all primary school children to undergo a 10-hour coding program. The initiative is to help students to develop an appreciation for CT and coding concepts. For the implementation of the program, schools often engage the services of external vendors to run workshops for students across the level. This is a pragmatic reason considering that schools do not have manpower resources to run the workshops for large numbers of students across the level. Furthermore, schools may not have teachers that have the knowledge or experience to conduct the workshops for a large group of students. The vendors can offer various programs to introduce coding such as Scratch and Micro:bit. Whilst the intent is to help students to learn about CT through coding, there is much more emphasis in getting the students to code than developing CT skills. Students are not involved in thought processes of formulating problems that can be solved computationally which is the essence of CT. From our prior study (Seow et al., 2020), such workshop program may not help students to develop CT skills as students do not explicitly engage in using cognitive skills such as abstraction, or practice such algorithmic or system building.

### 2.3. Leveraging on Community resources

Community resources, e.g., volunteers, can play the part of an informal educator. These educators come up with programs, choose and modify training material, as well as expedite learning tasks (Fritz, Karmazin, Barbuto Jr & Burrow, 2003). The pedagogies adopted by volunteer educators play a critical role in influencing how good the

learning program is and the kind of outcome they have on young students (Worker, 2017). Volunteer educators (or mentors) prominently affect the organisation of the learning environment (Borden, Schlomer & Wiggs, 2011; Evans, Ching & Ballard, 2012). They can also encourage children to participate willingly by making activities fun (Worker, 2017). In a study by Worker (2017), volunteer educators used a range of pedagogies when students designed and constructed a device. These volunteers used targeted questions and gave precise design suggestions. In another study by Benander and Benander (2008), volunteers performed active demonstration to their students. This resulted in the students grasping a better understanding of the computing concept. The use of class time was productive as well.

## 2.4. Informal Mentoring

According to Hidi and Renninger (2006), mentors play a crucial part in coming up with learning experiences to spark and sustain interest in students. They push students to learn even more, link the process of learning to their individual identities, as well as transform a situational, fleeting interest into a personal one. Many youths have an informal mentor (Beam, Chen & Greenberger, 2002) who can be a major part of their own lives (Klaw, Rhodes & Fitzgerald, 2003). In informal mentoring, it is important for mentors to communicate effectively with their mentees and have a suitable character that matches that of their mentee (Norling, 1995; Pisimisi & Ioannides, 2005; Townsend, 2002).

# 3. DESCRIPTION OF STUDY

## 3.1 Participants (Students, mentors)

In this study, students were part of a community student day care centre. The learning setting was outside of the classroom, instead of in a formal primary school or institution. Students consisted mainly of lower SES students, ranging from Primary 4 to 6 (10 to 12-year-olds). Our volunteer mentors, also known as tutors, were 18-year-old students from Junior Colleges. Although only a few of them had little experience in computing, not everyone was familiar with the micro:bit or had any form of computing experience.

## 3.2 Data Collection (Methods)

Observations of the students and their interaction with the tutors were made for all lessons. On the final weeks of the workshops, we selected 2 groups for more in-depth observation. For these groups, we observed the interaction between the students and tutors, and recorded their discussion and implementation of the project. For the group project, tutors wrote down field notes on how the students behaved in terms of 4 dimensions—say, write, do, and make. Lastly, photos, audio and video recordings of the students were collected.

We analysed the CT skills of students in terms of (i) cognitive processes and (ii) practices based on the definition of CT by Digital Promise (2020). The 4 types of cognitive processes are essential CT skills:

- (i) *Abstraction* is finding out and illustrating the most relevant portions of a complicated structure. It includes forming a procedure or categorising concepts.
- (ii) *Decomposition* is getting down to the basics and disentangling the whole structure into small blocks, enabling simplicity and clarity.
- (iii) *Pattern recognition* is noticing the associations or interconnections between pieces of information. It sieves out interactions between a cause and effect, enabling one to foresee or expect what would happen next time.
- (iv) *Testing and debugging* refers to making sure that the newly created system is working. *Testing* is double-checking if the intended process runs smoothly. *Debugging* is fixing the issues discovered through testing that were deemed faulty or wrong.

The 4 types of computational practices are:

- (i) *Creating algorithms* refers to coding the program or constructing a chart that depicts key ideas, and procedures to solve a problem.
- (ii) *Working with data* refers to collating, working with figures, organising, making sense of the numbers and presenting it meaningfully.
- (iii) *Understanding systems* means simplifying and comprehending complicated plans by applying the practices of – *abstraction, decomposition, pattern recognition, and testing and debugging*.
- (iv) *Creating computational models* involves piecing together the codes, application, information and everything that depicts the comprehension of the system.

## 3.3 Activity: The 8-week programme for students

We started off with a basic training session for tutors, introducing them to the micro:bit. It was followed by the 8-week student programme. There was one other intermediate training session at the end of lesson 4, as seen in *Table 1*.

The first 3 student sessions were on basic micro:bit projects. There was a 1 to 1 mentoring (or tutoring) approach. Tutors (or mentors) communicated with their students in an engaging way, prompting them using scaffolds. The student-tutor interaction enabled students to build rapport with their tutors and create a tiny community to learn programming with the micro:bit. Tutors were encouraged to modify or adjust the lesson content according to the student's learning pace. They were not required to explicitly follow the lesson slides or guidelines in the given sequence. Most tutors were learning together hand in hand with the students. Students were leading the learning. Tutors followed the students' learning patterns and attention spans. This has encouraged self-directed learning.

Students were organized into 4 groups. In some lessons students moved around to different learning stations to learn how to use different micro:bit tools like sensors.

In this program, the learning outcomes of the students were difficult to be assessed as compared to formal settings. Instead of being officially graded in exams, students completed a hands-on project—building a Smart Home. With this context, tutors provided an example by giving them a problem and a solution on how to improve a home, using the micro:bit. While in groups, they were told

to construct a smart home model, programming for any smart feature of their choice. The instructors probed students to think of an issue and generate alternative solutions. They then presented their model and idea on the last session. The approach was flexible and more open-ended, and the projects allowed students to freely explore programming in the real-world context. Students discussed in their groups, experimented with different codes and designs, as well as tinkered.

Table 1. Programme Structure

Session	Activity	Description
T1	Training session for Tutors—Basic	Introduction to the micro:bit
S1	Student Basic Training — Calming LEDs	Pair Work- One to one learning with the tutors
S2	Student Basic Training—Rock Paper Scissors	Pair Work
S3	Student Basic Training—Pressure Switch Alarm - LED	Pair Work
S4	Student Basic Training—Pressure Switch Alarm - Buzzer, Radio	Pair Work
T2	Intermediate Training session for Tutors	
S5	Student Intermediate Training—Micro:bit Extensions and Tools - Breadboard - Input Sensors: Distance, Light, Water, Temperature, Sound, Motion - Output: Rainbow LED, Servo Motor	Learning Stations— Students move as a group in a rotation basis
S6	Student Intermediate Training	Learning Stations
S7	Student Group Project—Create a Smart Home	Group 1: Automatic Sliding Door
S8	Student Group Project	Group 2: Anti-theft Phone Sensor Group 3: Card-Swiping System Group 4: Safe Distance Sensor

## 4. OBSERVATIONS and ANALYSIS

There were 12 students observed for lessons and project. For training lessons, each student was paired with one dedicated tutor. For the project, students were put in groups. They worked in 4 groups. Each group had 2-4 students and was mentored by 2-3 dedicated tutors. Tutor observations during training lessons were captured during the end of lesson debrief sessions.

A structured observation approach was followed for the group project portion of the program to assess CT competence through their behaviour, verbal cues, visual cues and artefacts created. We tried to capture what students said, wrote, skills demonstrated, and artefacts created, with a focus on understanding their CT cognitive processes and practices.

Typically, students brainstormed with tutors by throwing out suggestions. They then decided on the problem they want to solve. Next, they conceptualised the solution by drawing, listing out sensors and designing how the prototype should look like, in relation to the placement of other components. In this step, they brought in previous experiences and built on them gradually. Lastly, they built the prototype, coded and tested their solutions.

Below, we present our observations and analysis on 2 participant groups—Group 2 and Group 4, followed by examples of the tutor observations based on the above framework.

### 4.1 Observations of Group 2

Group 2, making an anti-theft phone sensor, used input from a light sensor and programmed it to output sound(buzzer) and light (LED).

#### 4.1.1 Cognitive Processes

##### Abstraction

Students demonstrated good *abstraction* skills by (i) choosing to focus on the light sensor, among all the sensors available to them, (ii) narrowing down the project scope by iteratively communicating the concept and illustrating the complex ideas of the alarm system. As seen in *Figure 1*, during brainstorming, a student briefly drew out the phone holder, placement of the light sensor, LEDs and buzzer. He explained it verbally through pointing out the placement to the tutor, instead of labelling parts of the model.

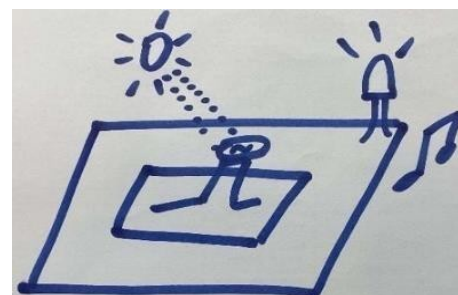


Figure 1. Layout Drawn by a Student during Brainstorming.

##### Decomposition

Students also demonstrated *decomposition* skills by systematically breaking down the algorithmic tasks of



coding input from the light sensor and coding the part of after receiving the input. They could explicitly explain each step e.g., the first code step was articulated as “We would like to tell the micro:bit to register the amount of light.” The input transformation was further decomposed into the code for LED and the code for the buzzer. It was guided by tutors though. For example, prompts like “So now when there is a lot of light, what do you want the micro:bit to do?” or “We want the micro:bit to make sound. What should we do?” resulted in students pondering through the next step.

### Pattern Recognition

Students showed a good understanding of the *pattern* between the placement of the phone, light received, and input reading obtained. An interesting point was the group placing the light sensor underneath the phone. When the phone is placed on the phone holder, the light sensor is covered and does not receive any light. On the contrary, once the phone is lifted from the phone holder, the light sensor is exposed to light and receives light. This *pattern recognition* forms the basis of this project. Tutors’ scaffolding with questions e.g., “When the phone is on top of the light sensor, is there a lot of light or very little light?” helped students see the reason to link the concepts of input with multiple outputs. A student said “This is where you place the phone, then after that LED lights will be placed around it. When someone takes away the phone, there will be sound, and the lights will also light [up].” *Pattern recognition* was also reinforced through drawing students’ attention to making connection with the past lessons e.g., guidance like “Remember the last time when we tried using the light sensor, what was the threshold that you used in the code? Was it 700-800?” helped students to recall. They also tested with different values outside the threshold range.

### Testing and Debugging

Students continuously *tested and debugged* in resolving the issues e.g., if the volume of the buzzer was too soft, they *tested* with various values to get the optimum. In one case, they isolated the problem to hardware and not code-software and switched to a different buzzer. Another instance of isolating the problem was when they *tested* the code using digital and analog blocks and concluded that the digital block was the correct one. They similarly *tested* for different light thresholds as well.

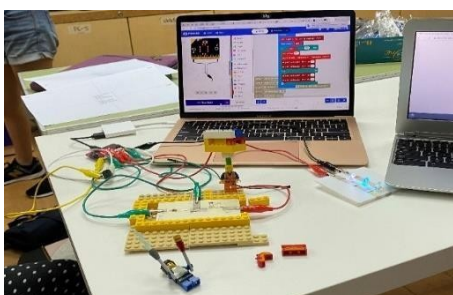


Figure 2. Final Model of the Anti-theft Phone Sensor.

#### 4.1.2 Practices

Students demonstrated their *algorithmic thinking* by handling multiple conditions in the logic, and appropriately using e.g., the “if-else” and “digital write pin” blocks. We

also observed *systemic thinking* as the students demonstrated a good understanding of the problem and solution in a wider context of Smart home. Furthermore, as seen in Figure 2, for *creating computational models*, the group made use of Lego blocks to place and secure the light sensor in it. They also connected the light sensor, LED and buzzer to the correct pins to piece the whole smart home.

### 4.2 Observations of Group 4

Group 4 worked on a self-conceptualized project idea based on the ‘safe distancing’ concept, very relevant to the present COVID context. They made use of the concept of “On pin pressed”, which is similar to the pressure switch alarm, covered in the third lesson. If detected, triggered by someone violating the marked line in a human queue, it would send a warning sound message indicating a violation of safe distancing rule.

#### 4.2.1 Cognitive Processes

##### Abstraction

As seen in Figure 3, students demonstrated *abstraction* skills by (i) simplifying and illustrating their concept and plan by drawing out the complex idea, (ii) narrowing the project scope to a single objective of preventing people from being too close or cutting queues, and (iii) listing their needs for carrying out the project. A student aptly verbalized “So if anyone steps on the carpet, there is this light and something saying [a message], so he will go all the way to the back of the queue”.

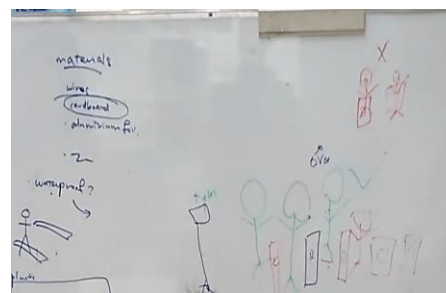


Figure 3. List of Materials and Layout of People in a Queue.

##### Decomposition

Students demonstrated *decomposition* skills by (i) systematically dividing the logic into 3 chunks—physically creating the distance sensor by placing the micro:bit underneath the carpet (i.e., in-between 2 people 1 meter apart), carrying out the appropriate action upon pin pressed, and resetting the state and (ii) dividing the output into 3 parts—the warning message, LED, and alarm tone on the buzzer. These parts are seen in the code in Figure 4.

##### Pattern Recognition

The group demonstrated *pattern recognition* skills by applying the patterns they have seen, e.g., in real life human queues and in the example from an earlier lesson.

##### Testing and Debugging

Students in this group relied on tutors for coding though, and therefore did not demonstrate much of *testing and debugging* skills.

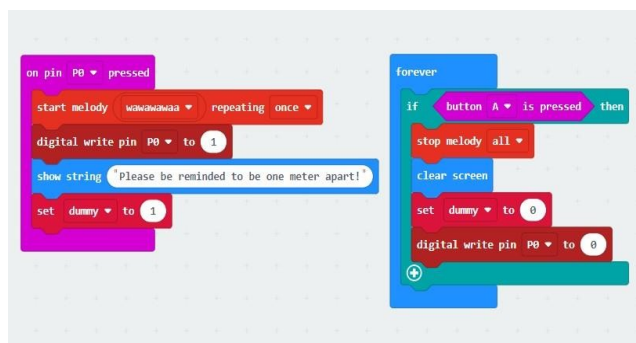


Figure 4. The Code of the Safe Distance Sensor.

#### 4.2.2 Practices

The *systemic skills* were demonstrated, similar to that in Group 2, by seeing safe-distancing application in context of a smart environment. They also showed a good understanding of constraints and resources needed for a real application e.g., needing many micro:bits for a long queue. Lastly, students *created computational models* successfully. This was evident in their model in Figure 5 linking the pins to the extensions—buzzer or LED, or in piecing the whole setup of the safe distance sensor together.

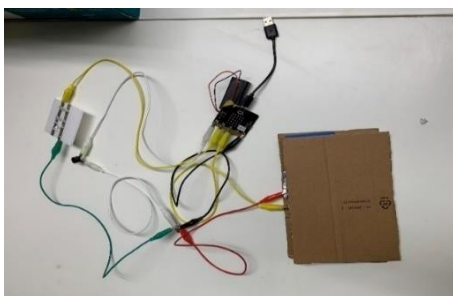


Figure 5. Final Model of the Safe Distance Sensor.

#### 4.3 Tutors' Observations

Our observations were based on the framework shown in Figure 6. The cognitive processes and practices were analysed based on the 4 behaviors—say, write, do, make.

From the tutors' observation framework, *abstraction* skills were more developed as (i) students summarised the concepts in their own words, and (ii) conceptualised the idea by drawing it. Students could successfully *recognise the patterns* of (i) the relationship between input and output, and (ii) labels on the sensor as corresponding to the pins on the micro:bit. Students were good in *testing and debugging* as (i) despite not being given any prompts, they could still distinguish when to use "If" and "On start", correcting their mistakes automatically. Tutors observed that a student could *create algorithms* by (i) drawing diagrams, and (ii) using "if" code. Students could *work well with data* as they (i) experimented with different threshold numbers until they got their ideal value. Students could *understand systems* competently as they (i) adapted the code to the context of a theft case, (ii) linked past experiences of the light sensor to estimate the light sensor threshold, and (iii) linked the light sensor to a real world application. All students could *create computational models* by (i) building the structure together, and (ii) suggesting the positioning of LED lights.

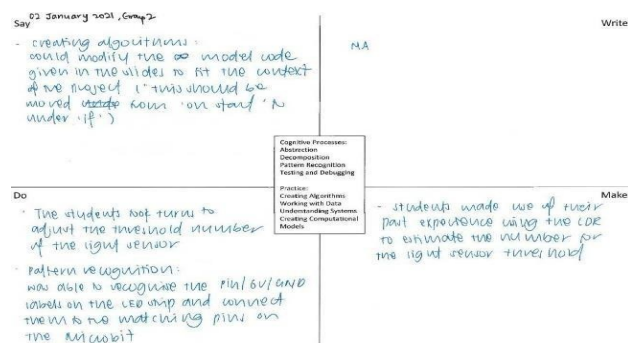


Figure 6. Tutors Observation Framework for CT

## 5. FINDINGS AND DISCUSSION

This study is a first-hand experience of designing a physical computing program for an underserved community in an out-of-school context. Here are a few things that made this volunteer run program work. First, the program was held at a place next to students' homes making it easier for them to attend. Students and tutors knew each other through other programs and were familiar with the rules and regulations of the place. Secondly, we conducted a survey to know the experience students have with physical computing programs and device usage in general. This informed us what needs to be provided. We also checked whether students have access to internet and devices, school supplies, and in identifying the right content-area tutor. Thirdly, the program was designed to have collaboration opportunities among tutors as well among students to encourage attendance, facilitate interaction and peer learning. Lastly, observing the individual student needs and pace closely through 1 to 1 student-tutor pairing helped monitor each student's progress. Innovating the project concepts and providing for additional resources made the program meaningful and enjoyable to students.

Mentors played a significant role in developing CT skills for students. Mentors not only helped provide a good foundation of problem-solving and decomposition by helping students through a design thinking process, but they also contributed to students' learning by coming up with an example or adding a new concept. They built on the student's interests, keeping them learning and engaged. They also helped develop an appreciation and confidence in student abilities by answering their questions, providing moral support, and learning together with them. They also taught them how to be responsible by helping them understand the importance of managing resources in a project. We find that a close relationship between mentor and student may be the key to the effectiveness of the program. Greater attention to the importance and building of a close relationship between students and mentors could help inform the design of such out-of-school programs.

Students demonstrated sufficiently developed problem-solving and *abstraction* skills. They were new to sensors and related coding. Therefore, their *pattern recognition* and coding skills were not seen to be at the same level. They showed a keen interest in creating a project for a real-life context, and in *testing and debugging* code. Our experience shows that well designed, scaffolded physical computing activities have the potential to improve students' CT skills.

It is true that some of the students do not engage 100% with the experience in such informal settings, but for the majority, it served as an opportunity to add CT skills to their set of skills.

Although this exploratory study is susceptible to some limitations, it explored an opportunity in a non-formal education situation. In this study, we have been able to explore variables and factors to be addressed in future research works related to the acquisition of CT through physical computing. We believe that such studies are important in achieving broader goals of studying technology as means to reduce socio-economic gaps in educational achievement.

## 6. ACKNOWLEDGMENTS

We would like to thank Ulu Pandan Stars Centre (UPSTARS) for collaborating with us. This project was funded by the grant OER 03/18 PS, under the National Institute of Education, Nanyang Technological University, Singapore, IRB Ref. No. IRB-2019-02-018.

## 7. REFERENCES

- Beam, M. R., Chen, C., & Greenberger, E. (2002). The nature of adolescents' relationships with their "very important" nonparental adults. *American journal of community psychology*, 30(2), 305-325.
- Benander, A. C., & Benander, B. A. (2008). Student monks—Teaching recursion in an IS or CS programming course using the Towers of Hanoi. *Journal of Information Systems Education*, 19(4), 455.
- Borden, L. M., Schlomer, G. L., & Wiggs, C. B. (2011). The evolving role of youth workers. *Journal of Youth Development*, 6(3), 124-136.
- Digital Promise. (2020). *Key Concepts of Computational Thinking*. Retrieved January 26, 2021, from <https://digitalpromise.org/initiative/computational-thinking/key-concepts-of-computational-thinking/>
- Evans, E., Ching, C. C., & Ballard, H. L. (2012). Volunteer guides in nature reserves: exploring environmental educators' perceptions of teaching, learning, place and self. *Environmental Education Research*, 18(3), 391-402.
- Fritz, S., Karmazin, D., Barbuto Jr, J. E., & Burrow, S. (2003). Urban and rural 4-H adult volunteer leaders' preferred forms of recognition and motivation. *Faculty Publications: Agricultural Leadership, Education & Communication Department*, 24.
- Hidi, S., & Renninger, K. A. (2006). The four-phase model of interest development. *Educational psychologist*, 41(2), 111-127.
- Klaw, E. L., Rhodes, J. E., & Fitzgerald, L. F. (2003). Natural mentors in the lives of African American adolescent mothers: Tracking relationships over time. *Journal of Youth and Adolescence*, 32(3), 223-232.
- Nardelli, E. (2019). Do we really need computational thinking?. *Communications of the ACM*, 62(2), 32-35.
- Norling, E. (1995). Encouraging networking through informal mentoring: a look at a newly-established mentor scheme. In *Second Australasian Women in Engineering Forum*. RMIT, Australia.
- Papert, S. (1972). Teaching children thinking. *Programmed Learning and Educational Technology*, 9(5), 245-255.
- Pisimisi, S. S., & Ioannides, M. G. (2005). Developing mentoring relationships to support the careers of women in electrical engineering and computer technologies. An analysis on mentors' competencies. *European journal of engineering education*, 30(4), 477-486.
- Rogers, Y., Shum, V., Marquardt, N., Lechelt, S., Johnson, R., Baker, H., & Davies, M. (2017). From the BBC Micro to micro: bit and Beyond: A British Innovation. *interactions*, 24(2), 74-77.
- Seow, P., Wadhwa, B., Lim, Z-X., & Looi, C-K. (2020). Towards using Computational Modeling in learning of Physical Computing – An Observational Study in Singapore Schools. In Kong, S-C (Ed.), *Proceedings of the Fourth International Conference on Computational Thinking Education 2020* (pp. 100-107). Hong Kong, Hong Kong (China): The Education University of Hong Kong.
- Townsend, G. C. (2002). People who make a difference: Mentors and role models. *ACM SIGCSE Bulletin*, 34(2), 57-61.
- Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), 3717-3725.
- Worker, S. (2017). Volunteer educators bring their own ideas about effective teaching to a 4-H curriculum. *California Agriculture*, 71(4), 208-213

# Combining Maker Technologies to Promote Computational Thinking and Heartware skills through Project-based Activities: Design Considerations and Empirical Outputs

Ali HAMIDI<sup>1\*</sup>, Sepideh TAVAJOH<sup>2\*</sup>, Marcelo MILRAD<sup>3\*</sup>

<sup>1,2,3</sup> Faculty of Technology, Linnaeus University, Sweden  
ali.hamidi@lnu.se, st222yd@student.lnu.se, marcelo.milrad@lnu.se

## ABSTRACT

Many countries have started to integrate Computational Thinking (CT) as an essential 21st century skill into different schools' STEM related subjects. Despite positive developments in terms of CT integration into schools' curricula, there are still important issues and challenges to address on how to teach and use programming and CT in the classrooms. As part of our ongoing efforts to introduce and to apply different maker technologies to foster CT, this paper describes the results of an exploratory study aiming at designing and implementing learning activities in informal settings using the Engino® Robotics Platform (ERP) and the BBC micro:bit. We conducted a one-week-long workshop with the participation of 22 children aged 10-15 years old. The constructionist theoretical perspective and the Four P's Creative Learning theory (projects, peers, passion, and play) were applied for conceptualizing and designing our activities. The initial results of our efforts indicate that firstly, learning contexts enriched by the combination of different maker technologies can help students to develop CT skills; Secondly, remixing learning experiences can bring CT into STEM subjects; and lastly, the design of the proposed workshop and the planned activities serve as the basis of a learning environment that can foster problem solving, creativity, and heartware skills when the four P's are taken into account. The current study contributes with empirical knowledge that can be used for the advancement of design practices to promote CT development in connection to STEM-related subjects both in informal and formal learning settings.

## KEYWORDS

computational thinking (CT), STEM education, constructionism, informal learning, four P's creative learning theory

## 1. INTRODUCTION

Many countries have recently started to make the necessary changes in the primary and secondary school curriculum to integrate Computational Thinking (CT) as part of the 21st century skills included in their different school subjects (Grover & Pea, 2018). In Sweden, the integration of CT and programming in the school curricula has started in 2017 based on a National IT strategy, which builds upon the suggested proposal by the National Agency of Education. Accordingly, schools' syllabuses were updated by positioning CT in the revised curricula and focusing on programming, algorithmic thinking, and problem-solving in the subjects of Mathematics and Technology (Skolverket, 2020). Despite these positive developments, there are still important issues and challenges with regard to how to put in

practice these changes into the curriculum. It is worth to mention, for example, that there is not yet a national strategy that provides recommendations on how to practically implement these curricular changes (Kohen-Vacs & Milrad, 2019) and operationalize them in schools. An evidence of such claims is a recent survey answered by more than 550 teachers conducted by the Sweden teachers' union. It reveals that more than 70% of the teachers who took part in CT and programming related courses still feel very uncertain about how to teach and use programming and CT in schools (Dagens Nyheter, 2020). Consequently, there is a need for carrying out research and implementation efforts related to how to address these topics and put them into practice in schools.

In line with the efforts mentioned before, this paper presents the results of an exploratory study aiming at to introduce different maker technologies (Fitton et al., 2015) to design CT practices and activities. For this purpose, we use the ERP and the BBC micro:bit. One of our aims is to design and foster educational activities in order to develop CT skills and abilities for students in K-12 schools both in formal and informal learning settings (Kynigos & Grizioti, 2020; Yilmaz Ince & Koc, 2020; Lee & Low, 2020). Our approach can be characterized by the combination of constructionist views of learning supported by the use of complementary maker technologies and materials. The specific focus here is on STEM-related subjects in informal learning settings that support the aims described above. Considering the increasing need for designing and developing educational activities and support for teachers in terms of CT educational materials (Tyrén et al., 2018), the use of maker technologies and software systems may help to compensate the limitations of each one of these tools and systems if they were used separately. When different maker technologies and materials are used in CT related activities, a positive relation is exerted between designing and learning throughout structural aspects of CT that expands users' perceptions and understanding of it (Sung et al., 2017). Remixing experiences that are referred to *sharing*, *modifying*, *embedding*, or *adapting* an object within another object, are considered as an element of a CT pedagogical framework as proposed by Kotsopoulos et al., (2017). This framework covers other aspects of pedagogical experiences in connection to constructionist education such as *making* and *tinkering*. To date, with the high expectations of developing STEM skills (Sung et al., 2017), there have been attempts to apply different approaches to teach these areas together with CT development (Kohen-Vacs et al., 2020). Thus, the research question that is at the core of this work can be formulated as follows: *What are the effects of*

*combining different Maker Technologies while trying to promote CT skills in informal learning settings?*

The remaining of the paper is organized as follows, in section II we describe our approach and the theoretical perspectives that guide our research efforts. In section III we present the settings and the activities carried out as part of these efforts. The data collection methods we have used are described in section IV while section V presents the analysis and outcomes of our work in the findings & discussion section. Finally, the conclusions are presented in section VI.

## 2. METHODOLOGICAL APPROACH

In this section we present both the theoretical perspectives that guide our work followed by the technological approach used in the activities described later in the paper.

### 2.1. Theoretical Approach

Considering the challenges related to teachers' CT and programming competences and the lack of suitable knowledge and infrastructure in schools (Kohen-Vacs & Milrad, 2019), we have decided to explore our research ideas within the context of informal learning. According to Lee et al., (2019), heart-ware skills can be developed in informal learning settings beyond the school's curricula. Heart-ware skills refer to the holistic development of students in terms of active contribution, gaining confidence, and leadership opportunities. As part of our ongoing practices to introduce and apply different maker technologies (Fitton et al., 2015) for CT development we considered different aspects of an appropriate activity from a constructionist perspective which is used as a theory of learning and a theory of design too (Kynigos, 2015). However, as argued by Kynigos & Grizioti (2018), the challenges for designers and teachers still remain when new affordances of maker and digital technologies are introduced. In order to increase the level of the participants' engagement with our practice, we considered also the theory of Four P's Creative Learning as suggested by Resnick (2014). According to this theory, four factors related to *Projects, Peers, Passion, and Play* are the guiding principles for the active engagement of pupils in the construction and explorative approaches to CT.

Guided by these ideas, we have designed several learning activities as part of a one-week-long workshops for engaging students with a few CT tools related to *maker activities*, design and programming in connection to STEM related subjects. The ERP and the BBC micro:bit have been used in a summer camp workshop with the participation of students to integrate the potentials and affordances of each system, and also by the combination of those two together. We intended to cover most of the STEM elements through the application of the functionality and affordances of these two systems in the context of authentic learning scenarios. While using the Engino robotics sets in some of our activities brings together the *Technology* and *Engineering* parts of the STEM (Yilmaz & Koc, 2020), the workshop on the use of micro:bit was designed with a particular focus on the *Science* part. The technological aspects of the two technologies and software tools we used in our workshops are presented in the coming subsection.

### 2.2. Technological Approach

Engino<sup>1</sup> proposes a novel building system of modular connectors that provides a three-dimensional building structure. It enables users to simply snap-fit on various locations of building blocks in order to construct functional models quite easily and quickly. The Engino sets are designed for different age groups from simple constructing level with building blocks to advanced wireless robotics that include peripherals and sensors for students of all ages (Engino, 2020). The controllers can be programmed either manually or through a scratch-like programming software that is named KEIRO™ which is a key element of the system. The drag-and-drop block based graphical programming platform can be switched to an Arduino IDE environment providing both C/C++ and Python programming languages.

BBC micro:bit<sup>2</sup> is a small microcontroller that was developed in 2015 to encourage students to become creative in the digital world in connection to STEM subjects with the possibility to be connected to other devices or sensors. The matrix display, programmable buttons, and built-in sensors are providing an educational platform with a lot of potential to be used in educational settings (Tyrén et al, 2018). The Microsoft MakeCode editor is a free programming environment that is used for coding with the micro:bit by snapping different blocks together. In the next section we described the settings and activities in which our exploratory study took place.

## 3. SETTINGS AND ACTIVITIES

During the summer of 2020, 22 children aged 10-15 years old gathered at the Innovation Lab located at Videum Science Park in Sweden, to carry out different activities by using the technologies mentioned above. The children took part in 5 workshops, conducted on daily basis for one week, each one lasting 7 hours. Two researchers in collaboration with five tutors from Linnaeus University (LNU) participated and worked together to carry out the workshops. As shown in Table 1, students were divided in two groups, each one consisting of 11 children. Each group took part in four different workshops including *Web design*, *3D printing*, *Engino (focus on Robotics)*, and *micro:bit (focus on STEM)* in order to get familiar with different digital tools and technologies. However, the focus of this study is on the two latter workshops as our emphasis is on CT skills and programming. While one group of children joined for example the Engino workshop, the another group of children worked with the micro:bit. Each daily workshop comprised a threehours morning session, onehour lunch break, and a threehours afternoon session. In the morning, we conducted an introduction and instructions to simple practices on how to use the different tools and continued with the students' practical activities and experience in the afternoon. While the first four days of the workshops followed the structure described in table 1, the last day was specified to conduct a competition and students were free to work on their previous products, to develop them, and to have group collaboration. Students' parents were also invited to join their children in the last day.

<sup>1</sup> <https://enginoeducation.com/>

<sup>2</sup> <https://microbit.org/>



Table 1. Workshops' Schedule (Each Day 7 Hours)

	Engino	Micro:bit	3D printing	Web design
Day#1	Group I	Group II	-	-
Day#2	Group II	Group I	-	-
Day#3	-	-	Group I	Group II
Day#4	-	-	Group II	Group I
Day#5	Group I + Group II			

### 3.1. Robotics Workshop

This workshop began with an introduction to the ERP equipment and instructions on how to use it. The workshop was designed based on two main activities: *constructing robots* and *programming them*. According to our previous experiences, in order to maximize the productivity of this workshop (given the time constraints we had) we provided half-built models and asked the participants to add other required peripheral devices and make them ready for programming. The idea here was to increase children's motivation when they change or redesign the models according to their own interests to increase their sense of ownership of their designs. Different sensors were also available to control the robot by the means of programming. As shown in Figure 1, two main tasks were designed.



Figure 1. Tasks to perform in the Engino Workshop

In the first one, we designed a lane in which the robot needs to be programmed to pass on the road without hitting the barriers. In the next task, a line tracker robot (car or train) should be controlled to follow the black line that was taped on the floor. In both tasks, two students created a group and worked together to carry out the assignments. In order to perform that, two sensors including an infrared (IR) sensor and an ultrasonic sensor were used. To get familiar with the devices a brief instruction was presented on how to calibrate and set them up when they were connected to the software.

### 3.2. STEM Workshop

The aim of conducting the STEM workshop by using the micro:bit was to design an activity in which students could be involved in authentic and meaningful activities such as building a greenhouse and controlling it using different sensors and actuators (see Figure 2). This workshop was conducted one day after the robotics workshop. The idea was that students' experience and knowledge gained from the first workshop could be used in the second one. In addition, the physical structure of the greenhouses that were used in this workshop was built using the Engino construction physical materials and blocks. Thus, this approach provided an opportunity to think and compare different affordances provided by each one of the proposed technologies and their use in connection to STEM, programming environments, and CT. This workshop began with an introduction about the micro:bit and the Microsoft MakeCode. Thereafter, simple

tasks were explored and examined with the students to make them familiar with the materials and programming software.

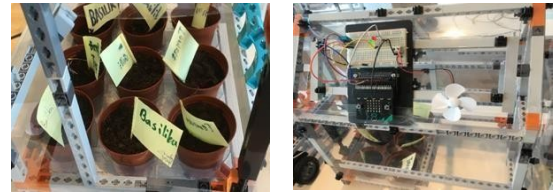


Figure 2. Application of the Maker Technologies in the Context of a Greenhouse

As shown in Figure 2, the children were asked to fill the small pots with soil and to plant seeds for better making sense of this authentic situation. Thereafter, they mounted the micro:bit in the greenhouse and measured the temperature and light level. A fan also was connected to the micro:bit that should be set to switch on when the temperature exceeds a pre-defined value. Similar to the previous workshop, children worked in pairs to carry out the given tasks. Table 2 below summarizes the focus and description of the activities for each one of the workshops.

Table 2. Workshops' Focus and Description

Focus	Activity Description
Robotics	Constructing experience
ERP	Developing & redesigning half-built robots Connecting sensors and other peripheral devices and setting them up Programming the robots as line tracker
STEM	Working with a ready built greenhouse and planting seeds
BBC micro:bit	Working with sensors for measuring humidity, temperature, and light level Connecting the sensors and calibrating them Programming to activate the fan depending on temperature and humidity Technology and software integration to the greenhouse

## 4. DATA COLLECTION METHOD

Data was collected from three sources namely: *a pre-questionnaire*, observations and informal discussions during the workshops, and *a post-questionnaire*. The pre-questionnaire was filled online before the start of the workshops by 16 students with the aim of getting information about the participants' background in programming and CT basics. We asked them to provide brief information about their knowledge and experience in visual programming languages, what programming is used for, and how to control a device through programming. Due to the nature of our exploratory study and the oral and written skills of the participants, we decided to conduct the data collection process through a participant observation approach where the researchers observed the students' activities and challenges while interacting with them through informal discussions. The latest helped us to collect data through a close familiarity with the individuals. The paper-basedpost-



questionnaire was answered after the completion of all workshops to collect students' ideas, feedback, suggestions, and reflections. We collected 18 responses (4 students did not provide written answers) related to the children's achievement from the workshops, their ideas about different maker technologies, and their suggestions for the content of future workshops. Thereafter, a qualitative data analysis (QDA) has been applied to analyze the data collected that includes our interpretation and understanding of the data and then sorting it into categories based on contextual factors such as participants' enrolment, intentions, and hands-on experiences. The theoretical perspectives mentioned earlier in the paper (heart-ware skills & four P's creative learning theory) played an important role with regard to the data analysis in term of comparison and categorization.

## 5. FINDINGS & DISCUSSION

In this section, we first describe our findings based on the analysis of the data we collected. Thereafter, we discuss our results that are presented in three sub-sections.

The pre-questionnaire that was answered one week before conducting the workshops helped us to have a better view on the participants' previous knowledge in connection to programming and CT. The questions were sorted in a way to gather students' perceptions on the general use of programming to more detailed questions about central CT concepts. Students' responses revealed that most of them use computers for gaming. Accordingly, those who used computers for gaming basically defined programming as the means of developing games and similar too websites. On the other hand, some children looked at programming from a much broader view as two of them stated:

*Programming is used for almost everything. Programming is used to make the world better.*

When it comes to CT concepts, a variety of opinions were listed for *algorithms*, *variables*, *input/output*, *conditional*, and *functions*. While concepts like algorithms and variables were more familiar to the children, some others (such as functions) were not well known. For example, an answer to a question about variables was quoted that it is like a virtual box with changeable values. Another example is regarding *conditional* that was again referred to a game:

*Conditional is something in a game that you lose a life if you touch something.*

Based on our observations and dialogues with the children, the construction part and hand-on activities were very interesting to them. Although the half-built models were ready to use for programming through adding some extra parts and peripherals, students liked to redesign, detach and connect different parts again and again. The latest is much aligned to the constructionist theory view on design (Kynigos, 2015). Looking at the children's responses in the post-questionnaire indicates that both sessions were interesting for the students. Some of them were more motivated in working with Engino and some preferred to work with micro:bit. Nevertheless, the combination of the two maker technologies were pointed out as the most interesting part for some children.

The design approach we applied in both activities was similar to what Kynigos & Grizioti (2020) referred to as "half-baked games" to be changed and completed in a process of try and error. The latest provides some initial insights on how to design an activity to deploy the potential affordances of both, the Engino and micro:bit together. While in the first activity children examined and got inspired with the Engino constructions and programming, they brought their experience to the next workshop with the micro:bit working on a pre-built Engino greenhouse. The qualitative analysis of the collected data, particularly the observation notes, revealed three preliminary results as describe below.

### 5.1. Contexts Enriched by Combining Different Maker Technologies Help Students to Develop CT Skills

According to the collected data through our observations, children learned some CT concepts and practices (Grover & Pea, 2018) in an authentic integrated context. The combination of digital technologies aligned with the constructionist approach and the 4P's constructs (Resnick, 2014) offers a playful experimentation that facilitates CT development through its integration in different activities. For example, the logic and logical thinking was practiced by the children when they set out their cars in the Engino workshop, and where they tried to make a *smart fan* in the micro:bit workshop. In both cases, they applied the *If statement* by using true/false values in order to control the artifact. That can be seen clearly when children used two IR sensors on both sides of the car and applied AND & OR expressions in the programming to avoid the car hitting obstacles. The algorithm and algorithmic thinking were followed by the children with sequence, selection, and repetition. While sequence and selection were seen in the loops of the KEIRO software environment and the MakeCode programming, the repetition was applied by most of the children where they used *While forever* to repeat the command unlimitedly.

We also recognized a process of abstraction that was conducted by one of the children in the micro:bit workshop. She first displayed a sequence of arrows in different directions by using the *On Start* block of the MakeCode programming environment. Then, she used the *repeat loop* block instead of using the same commands repeatedly. The result was interesting because she used the following block as shown in Figure 3 with a forever statement. Since the repeat block does not take a string input, so she learned how to use forever block instead.

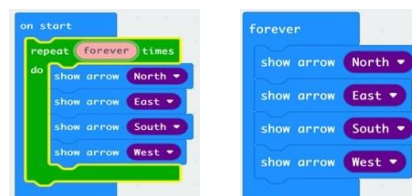


Figure 3. Abstraction Learning Sample by MakeCode

We also found several indications of CT practices while children worked on their digital artifacts. For example, *problem decomposition* has been frequently grasped when

the robots hit the barriers, so that students should check the program, sensors calibration, connections etc. The same was seen in the micro:bit workshop when a children could not work with the humidity sensor, so he first checked the code and then realized that the problem comes from the incorrect connection of the wires to the sensors' pins. *Testing and debugging* were the other elements that were tried repetitively by the participants. Specifically, the IR sensor was the most challenging part that required lots of tries and errors in terms of setting the accuracy. The critical competencies of collaboration, creativity, and promotion of heart-ware skills (Lee et al., 2019) were also an inseparable part of our workshops since children worked together and presented their results to each other to promote also collaborative learning.

## 5.2. Bringing CT into STEM Subjects

As argued by Grover & Pea (2018), CT skills can be developed in learning contexts outside the classroom in terms of generativity. The integration of CT concepts within the STEM subjects in after-school activities can be seen as viable way to enrich CT learning as well as developing CT competencies. The analysis of our data indicates that combining the Engino greenhouse with the micro:bit toolkit is an effective effort towards bridging CT and STEM in a context that provides the integration and interaction of both physical and digital objects. The activity we have illustrated above led to children's engagement with STEM subjects in connection to CT development while they executed several attempts to redesign the models to benefit from the potentials of maker technologies (Fitton et al., 2015). The examples described in the previous section such as calibrating the sensors and using them to control the robot address the engineering and technology aspects of STEM. For example, students' practices on how to turn the robot depending on the velocity and position of motors help students to understand engineering and technology concepts. Moreover, the greenhouse activity provides a context for developing the science part of STEM. For example, children were curious to test whether their code for air circulation was working by the use of the fan. The code they created with MakeCode program was intended to switch it ON when the indoor temperature of the greenhouse changed. In order to do that, children were changing the location of the greenhouse to make the change of temperature noticeable. They examined even that by placing the greenhouse inside the refrigerator.

Our observations, following the children activities also revealed another important result that was beyond our initial plan for these workshops. Although the research design was planned to use only the Engino physical parts with the micro:bit programming microcontroller, some groups of children were very enthusiastic to integrate the micro:bit potentials with Engino controllers and motors and even the KEIRO programming platform to open the windows and doors while the micro:bit sensors send the signals. A sample of such effort is shown in Figure 4. The children's response to our question while talking with them about their intention to combine the two platforms was that they wanted to benefit from Engino's possibilities and peripherals like the motors and gears to be connected to physical objects that were

provided by micro:bit sets like fans. However, they figured out that they cannot connect the MakeCode programming environment to KEIRO that is the software system used by Engino. In order to solve the problem above, they decided to use the manual controlling functions of the Engino motors. They used ERP controller's buttons to open or close the window when the micro:bit displayed a high or low temperature.

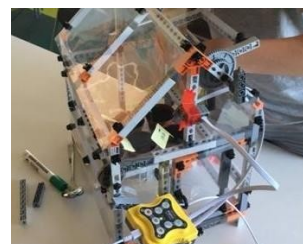


Figure 4. A Sample of Digital Tools Combination

## 5.3. Providing a Creative-learning Environment

The example given above for problem-solving indicates that children's interest and ability to solve a problem may increase when they have more functionalities in the technologies they use, so that they can combine, share, adapt, or place one physical and virtual tool within another one (Kotsopoulos et al., 2017). Considering the design scenario of the workshops, children's engagement in the different activities from filling the pots and planting the seeds to controlling the environmental parameters through the remixing digital tools are providing a creative-learning environment as suggested by Resnick (2014). He emphasizes the value of the 4P's elements as *Projects, Peers, Passion, and Play*. While the meaningful projects introduced by Engino and the micro:bit provide the opportunity for sharing the ideas and collaboration between peers, combining different tools increases pupils' passion. The creative solutions they applied to compensate the shortcomings of each one of the tools through adapting to another platform is an evidence of such claim. For instance, while trying to control the Engino motors' rotation based on the micro:bit signals, a group of students experienced moments of disappointment and discouragement when they could not make it work as they expected, but also a feeling of joy and happiness when they discovered a solution to control it manually. The latest increased students' confidence to share their learnings with their friends. Such emotions show that students were indeed engaged with the activity that illustrates heart-ware skills development of students at the same time (Lee et al., 2019). Moreover, we believe that the last day of the workshop devoted to playing and contesting in an informal learning environment acted as an important role to follow the children's goals where their active contribution was observed. Some parents also joined their children in the last day where students shared their findings, delights, feelings and emotions with them that highlights the value of the passion element in our design.

## 6. CONCLUSION

The learning design strategies used in this CT activity brought an opportunity for children to follow the different steps towards designing, building, and programming where creative learning took place through testing and debugging

of different solutions including software programs. In general, by using different maker technologies in the domains of biology and programming, our preliminary results illustrate that firstly, a range of CT skills emerged from the concepts to practices both in the construction and the programming parts such as problem decomposition, logical and algorithmic thinking, abstraction, testing and debugging. Secondly, it improved the link between CT and a couple of STEM relevant aspects through the application, manipulation and control of both physical and virtual objects. Thirdly, the design of the workshops, the planned activities, and the results illustrate that a creative learning environment can emerge when heart-ware skills are promoted and the four factors of *projects, peers, passion, and play* are taken into account.

Looking back to our research question on the effects of maker technologies on CT skills development, we consider the main contribution of this research from two perspectives. First, as a practice for researchers to explore how to use and combine different maker technologies and educational materials in an informal learning context in order to foster and develop children's CT skills. The design and implementation of the different learning activities make it possible to see the similarities and differences of interacting with different technologies and their combination in terms of what they can offer to both educators and students. It provides a free space for students to practice problem solving where they can choose alternative solutions. Second, the results of this study could be also helpful for teachers who are willing to design educational activities that include knowledge about CT and programming in connection to STEM related subjects. Although teachers did not participate in our study, the results presented here can be considered as a point of departure for post activities with close collaboration of teachers where a seamless learning view can be fostered in school settings as well as in informal educational contexts. Finally, the informal learning setting described in our study would cover inadequate infrastructure of the schools.

## 7. REFERENCES

- Dagens Nyheter (2020). *Sju av tio lärare osäkra på att undervisa i programmering*. Retrieved May 27, 2020, from <https://www.dn.se/nyheter/sverige/sju-av-tio-larare-osakra-pa-att-undervisa-i-programmering/>
- Engino (2020). *Total Educational Solution: Combination of STEM & Robotics*. Retrieved Aug 20, 2020, from <https://www.engino.com/w/index.php>
- Fitton, D., Read, J. C., & Dempsey, J. (2015, June). Exploring children's designs for maker technologies. In *Proceedings of the 14th International Conference on Interaction Design and Children*, 379-382.
- Grover, S., & Pea, R. (2018). Computational Thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*. London: Bloomsbury Academic, 19-37.
- Kohen-Vacs, D., & Milrad, M. (2019). Computational Thinking Education for In-Service Elementary Swedish Teachers: Their Perceptions and Implications for Competence Development. In *International Conference on Computational Thinking Education 2019*. Hong Kong: The Education University of Hong Kong, 109-112.
- Kohen-Vacs, D., Kynigos, C., & Milrad, M. (2020). On the Integration of Learning Mathematics and Programming. *Proceedings of International Conference on Computational Thinking Education 2020*. Hong Kong: The Education University of Hong Kong., 53-56.
- Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A pedagogical framework for computational thinking. *Digital Experiences in Mathematics Education*, 3(2), 154-171.
- Kynigos, C. (2015). Constructionism: Theory of Learning or Theory of Design? In *Selected regular lectures from the 12th International Congress on Mathematical Education*, Cham: Springer, 417-438.
- Kynigos, C., & Grizioti, M. (2020). Modifying games with ChoiCo: Integrated affordances and engineered bugs for computational thinking. *British Journal of Educational Technology*.
- Lee, P. T., Lee, X. R., Low, C. W., & Kokila, A. (2019, June). Implementing Computational Thinking through Nonformal Learning in after School Activities at Students Society Club. In *Proceedings of the International Conference on Computational Thinking Education 2019*. Hong Kong: The Education University of Hong Kong, 201-202.
- Lee, P. T., & Low, C. W. (2020). Implementing a Computational Thinking Curriculum with Robotic Coding Activities through Non-formal Learning. *CoolThink@JC*, 150.
- Resnick, M. (2014, August). Give P's a chance: Projects, peers, passion, play. In *Constructionism and creativity: Proceedings of the third international constructionism conference*. *Austrian computer society, Vienna*, 13-20.
- Skolverket (2020). *Digitalization*. Retrieved May 26, 2020, from <https://www.skolverket.se/temasidor/digitalisering>
- Sung, W., Ahn, J., & Black, J. B. (2017). Introducing computational thinking to young learners: Practicing computational perspectives through embodiment in mathematics education. *Technology, Knowledge and Learning*, 22(3), 443-463.
- Tyrén, M., Carlborg, N., Heath, C., & Eriksson, E. (2018, June). Considerations and Technical Pitfalls for Teaching Computational Thinking with BBC micro: bit. In *Proceedings of the Conference on Creativity and Making in Education*, 81-86.
- Yilmaz Ince, E., & Koc, M. (2020). The consequences of robotics programming education on computational thinking skills: *An intervention of the Young Engineer's Workshop (YEW)*. *Computer Applications in Engineering Education*, 191-208.

# **Computational Thinking and Psychological Studies**

# Influential Factors of Hong Kong Secondary School Students' Intrinsic Motivation to Coding Education during the COVID-19 Epidemic: A Correlational Analysis

Xin ZHANG<sup>1\*</sup>, Gary K. W. WONG<sup>2</sup>, Qiaobing WU<sup>3</sup>, Bill Y. P. TSANG<sup>4</sup>

<sup>1,2</sup>Faculty of Education, The University of Hong Kong, Hong Kong

<sup>3</sup>Department of Applied Social Sciences, The Hong Kong Polytechnic University, Hong Kong

<sup>4</sup>The Youth Global Network, Hong Kong

zhangxsid@gmail.com, wongkwg@hku.hk, qiaobing.wu@polyu.edu.hk, bill.tsang@ygn.org.hk

## ABSTRACT

This study explores the relationship among Hong Kong secondary school students' demographics, psychosocial attributes, social interaction and subjective experience with the COVID-19 epidemic, and their intrinsic motivation to coding education under the educational context of school closure on a worldwide scale due to the pandemic. An online questionnaire survey was carried out in three subsidized secondary schools of Hong Kong before the face-to-face teaching and learning resumed in early June 2020. 204 participants were from Form 2 (equivalent to Grade 8). The results of correlational analysis showed that:

(1) students' psychosocial attributes during the pandemic was significantly positively correlated with their intrinsic motivation to coding education; (2) students' social interaction during the pandemic were significantly positively correlated with their intrinsic motivation to coding education; and (3) students' subjective experience with the pandemic was significantly positively correlated with their intrinsic motivation to coding education. The findings may have implications for educators and academics of computational thinking to identify the factors which will enhance student engagement and learning outcome in a time of uncertainty and crisis.

## KEYWORDS

intrinsic motivation, coding education, psychosocial attributes, social interaction, COVID-19

## 1. INTRODUCTION

Beginning from early 2020, the COVID-19 epidemic rampaged throughout much of the globe. By April 2020, around 91.3% of the world's learners, almost 1.6 billion of total enrolled couldn't go to school due to 194 country- wide closures of schools, according to the United Nations Educational, Scientific, and Cultural Organization (2020). The months-long suspension of face-to-face classes and students' home confinement, and the subsequent arrangement of remote learning have imposed many knotty and unprecedented challenges for teachers, parents, and students, e.g. public concern on the equity in access to remote learning occupied many headlines, teachers' helplessness on the growing absenteeism and late submission of homework that were uncommon for conventional learning were seen around in most of the online forums or discussion groups for educators.

Thus, understanding the influential factors of students' learning motivation during these unprecedented times is a timely move for the education sector, as the rapid shift in

the delivery mode of instruction has led educators to explore effective ways to provide appropriately supportive environments to maintain students' motivation. In this study, we surveyed Hong Kong secondary students during late May and early June of 2020 to explore the association between demographics, psychosocial attributes, social interaction and subjective experience with the pandemic during the COVID-19 lockdown to get a broader sense of possible buffering role of various factors for students' intrinsic motivation to coding education.

## 2. INTRINSIC MOTIVATION

The self-determination theory (SDT) provides an account for human motivation and personality that focuses on people's inherent growth tendencies and innate psychological needs (Ryan & Deci, 2000a). Based on the different underlying reasons and goals of human actions, SDT also distinguishes between different types of motivation such as intrinsic motivation versus extrinsic motivation. Being intrinsically motivated means that individuals engage in activities out of the inherent interest and enjoyment of the behavior itself, and such internal locus of causality leads to functional differences of intrinsic motivation from other types of motivation (Ryan & Deci, 2000b). When individuals feel their psychological needs are satisfied, they tend to be more intrinsically motivated. Therefore, it is important to detail the factors and psychosocial attributes that engender it, as in a learning environment where teachers and peers are not physically present, and parents may not be around supporting, intrinsic motivation is particularly relevant for catalyzing students' commitment to learning out of nothing but only rewards in learning behavior itself and protecting them from maladjustment.

The crucial importance of student motivation to a series of positive development outcomes has been well-documented (Cerasoli et al., 2014; De Naeghel et al., 2012; Lazowski and Hulleman, 2016); however, student motivation under a typically uncertain time has been scantily address. Under the new learning environment shaped by various restrictive measures such as school closure, remote learning, and home confinement, what makes students more intrinsically motivated? To step into this void, we investigated Hong Kong secondary students' intrinsic motivation in coding education during the COVID-19 epidemic and focused on the connections between a variety of personal and contextual factors with the intrinsic motivation, which may provide important insights into potential avenues for ameliorating the negative effects of various on the vulnerable students during uncertain times.

### 3. METHODS

#### 3.1. Participants and data collection

The data was drawn from the Jockey Club Coding for Community Project, which aims to provide the underprivileged youths in Hong Kong with coding courses to develop their computational thinking skills, and an adult-youth partnership scheme to promote their relational development and community involvement through designing mobile applications to address practical issues in local communities. Participants of this project were invited to respond to an online questionnaire before face-to-face classes resumed on June 8, 2020. The sample in this study comprises 204 Form 2 (equivalent to Grade 8) students (78 females) from three Hong Kong subsidized secondary schools, with an average age of 13 (age range: 12 to 15).

#### 3.2. Instruments

An adapted version of the Intrinsic Motivation Inventory (Jiang & Wong, 2017; McAuley, Duncan, & Tammen, 1989) was used where the mean score was calculated out of 10 items on a five-point Likert scale (sample item: 'Programming activity was fun to do',  $M = 2.71$ ,  $SD = .93$ ,  $\alpha = .96$ ). For psychosocial attributes, we measured students' general self-efficacy (sample item: 'I am confident that I could deal efficiently with unexpected events',  $M = 12.92$ ,  $SD = 3.72$ ,  $\alpha = .87$ ), grit (sample item: 'I often set a goal but later choose to pursue a different one',  $M = 3.10$ ,  $SD = .54$ ,  $\alpha = .65$ ), resilience (sample item: 'I am able to solve problems without harming myself or others',  $M = 37.69$ ,  $SD = 8.89$ ,  $\alpha = .87$ ), sense of community (sample item: 'I feel like a member of this neighborhood',  $M = 35.13$ ,  $SD = 7.68$ ,  $\alpha = .92$ ), and youth social responsibility (sample item: 'It's important for people in their teens to know what's going on in the world',  $M = 12.04$ ,  $SD = 11.68$ ,  $\alpha = .81$ ) with validated scales (Bollen & Hoyle, 1990; Duckworth et al., 2007; Liebenberg, Ungar, & LeBlanc, 2013; Pancer et al., 2007; Peterson, Speer, & McMillan, 2008; Romppel et al., 2013).

For contextual factors, we asked students to rate their communication frequency and change in communication with different people during the school closure with a five-point Likert scale. We also asked if students had certain interaction with their parent(s) such as 'go shopping', 'play games together' etc. A binary response was used in each activity with 1 (yes) and 0 (no), and the sum of responses to eight items were computed for analysis.

For students' subjective experience with the pandemic, we asked if they felt generally anxious about the pandemic and if their studies and emotions were bothered by the pandemic. Students also were asked to evaluate the impact of their longer time staying at home on their relationship with family members and their school-based studies, and the effectiveness of a list of 11 major anti-pandemic measures to combat the spread of the disease such as the 'compulsory quarantine for a period of 14 days upon arrival at Hong Kong', 'prohibition of any group gathering of more than four or eight persons in any public places during a specific period', etc.

We also documented students' demographic information such as sex and age, parental educational attainment, housing types, financial assistance scheme status, and their Wi-Fi ownership at home.

### 4. BIVARIATE ANALYSIS RESULTS

The objective of this study was to explore the factors associated with students' intrinsic motivation in coding education during the COVID-19 epidemic. We computed correlation coefficient of intrinsic motivation and a set of demographic variables (i.e., age, sex, parental educational attainment, etc.), psychosocial attributes (i.e., general self-efficacy, resilience, sense of community, etc.), and contextual factors that reflected students' lives and subjective experience on the pandemic during the months-long school closure.

The Pearson correlation coefficient indicated no significant relationships between sex, age, parental educational attainment, housing type and students' intrinsic motivation in coding education during the COVID-19 epidemic. However, a significant correlation was identified between students' financial assistance scheme status ( $\beta = -.18$ ,  $p < .05$ ), meaning that students from families receiving the Comprehensive Social Security Assistance tended to show lower intrinsic motivation in coding education. Besides, students' satisfaction of their internet condition at home was significantly correlated with their intrinsic motivation in coding education ( $\beta = .16$ ,  $p < .05$ ).

For psychosocial attributes, general self-efficacy ( $\beta = .21$ ,  $p < .01$ ), Grit ( $\beta = .15$ ,  $p < .05$ ), resilience ( $\beta = .43$ ,  $p < .01$ ), sense of community ( $\beta = .45$ ,  $p < .01$ ), and social responsibility attitude ( $\beta = .18$ ,  $p < .05$ ) were all found to be significantly related to students' intrinsic motivation to coding education during the COVID-19 epidemic, which means if students had higher level of beliefs about their capabilities to learn or perform behaviors at designated level (Bandura, 1986, 1997), higher level of effort perseverance and passion for long-term goals, higher ability to cope with a crisis, higher level of belonging to their communities, and more affirmative attitude toward commitment to the communities and civic engagement, then they would be more likely to evaluate coding education as interesting, enjoyable, and desirable during the COVID-19 epidemic.

Among different people, students' communication frequencies with parents ( $\beta = .17$ ,  $p < .05$ ), non-classmate friends ( $\beta = .17$ ,  $p < .05$ ), classmates ( $\beta = .19$ ,  $p < .01$ ), and teachers ( $\beta = .31$ ,  $p < .01$ ) were found to be significantly related to their intrinsic motivation. Although it is a relatively weak positive linear relationship, the correlation coefficient for the relationship between students' communication frequency with teachers and their intrinsic motivation is greater than others, indicating that even in a remote learning setting, teachers still play an important role. For students' perceived change in communication with different people, only the change in communication with friends ( $\beta = .16$ ,  $p < .05$ ) was found to be positively correlated with their intrinsic motivation, meaning that if students tend to perceive their communication with friends had increased during the school closure period, they will be



more likely to be intrinsically motivated. Parent-child interaction ( $\beta = .16, p < .05$ ) was also found to be significantly associated with students' intrinsic motivation in coding education. If students had done more activities with parents, such as went shopping, did physical exercises, went outing, talked about their studies, emotions, current affairs or family issues, they would display higher intrinsic motivation.

The subjective experience with the pandemic was also significantly related to students' intrinsic motivation in coding education. First, students who were more anxious about the pandemic showed greater intrinsic motivation. If they tended to agree they felt anxious about the pandemic ( $\beta = .29, p < .01$ ), or they tended to agree the pandemic had bothered their emotions ( $\beta = .16, p < .05$ ), they would be more likely to be intrinsically motivated. Second, if students tend to hold positive evaluation on the impact of their longer time staying at home on their relationship with family members ( $\beta = .18, p < .05$ ) and their school-based studies ( $\beta = .20, p < .01$ ), they will have more intrinsic motivation in coding education. At last, their evaluation on the effectiveness of various anti-pandemic measures ( $\beta = .27, p < .01$ ) was also found to be significantly correlated with the intrinsic motivation, which means if students tended to perceive the anti-pandemic measures were effective in protecting their health or facilitating their studies, they would display more intrinsic motivation in coding education.

## 5. DISCUSSION

The purpose of this study was to examine the influential factors associated with Hong Kong secondary school students' intrinsic motivation to coding education under the COVID-19 epidemic. The results revealed some links between students' socio-economic status, social environmental factors, subjective experience in the pandemic, and their intrinsic motivation.

### 5.1. More positive psychological states, higher intrinsic motivation

According to self-determination theory, it is essential for individuals to develop intrinsic motivation based on the satisfaction of three basic psychological needs: competence, autonomy, and relatedness (Ryan & Deci, 2000a). When students feel satisfied with these three psychological needs, they would become more intrinsically motivated, which in turn leads to higher level of engagement and learning outcomes. The psychosocial attributes concerned in this study covered most of these three basic psychological needs, and therefore supporting intrinsic motivation. Intrinsic motivation's relatively stronger relationship with sense of community compared to other variables is noteworthy. In this development project, it is expected that students would engage in collaborative and affective relationships with their classmates, teachers, and adult mentors to solve practical problems for community good and therefore the concept of relatedness is thought to hold importance for their well-being throughout the project.

### 5.2. More social interaction, higher intrinsic motivation

Variables of interaction with parents, teachers, non-classmate friends, and classmates were consistently found to be positively associated with higher intrinsic motivation. A plausible explanation could be related to their received social support from such interaction in both online and offline social encounters (Wang & Wang, 2013), and the perceived social presence of their teachers and classmates in imaging an online learning community. The idea that adolescents' supportive relationship with parents, teachers, and peers is related to their motivation at school is not new (Wentzel, 1998); however, in remote learning where teachers and peers no longer physically present, the subjective experience mediating such interaction and their intrinsic motivation is worthy further investigation.

### 5.3. More positive subjective experience with the crisis, higher intrinsic motivation

Many researchers have examined the role of anxiety in learning and their findings generally have shown that there is a negative relationship between anxiety and academic outcomes (Wolf & Smith, 1995). Past research often investigated anxieties to specific subjects and exams, and their relationship with certain academic outcomes, if experiencing the COVID-19 epidemic can be considered as a stressful event, then how feeling anxious about the pandemic affected academic outcome has been little addressed. Unexpectedly, a positive relationship between anxiety and students' intrinsic motivation was identified in this study, indicating that the current sample can use the anxious experience about the pandemic to motivate themselves. This implies that anxiety is not necessarily a handicap, in fact, students may be able to use anxiety as a source of self-motivation, and thereby alleviating negative issues brought about by anxiety and make the most of it. Further studies can look into how students experience and respond to anxiety may influence their academic outcomes.

Due to the reversal of COVID-19 pandemic, everything became uncertainties. If students perceived the longer time of staying at home has positive influence on their relationship with family members, and their school-based studies, in other words, if they perceived such longer time at home is beneficial for their family relationship, and they also could be adaptive to remote learning, they would have more intrinsic motivation in code learning during the pandemic. Besides, if they perceived the various anti-pandemic measures are effective in protecting them, they will be more intrinsically motivated. The positive subjective experience with the pandemic may reflect certain active coping strategies adopted by students to promote their sense of control over an uncertain time and safeguard their mental health, which therefore played a protective factor for their intrinsic motivation.

The results of the present study need to be interpreted with several limitations. First, the cross-sectional survey makes it difficult to evaluate the temporal relationship among the variables concerned, so it is not sufficient to establish a cause-and-effect relationship based on the available data. To remedy this, longitudinal data will be collected with the progressing of the project in the next two academic years, to further illuminate the direction of influence among these

variables. Another limitation is that we adopted a translation and back-translation process (Brislin, 1970) for the original scales that had not been previously used in Chinese language, in future studies, the full set of the scales will be validated.

## 6. ACKNOWLEDGEMENT

This study was funded by the Hong Kong Jockey Club Charities Trust (Project S/N# 2019/0114). The authors would like to thank the Hong Kong Jockey Club Charities Trust for the generous support in the development of Hong Kong youths' computational thinking skills and psychosocial skills, and their relational development and community involvement.

## 7. REFERENCES

- Bollen, K. A., & Hoyle, R. H. (1990). Perceived Cohesion: A Conceptual and Empirical Examination. *Social Forces*, 69(2), 479–504.
- Brislin, R. (1970). Back-translation for Cross-cultural Research. *Journal of Applied Psychology*, 1(3), 185–216.
- Cerasoli, C. P., Nicklin, J. M., & Ford, M. T. (2014). Intrinsic Motivation and Extrinsic Incentives Jointly Predict Performance: A 40-year Meta-analysis. *Psychological Bulletin*, 140(4), 980–1008.
- De Naeghel J., Van Keer H., Vansteenkiste M., Rosseel Y. (2012). The Relation between Elementary Students' Recreational and Academic Reading Motivation, Reading Frequency, Engagement, and Comprehension: A Self-determination Theory Perspective. *Journal of Educational Psychology*, 104(4), 1006–1021.
- Duckworth, A. L., Peterson, C., Matthews, M. D., & Kelly, D. R. (2007). Grit: Perseverance and Passion for Long-term Goals. *Journal of Personality and Social Psychology*, 92(6), 1087–1101.
- Jiang, S., & Wong, G. K. (2017, December). Assessing Primary School Students' Intrinsic Motivation of Computational Thinking. In *2017 IEEE 6th International Conference on Teaching, Assessment, and Learning for Engineering (TALe)* (pp. 469–474). IEEE.
- Lazowski, R. A., & Hulleman, C. S. (2016). Motivation Interventions in Education: A Meta-analytic Review. *Review of Educational Research*, 86(2), 602–640.
- Liebenberg, L., Ungar, M., & LeBlanc, J. C. (2013). The CYRM-12: A Brief Measure of Resilience. *Canadian Journal of Public Health*, 104(2), e131–e135.
- McAuley, E., Duncan, T., & Tammen, V. V. (1989). Psychometric Properties of the Intrinsic Motivation Inventory in A Competitive Sport Setting: A Confirmatory Factor Analysis. *Research Quarterly for Exercise and Sport*, 60(1), 48–58.
- Pancer, S. M., Pratt, M., Hunsberger, B., & Alisat, S. (2007). Community and Political Involvement in Adolescence: What Distinguishes the Activists from the Uninvolved? *Journal of Community Psychology*, 35(6), 741–759.
- Peterson, N. A., Speer, P. W., & McMillan, D. W. (2008). Validation of A Brief Sense of Community Scale: Confirmation of the Principal Theory of Sense of Community. *Journal of Community Psychology*, 36(1), 61–73.
- Romppel, M., Herrmann-Lingen, C., Wachter, R., Edelmann, F., Düngen, H. D., Pieske, B., & Grande, G. (2013). A Short Form of the General Self-Efficacy Scale (GSE-6): Development, Psychometric Properties and Validity in an Intercultural Non-clinical Sample and A Sample of Patients at Risk for Heart Failure. *GMS Psycho-Social-Medicine*, 10.
- Ryan, R. M., & Deci, E. L. (2000a). Self-determination Theory and the Facilitation of Intrinsic Motivation, Social Development, and Well-being. *American Psychologist*, 55(1), 68–78.
- Ryan, R. M., & Deci, E. L. (2000b). Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. *Contemporary Educational Psychology*, 25(1), 54–67.
- UNESCO (2020). School Closures Caused by Coronavirus (Covid-19). Retrieved August 15 2020, from <https://en.unesco.org/covid19/educationresponse>
- Wang, E. S. T., & Wang, M. C. H. (2013). Social Support and Social Interaction Ties on Internet Addiction: Integrating Online and Offline Contexts. *Cyberpsychology, Behavior, and Social Networking*, 16(11), 843–849.
- Wolf, L. F., & Smith, J. K. (1995). The Consequence of Consequence: Motivation, Anxiety, and Test Performance. *Applied Measurement in Education*, 8(3), 227–242.

# **STEM Learning in the Classroom**

# An Evolving Definition of Computational Thinking in Science and Mathematics Classrooms

Amanda PEEL<sup>1\*</sup>, Sugat DABHOLKAR<sup>2</sup>, Sally WU<sup>3</sup>, Michael HORN<sup>4</sup>, Uri WILENSKY<sup>5</sup>  
<sup>1,2,3,4,5</sup> Northwestern University, USA

amanda.peel@northwestern.edu, sugat@u.northwestern.edu, sally.wu@northwestern.edu, michael-horn@northwestern.edu, uri@northwestern.edu

## ABSTRACT

Computational Thinking (CT) curricula are increasingly being integrated into K-12 education across multiple subject areas. Our approach to this integration is to define Computational Thinking in terms of prevalent practices of professional disciplines. As our understanding of these practices evolve, so too must our operational definition of CT. Here we present a refined definition of CT in science and mathematics classrooms. Based on our extensive research designing and studying CT curricular units in collaboration with science and mathematics teachers, we have arrived at a working draft of a revised computational thinking in science and math taxonomy. We present the new version of taxonomy which has six revised categories of practices: computational modeling and simulation, computational visualization, algorithms, data practices, programming, and computational problem solving. We describe each category and how they are related.

## KEYWORDS

Computational Thinking, Science and Mathematics education, Taxonomy, Practices

## 1. INTRODUCTION

The concept of Computational Thinking (CT) has been evolving as researchers, educators, and policymakers devise new ways to support the development of computational literacy in K-12 education. One popular approach has been to integrate CT into core classes, such as science and mathematics (Heintz, Mannila, & Färnqvist, 2016). In this effort, our team developed a definition of CT as a taxonomy of practices specific to science and mathematics contexts (Weintrop et al., 2016). This taxonomy is comprised of four categories: data practices, modeling and simulation practices, computational problem-solving practices, and systems thinking practices. This taxonomy has been widely cited and used to frame many CT integrations (e.g., Ketelhut et al., 2020; Suters & Suters, 2020).

Our team has used this taxonomy since its publication to create curricular units for middle school and high school students and provide professional development for in-service teachers (<https://ct-stem.northwestern.edu>). Our extensive research regarding designing CT-integrated curricula, teacher practices, and student learning using the initial version of taxonomy has prompted us to revise it into a second version (Swanson, Anton, Bain, Horn, & Wilensky, 2019; Arastoopour Irgens, et al., 2019; Peel, Dabholkar, Anton, Wu, Wilensky, & Horn, 2020; Dabholkar, Arastoopour Irgens, Horn, & Wilensky, 2020). In this version, we make two major changes that address issues that emerged from our work with the taxonomy.

First, we identified three new practices that are central to CT in science and mathematics but were not explicitly developed into separate categories in the first taxonomy: algorithms, programming, and visualization practices. We realized that creating separate categories for these practices is important for integration and scaffolding in CT-integrated curricula. For example, students sometimes struggle to program and understand algorithms when creating computational models. The inclusion of algorithm and programming practices as top-level categories support skills and knowledge necessary to engage in computational creation. These categories make these skills and associated knowledge explicit in the conceptualization of CT in science and math. Moreover, algorithms are key tools used in modern science and math, and algorithms cannot be implemented without programming. We also broadened our account of visualization to include more diverse set of representations and practices.

Second, we revised the prior four practices to improve clarity in terms of their interpretations for curricular integration. After working with teachers for several years and iterations of professional development and co-design, we found there were issues with the interpretation of some of the practices. For example, many teachers see “data practices” and confound it with traditional science data practices, rather than computational data practices. Similarly, we found that systems thinking practices were often disconnected from computational thinking. In these cases, the development of systems thinking and data practices in science classrooms are essential, but the activities used to teach these are not always CT. We take the position that systems thinking practices are developed with CT when students engage with computational models. As such, we combined the computational modeling practices and systems thinking practices.

To streamline the sub-practices within each overarching category, we re-designed the sub-practices to represent a spectrum within each category. In this way, the sub-practices increase in sophistication from top to bottom and are self-contained within each practice category. To shape the spectrum, we created sub-practices that represent a use, modify, assess, and create spectrum (modified from Lee et al., 2011). Students can use a computational tool to understand a phenomenon, engage in investigations, or solve a problem. However, this represents the baseline for CT practices within the taxonomy. While these practices support student learning, we argue it is essential for the development of CT practices and knowledge to move beyond the use of computational tools to the modification, assessment, and creation of computational tools.

Our second version of the taxonomy of practices is an evolving draft, and formal feedback will be collected from stakeholders in the future. The goal of this paper is to describe the current draft of the second taxonomy of CT in science and mathematics practices and how these practices are operationalized in the classroom. Our final goal is to provide a taxonomy that supports the integration of CT in K-12 science and mathematics classrooms.

## 2. NEW TAXONOMY DESCRIPTION

Integration of CT into science and mathematics curricula is intended to promote: 1) understanding the ubiquity and importance of computing in STEM, 2) understanding how modern STEM professionals use computing, 3) access to computing-related content and practices for all students, and 4) science and mathematics learning in new and deeper ways. There are six practice categories represented as columns in the taxonomy: computational modeling and simulation, computational visualization, algorithms, computational data practices, programming, and computational problem-solving (Figure 1). We view the sub-practices within each category as target competencies for students by the end of their K-12 education.

Computational Modeling and Simulation Practices	Computational Visualization Practices	Algorithm Practices	Computational Data Practices	Programming Practices	Computational Problem-Solving Practices
Using computational models to understand a complex phenomenon	Using a computational visualization to understand a phenomenon	Using an algorithm to solve a problem or understand a phenomenon	Using computation to collect and create data	Reading and understanding code	Choosing effective computational tools to solve a problem
Using computational models to hypothesize and test predictions	Using a computational visualization to identify and predict trends	Selecting an appropriate algorithm to solve a problem	Using computation to transform, manipulate, and clean data	Modifying code	Using a computational tool to solve a problem
Using a computational tool to understand a system's components and dynamics	Assessing computational visualizations	Assessing algorithms	Using computation to analyze data	Writing elegant, readable, and maintainable code	Preparing problems for a computational solution
Using a model to understand how positive and negative feedback function and impact a complex system	Modifying a computational visualization to better fit a phenomenon/data	Modifying an algorithm to better address a problem	Using computation to explore and draw insight from large data sets	Debugging programs	Decomposing complex problems into smaller solvable pieces
Assessing computational models	Designing and constructing computational visualizations	Designing and constructing algorithms	Modifying a computational approach to better fit data	Developing modular solutions and abstractions	Systematically troubleshooting a solution
Modifying a computational model to better fit a phenomenon				Iteratively designing and testing programs	Modifying a computational tool to solve a problem
Designing and constructing computational models					Creating a computational tool to solve a problem

Figure 1. Taxonomy of CT Practices in Science and Mathematics

There are inherent linkages between the practice categories, in that some sub-practices require the use of other sub-practices and several sub-practices can be used in any one activity. Linkages between practices are discussed in more detail in section 3 of this document. The practice categories are structured with a use, modify, assess, and create spectrum. As students engage in practices on this spectrum, the practices become more sophisticated. Engaging in practices related to modifying and creating begins to require engaging in practices from other categories. For example, creating a computational model requires programming practices and algorithm creation practices. The following sections describe each practice category.

### 2.1. Computational Modeling and Simulation Practices

Computational models and simulations are useful tools to understand complex systems and reason about phenomena. Students competent in these practices will be able to use, modify, assess, and create dynamic computational models in order to understand complex phenomena and solve problems. Computational models and simulations in this

category are conceptualized as dynamic models and simulations, not static models. Examples of classroom computational modeling tools include NetLogo (Wilensky, 1999), SageModeler (2020), etc. When engaging students in computational modeling practices, models can be used to understand a phenomenon, to understand a system's dynamics, and to test predictions and hypotheses. Simple animations of phenomena are not considered computational models or simulations because they have neither parameters that students can tinker with and alter when running the model or simulation nor can they be modified or elaborated.

We argue that using, modifying, assessing, and creating computational models supports the development of systems dynamics competencies. Systems dynamics competencies may include understanding: 1) positive and negative feedback and their impact on the system, 2) stocks and flows of a system, 3) micro and macro levels and how changes in the micro level impact the macro level, and 4) emergent phenomena. People have mental models that represent their understanding of a phenomenon or system. When people simulate that mental model of the system in their minds, their reasoning is often incomplete or fails to match the system in all of its complexity (Forrester, 1993). Using computational models to simulate systems and phenomena allows students to reason about systems in more sophisticated and nuanced ways. These computational simulations can then impact students' mental models and bring their understanding of the system and phenomenon more in line with canonical definitions and representations.

Designing and constructing computational models allows for deeper understanding of phenomena (Wilensky & Reisman, 2006). Computational models can take on many forms and focus on different aspects of a phenomenon. For example, one might use an agent-based model (e.g., NetLogo) to describe interactions between many things at a micro level while a systems dynamic model (e.g., SageModeler) might be used to describe interactions at higher system level. Assessing computational models can be done at every stage (using, modifying, and creating). Students should be able to assess the limitations and affordances of a model, the simplifications made in the model, and how well the model represent reality. Assessing computational models also includes meta-modeling knowledge, such as understanding how models are used in science and mathematics, understanding the value of computational models, and critically reflecting on how models are used are interpreted.

### 2.2. Computational Visualization Practices

Visualization is a metacognitive skill in science and science education (Gilbert, 2005) as well as mathematics education. Students competent in these practices will be able to use, modify, assess, and create computational visualizations in order to understand and represent complex phenomena, analyze and interpret data, and solve problems. We conceptualize computational visualizations as graphs, tables, diagrams, static models, models (dynamic) as long as they are made within a computational medium. This can range from something that is completely made by the computer (e.g., graphs in NetLogo), to something made by the user with a computational tool (e.g., graphs in Microsoft

Excel), to something made through programming (graphs in R or Python). This also includes technical drawings used for creating computational drawings and designs.

When students engage in computational visualization practices, they think about how to represent a phenomenon, or a part of the phenomenon. A visualization can be used as a way to think about and understand a phenomenon in different and new ways, especially when the visualization provides a new way of looking at the phenomenon. Computational visualizations can be used for identifying and predicting trends. Assessing computational visualization can be done at every stage (using, modifying, and creating). Students should assess the affordances and limitations of specific kinds of visualizations and be able to choose tools based on knowledge of strengths and weaknesses of visualization types.

### 2.3. Algorithm Practices

The ability of researchers to make sense of large amounts of data often comes down to the sophistication of algorithms available to process those data. Algorithm practices involve using, modifying, assessing, and creating algorithms to solve problems and understand phenomena. Much of the computational power harnessed in STEM comes from using computation to complete a multitude of small tasks times in a short period of time and to make such work as computationally efficient as possible. In the case of K-12 education, we expect that students can understand and use algorithmic logic and concepts, such as loops, conditionals, logic, procedures, recursion, and variables.

Algorithms can be developed and implemented in classrooms using a variety of formats ranging including unplugged approaches, block-based coding environments, or interactive computational notebooks (e.g., Jupyter, <https://jupyter.org/>). Algorithms can be assessed at each stage (using, modifying, and creating). Students can assess correctness by asking, “Does the algorithm accomplish the task?” or “How well does the algorithm accomplish the task?” Efficiency can be assessed by asking, “Can the algorithm complete the same task in less steps or more concise steps?”, “How much time does it take to run the algorithm?”, and “How much memory does the algorithm require?” For example, when the number of agents in an agent-based computational model are doubled, does it take twice as long (linear) to run? Four times as long (quadratic) to run? Students can also assess clarity by asking, “Is the algorithm written clearly?” and “Can others understand my algorithm?” While the initial algorithm writing process can be messy, the final algorithm should be as clear as possible. There are also opportunities to critically reflect on the role of algorithms in society.

### 2.4. Computational Data Practices

Data practices are central to scientific inquiry and mathematics. As data sets become larger and calculations become more complex, computational tools can help in a range of ways including data collection, cleaning, transformation, analysis, and visualization. As such, it is important that students learn computational data practices. Students engage in computational data practices when they data create, collect, manipulate, and analyze data with

computational tools. This can range from using pre-programmed algorithms to writing code in order to complete the data-related task. Students with computational data practice competencies will be able to use, modify, and create computational approaches and tools to collect, manipulate, visualize, and analyze data.

### 2.5. Programming Practices

We define programming as the act of writing code on a computer, creating a distinct practice category from algorithm practices, which include a broader range of practices related to, but beyond coding. Programming practices are central to designing and constructing computational tools. Much of the work of STEM professionals consists of modifying existing tools and approaches and coding new tools and approaches to meet their needs. Students with programming competencies will be able to read, understand, and write code in order to solve problems and understand phenomena. Learning to program involves reading and understanding existing code, learning to modify code to meet the user’s needs, and learning to write code. Writing code involves testing and debugging the code, developing abstractions, and iteratively testing and designing the program. When writing code, attention should be given to the creation of readable and maintainable code. Following programming conventions and using comments to annotate the code is key for readability and maintenance. Comments can communicate a programmer’s intent, which may or may not be clear in the code itself.

### 2.6. Computational Problem-solving Practices

Computational tools have become useful to solve problems in science and math. Engaging in these practices should help students solve problems with computational tools and approaches and understand computation’s role in scientific and mathematic problem solving. Computational problem-solving practices involve using computation to solve a problem. This requires understanding different approaches to solve the problem computationally, and the ability to choose a computational tool or approach that is appropriate and effective in solving the problem. Problems often have to be decomposed into smaller solvable pieces and prepared for a computational solution (e.g., raw data may need to be manipulated in order to run an algorithm that solves the problem). Computational solutions to problems also need to be iteratively designed, tested, and troubleshoot in order to solve the problem effectively.

Students should engage in these practices with different computational tools to support skills and understandings of specific tools and how they can be used in different problem contexts. For example, students can create a computational model in order to predict how a new pesticide will impact crop yield. When creating the model, students engage in computational modeling practices, algorithm practices, and programming practices. Then, using the model to predict the yield outcomes of using the pesticide engages students in computational data and modeling practices. We propose that when students use computational tools to solve a problem, they may better understand the purpose of those tools and how scientists and mathematicians use them.

## 3. LINKAGES BETWEEN PRACTICES



While each practice category represents a contained spectrum of knowledge and skill specific to that category, there are inherent linkages between practice categories. A CT-integrated activity can potentially engage students in a combination of interlinked sub-practices. In some cases, it is impossible to avoid separating multiple sub-practices from different practice categories. For example, the modify and create sub-practices within computational modeling, computational visualizations, and computational data practices require algorithm and programming practices. Designing and constructing computational models involves designing and constructing algorithms and programming practices when coding the model. There is a linkage between programming practices and others when reading or writing code is involved. For example, reading the code of a computational model engages students in programming practices and computational modeling practices.

There are linkages between data and visualization practices when a computational visualization is made to analyze data (e.g., graphing data to identify trends) or to collect data (e.g., collecting data from a computational model). There are also linkages between computational visualization and computational modeling practices because computational models are a type of computational visualization. However, not all computational visualizations are computational models. Students use a computational visualization when they use a computational model. Students modify a computational visualization when they modify the visual aspects of a computational model (e.g., how agents look, how to represent the phenomenon, graphing data from the model). Students design and constructing a computational visualization when they design and construct a computational model. Computational problem-solving practices are linked with other practices when the practices are used to solve a problem. For example, if students use, modify, or create an algorithm to solve a problem, they are engaging in both algorithm and problem-solving practices.

While designing CT integrated curricular activities, we recommend viewing the practices as interconnected sets and not in isolation. The 10 new CT-integrated science and math units designed with this draft taxonomy will be available on the project webpage after implementations are complete (<https://ct-stem.northwestern.edu/>).

## 4. CONCLUSION

This paper has presented a draft second version of a taxonomy of practices that defines CT in science and mathematics classrooms. The six practice categories represent CT practices specific to science and mathematics contexts. We have expanded and revised the taxonomy to include more key practices and clarify their roles in the classroom context. We plan to present this draft taxonomy to stakeholders, including practicing scientists and mathematicians, teachers with CT experience, and CT researchers. The feedback will inform another round of revisions, and a final version of the revised taxonomy will be disseminated. We have piloted this version of the taxonomy with a professional development program that resulted in 10 new CT-integrated science and mathematics units designed by teacher-researcher co-design teams,

which are currently being implemented in schools.

Our goal is to define and characterize CT in science and mathematics contexts. We believe this will help facilitate the integration of CT and provide a resource for those who are unfamiliar with CT practices. Further, we believe the taxonomy can help shape CT in science and mathematics practices, which may enhance student outcomes related to authentic science and mathematics learning.

## 5. REFERENCES

- Arastoopour Irgens, G., Dabholkar, S., Bain, C., Woods, P., Hall, K., Swanson, H., Horn, M.S., Wilensky, U. (2020). Modeling and Measuring High School Students' Computational Thinking Practices in Science. *Journal of Science Education and Technology*, 29(1), 137-161.
- Dabholkar, S., Arastoopour Irgens, G., Horn, M., & Wilensky, U. (2020). Students' epistemic connections between science inquiry practices and disciplinary ideas in a computational science unit. *Proceedings of the International Conference for the Learning Sciences (ICLS 2020)*, Nashville, USA: ISLS.
- Forrester, J. W. (1993). System dynamics and the lessons of 35 years. In *A systems-based approach to policymaking* (pp. 199-240): Springer.
- Heintz, F., Mannila, L., & Färnqvist, T. (2016). A Review of Models for Introducing Computational Thinking. *Computer Science and Computing in K-12 Education*.
- Ketelhut, D. J., Mills, K., Hestness, E., Cabrera, L., Plane, J., & McGinnis, J. R. (2020). Teacher change following a professional development experience in integrating computational thinking into elementary science. *Journal of Science Education and Technology*, 29(1), 174-188.
- Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *Acm Inroads*, 2(1), 32-37.
- Peel, A., Dabholkar, S., Anton, G., Wu, S., Wilensky, U., & Horn, M. (2020). A Case Study of Teacher Professional Growth Through Co-design and Implementation of Computationally Enriched Biology Units. *Proceedings of the International Conference for the Learning Sciences (ICLS 2020)*, Nashville, USA: ISLS.
- SageModeler [Computer software]. (2020). Concord, MA: The Concord Consortium and the CREATE for STEM Institute at Michigan State University.
- Suters, L., & Suters, H. (2020). Coding for the Core: Computational Thinking and Middle Grades Mathematics. *Contemporary Issues in Technology and Teacher Education*, 20(3), 435-471.
- Swanson, H., Anton, G., Bain, C., Horn, M., & Wilensky, U. (2019). Introducing and assessing computational thinking in the secondary science classroom. In *Computational Thinking Education* (pp. 99-117): Springer.
- Wilensky, U. (1999). NetLogo [Computer Software]. Retrieved December 1, 2019, from <http://ccl.northwestern.edu/netlogo/>
- Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology*, 25(1), 127-147.
- Wilensky, U., & Reisman, K. (2006). Thinking like a wolf, a sheep, or a firefly: Learning biology through constructing and testing computational theories—an embodied modeling approach. *Cognition and instruction*, 24(2), 171-209.

# Action Research on Engineering Design-oriented and Project-based STEM Teaching Model

Hong YU<sup>1</sup>, Lu ZOU<sup>2\*</sup>

<sup>1,2</sup> South China Normal University, China  
454534852@qq.com, zoulu\_98@qq.com

## ABSTRACT

From the perspective of engineering design and based on the LBD's Binary Cycles Model, this study constructed the engineering design-oriented and project-based STEM teaching model after three rounds of action research. This model includes four process modules as the startup, the preparation, the practice and the summary of the engineering project. More, it deconstructs the behavioral activities in each module from the perspective of the teachers and the students. Teachers mainly carry out the teaching activities of establishing situation, describing task, providing scaffold, guiding methods, evaluating artifacts and summary/migrating, while students carry out the learning activities of understanding challenges, clarifying projects, investigation/inquiry, engineering design, demonstration/communication and reflection/improvement. This model provides a new paradigm for the implementation of STEM teaching in basic education, and the results proved that students' STEM literacy was improved, especially in engineering and technology.

## KEYWORDS

STEM, STEM teaching model, engineering design, project-based learning

## 1. INTRODUCTION

STEM education is a new educational form that organically integrates four disciplines of Science, Technology, Engineering and Mathematics, which is conducive to the cultivation of the core skills of talents in the 21st century and enhancing the talents' competitiveness. Since the Undergraduate Science, Mathematics and Engineering Education report issued by the National Science Board (NSB) firstly proposed the concept of "science, mathematics, engineering, and technology" (National Science Board, 1986), countries around the world have gradually incorporated STEM education into their talent training program and curriculum system. China also attaches great importance to the localization of STEM education. However, basic education in China has long existed in the phenomenon of "teacher-oriented", "theory rather than practice" and so on, as well as the curriculum lacks engineering education content connected with higher education. As the result, students' motivation to learn and innovate cannot be effectively stimulated, which is contrary to the original intention of STEM education. Studies have pointed out that engineering design-oriented STEM project teaching is conducive to promoting students' interest in STEM subjects and STEM careers (Shahali et al., 2016). Therefore, it's necessary to carry out project-based STEM education oriented by engineering design in primary and secondary schools from the perspective of talent

cultivation, which is intended to improve talents' comprehensive literacy. Furthermore, the research question is: How to implement the engineering design-oriented and project-based STEM teaching?

## 2. RELATED WORK

### 2.1. Engineering Design and STEM Education

From the perspective of teaching strategies, STEM education can be divided into two orientations currently: scientific inquiry orientation and engineering by design orientation. Scientific inquiry-oriented STEM education focuses on the generation of intellectual outcomes and provides a standardized thinking way to solve scientific problems; engineering design-oriented STEM education is a practical process of applying engineering methods to solve practical problems, which is a real sense of putting science into practice, and its main activities are design, manufacturing and improvement (Huang et al., 2020), focusing on the generation of learning outcomes in materialized form. In the study of curriculum reform, engineering-oriented STEM courses have proven to be the most appropriate form of implementing the concept of integrated STEM education, containing three core elements as contextual learning, engineering design, and scientific inquiry (Xie et al., 2017). The typical curriculum design model with engineering design orientation is the 6E design- based learning model proposed by the International Technology and Engineering Educators Association (ITEEA). This model combines the thinking of scientific inquiry with the practice of engineering design, which mainly includes six stages as engagement, exploration, explanation, engineering, enrichment and evaluation (Barry N B., 2014).

### 2.2. Project-based Learning and STEM Education

Thomas argues that Project-based learning is an experiential learning approach that engages students in projects and is an active learning that allows students to promote their own understanding of an area of knowledge (Thomas, 2000). He defines five characteristics of project- based learning: (1) The projects are the core to the curriculum; (2) A method to guide students' understanding of the core disciplinary knowledge; (3) Students' activities involve constructive investigation; (4) A certain degree of students' drive; (5) The topic, context, and tasks of the project based on real situations. Participation in project- based STEM courses influences students' attitudes toward STEM skills, practical values, and career aspirations (Beier et al., 2018). A STEM project design model was proposed based on the constructivist perspective, which takes the "project or problem" as the core point and focuses on the corresponding intensive practice and summary

improvement after the completion of the project (Yu & Hu, 2015).

Based on the above theories, this study combining engineering design concepts and STEM project-based teaching methods constructs an engineering design- oriented and project-based STEM teaching model and conducts three rounds of action research, with the intention of exploring a STEM teaching model in basic education suitable for China's national conditions, and opening up new paths for articulation with higher education.

### 3. PRIMARY CONSTRUCTION

In order to promote deep and sustained interdisciplinary learning and develop students' ability to solve complex and unstructured problems, Kolodner (2002) proposed the "Design & Exploration" double- cycle model of Learning By Design™ (LBD), which consists of Design/Redesign cycle and Investigate & Explore cycle. The Design/Redesign cycle includes all activities needed by

completing a design task, such as planning design, understanding challenge, presenting & sharing gallery walk, analyzing & explaining, construction & test, Presenting & sharing pin-up session, while the Investigate & Explore cycle includes designing investigation, making hypothesis, clarifying question, presenting & sharing poster session, analyzing results, and conducting investigation, which are a series of investigation activities based on specific design content. In addition, "Need to Do" and "Need to Know" are the links between the two circles. The model integrates multiple designs and multiple investigations, which means that integrates doing practice with learning knowledge, which is in line with the engineering and project-based STEM teaching concept. Based on the LBD's Cycles by Kolodner, this research constructs the engineering design-oriented and project-based STEM teaching model from the perspectives of both teachers and students. The preliminary construction results are shown in Figure 1.

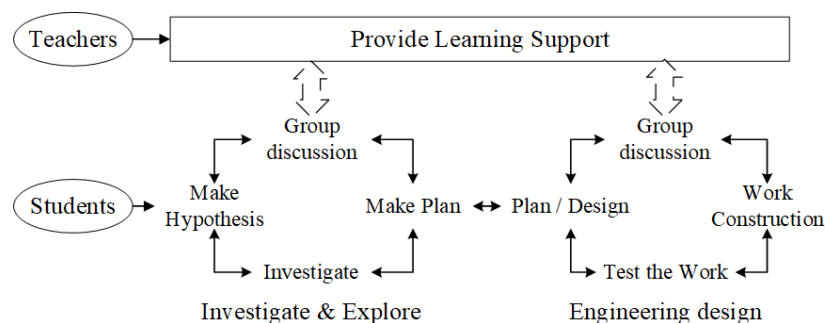


Figure 1. The Preliminary Construction of Engineering Design-oriented and Project-based STEM Teaching Model.

The preliminary construction of engineering design-oriented and project-based STEM teaching model takes a specific engineering project as the starting point, and students engage in STEM learning through "Investigate & Explore" circular process and "Engineering Design" circular process, with the works constructed by students being the final resulting output. The main purpose of "Investigate & Explore" process is to let students acquire the knowledge needed to complete this engineering design and know "how to do it", which mainly includes the steps of making hypothesis, brainstorming of group discussion, making investigation plan and implementing investigation; In addition, "Engineering Design" process's main purpose is to let students construct project works through design and hands-on practice, and discover "what they need to know" in the practice before obtaining the required knowledge through investigation and exploration again. It mainly includes the steps of planning/designing the engineering projects, brainstorming of group discussion, constructing engineering design works, and testing the works. In addition, reflection is integrated throughout the students' activities, and teacher's main task is to provide support services such as learning guidance for the students' learning.

## 4. METHODOLOGY

### 4.1. The Research Object

In this study, students from grade 4 to grade 6 in a primary school in Guangzhou province were selected as the object

of action research. The school is well-equipped with the infrastructure for STEM teaching. In addition, according to Piaget's theory of cognitive development, primary school students have some logical thinking skills, but still need the support of specific content when they engage in thinking activities. Engineering design-oriented and project-based STEM teaching allows students to carry out project-based engineering design activities in specific contexts, which meets the demands of students' development of hands-on practical ability, problem-solving ability and innovative thinking.

### 4.2. The Research Method

Action research is an important method for educational research, and its conducting process is a spiral cycle consisting of the four components as planning, action, observation and reflection. Educational action research is a research method based on a certain purpose and plan to systematically investigate specific issues in educational action for the purpose of improving the effectiveness of educational action (Kemmis & Zhang, 1994). In this study, three rounds of action research were used to conduct an exploration of an engineering design-oriented and project-based STEM teaching model.

### 4.3. The Research Process

#### 4.3.1. The First Round of Action Research

Research Objective: To analyze the process module of engineering design-oriented and project-based STEM teaching model.

Teaching Content: “The Design of Sweeping Robot” (4 lessons)

Research processes: (1) Planning: Make an instructional design plan for “The Design of Sweeping Robot” to summarize the process modules of teaching; (2) Action: Carry out STEM teaching practice according to the developed instructional design; (3) Observation: The main processes of the classroom; (4) Reflection: Reflect on whether the process modules of the engineering design-oriented and project-based STEM teaching model are reasonable according to the classroom videos and teaching logs.

Summary: After the first round of action research, we found that the engineering design-oriented and project-based STEM teaching model can be divided into four main processes: project startup, project preparation, project practice and project summary, so that the teaching model can be initially divided into four modules.

#### **4.3.2. The Second Round of Action Research**

Research Objective: To analyze the pedagogical elements of teachers in engineering design-oriented and project-based STEM education.

Teaching content: “the Making of Arduino Light Painting” (4 lessons)

Research processes: (1) Planning: Make an instructional design plan for “the Making of Arduino Light Painting” to analyze the teaching elements; (2) Action: Carry out STEM teaching practice according to the developed instructional design; (3) Observation: teachers’ classroom behaviors; (4) Reflection: Reflect on whether the summary of teachers’ teaching activities in the STEM education is reasonable according to classroom videos and teaching logs.

Summary: After the second round of action research, we found that the preliminary teaching activities of teachers lack the process of “providing learning scaffolds”. Learning scaffolds should be provided for students’ investigative exploration and engineering design in the project preparation session following the description of the task in order to guide them to explore more possibilities in the heterogeneous problems.

#### **4.3.3. The Third Round of Action Research**

Research Objective: To analyze the composition of the students’ activities in engineering design-oriented and project-based STEM education.

Teaching Content: “Building A Bridge for Bay Area” (4 lessons)

Research processes: (1) Planning: Make an instructional design plan for “Building A Bridge for Bay Area” to analyze the activity composition of students; (2) Action: Carry out STEM teaching practice according to the developed instructional design; (3) Observation: students’ learning activities; (4) Reflection: Reflect on whether the division of the elements of the students’ activities in engineering design-oriented and project-based STEM education is reasonable according to classroom videos and the classroom observation forms.

Summary: After the third round of action research, we found that students’ learning activities lack the transfer and improvement link after summary and the adjustment link after reflection. Therefore, homework should be assigned at the end of the class so that students can consolidate the knowledge and skills they have learned as well as seek the expansion and deeper construction of knowledge. In addition, students should not only to reflect on the whole learning process, but also apply the reflection results to the practice and conduct engineering project practice again.

## **5. MODIFICATION OF MODEL**

### **5.1. Engineering Project Startup**

Establish situation and understand challenges: Situational learning is one of the characteristics of engineering-oriented STEM education. Therefore, engineering design-oriented STEM projects need to be carried out in a specific context. Teachers need to create specific engineering situations based on reality before students begin to study STEM projects, so that students can relate to their own experience and understand the problems or challenges they will face in the project.

Describe the task and clarify project: Teachers can assign tasks and make a certain task description after students understand the project’s background and challenges in a specific context, so that students can make clear to the theme of the STEM study and the specific project they need to undertake.

### **5.2. Engineering Project Preparation**

Provide scaffolds and investigate & Explore: After assigning tasks, teachers should provide certain support services for students’ learning, such as listing questions and providing examples for reference, so that students can conduct investigation & exploration activities in a targeted and structured way after clarifying specific engineering projects. During the link of investigation & exploration, students firstly put forward relevant hypotheses based on the specific issues, and conduct brainstorming in the form of group discussion to determine the hypothesis and formulate a specific investigation plan, finally conduct investigation according to the plan with adjusting the behaviors of each link through constant reflection, laying the knowledge foundation for the subsequent practical sessions of the project.

### **5.3. Engineering Project Practice**

Method guidance and engineering design: In this part, students first plan and design the project based on the acquired knowledge, then modify the planning and design as well as construct the works with group discussion, finally test the practical results and optimize them. In the entire processes, students continue to reflect on each stage and modify the actions of the previous stage, while teachers are responsible for providing methodological guidance.

### **5.4. Engineering Project Summary**

Evaluate the works, demonstrate and communicate: Students present the constructed works, and each group will conduct mutual evaluation and exchange experience,

while teachers are responsible for reviewing students' works and giving feedback and suggestions.

Summarize and migrate, reflect and improve: Teachers comment and summarize the constructed works displayed by students, and finally assign extracurricular homework

for students to transfer knowledge; Students communicate with other groups' members when they show their works, and finally they reflect on the deficiencies in the whole project, modify the deficiencies, and re-practice the engineering design after class.

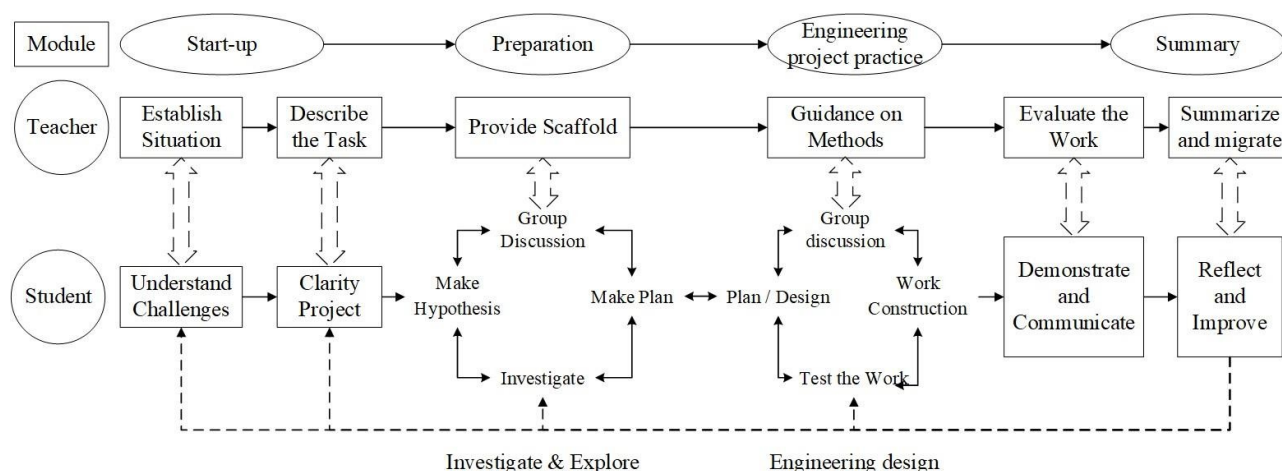


Figure 2. The Engineering Design-oriented and Project-based STEM Teaching Model.

## 6. CONCLUSION

After three rounds of action research, we constructed an evaluation index system for the teaching effect of cultivating STEM literacy from three dimensions: STEM knowledge, STEM skills and STEM emotional attitude. Then 30 students who were action research object of the third round were asked to evaluate the STEM teaching results of "Building A Bridge for Bay Area". As the result, 28 of them showed significant improvement in STEM knowledge and skills, especially in the field of engineering and technology, and they showed great interests in engineering design-oriented and project-based STEM education as they gained a great sense of achievement in engineering project practice. However, the remaining 2 students barely participated in the overall STEM learning activities because they didn't like learning and lacked social skills, as the interviews showed.

In conclusion, this study summarizes an engineering design-oriented and project-based STEM teaching model based on the LBD's Cycles by Kolodner after three rounds of action research, which provides a new paradigm for STEM teaching development in basic education. However, the validity of the model still needs to be validated on a larger scale due to the current limitation of the sample size. Furthermore, increasing students' engagement in engineering design-oriented and project-based STEM learning is also a factor that should be considered.

## 7. REFERENCES

- Barry N B. (2014). The ITEEA 6E Learning by DeSIGNTM Model. *Journal of Technology and Engineering Teacher*, 2014(3), 14-19.
- Beier, M. E., Kim, M. H., Saterbak, A., Leautaud, V., Bishnoi, S., & Gilberto, J. M. (2018). The effect of authentic project-based learning on attitudes and career aspirations in STEM. *Journal of Research in Science Teaching*, 1-21.
- Huang, X. D., & Yu, R. W. (2020). STEM Engineering Teaching Model: Meaning, Construction and Application. *Chinese Journal of Educational Science Research*, 2020(7), 60-66.
- Kemmis, S., & Zhang, X. Y. (1994). Action Research. *Chinese Journal of Educational Science Research*, 1994(4), 32-36.
- Kolodner, J. L. (2002). Learning by Design™ Iterations of Design Challenges for Better Learning of Science Skills. *Journal of Cognitive Studies*, 9(3), 338-350.
- National Science Board. (1986). *Undergraduate Science Mathematics and Engineering Education*. Retrieved October 12, 2020, from <http://www.nsf.gov/nsb/publications/1986/nsb0386.pdf>
- Shahali, E. H. M., Halim, L., Rasul, M. S., Osman, K., & Zulkifeli, M. A. (2016). STEM Learning through Engineering Design: Impact on Middle Secondary Students' Interest towards STEM. *Eurasia Journal of Mathematics, Science and Technology Education*, 13(5), 1189-1211.
- Thomas, J. W. (2000). *A review of research on project-based learning*. Retrieved from <http://www.dl.icdst.org/pdfs/files1/aac48826d9652cb154e2dbf0033376fa.pdf>
- Xie, L., & Li, C. M. (2017). The Study of Curriculum Reform based on Integrated STEM Education. *Chinese Journal of Curriculum, Teaching Material, and Method*, 37(6), 63-68+62.
- Yu, S. Q., & Hu, X. (2015). STEM Education and Its Model for Interdisciplinary Integration. *Chinese Journal of Open Education Research*, 21(4), 13-22.

# A Case Study of 7<sup>th</sup> Grade Students Learning Programming to Solve Mathematics Problems

Wendy HUANG<sup>1\*</sup>, Chee-Kit LOOI<sup>2</sup>, Mi Song KIM<sup>3</sup>

<sup>1, 2</sup> National Institute of Education, Nanyang Technological University, Singapore

<sup>3</sup>University of Western Ontario, Canada

wendy.huang@nie.edu.sg, cheekitlooi@nie.edu.sg, mkim574@uwo.ca

## ABSTRACT

Unlike previous research that focused on transfer of cognitive skills gained in programming to problem solving, we contend that students can learn programming to directly solve mathematics problems. Our study begins a line of inquiry on whether computational thinking (CT) can be considered a distinct approach for mathematics problem solving in schools. This paper presents analysis of an episode that featured two students who conceptualized and coded an algorithmic solution to a textbook word problem. A comparison is made with the standard deductive approach. We generalize the differences in the two approaches in terms of knowledge types. The case is paradigmatic of a broader phenomenon in which CT makes a difference in how students approach problem solving in school mathematics.

## KEYWORDS

computational thinking, mathematics education, problem solving, computer programming, secondary education

## 1. INTRODUCTION

Enthusiasm around teaching programming in connection to mathematics led researchers to empirically investigate the effect of computer programming on K-12 mathematics learning. In 1989, McCoy and Dodl published a quantitative study of 800 high school students that concluded transfer of skills from computer programming experience to mathematical problem solving. In 1995, Yelland reviewed research on the LOGO experiments conducted in the 1980s and found mixed results on cognitive gains in mathematics achievement and problem solving. Schanzer et al (2018) reported that the Bootstrap programming curriculum improved students' ability to solve algebra word problems on pre/post-tests. In their review of 15 studies, Forsström and Kaufmann (2018) found that under certain circumstances, programming in mathematics education could improve student motivation and performance in mathematics. There have been many studies relating programming and learning mathematics concepts, especially in geometry (e.g., Sung et al., 2020; Benton et al., 2017), but very few have specifically related programming to school mathematics problem solving.

Unlike previous research that focused on transfer or bridging of cognitive skills gained in programming to problem solving, we contend that students can learn programming to directly solve mathematics problems. Therefore, our study begins a line of inquiry on whether CT can be considered a distinct approach for mathematics problem solving in schools, such that students formulate

problems computationally and then effectively solve them by writing programs. We also contribute to designing for such learning in secondary level education.

## 2. RESEARCH QUESTIONS

In design-based research (DBR), we strive to engineer certain learning outcomes while “building theories about why designs work and how to adapt them to new circumstances” (p. 9, Cobb, et al., 2003). In our study, we designed instructional materials to support students learning programming to solve mathematics problems in order to answer the following questions:

1. What counts as evidence of CT?
2. Does CT make a difference in how students approach mathematics problems?

## 3. INTERVENTION DESIGN

The instructional design being tested was based on four principles.

### 3.1. CT for programming. Programming for math.

Among many contested definitions of CT, we chosen one that aligns well with learning to program. CT is defined as “the thought process involved in formulating a problem and expressing its solution(s) in such a way that a computer—human or machine—can effectively carry out” (Wing, 2014). Computers are defined as information processing agents that carry out numerical calculations or symbolic manipulations (Denning and Tedre, 2019).

While it is possible to design algorithms without implementing them on machines, there are compelling reasons to learn computational problem solving via programming. As students give instructions (i.e., code) to a machine, they learn how computers blindly and mechanistically process the code. It is harder to learn this from giving verbal commands to a human computer. Humans can tolerate ambiguity when interpreting natural language.

### 3.2. Programming to solve mathematics problems

Problem solving is an important instructional goal in school mathematics (e.g., National Council of Teachers of Mathematics, 2000; Common Core State Standards, 2010). In Singapore syllabus documents, it is considered “central to mathematics learning” (MOE, 2006) and involves “applying mathematics in practical tasks, in real life problems and within mathematics itself” (MOE, 1990).

Conventionally, students solve problems directly using the technologies of paper, pen, and sometimes a calculator. A



computational thinker designs algorithms for an intermediary, a computing agent, to find the solution.

### 3.3. Learning programming should be embedded in math contexts

Rather than have students take a generic programming class before applying the skill to mathematics, we designed the instructional materials so that students learn programming within a mathematics context. The materials assume no prior knowledge in programming and uses mathematics examples that most 7th grade students are already familiar with in the Singapore context.

### 3.4. Focus on using the language, rather than learning the language.

We selected and sequenced lessons so that students learn programming concepts at a level essential to doing something meaningful with mathematics. The Code by Math web-based platform uses Lua, a programming language that novices can learn quickly. We determined the following to be essential programming concepts for computational problem solving: output, sequencing, arithmetic operators, basic mathematical functions, variables, loops, and selection.

## 4. METHOD

The case study reported is from the second DBR cycle and took place at a typical public school. The teacher assisted us in getting four 7<sup>th</sup> grade students to volunteer and obtained consent from the parents. The program lasted four days and each day's session lasted 1 hour.

On day 1, we conducted a 10-minute focus group to find out about their interest in mathematics and programming. The students expressed minimal interest in the mathematics subject, due to a history of getting poor grades, finding the subject difficult and boring, and having trouble understanding their teachers' instruction. None of the students had any experience with computer programming but were cautiously open to trying it.

After explaining pair programming roles and how to use the instructional materials, students worked through the lessons in a self-guided way. The printed and soft copies of the instructional documents provided links to select lessons on the Code by Math website ([www.codebymath.com](http://www.codebymath.com)), and additional exercises.

For data collection, we recorded students' screens to capture what they were typing and their facial expressions. As backup, we used separate audio and video recorders.

## 5. ANALYSIS

We chose to analyze the following 10-minute episode because it drew attention to the distinctive influence of CT. We describe the episode in two segments: (1) developing an algorithmic solution, and (2) coding the algorithmic solution. Lastly, we compare the computational solution to the pen-and-paper solution that students would normally be expected to produce.

On the third day, Isaac and Fei Hong (pseudonyms) had just completed a brief introduction on loops, and learned

how to evaluate and display values based on the loop counter, as shown in Figure 1.

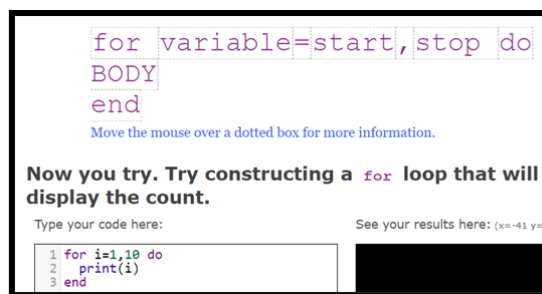


Figure 1. Screenshot of the lesson on loops from the Code by Math website

Next, they were given the math problem shown in Figure 2, which came from the New Syllabus Mathematics book 7th edition, and marked as an advanced exercise.

*A class has between 30 to 40 students. Each boy in the class brings 15 chocolate bars for a class party to celebrate Teacher's Day. The chocolate bars are shared equally among the 20 girls of the class and their form teacher with no leftovers. (p. 23)*

- How many students are there in the class?*
- How many chocolate bars does their form teacher receive?*

Figure 2. Textbook word problem

### 5.1. Developing an algorithmic solution

After 4 minutes of discussing the problem on their own, students began using a "guess-and-check" method. They had deduced that the number of possible boys was between 10 and 20. However, they seemed to be guessing randomly rather than in a systematic way. After a while, they could not proceed to solve the problem, so I (first author) guided them through clarifying "what we know" and listing these on the paper.

After a pause, one of the students said: "Can't we just take every number between 10 and 20 and then times 15 and divide by 21?" I realized that this student had just articulated an algorithmic solution, so I said: "...and you want to try every number so you can use what?" They responded, "code". I pressed them to be more precise and we had the following exchange:

WH (first author): "...a loop, right? so where would you want the loop to start at?"

Isaac: "ten"

WH: "and it's going to go until..."

Isaac: "eleven"

FH (Fei Hong): "twelve, thirteen"

WH: "so it's going to go from ten to..."

Isaac: "twenty"

WH: "and like he said, first you're going to multiply by..."

Isaac: “fifteen”

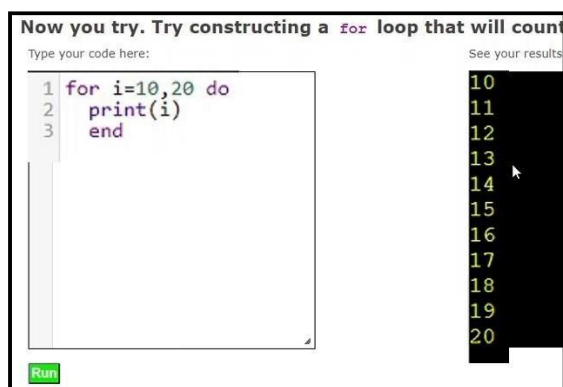
WH: “and then divide by...”

FH: “twenty one”

WH: “and see if you get a whole number. So now write code to do this, ok? But you kind of get the idea. That’s your set of instructions...The idea is that we’re going to be repeating the instructions over and over.”

## 5.2. Coding the algorithmic solution

Isaac constructed a loop for the range of possible number of boys, as shown in Figure 3. He said “oh” after clicking the Run button and his partner nodded, perhaps realizing that the output confirmed what they expected and also showed numbers that corresponded with the range of possible numbers of boys.



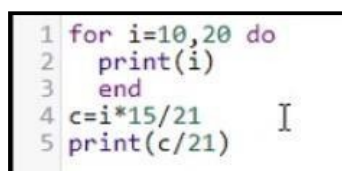
```

Now you try. Try constructing a for loop that will count
Type your code here:
1 for i=10,20 do
2   print(i)
3   end
Run
See your results
10
11
12
13
14
15
16
17
18
19
20

```

Figure 3. Isaac’s code and output

As shown in Figure 4, Isaac added lines 4 and 5 to reflect the two computation steps to be carried out for each value of  $i$ , where  $i$  is the possible number of boys. The variable  $c$  is an intermediate variable that stores the number of chocolates brought by  $i$  number of boys. However, there were some errors which reflected an immature understanding about encapsulating commands inside loops.



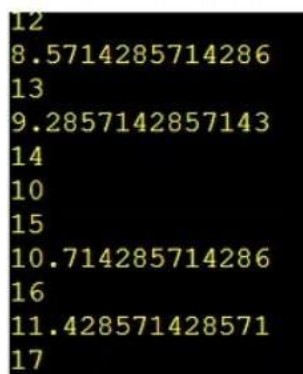
```

1 for i=10,20 do
2   print(i)
3   end
4   c=i*15/21
5   print(c/21)

```

Figure 4. Code showing errors

After fixing the errors in the code, students got the output as shown in Figure 5.



```

12
8.5714285714286
13
9.2857142857143
14
10
15
10.714285714286
16
11.428571428571
17

```

Figure 5. Output from the correct code

The students and I took a moment to interpret the results. It was striking that only when  $i = 14$ , was the result of  $c/21$  a whole number (10). This meant that the number of chocolates brought by 14 boys were divisible by 21 (representing the 20 girls + 1 form teacher). So each girl and teacher received 10 chocolates, and there were 34 students in the class (20 girls + 14 boys).

## 5.3. Comparison with conventional problem solving

The problem we gave the students came from a textbook chapter on the topic of least common multiples (LCM), along with worked examples on solving similar problems.

According to the textbook, students should deduce that the solution must be divisible by both 21 and 15, and therefore must be a multiple of the two numbers.

Students could find the common multiples using prime factorization:

$$21 = 7 \times 3$$

$$15 = 5 \times 3$$

$$\text{LCM} = 3 \times 7 \times 5 = 105$$

Students could either deduce or recall that common multiples are always multiples of the LCM, so the next multiple is  $105 \times 2 = 210$ . This corresponds to 14 boys and each girl and teacher receiving 10 chocolates each. 14 boys + 20 girls = 34 students in the class.

Another way to frame the solution is that the number of chocolates must be between 150 and 300, because the least number of boys is 10, so 10 boys \* 15 chocolates per boy = 150 chocolates and the most number of boys is 20, so 20 boys \* 15 chocolates per boy = 300 chocolates. Therefore, the number of chocolates must be a number between 150 and 300 that is divisible by 21. This formulation would exclude the LCM, which was 105 chocolates (7 boys \* 15 chocolates per boy).

## 6. DISCUSSION

We expected to see evidence of CT, but did not know from the outset what it would look like (RQ1). It was not sufficient for students to simply demonstrate CT by completing programming exercises. What is novel was the possibility that learning programming could have “primed” students to conceptualize the textbook problem in computational terms, rather than to apply the concept of “common multiples” (a problem feature that they failed to recognize). We theorize that learning loops, in particular, might have inspired students to “discover” a method to have the computer systematically calculate all the possible solutions and then to select the result that met the problem constraints. A “brute force” approach is easy to grasp and can therefore provide entry into the world of computational problem solving. We are currently analyzing other cases where we see students doing something similar, as evidence of algorithmic thinking specifically for mathematics problem solving. Another open question is whether the order of first conceiving the algorithm off-computer and then implementing on-computer is always so distinct.

To address RQ2, we showed that CT made a difference in what was included and left out of the thinking process

when compared to similar worked problems in the math textbook. In this case, the textbook examples required that students apply the concept of “common multiples”. Students did not need this concept for the computational formulation. However, they did need to specify the lower and upper bounds of the solution space and develop computations that would be repeated between those bounds. There is, of course, overlap in some areas of thinking, such as understanding the mathematical features of the problem so that key parts are abstracted and translated into numbers and operations.

We are still exploring how to theoretically describe the differences between the computational approach and the school math approach. One starting point is in epistemology. According to Abelson and Sussman (1996), *The computer revolution is a revolution in the way we think and in the way we express what we think. The essence of this change is the emergence of what might best be called procedural epistemology—the study of the structure of knowledge from an imperative point of view, as opposed to the more declarative point of view taken by classical mathematical subjects. Mathematics provides a framework for dealing precisely with notions of ‘what is.’ Computation provides a framework for dealing precisely with notions of ‘how to’.* (Structure and Interpretation of Computer Programs, 1996, p.xxiii)

Declarative knowledge is presented as a statement of fact: “the number of chocolates brought by the boys is between 150 and 300 such that the number is divisible by 21”. Stated imperatively, a computational formulation to the same problem is: “for each number between 10 and 20, multiply by 15, then divide the result by 21. If the dividend is a whole number, the number of chocolates per girl or teacher is found”. Although an oversimplification, the knowledge categories represent our early attempt to describe the phenomenon.

## 7. LIMITATIONS

DBR studies are validated through the accumulation of *storied truths* (Gee, 2013), as explanations of underlying mechanisms within rich, contextualized cases. This paper provides one case that we theorize to be paradigmatic of a broader phenomenon where CT enables secondary level students to approach a math problem differently from what they usually do in school. As we iteratively develop analytical labels and relationships based on the first two rounds of the study, we can use them to describe new cases. Future DBR cycles will take place in natural classroom settings, which would provide increased sample size and diversity.

## 8. CONCLUSION

We set out to investigate the broad phenomenon of how students develop CT while learning programming to solve mathematics problems. We designed the instructional materials to embody principles that reflected how we viewed the relationship between CT, programming, and mathematics problem solving, as well as a commitment to making the materials accessible to students of all abilities.

When interacting with students during a particular session, we identified a critical moment in which a student clearly articulated an algorithmic solution, which prepared them to write code that expressed the solution. Therefore, the case is paradigmatic of conditions where CT can make a difference in how students conceptualize a mathematics problem. A comparison of students’ computational solution and the expected textbook-based one revealed a possible area for theory-building according to knowledge types.

## 9. REFERENCES

- Abelson, H., Sussman, G. J., & Sussman, J. (1996). *Structure and Interpretation of Computer Programs - 2nd Edition*. The MIT Press.
- Cobb, P., Confrey, J., diSessa, A., Lehrer, R., & Schauble, L. (2003). Design experiments in educational research. *Educational Researcher*, 32(1), 9–13.
- Common Core State Standards. (2010). *Standards for Mathematical Practice*. Retrieved January 29, 2021, from <http://www.corestandards.org/Math/Practice/>
- Denning, P. J., & Tedre, M. (2019). *Computational Thinking*. MIT Press.
- Forsström, S. E., & Kaufmann, O. T. (2019). A literature review exploring the use of programming in mathematics education. *International Journal of Learning, Teaching and Educational Research*, 17(12). <https://doi.org/10.26803/ijlter.17.12.2>
- Gee, J. (2013). *The anti-education era: Creating smarter students through digital learning*. New York: Palgrave Macmillan.
- McCoy, L. P., & Dodl, N. R. (1989). Computer programming experience and mathematical problem solving. *Journal of Research on Computing in Education*, 22(1), 14–25.
- Ministry of Education. (1990). *Mathematics syllabus (lower secondary)*. Singapore: Author.
- Ministry of Education. (2006). *Secondary mathematics syllabus*. Singapore: Author.
- National Council of Teachers of Mathematics. (2000). *Principles and standards for school mathematics*. Reston, VA.
- Schanzer, E., Fisler, K., & Krishnamurthi, S. (2018). Assessing Bootstrap: Algebra students on scaffolded and unscaffolded word problems. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 8–13.
- Wing, J. M. (2014). Computational thinking benefits society. Retrieved January 28, 2021, from <http://socialissues.cs.toronto.edu/index.html%3Fp=279.html>
- Yelland, N. (1995). Mindstorms or a storm in a teacup? A review of research with Logo. *International Journal of Mathematical Education in Science and Technology*, 26(6), 853–869

# **STEM Activities in Informal Contexts**

# Developing STEM Makers with Mentoring and Authentic Problem-Solving Strategies

Xiaojing WENG<sup>1\*</sup>, Thomas K.F. CHIU<sup>2</sup>, Morris S.Y. JONG<sup>3</sup>

<sup>1,2,3</sup>Department of Curriculum and Instruction & Centre for Learning Sciences and Technologies

The Chinese University of Hong Kong, Hong Kong

xweng@link.cuhk.edu.hk, tchiu@cuhk.edu.hk, mjong@cuhk.edu.hk

## ABSTRACT

Maker education is regarded as a global initiative. Empowering STEM makers in schools becomes an important task for K-12 educators. Creativity, critical thinking, STEM identity, and STEM interest are all vital attributes of a good STEM maker. While researchers have proposed various pedagogical approaches to supporting the processes of STEM making, rare studies have been carried out to compare the effectiveness of these approaches. To fill the research gap, we have conducted a quasi-experimental study (with 63 secondary school students in Hong Kong) to explore the pedagogical effects of mentoring and authentic problem-solving strategies in supporting STEM making. Implications for the development of maker education and future work are also discussed in this paper.

## KEYWORDS

STEM education, maker, mentoring, authentic problem-solving, instructional design

## 1. INTRODUCTION

Maker movement has started to spread widely since the early of this century (Sang & Simpson, 2019). The advocacy of this cultural phenomenon, which is to cultivate learners to become "makers rather than consumers" of products (Chiu et al., accepted; Marshall & Harron, 2018), matches the pursue of developing students' multiple capabilities in Science, Technology, Engineering, and Mathematics (STEM) areas (Honey et al., 2014). Correspondingly, making activities have been adopted to promote STEM education. Many researchers have reported the effects of maker-centered method in STEM education. For example, to develop learners' creativity, critical thinking, and algorithmic thinking (Jeng et al., 2020), and to improve learners' psychological perceptions towards STEM (Chiu, et al., 2020; Lin et al., 2019; Schlegel et al., 2019). Among these intervention outcomes, there are four significant attributes for youth to be identified as STEM makers, including creativity, critical thinking, STEM identity, and STEM interest.

Meanwhile, educators are trying to enrich learners' STEM making experience. One of the strategies used is to infuse other instructional mechanisms into making activities

(Geng et al., 2019; So et al., 2020). For instance, mentoring and authentic problem-solving approaches are two choices of them (Hernandez et al., 2017; Musavi et al., 2018). These two instructional designs have been highlighted because of their potential in developing student learning. For instance, mentors can help learners get access to a variety of learning resources and provide them with capability development

opportunities; authentic problem-solving method exposes students to ill-structured real-world problems and enables them to practice their creativity and critical thinking skills when exploring solutions. Some researchers have reported their experience of integrating mentoring and/or authentic problem-solving approaches in students' making activities (Carbonell-Carrera et al., 2019; Kuo et al., 2019), but few of them have compared the outcomes of adopting these two methods in developing student makers. To fill the research gap, this study aims to investigate the effectiveness of using mentoring and authentic problem-solving strategies in cultivating students to become STEM makers. Accordingly, the main research question of the study is: "Which of the two instructional designs is more effective in improving students' creativity, critical thinking, STEM identity, and STEM interest, mentoring or authentic problem-solving?"

## 2. RESEARCH DESIGN

### 2.1. Research Participants

Two classes of students from two different secondary schools in Hong Kong have participated in this study. These two schools shared a similar academic background. Besides, teachers of these two classes were both certificated teachers with PGDE and around 5 years of Mathematics teaching experience. These two classes were randomly assigned to be the mentoring class (n=32) and the authentic problem-solving class (n=31). In addition, 8 mentors, who were undergraduate students in STEM-related majors, were recruited to help students in the mentoring class.

### 2.2. Research Intervention

There were three phases of the study. Step 1: Pre-test. A week before the STEM making courses, a questionnaire was distributed to all student participants. The questionnaire items, which included creativity, critical thinking, STEM identity, and STEM interest scales, were adopted from the previously published works (Kelley et al., 2019; Tyler-Wood et al., 2010; Young et al., 2013), and the Cronbach Alpha (CA) for each scale was adequate in general (see Table 1).

Table 1. Scale characteristics

Scales	Items	CA
Creativity	Q1: I am confident in my ability to understand how knowledge or insights might transfer to other situations or contexts.	0.8
	Q2: I am confident in my ability to find sources of information and inspiration when others do not.	
	Q3: I am confident in my ability to elaborate and improve on ideas.	

Critical thinking	Q4: I am confident in my ability to evaluate reasoning and evidence that support an argument.	0.7
	Q5: I am confident in my ability to identify in detail what needs to be known to answer a question.	
	Q6: I am confident in my ability to justify choices of evaluation criteria.	
STEM identity	Q7: My classmates ask me for help with STEM.	0.9
	Q8: My teachers expect me to study STEM in the future.	
	Q9: My parents think I am good at STEM.	
STEM interest	Q10: I find STEM fascinating.	0.9
	Q11: I find STEM exciting.	
	Q12: I find STEM appealing.	

Step 2: Making programme implementation. The making courses lasted for 5 weeks. Both participating classes organized students into groups (with 3-4 students in a team). Learners used Arduino kits to design and make their STEM projects. The difference between the two classes was that students in the mentoring class were assigned one mentor for each student team. On the other hand, the students in the authentic problem-solving class were assigned a real-world problem, which was to design a smart traffic light for the community, at the beginning of the programme. Table 2 shows the themes of the learning activities. Step 3: Post-test. A week after the making courses, the questionnaire was

distributed to the two participating classes again.

Table 2. Themes of Classroom Activities

Time	Theme
Week 1	Meet Arduino and my first Arduino program
Week 2	Change the brightness of LED
Week 3	Use variable resistance to change the brightness of LED
Week 4	Use the photoresistor (LDR) to change the brightness of LED
Week 5	Arduino for Problem-solving

### 3. RESULTS

Participants' pre-test scores of creativity, critical thinking, STEM identity, and STEM interest were shown in Table 3. They were covariates used to exclude the effects of students' pre-test on their post-test performance.

Table 3. Students' Pre-test Results

Groups	Variables	Mean	SD
	Creativity	2.78	0.64
Mentoring (n=32)	Critical thinking	3.16	0.58
	STEM identity	3.02	0.80
	STEM interest	3.02	0.88
Authentic problem- solving (n=31)	Creativity	2.85	0.72
	Critical thinking	2.85	0.61
	STEM identity	2.67	0.56
	STEM interest	2.87	0.85

The homogeneity of the regression coefficient of the two groups was analyzed, results showed that these two groups have no difference in creativity ( $F(1, 59) = 0.67, p = .42$ ), STEM identity ( $F(1, 59) = 0.01, p = .91$ ), and STEM interest ( $F(1, 59) = 2.38, p = .13$ ), which confirmed the assumption of homogeneity. The analysis results of critical thinking did not pass the homogeneity test. In the next step, ANCOVAs were performed to analyze the scores in the four dimensions of the post-tests.

For the dependent variable creativity, there was no significant difference in the post-test scores for creativity (see Table 4). For the dependent variable STEM identity, the adjusted means of mentoring and authentic problem groups were 4.55 and 3.24, respectively. The post-test scores of the two groups achieved significance ( $F(1, 60) = 187.09, p < .001, \eta^2 = .76$ ), showing a large effect size. For the dependent variable STEM interest, the adjusted means of mentoring and authentic problem groups were 4.59 and 3.87, respectively. The post-test scores of the two groups achieved significance ( $F(1, 60) = 19.52, p < .001, \eta^2 = .25$ ), showing a large effect size. Therefore, we conclude from the data analysis that students developed better STEM identity and STEM interest with the mentoring approach.

Table 4. Post-test Descriptive Data and ANCOVA Results

Variable	Group	N	Mean	Adjusted Mean	F	$\eta^2$
Creativity	Mentoring	32	4.22	4.22	0.15	0
	Authentic problem	31	4.27	4.27		
Critical thinking	Mentoring	32	3.22	3.15	104.39	0.64
	Authentic problem	31	4.42	4.49		
STEM identity	Mentoring	32	4.60	4.55	187.09	0.76
	Authentic problem	31	3.18	3.24		
STEM interest	Mentoring	32	4.60	4.59	19.52	0.25
	Authentic problem	31	3.85	3.87		

\* $p < 0.05$ , \*\* $p < 0.01$ , \*\*\* $p < 0.001$

### 4. DISCUSSION AND CONCLUSION

This research shows that, in the context of STEM making activities, mentoring is more capable than authentic problem-solving in cultivating students' STEM identity and STEM interest. It might be due to the formation patterns of interest and identity. Once students' interest has been developed, it can sustain when they know more about the representative professionals in the related area through instruction and/or out-of-school experiences (Jong et al., 2006; Krapp, 2007). Likewise, when students are developing their identity, they tend to make use of the resources available in the environment (Chiu & Churchill, 2015, 2016; Chiu & Mok, 2017; Dong et al., 2020), especially the external support offered by the STEM professionals. Infusing the mentoring strategy into students' STEM making activities fits learners' interest and identity development tracks. It allows learners to approach in-school or out-of-school STEM mentors who can guide their STEM interest and STEM identity development.



Though the advantages of adopting the mentoring approach have been proved in this study, the potential of using authentic problem-solving method has not been identified. We regard that there is a need to further probe into the effectiveness of the authentic problem-solving strategy for STEM maker development.

## 5. REFERENCES

- Carbonell-Carrera, C., Saorin, J. L., Melian-Diaz, D., & De la Torre-Cantero, J. (2019). Enhancing creative thinking in STEM with 3D CAD modelling. *Sustainability*, 11(21), 6036.
- Chiu T. K. F., Chai C. S., Williams, J., & Lin T. J. (accepted) Teacher professional development on self-determination theory-based design thinking in STEM education, *Education Technology & Society*.
- Chiu, T. K. F., Jong, M. S. Y., & Mok, I. A. C. (2020). Does learner expertise matter when designing emotional multimedia for learners of primary school mathematics? *Educational Technology Research and Development*, 68, 2305–2320.
- Chiu, T. K. F., & Mok, I. A. (2017). Learner expertise and mathematics different order thinking skills in multimedia learning. *Computers & Education*, 107, 147–164.
- Chiu, T. K. F., & Churchill, D. (2016). Design of learning objects for concept learning: Effects of multimedia learning principles and an instructional approach. *Interactive Learning Environments*, 24(6), 1355–1370.
- Chiu, T. K. F., & Churchill, D. (2015). Exploring the characteristics of an optimal design of digital materials for concept learning in mathematics: Multimedia learning and variation theory. *Computers & Education*, 82, 280–291.
- Dong, A., Jong, M. S. Y., & King, R. B. (2020). How does prior knowledge influence learning engagement? The mediating roles of cognitive load and help-seeking. *Frontiers in Psychology*, 11, 591203.
- Geng, J., Jong, M. S. Y., Chai, C. S. (2019). Hong Kong teachers' self-efficacy and concerns about STEM education. *The Asia-Pacific Education Researcher*, 28(1), 35–45.
- Hernandez, P. R., Bloodhart, B., Barnes, R. T., Adams, A. S., Clinton, S. M., Pollack, I., & Fischer, E. V. (2017). Promoting professional identity, motivation, and persistence: Benefits of an informal mentoring program for female undergraduate students. *PLoS One*, 12(11), e0187531.
- Honey, M., Pearson, G., & Schweingruber, H. A. (Eds.). (2014). *STEM integration in K-12 education: Status, prospects, and an agenda for research*. National Academies Press.
- Jong, M. S. Y., Shang, J. J., Lee, F. L., Lee, J. H. M., & Law, H. Y. (2006). Learning online: A comparative study of a game-based situated learning approach and a traditional web-based learning approach. In Z. Pan, R. Aylett, H. Diener, X. Jin, S. Gobel, & L. Li (Eds.), *Lecture notes in computer science: Technologies for e-Learning and digital entertainment* (pp. 541–551). Springer.
- Jeng, Y. L., Lai, C. F., Huang, S. B., Chiu, P. S., & Zhong, H. X. (2020). To cultivate creativity and a Maker mindset through an Internet-of-Things programming Course. *Frontiers in Psychology*, 11, 1572.
- Kelley, T. R., Knowles, J. G., Han, J., & Sung, E. (2019). Creating a 21st century skills survey instrument for high school students. *American Journal of Educational Research*, 7(8), 583–590.
- Krapp, A. (2007). An educational-psychological conceptualisation of interest. *International Journal for Educational and Vocational Guidance*, 7(1), 5–21.
- Kuo, H. C., Tseng, Y. C., & Yang, Y. T. C. (2019). Promoting college student's learning motivation and creativity through a STEM interdisciplinary PBL human-computer interaction system design and development course. *Thinking Skills and Creativity*, 31, 1–10.
- Lin, H. C. S., Yu, S. J., Sun, J. C. Y., & Jong, M. S. Y. (2019). Engaging university students in a library guide through wearable spherical video-based virtual reality: Effects on situational interest and cognitive load. *Interactive Learning Environments*, 1–16.
- Marshall, J. A., & Harron, J. R. (2018). Making learners: A framework for evaluating making in STEM education. *Interdisciplinary Journal of Problem-Based Learning*, 12(2), Article 3.
- Musavi, M., Friess, W. A., James, C., & Isherwood, J. C. (2018). Changing the face of STEM with stormwater research. *International Journal of STEM Education*, 5(1), Article 2.
- Sang, W., & Simpson, A. (2019). The Maker movement: A global movement for educational change. *International Journal of Science and Mathematics Education*, 17(1), 65–83.
- Schlegel, R. J., Chu, S. L., Chen, K., Deuermeyer, E., Christy, A. G., & Quek, F. (2019). Making in the classroom: Longitudinal evidence of increases in self-efficacy and STEM possible selves over time. *Computers & Education*, 142, 103637.
- So, H. J., Jong, M. S. Y., & Liu, C. C. (2020). Computational thinking education in the Asian Pacific region. *The Asia-Pacific Education Researcher*, 29(1), 1–8.
- Tyler-Wood, T., Knezek, G., & Christensen, R. (2010). Instruments for assessing interest in STEM content and careers. *Journal of Technology and Teacher Education*, 18(2), 345–368.
- Young, D. M., Rudman, L. A., Buettner, H. M., & McLean, M. C. (2013). The influence of female role models on women's implicit science cognitions. *Psychology of Women Quarterly*, 37(3), 283–292.

# **STEM Education Policies**

## Euro-Asia Collaboration for Enhancing STEM Education

Anders BERGLUND<sup>1</sup>, Valentina DAGIENE<sup>2\*</sup>, Mats DANIELS<sup>3</sup>, Vladimiras DOLGOPOLOVAS<sup>4</sup>, Siegfried ROUVRAIS<sup>5</sup>, Miriam TARDELL<sup>6</sup>

<sup>1, 3, 6</sup> Uppsala University, Sweden,

<sup>2, 4</sup> Vilnius University, Lithuania

<sup>5</sup> IMT Atlantique, Lab-STICC, UMR CNRS 6285, France

anders.berglund@it.uu.se, valentina.dagiene@mif.vu.lt, mats.daniels@it.uu.se, vladimiras.dolgopolovas@mif.vu.lt, siegfried.rouvrais@imt-atlantique.fr, miriam.tardell@uadm.uu.se

### ABSTRACT

EASTEM is a capacity-building project funded by Erasmus+ with the aim of improving employability of STEM (Science, Technology, Engineering and Mathematics) graduates from partner universities by ensuring students acquire skills needed in the workplace. EASTEM uses approaches from student-centred STEM education to develop the competence of lecturers and bridge the gap between industry and universities. Over the course of three full years (2019-2022) the project brought together ten universities from Asia and three universities from Europe to work together on improving STEM education, creating a platform for partner universities to exchange best practices on student-centred STEM education. Two associate partners, the Ministry of Education and Training of Vietnam and Vietnam Electronics Industries Association are supporting EASTEM activities.

### KEYWORDS

STEM education, STEM centres, academia and industry collaboration, student-centred approach, computational thinking

### 1. INTRODUCTION

Universities across the world are seeking to form global partnerships and fostering relationships with other institutions.

EASTEM project (<http://eastemproject.eu/>) is focused on advances in the quality of teaching. Increased connections between universities, corporate partners and schools brought about by the EASTEM project are set to improve the employability of graduates, fulfilling industry needs of the workforce in Indonesia, Thailand and Vietnam. The recent shift to online teaching and learning in all our partner universities have further increased the need to teach in ways that engage students. Improving the competence of teachers and the quality of higher education in STEM to ensure that graduates can make the best of their abilities are considered crucial measures for industrial competitiveness in partner countries.

There is an increase in demand for skilled professionals within the STEM field across the globe, and a high number of STEM workers are reaching retirement age, adding further pressure to an already skill-short area. Thus, a high quality STEM education is seen as a critical success factor for Asian countries in light of the fourth industrial revolution. Development of professional skills such as teamwork, communication and leadership skills, quality of education in relation to the demands of the job market, and

employability are skills that we can learn by collaborating and cooperating.

The project partners are drivers for change in educational approaches in their local and national context. Still the majority of teaching is based on traditional methods rather than meeting today's need of the students. This is in spite of government efforts. For example, in Vietnam the Ministry of Education and Training has identified STEM education as a key factor for development. In Indonesia, all of higher education curriculum must refer to Kerangka Kualifikasi Nasional Indonesia (Indonesia National Qualification Framework), focusing in part on competence development. As part of the Thailand 4.0 Policy of the Thai government and the strategy of the Ministry of Education, Thailand is hoping to develop a holistic strategy to prepare teachers and school leaders to deliver education reform, with a strong emphasis on improving teachers' skills to make the best use of technology in the classroom.

The project partner universities have established policies and strategies for moving their institutions forward into the next decade, with a strategy to proactively support the needs of the communities and society, to produce quality graduates who have a mind to serve the society and to develop research and create innovations for the development of economy, society and local security. Nevertheless, the institutions involved in EASTEM is at the forefront of this movement, they have expressed a need for the activities planned in this project. In spite of the focus on development of professional skills development for employability in recent years nationally, regionally and locally, the penetration rate remains low, particularly in STEM education.

With the knowledge exchanged through the partnership, each partner's expertise and experiences from different contexts can synergistically enrich each other, and will in turn subsequently benefit the members within this resulting strengthened network. This includes developing strategies for enhancing each university's own STEM education system, to establish a platform for networking on STEM education, and to safeguard the pitfalls of education in rapid changes of science and technology.

EASTEM structure and activities are built on a European model for how learning and teaching in the STEM area should be enhanced. These strategies can (and should) be contrasted to formalized "teaching methods", where certain predefined protocol for how the teaching should take place is to be followed. Such formalized teaching methods lack the flexibility to follow the development of the discipline,

the students, the students' future employers, and the needs of these entities, and are difficult to adapt to new student groups and/or new environments. These strategies should be adapted to the local cultural, social, economic and disciplinary environment. By doing that, they turn out to be powerful tools for enhancing the quality of STEM education.

There have been several initiatives focused on skills development and employability. EASTEM, described in this paper, differentiates itself on several points, grounded in a European model for enhancing STEM education. Firstly, EASTEM focus on approaches for making education more focused on the students and their needs, not through specific teaching methods. Secondly, the main attention is to integrated STEM education. Thirdly, in order to capitalize on previous and current initiatives we address student competence development on three levels in order to vitalize the student-centred STEM education. Fourthly, EASTEM lays the organizational foundation for a STEM Education network, providing visibility to initiatives within the field. Finally, in being rooted in staff development in two phases, the Asian universities conduct and establish their own staff development adaptation to local contexts.

## 2. COLLABORATION AND EXCHANGE OF GOOD PRACTICES

Working together in the EASTEM project, we are currently a part of the ongoing national and European reform of STEM in schools and universities. The motivation is to move from subject-oriented STEM to transdisciplinary and project-oriented STEM (Pears et al, 2019). The political goal is to increase the motivation of students in STEM and increase the number and diversity of students interested in STEM university subjects and professional STEM and engineering careers. The educational goal is to focus on transdisciplinary aspects and promote research-based education (Cook, Bush, 2018). At the same time, a number of international, regional and national research-based initiatives are underway to improve university and high school STEM education, focusing on student skills development and related aspects.

Synergistic learning combining Computational Thinking (CT) and STEM has proven to be an effective method for advancing learning and understanding in a number of STEM domains and simultaneously helping students develop important computer science concepts and practices (Park, Green, 2019).

Many computational environments and tools have been developed to promote CT competencies in STEM education. The way scientists and engineers approach problems is very similar to CT methodology: Identify problems and do research; Decompose the problem; Design the algorithm or create plan; Analyse results; Debug and modify, etc. (Palts, Pedaste, 2020).

Educators are often confused about CT and STEM and have difficulties to see the link. However, CT is a way of solving problems and can be integrated with various disciplines. Especially STEM contexts are very suitable for this. CT skills incorporate analytical thinking, engineering thinking, and scientific thinking. Thus, they could be positioned as a

kind of universal skill for the modern student, and this is especially true for STEM education.

Besides critical thinking, creativity, communication and collaboration, CT can be seen as an important part of 21st century learning. The importance of CT is still underestimated in education. CT is a set of problem-solving methods that involve expressing problems and their solutions in ways a computer could execute (Denning, Tedre, 2019). Modern computation tools are changing the way science and mathematics are practiced. CT encompasses a wide range of mental processes, which are considered necessary supplies for the 21st-century children.

The aim of EASTEM project is to provide a European-Asian insight on student-centred STEM education research practice. It is based on a collection of best practices, case studies, analytical reviews, theoretical contributions focused on approaches to students' skills development and university-industry collaborative practices as related to university STEM education. The motivation is as follows. A look at the university STEM in terms of institutional development, focusing on: (a) country-specific STEM results for a range of unique experiences and best practices; (b) a look at collaborative practices and outcomes associated with global and international STEM activities. Specific topics could include: educational policies and managerial approaches to university STEM development and research; research on curriculum development and integration focusing on students' STEM professional competencies; Euro-Asian University and university-industry collaboration in research and best practices as related to university STEM. The project activities and outcomes are organized in three main strands.

### 2.1. Train lecturers in student-centred competence development

Partners developed trainings in student-centred STEM education approaches with the help of Uppsala University, Sweden. In the first phase, a number of lecturers from Asian partner institutions participated in a course in student-centred competence development. Then they set up a pilot course where students solve problems from local industries and communities in international teams at the different partner institutions. In the second phase, lecturers trained in the first phase conducted staff development sessions for other lecturers both within and outside of their institutions. At some occasions this was made for lecturers at the local institutions, but at some institutions those courses were also regional or national

In conclusion, the European and Asian partners have jointly developed and implemented student-centred STEM education staff trainings at the Asian partner institutions. At the same, a quality revision of the Asian partners teaching of STEM is taking place.

During the pilot course, the course participants (lecturers) taught modules for students applying their new ideas. In this way, the pilot module served as a test bed at the same time as it is an occasion for the lectures to apply student-centred teaching approaches. Lecturers who have participated in the Training of Trainers courses are now applying SCL approaches that we have learnt in regular teaching with

students. A particular focus has been on attitudes, both of the staff towards their new roles and the learners towards being in control of their own learning.

## 2.2. Establish STEM education centres

Vilnius University, Lithuania is supporting lecturers, deans and administrative staff at Asian partner universities to establish, staff and run STEM education centres to ensure the sustainability and increase the visibility of student-centred STEM activities. These centres should anchor STEM activities firmly within the university structure and serve as focal points for each university's STEM initiatives. By engaging both university students and external partners such as local high schools in centre activities, the centres have the potential to develop into hubs for STEM education and learning in each city or region. The establishment centre for excellence in STEM education is based on each partner university sharing of current status and good case practices.

The establishing of the Centre for Excellence in STEM Education allows:

- to improve study programmes quality through integrated style of study and to modernize the curricula of study programmes by including innovative learning and teaching tools;
- to engage students with STEM disciplines and to allow students to get acquainted with different STEM disciplines;
- to strengthen the link between academic environment (university) and work life, to develop competences needed for the job market;
- to develop the students' and teachers' soft skills;
- strengthen partnerships between university-industry-school;

Preparation of the feasibility study based on each partner's needs and best practices in STEM Centres. Mapping STEM Education centre conceptions in each partner's institution: preparation of the guidelines, recommendations, strategical plan of STEM centre establishment. Implementation part consists of preparation of training material, developing training modules, innovative methodological tools and pilot implementation:

1. Staff development in STEM Education Centre Management;
2. Staff development for Centre activities coordinators (university lecturers, researchers, who organize and implement activities in centre);
3. Launch of STEM centres (in each Partner University);
4. Piloting the STEM centres platform:
  - Activities (at university level);
  - Lecturers: 1) study and research organization for students; 2) consultation, expertise, research of educational process; 3) preparation of methodological material and tools for teachers;
  - University students' involvement in centre activities integrative modules, research, practice supervised by lecturers;
  - Activities (at the K-12 level);

- School teachers training;
- School students: formal and non-formal education activities in STEM centre;

5. Evaluation of launched centres activities;

6. Centres for Excellence in STEM Education consortium establishment.

An interdisciplinary platform for STEM education at universities provides sustainability for the project network, activities, increasing visibility of student-centred educational approaches and research in STEM education (<https://www.fsf.vu.lt/en/eastem-centres-platform>).

## 2.3. Facilitate industry engagement and competence integration into STEM educational programmes

New skills are required in the era of the Fourth Industrial Revolution and recognizing the importance of competence development for students, institutions are to facilitate education focused on students' needs but also offer STEM programmes that better align with labor market needs. Based on major accreditation requirements in the six partner countries, several University-Industry collaboration formats were categorized in themes (Rouvrais et al, 2020). They lay the foundation of a structured relationship model for STEM universities, which now permits to build on good case practices from all partner institutions. It thus contributes to advancing STEM-educational frameworks for curriculum guidelines aligned with skills for industry.

Going beyond concerns and models of an EASTEM educational framework, incl. curriculum development, SCL, industry collaboration, training of trainers and STEM centres, a more strategic level is to be reached. EASTEM aims to provide partner institutions with the knowledge to develop their own processes for continuous integration of good practices into their STEM educational ecosystem. With support on a strategic level from university management, STEM activities are more sustainable.

IMT Atlantique, France engages with partners on how to better reach university management (e.g. programme leaders and deans, vice-rectors and rectors). They work to develop a strategy, canvas and maturity tools for continuous integration of competence development and EASTEM models into various levels of university education. Anchoring the need for support on a strategic level leads to sustainability of the various action plans, at short to longer terms. In addition, designed tools should lead to a new way to interact with stakeholders in the design, development, operation and revision of STEM university education according to various needs or more formal requirements.

## 3. CAPACITY BUILDING IMPACTS

EASTEM addresses university-enterprise cooperation, entrepreneurship and employability of graduates for the Asian region. Student-centred competence development within STEM education and active engagement with industry should help bridge the skills gap in our partner countries and improve graduate employability.

STEM education centres, similar supporting units or established groups of like-minded lecturers have been set up at partner institutions and provided a focal point for STEM education activities including external stakeholders

such as high schools and companies. Through a questionnaire and interviews, we have developed a framework aimed to support the Asian partners including eight themes for university-industry collaboration (Rouvrais et al, 2020), which provides the foundations for improving local industry engagement strategies and processes in partner institutions. In 2020-2021 the STEM education centres should incorporate and pilot various student-centred STEM activities involving corporate partners, lecturers, teachers, and university and high school students.

According to our Asian partners, the EASTEM project has created value by building strong national and international networks, promoting cooperation between the EU and the partner countries, between partner countries and within partner countries. Elevating the Training of Trainers courses to a national level as our partners have done in Thailand and Indonesia and intend to do in Vietnam strengthens the potential for wider dissemination of methodologies inspired by European universities, thus promoting voluntary convergence with EU developments in higher education.

With our focus on improving the quality of higher education and enhancing its relevance for the labour market and society, EASTEM objectives are also in line with the new EU Skills Agenda, more specifically increasing STEM graduates and fostering entrepreneurial and transversal skills. Lecturers from our three European partners gain additional insight and perspective on SCL teaching approaches and industry engagement and establish new partnerships with colleagues in Asia.

Through the STEM centres, partner universities have strengthened relationships with industry partners and high schools. For example, when launching a STEM centre in November 2019, Mahidol University (Thailand) signed a memorandum of understanding with Imagineering Education Company. Partners in Vietnam and Thailand emphasize the potential for their STEM centres to establish cooperation between academia and industry. Partners in Thailand have also pointed out that the EASTEM Training of Trainers courses help improve the quality of education, with the potential to transform teaching and learning philosophy and inspire lifelong learning.

Vietnam's government recognizes STEM education as a driving factor for a strong labour workforce that meets the requirements of the 4th Industrial Revolution. Foreexample, in Thua Thien-Hue province, EASTEM's activities connecting universities with high schools and industry partners are also in line with the province's ambition to develop a smart city urban cluster.

#### 4. CONCLUSIONS

In EASTEM we focus on STEM education and we go beyond methods. Drawing from successful strategies to

address the skills gap, we aim to strengthen student-centred competence development by taking a holistic approach also by including computational thinking.

By jointly developing the Training of Trainers course methodology, Asian partner university lecturers have been trained to design, teach and assess STEM classes using student-centred approaches. The Training of Trainers course evaluation results have shown how participating lecturers have increased their knowledge and skill in applying student-centred approaches into their teaching.

These core strategies can be taken as the European model for how learning and teaching in the STEM area could be enhanced. They are based on research on students' learning of the discipline and do not prescribe certain teaching methods as being better than others. Instead, these strategies can (and should) be contrasted to formalized "teaching methods", where certain predefined protocols for how the teaching should take place are to be followed.

#### 5. ACKNOWLEDGMENTS

The authors would like to acknowledge all their colleagues from the EASTEM project, co-funded by the Erasmus+ Programme of the European Union (reference 598915-EPP-1-2018-1-SE-EPPKA2-CBHE-JP).

#### 6. REFERENCES

- Cook, K. L, Bush, S. B (2018). Design thinking in integrated STEAM learning: Surveying the landscape and exploring exemplars in elementary grades. *School Science and Mathematics*, v. 118, no. 3-4, 93-103
- Denning, P.J., Tedre, M. (2019). *Computational Thinking*. The MIT Press, Cambridge.
- Palts, T., Pedaste, M. (2020). A Model for Developing Computational Thinking Skills, *Informatics in Education*, v. 19, no. 1, 113-128, Retrieved from: <https://infedu.vu.lt/journal/INFEDU/article/27/info>
- Park, Y., Green, J. (2019). Bringing Computational Thinking into Science Education. *Journal of the Korean Earth Science Society*, v. 40, no. 4, 340-352.
- Pears, A., Barendsen, E., Dagienė, V., Dolgopolas, V., Jasutė, E. (2019). Holistic STEAM education through computational thinking: a perspective on training future teachers. *Informatics in schools. New ideas in school informatics*: In proceedings of the 12th Intl. Conf. on informatics in schools: situation, evolution, and perspectives, Larnaca, Cyprus, Nov 18–20, Springer, LNCS vol. 11913, pp. 41-52.
- Rouvrais, S., Jacovetti, G, Chantawannakul, P., Suree, T., Bangchokdee, S. (2020). University-Industry collaboration themes in STEM higher education: An Euro-ASEAN perspective. In *16th International CDIO Conference*, Gothenburg, Sweden, 91-102.



# **STEM Pedagogies and Curriculum**

# Designing an Interdisciplinary Social-scientific STEM Curriculum on Students' Empathy, Efficacy, and Interest

Biyun HUANG<sup>1\*</sup>, Morris Siu-Yung JONG<sup>2</sup>, Ching Sing CHAI<sup>3</sup>, Yun DAI<sup>4</sup>, Darwin LAU<sup>5</sup>

<sup>1, 3, 4</sup> Centre for Learning Sciences and Technologies & Department of Curriculum and Instruction

<sup>2, 5</sup> Department of Mechanical and Automation Engineering

The Chinese University of Hong Kong, Hong Kong

lucyhuang99@cuhk.edu.hk, mjong@cuhk.edu.hk, cschai@cuhk.edu.hk, yundai@cuhk.edu.hk, darwinlau@cuhk.edu.hk

## ABSTRACT

More and more countries have regarded that STEM education is one of the best pathways to develop future citizens optimally equipped for the needs of future industries. Students can develop 21st century skills such as communication, collaboration, design thinking, and innovation through learning STEM-related subjects. However, few studies focus on combining STEM and social care education to enhance students' empathy and STEM competencies. The present work aimed to design a social-scientific STEM curriculum based on students' abilities and backgrounds. Apart from that, it probed into the students' changes in the areas of empathy, self-efficacy, and interest after learning the curriculum via a quantitative survey. The results showed that the students made positive changes in the areas concerned.

## KEYWORDS

STEM, social care, interdisciplinary, empathy, self-efficacy

## 1. INTRODUCTION

In recent years, STEM education has been increasingly advocated and implemented in more and more countries and regions (Lee et al., 2019; MartínPáez et al., 2019). It is widely believed that STEM education can enhance students' communication, cooperation, design thinking, innovation, and other skills needed in the 21st century (Honey et al., 2014; Geng et al., 2019). Consequently, the number of studies on STEM has been growing, including those on STEM pedagogy (e.g., Simeon et al., 2020), learning effectiveness evaluation (e.g., Huang & Jong, 2020), teacher preparation and development (e.g., Chai et al., 2020; So et al., 2020), and others. Simultaneously, there is a growing awareness that empathy can be an important component of STEM teaching and learning, especially when teaching design thinking. Empathy is the ability to understand and respond adaptively to others' feelings and sufferings, which is a vital step to compassionate actions (Preston & de Waal, 2002; Riess, 2017). Enhancing people's empathy will enable them to design products that are more attuned to users' needs and optimize user experience (Carlson & Dobson, 2020). However, the existing literature shows little experimental research investigating STEM and empathy integration (Gunckel & Tolbert, 2018).

In light of the lack of a robust body of literature in this regard, we designed an interdisciplinary social-scientific STEM curriculum that combines social care and STEM topics and evaluated its impact on empathy, self-efficacy, and interest in a secondary school.

## 2. THEORETICAL FRAMEWORK

### 2.1. Research background

In contrast to traditional didactic approaches, STEM education emphasizes the integration of learning with real-life problems to develop students' abilities to solve problems and build other skills needed in the 21st century (Nadelson & Seifert, 2017; Lee et al., 2019). Many efforts were made to study how different scientific disciplines, such as mathematics and technology, can be combined to promote the development of high-level thinking skills and the creation of high-quality STEM products (MartínPáez et al., 2019). Apart from integrating two or more science subjects to develop students' abilities, it would be prudent to consider integrating these science subjects with some humanistic subjects, e.g., social care. Through the interdisciplinary learning process, students can learn to care for others and apply their STEM knowledge to design products that are suitable for addressing and fulfilling a full spectrum of potential user requirements. It would be interesting to explore whether students' empathy, interest, and self-efficacy improve after participating in such programs.

### 2.2. The EDIPT design thinking model

This curriculum's design is informed by The EDIPT design thinking model proposed by the Hasso Plattner Institute of Design at Stanford University (Hasso Plattner Institute of Design [HPID], 2010). This model is a widely accepted design model in the STEM field. The model suggested that students can experience five stages in the design process: empathize, define, ideate, prototype, and test (HPID, 2010). In the empathize stage, students can visit and have a conversation with the users to understand their thoughts and their needs. In the definition stage, students can synthesize and select the needs they think are important to meet and then determine the one they will strive to address in their design. In the ideate stage, learners practice divergent thinking and propose a range of possible solutions to choose from. Next, students build their proposed solutions and prototype them, which could bring them closer to the final solution. In the test stage, students present the prototypes to users, garner feedback, and then refine them. Thoring and Müller (2011) conducted a study to observe how the EDIPT model was practiced in the Hasso Plattner Institute in Potsdam, Germany, and developed a process model to describe the process steps in detail. Henrisken et al. (2018) applied the model in a teacher education course and found that this model promoted creativity and empathy. In the design industry,

Da Silva et al. (2020) integrated the design thinking model with another design thinking tool to improve new product development and identified it as an appropriate framework in guiding product design.

### 3. METHODS

#### 3.1. Design of the social-scientific curriculum

We designed the course with reference to the EDIPT model. As the class time was limited, we focused on the first three aspects of the design process, empathize, define, and ideate. After ideating, we asked students to compile and present their written solutions. Then, expert and peer feedback were provided to help students know the direction for improving their work. The entire curriculum consists of three components: social care, STEM, and proposal writing. The duration of the interdisciplinary curriculum was six weeks.

The theme of the curriculum was social housing and product design. In the STEM course, students learned the STEM knowledge and skills necessary for realizing the solutions, such as coding through Thinkable, and Internet of Things (IoT). At the same time, in the social care course, students experienced the complete process of problem identification, solution proposal, and expert feedback. Students watched the videos of interviews with people living in social houses to understand the users' needs. Then, they analyzed and identified the problems they want to help solve. Next, the students brainstormed together to come up with possible solutions. Afterward, the groups refined their ideas and developed a more specific solution after the class. Class presentations were organized during the course, and experts and peers were invited to provide suggestions for improvements. Concurrently, in the proposal writing course, students learned to present their design ideas in the form of a proposal. See Figure 1 for the interdisciplinary social-scientific curriculum.

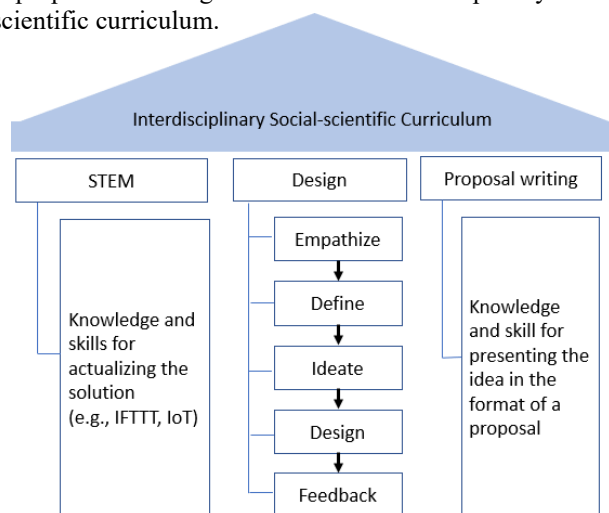


Figure 1. The interdisciplinary social-scientific curriculum.

#### 3.2. Participants

Two grade 8 classes from a secondary school participated in this study. Given that grade 8 students had laid some foundation in their STEM knowledge in the previous year (i.e., grade 7), it could be easier for them to adapt to the

interdisciplinary course and design social care products with their knowledge. All students underwent the curriculum. A total of 55 students and their parents co-signed the consent form. Only the students who signed the consent form were included in the analysis of this study. In order to track the changes in affective, the students were asked to complete a pre-questionnaire prior to the course. At the end of the course, students completed a post-questionnaire.

#### 3.3. Instrument

The questionnaire-based instrument consisted of three dimensions: empathy, self-efficacy, and interest (12 items in total, in a 6-point scale). In terms of empathy, we referred to the instrument of Vossen et al. (2015) for examining students' willingness to understand and contribute to community services. For example, one of the questions was, "I would try to understand how others feel about community service." In terms of self-efficacy, we referred to the instrument of Chen et al. (2001) for measuring students' confidence in applying STEM knowledge to serve their communities. One of the questions was, "I believe I can use STEM knowledge to come up with useful ideas for helping the community." In terms of interest, we referred to the instrument of Luo et al. (2019) for examining students' interest in STEM. For example, "I like to design products related to STEM." To establish content validity, three experts were invited to provide feedback about the measurement tool. Modifications on the content were conducted based on their suggestions.

### 4. RESULTS

Forty-two students completed both the pre- and post-surveys. The reliability test showed that the Cronbach Alpha of the subscales of empathy, self-efficacy, and interest were 0.83, 0.89, 0.79. The result indicated there was a high level of internal consistency for this instrument. Paired-sample t-tests were conducted to compare if there were any differences between students' pre- and post-survey scores. Results indicated that the post-survey scores were significantly higher than the pre-survey scores in all the examined dimensions. For example, in the empathy dimension, there was a significant difference in the pre-survey score ( $M=4.29$ ,  $SD=0.87$ ) and post-test ( $M=4.69$ ,  $SD=0.96$ ),  $t(41)=-2.63$ ,  $p=0.01$ . See Table 1.

Table 1. Paired-sample t-test results of the survey.

	Mean	SD	t value	df	Sig (two tailed)
Empathy					
Pre-test	4.29	0.87	-2.63	41	0.01
Post-test	4.69	0.96			
Self-efficacy					
Pre-test	3.88	1.06	-2.63	41	0.001
Post-test	4.60	0.98			
Interest					
Pre-test	4.13	0.90	-3.04	41	0.004
Post-test	4.63	1.03			

Note.  $n=42$

## 5. DISCUSSION AND CONCLUSION

This work explores how STEM and social-care education can be integrated for students to experience real-life problems and actively explore solutions by themselves. The study found that such a curriculum can potentially enhance students' empathy, self-efficacy, and interest. The innovative curriculum demonstrated how interdisciplinary courses could be designed to enhance students' social emotional competencies. As this study was implemented in a normal teaching environment, the practices are applicable in similar schools. A limitation of the study is that no control group was in the research setting. If possible, a control group could be introduced to size-up the effectiveness of using social-scientific curriculum and scientific-only curriculum on students' empathy, efficacy, and interest. Another limitation is that the data collected so far have been mainly quantitative. In the next round, more data (e.g., interview data) will be collected to triangulate the results.

## 6. REFERENCES

- Chai, C. S., Jong, M. S. Y., & Yan, Z. M. (2020). Surveying Chinese teachers' technological pedagogical STEM knowledge: A pilot validation of STEM-TPACK survey. *International Journal of Mobile Learning & Organisation*, 11(2), 203–214.
- Chen, G., Gully, S., & Eden, D. (2001). Validation of a new general self-Efficacy scale. *Organizational Research Methods*, 4, 62–83.
- Da Silva, R. H., Kaminski, P. C., & Armellini, F. (2020). Improving new product development innovation effectiveness by using problem solving tools during the conceptual development phase: Integrating Design Thinking and TRIZ. *Creativity and Innovation Management*, 29(4), 685–700.
- David Carlson, J., & Dobson, T. (2020). Fostering empathy through an inclusive pedagogy for career creatives. *International Journal of Art & Design Education*, 39(2), 430–444.
- Geng, J., Jong, M. S. Y., Chai, C. S. (2019). Hong Kong teachers' self-efficacy and concerns about STEM education. *The Asia-Pacific Education Researcher*, 28(1), 35–45.
- Gunckel, K. L., & Tolbert, S. (2018). The imperative to move toward a dimension of care in engineering education. *Journal of Research in Science Teaching*, 55(7), 938–961.
- Hasso Plattner Institute of Design (2010, January 1). *An introduction to design thinking process guide*. ALNAP. <https://www.alnap.org/help-library/an-introduction-to-design-thinking-process-guide>
- Henriksen, D., Richardson, C., & Mehta, R. (2017). Design thinking: A creative approach to educational problems of practice. *Thinking Skills and Creativity*, 26, 140–153.
- Honey, M., Pearson, G., & Schweingruber, H. (Eds.). (2014). *STEM integration in K-12 education: Status, prospects, and an agenda for research*. The National Academies Press.
- Huang, B., & Jong, M. S. Y. (2020). Developing a generic rubric for evaluating students' works in STEM education. *Proceedings of the 2020 International Symposium on Educational Technology (ISET)* (pp. 210–213). Institute of Electrical and Electronics Engineers.
- Lee, M. H., Chai, C. S., & Hong, H. Y. (2019). STEM education in Asia Pacific: Challenges and development. *The Asia-Pacific Education Researcher*, 28, 1–4. <https://doi.org/10.1007/s40299-018-0424-z>
- Luo, Z., Dang, Y., & Xu, W. (2019). Academic interest scale for adolescents: Development, validation, and measurement invariance with Chinese students. *Frontiers in psychology*, 10, <https://doi.org/10.3389/fpsyg.2019.02301>
- MartínPáez, T., Aguilera, D., PeralesPalacios, F. J., & VilchezGonzález, J. M. (2019). What are we talking about when we talk about STEM education? A review of literature. *Science Education*, 103(4), 799–822.
- Nadelson, L. S., & Seifert, A. L. (2017). Integrated STEM defined: Contexts, challenges, and the future. *The Journal of Educational Research*, 110(3), 221–223
- Preston, S., & De Waal, F. (2002). Empathy: Its ultimate and proximate bases. *Behavioral and Brain Sciences*, 25(1), 1–20. <https://doi.org/10.1017/S0140525X02000018>
- Riess, H. (2017). The science of empathy. *Journal of patient experience*, 4(2), 74–77.
- Simeon, M.I., Samsudin, M.A. & Yakob, N. (2020). Effect of design thinking approach on students' achievement in some selected physics concepts in the context of STEM learning. *International Journal of Technology and Design Education*. <https://doi.org/10.1007/s10798-020-09601-1>
- So, W. M. W., He, Q., Chen, Y., & Chow, C. F. (2020). School-STEM professionals' collaboration: A case study on teachers' conceptions. *Asia-Pacific Journal of Teacher Education*, <https://doi.org/10.1080/1359866X.2020.1774743>
- Thoring, K., & Müller, R. M. (2011). Understanding the creative mechanisms of design thinking: an evolutionary approach. *Proceedings of the Second Conference on Creativity and Innovation in Design (DESIRE'11)* (pp. 137–147). Association for Computing Machinery.
- Vossen, H.G.M., Piotrowski, J.T., Valkenburg, P.M. (2015). Development of the adolescent measure of empathy and sympathy (AMES). *Personality and Individual Differences*, 4, 66–71.

# A Co-design Approach for Developing Computational Thinking Skills in Connection to STEM Related Curriculum in Swedish Schools

Rafael ZEREGA<sup>1\*</sup>, Ali HAMIDI<sup>2\*</sup>, Sepideh TAVAJOH<sup>3\*</sup>, Marcelo MILRAD<sup>4\*</sup>

<sup>1,2,3,4</sup> Faculty of Technology, Linnaeus University, Sweden

rafael.zerega@lnu.se, ali.hamidi@lnu.se, st222yd@student.lnu.se, marcelo.milrad@lnu.se

## ABSTRACT

Computational thinking (CT) is a set of problem-solving methods which several scholars advocate for its inclusion in the educational curricula for K-12. Incorporating this knowledge into existing syllabuses is a challenge for both the educational community and researchers within the field of STEM education. The focus of this study is on the importance of co-design and constructionism in the process of planning and designing teaching modules to introduce students to CT using both robotic constructions and programming. This paper presents and discusses the design process of a series of workshops conducted with a group of middle school students during the fall of 2020. The main goal of these workshops was to introduce students to the main concepts and practices of CT, thus addressing the goals defined by the Swedish Agency for Education (Skolverket). Our initial findings indicate that co-designing educational activities (with a focus on constructionism and challenge-based learning) in close collaboration between teachers and researchers can lead to effective ways to foster the development of CT skills among students.

## KEYWORDS

Computational Thinking, Co-design, STEM Education, K-12 curricula, Challenge-based Learning, Constructionism.

## 1. INTRODUCTION

Computational thinking (CT) is an approach to problem-solving which many researchers within the computer science education community advocate for its inclusion in the current K-12 educational curriculum. One strong argument for such recommendations is to provide students with the required knowledge and necessary skills to face the challenges of our modern society (Wing, 2006; Grover & Pea, 2018). As Wing argued, we live in a society of ubiquitous computing, however, we do not yet live in a society of CT (Wing, 2006).

During the last few years, Sweden has started a process of adapting the curriculum of different subject matters, including mathematics and technology, so that K-12 students can acquire different skills for being able to produce creative and innovative solutions to solve authentic problems. Although the goals set by the Swedish National Agency for Education are clearly defined (Skolverket, 2018) in terms of the knowledge and skills that are to be developed by students in these particular subjects, it is, however, not

specified what learning strategies and methods should be used to reach these goals and effectively teach CT concepts in the classroom (Kohen-Vacs & Milrad, 2019). One of the aims of this paper is to explore possible ways in which CT learning activities and teaching modules can be designed in a collaborative way between teachers and researchers so that they can be integrated into the schools' curriculum for elementary and middle education within STEM-related subjects aiming at reaching the goals established by the Swedish Agency for Education (Skolverket, 2018).

This study is a continuation of the research activities we had started in the spring of 2020 related to validating different design approaches for teaching CT in Swedish schools taking into account the goals defined by Skolverket for the subject matter of technology. In order to reach this purpose, we are using the Engino® Robotics Platform (ERP)<sup>1</sup> which is an educational tool specially designed for primary and secondary STEM education. Considering all the above, the main research question that guides our research efforts in the focus of this paper can be formulated as follows: *How teaching modules for STEM related subjects in elementary and high schools should be designed and organized so that they can help students develop and practice CT concepts?*

The paper is organized as follows; in the coming section we present the theoretical framework on which this study is grounded. In section three and four we describe the design of our research interventions and the main findings. Finally, the discussions and conclusions are presented in sections five and six.

## 2. THEORETICAL FOUNDATIONS

In this section we discuss some theoretical aspects and concepts related to CT and its integration into K-12 STEM education. The concepts on which the learning theory of constructionism are based and on how children are builders of their own intellectual structures (Papert, 1980) have had a notable influence in the context of CT considering that learning to solve problems and to design solutions is a way of creating knowledge and a fundamental aspect of CT (Grover & Pea, 2018). These authors explain that the ultimate goal of CT is creating a computational artifact that could be a physical device, pure software or the combination of both. Another concept of importance within CT education is the use of *learning challenges* (Conde et al., 2019). Conde and colleagues argue that *challenge-based learning* is an

<sup>1</sup> <https://www.engino.com/w/index.php/products/robotics>

effective strategy for teaching CT because it allows students to learn to define and solve a problem, it promotes collaborative work, and it connects students with real world problems.

Two additional concepts that we would like to highlight in the context of this study is the importance of using co-design (Spikol et al., 2009) as an approach to create a common ground between educational practitioners and researchers as well as the TPACK framework for technological education (Wong et al., 2014). Designing teaching modules is a complex process that requires the involvement of all stakeholders. Engaging in co-design is, therefore, an essential aspect when planning learning activities. By actively involving different stakeholders (teachers, researchers, IT-developers, etc.) and working in direct contact with one another and assigning them specific roles based on their area of expertise, it is possible to yield educational innovation (Spikol et al., 2009). Wong et al. (2014) describes how the TPACK framework can be used in combination with co-design so that practitioners and researchers can combine their knowledge and expertise to allow teachers to integrate technological tools with their pedagogical and content knowledge in an integral manner to create rich learning environments.

### 3. RESEARCH DESIGN

In this section we present the main aspects concerning the research design approach used for this study.

#### 3.1. Participants and Settings

The workshops that are the central part in this study were conducted during the fall of 2020 at an international middle school in the south of Sweden. The activity took place as part of the weekly schedule in the subject of technology defined in the study curriculum of the school with the participation of four researchers, three teachers, and 25 students from eighth grade aged 13-14 years old. In three workshops of three hours, one session per week, students were divided by the teachers into two main groups. Within these two groups students were free to team up in smaller groups of 2-3 for conducting the activities together. The data we have collected for the analysis of the activities comes from field notes and pictures taken during the workshops. In addition, the students had to fill in a questionnaire regarding their perceptions about the activities they did and the way in which they came up with solutions to carry out the different tasks during the workshops.

#### 3.2. Workshop Design Approach

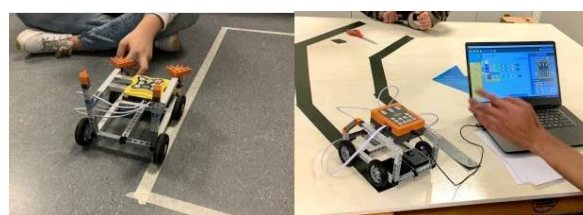
According to Mannila et al. (2014), there cannot be an appropriate development of CT-related ideas if teachers are not involved when designing learning activities. Likewise, Spikol et al. (2009) and Wu et al. (2020) highlight the importance of co-design when defining educational strategies for STEM-related subjects in K-12 classrooms. Based on these concepts, a couple of months before

conducting the workshops with the students, the research team met with the teachers during a series of sessions to introduce them to the ERP system and to plan together learning activities to bring CT concepts to their students.

#### 3.3. Workshop Activities with the Students

The first workshop was focused on hands-on activities with the Engino building parts with two main goals in mind. The first was to make them get familiar with the ERP construction tools as they had never worked with it before. The second goal was to put into practice some CT concepts such as *pattern recognition*, which has major relevance when engaging in construction activities. After receiving a brief instruction on how to work, students started building the models for the robot of their choice by using instructions printed on paper or a 3D interactive version of them that they could have on their computers. The second workshop, carried out one week after the first one, focused on programming in two different modalities: (1) manual programming, by using the physical buttons that the ERP robots have, to program basic functions only (Figure 1a), and (2) a block-based programming with the KEIRO software for designing more advanced algorithms (Figure 1b). Thus, the students had the chance to work with mechanical construction combined with programming and algorithmic thinking, which is another important CT concept. The main task here was to program the vehicles to move along a rectangular track that had been drawn on the floor.

The third workshop was held two weeks after the previous one and it had a higher level of complexity in the tasks which were focused on programming the robot vehicles and making use of infrared (IR) sensors that the students connected to the vehicles they had built in the previous workshop. The students were given a brief introduction where they learnt some basic concepts of programming such as using conditionals, loops and logical operators. The main task was to program the robot so that by using two IR sensors it would move along a track that was demarcated with dark tape (Figure 1b).



(a) (b)  
Figure 1. Manual Programming (a) and Software Programming (b).

The main goal with this task was to make students learn to calibrate the IR sensors and to acquire concepts of algorithm design. In addition, this workshop aimed to develop other relevant CT concepts and practices, such as logical thinking, problem decomposition, testing and debugging.



#### 4. EMPIRICAL FINDINGS

In the first workshop, which focused on the physical construction of the robot vehicles, students had to get familiar with the assembly method of the plastic pieces that the Engino ERP uses. At first, the students were very unfamiliar with this assembly method and it was especially difficult for them when they had to disassemble one piece from another. At times the students were frustrated when they had to struggle just to figure out how to disassemble a given piece. However, over time the students showed a clear improvement in their abilities to assemble and disassemble the pieces as they started to understand the method for assembling the pieces by trial and error.

During the second workshop the students were given a pre-assembled chassis and they had to build the rest of the vehicle in any fashion they deemed appropriate. The students showed great capacity for innovative design. When the students had to manually program the vehicles by using the physical buttons they have on the top (see Figure 1a) so that they would follow a rectangular track marked on the classroom's floor, the students faced a series of challenges. To begin with, they had to figure out what was the underlying principle that made the vehicles turn to either side and this required a lot of exploration as well as plenty of trial and error efforts. The students had to figure out how to program the vehicle so that it would move forwards just the necessary distance and then turn at just the right point in order to accurately follow the track. The third workshop posed several additional challenges for the students in two main areas: learning to use the IR sensors and creating a functional algorithm. The first major challenge was to make the IR sensors scan the edge of the track effectively so that the vehicle would not cross it. This task required the students to solve many different problems that arose. One of them had to do with the proper calibration of the IR sensor so that it would detect the edge of the track marked with a dark tape over a light-colored surface. The latter required the students figuring out how to place the sensor correctly so that it would effectively detect when the car was starting to trespass the dark colored tape that was marking the edge of the track.

#### 5. DISCUSSION

In this section we will elaborate on our findings and focus the discussion around two main issues, namely, the design of the teaching modules for CT skills development and how to incorporate CT education into the technology subject.

##### 5.1. Activity Design for CT Skills Development

The three workshops that were described in this paper have a strong theoretical foundation that rely on the principles of constructionism and CT concepts and practices (Kynigos, 2015; Grover & Pea, 2018). We planned the activities putting emphasis on the entire process of building the robots, starting with the construction of the physical structure and followed by the design of the algorithms and programming.

Teaching CT concepts does not necessarily require explicitly referring to them. We used a strategy based on exploration and learning by doing, thus allowing the students to acquire CT skills through their own practices. In the different activities, the teachers were only providing some general guidelines and serving as advisors when the students required their assistance. The activities we designed for the three workshops were based on a hands-on approach. The first workshop was designed with the goal of making the learning process both enjoyable and challenging to ensure a high level of engagement from the participants (Conde et al., 2019). This focus on constructionism and challenge-based tasks was essential for designing these activities. During the second and third workshops we added a focus on programming. Table 1 summarizes the main CT concepts and practices that were applied by students in the different workshops.

Table 1. CT Concepts and Practices Used by Students in each Workshop

Workshop	Activity	CT concepts and practices
1 <sup>st</sup>	<ul style="list-style-type: none"> <li>- Mechanical assembly and construction</li> <li>- Using instructions</li> </ul>	<ul style="list-style-type: none"> <li>- Pattern recognition</li> <li>- Follow instructions</li> <li>- Communication &amp; Collaboration</li> </ul>
2 <sup>nd</sup>	<ul style="list-style-type: none"> <li>- Connecting robots to controllers</li> <li>- Manual programming</li> </ul>	<ul style="list-style-type: none"> <li>- Logic and algorithmic thinking</li> <li>- Testing and debugging</li> <li>- Communication &amp; Collaboration</li> </ul>
3 <sup>rd</sup>	<ul style="list-style-type: none"> <li>- Connecting sensors and peripherals</li> <li>- Block based programming</li> <li>- Controlling robots</li> </ul>	<ul style="list-style-type: none"> <li>- Problem decomposition</li> <li>- Abstraction</li> <li>- Logic thinking and algorithmic thinking</li> <li>- Testing and debugging</li> <li>- Communication &amp; Collaboration</li> </ul>

##### 5.2. Incorporating CT into Swedish STEM Education

Looking at constructionism as a theory of learning and a theory of design (Kynigos, 2015), the suggested plans for integrating the educational practice of CT in the subject of mathematics and technology constitutes a relevant contribution towards reaching the goals defined by the Swedish National Agency for Education (Skolverket, 2018). According to the Swedish curricula for grades 7-9, the technology subject aims to develop students' curiosity in technology and help them handle technical problems through creative and innovative ways. The design approach used for this workshop series gives students the opportunity to learn by doing. The ideas described by Conde et al. (2019) focusing on the importance of challenge-based learning plays a relevant role when designing teaching instances where students have the chance to identify problems, propose solutions, and engage in an iterative process of design, testing and evaluation of the proposed solutions.

Another core content of the technology subject that is defined by Skolverket has to do with working out innovative and creative methods for developing technological solutions. A good case of innovative solutions could be witnessed when the students were working on calibrating the IR sensors. We used black tape to mark on the floor the edge of the track where the vehicles should move along. However, students had unsuccessful results when calibrating the IR sensors to detect the dark tape that marked the edge of the track. After many attempts the students finally realized that the problem was that the color of the floor was very similar to that of the tape and thus the IR sensor was not able to detect the difference in color between the floor and the tape. The students had to figure out how to increase the contrast between the track's surface and the dark tape marking its border to solve this unforeseen problem.

Lastly, it is also important to reflect on the importance of collaborative practices between teachers and researchers in order to elaborate effective teaching modules through an approach based on co-design (Spikol et al., 2009). This is especially relevant when identifying and defining in which ways the technological knowledge can complement the pedagogical and content knowledge that is addressed by the TPACK framework (Wong et al., 2014) to co-design teaching modules aiming at fostering CT skills among students. Designing teaching modules grounded on the core ideas of constructionism and challenge-based learning, like the workshops we conducted, allowed students to develop and put into practice relevant CT skills. Co-designing these learning activities between researchers and practitioners is an effective method to create learning instances where the technological knowledge can complement the pedagogical and content knowledge in a well-coordinated interplay.

## 6. CONCLUSION

CT is a thought process and an approach to problem-solving that is based on a set of concepts and practices that can provide students with the necessary skills to face the challenges of modern society. This study described our approach for designing teaching modules to introduce CT concepts and practices in STEM education for middle schools in Sweden. A strong emphasis was given to the exploration of these ideas in the subject of technology following the goals set by Skolverket. Constructionism and challenge-based learning are effective approaches to promote CT development. One of our goals was to demonstrate that students applied CT concepts and principles when facing the different challenges that they encountered during their experience building and programming robots. The three workshop sessions we conducted offered the students the possibility to apply CT concepts and practices. In addition, co-design is an effective approach to actively involve researchers within computer science education and K-12 teachers in the process of planning and designing teaching modules that bring CT knowledge and skills to the classrooms.

## 7. REFERENCES

- Conde, M. Á., Fernández, C., Alves, J., Ramos, M. J., Celis-Tena, S., Gonçalves, J., ... & Peñalvo, F. J. G. (2019, October). RoboSTEAM- A Challenge Based Learning Approach for integrating STEAM and develop Computational Thinking. In *Proceedings of the Seventh International Conference on Technological Ecosystems for Enhancing Multiculturality*, 24-30.
- Grover, S., & Pea, R. (2018). Computational Thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*, London: Bloomsbury Academic, 19-37.
- Kohen-Vacs, D., & Milrad, M. (2019). Computational Thinking Education for In-Service Elementary Swedish Teachers: Their Perceptions and Implications for Competence Development. In *International Conference on Computational Thinking Education, 13-15 June 2019, Hong Kong*. The Education University of Hong Kong, 109- 112.
- Kynigos, C. (2015). Constructionism: Theory of learning or theory of design? In *Selected regular lectures from the 12th International Congress on Mathematical Education*. Springer, Cham, 417–438.
- Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014, June). Computational thinking in K-9 education. In *Proceedings of the working group reports of the 2014 on innovation & technology in computer science education conference*. 1- 29.
- Papert, S. (1980). *Mindstorms: Children, Computers and Powerful Ideas* (1st Edition). New York, Basic Books.
- Skolverket (2018). *Curriculum for the compulsory school, preschool class and school-age educare*. Retrieved January 10, 2021, from <https://www.skolverket.se/getFile?file=3984>
- Spikol, D., Milrad, M., Maldonado, H., & Pea, R. (2009, July). Integrating co-design practices into the development of mobile science laboratories. *Ninth IEEE International Conference on Advanced Learning Technologies*. IEEE, 393-397.
- Wing, J. (2006). Computational Thinking. *Communications of the ACM*, 49(3), 33-36.
- Wong, L. H., Chai, C. S., Zhang, X., & King, R. B. (2014). Employing the TPACK framework for researcher-teacher co-design of a mobile-assisted seamless language learning environment. *IEEE Transactions on Learning Technologies*, 8(1), 31-42.
- Wu, S. P., PeeL, A., Bain, C., Anton, G., Horn, M., & Wileksky, U. (2020). Workshops and Co-design Can Help Teachers Integrate Computational Thinking into Their K-12 STEM Classes. *CoolThink@ JC*, 63

# Analysis of the Development Direction of STEM Curriculum in China

Lihua PENG

Shanghai International Studies University, China

1090537294@qq.com

## ABSTRACT

Due to the characteristics of interdisciplinary training of innovative talents, STEM education has been enriched in various countries from concept improvement to curriculum construction since it was put forward. Different countries also have different development modes. While China's STEM education has made some achievements, there is also room for some progress in STEM curriculum construction. This study searched the relevant literatures on the construction of STEM courses on CNKI (Chinese National Knowledge Infrastructure) and Web of Science, finding out the current status and trend of the development of STEM courses in China. Through an in-depth analysis on the basis of the differences between domestic and foreign studies, put forward the future research direction of STEM courses in China.

## KEYWORDS

STEM education, STEM course, curriculum integration

## 1. INTRODUCTION

In the 1980s, faced with the shortage of scientific and technological talents, American proposed STEM education based on Science, Technology, Engineering and Mathematics. STEM is different from traditional course, get rid of the knowledge system of single subject, focused on the tasks and projects. STEM guides students to interdisciplinary field using knowledge, integrate available resources, cooperation to complete learning tasks. Next Generation Science Standards points out that the goal of science education is to reflect the combination of practice and experience in the real world.

Under the background of STEM education, Chinese science and technology education workers get into STEM field and carry out the education of science and technology education. Chinese integrated STEM curriculum concentrated in the 3d printing science, robot, visual programming and so on. It mainly in the form of a comprehensive practice course or Mak-er activities. Pay attention to the students' participation and experience. However, there are also some problems on the basis of emphasizing the position of students and the learning and application of multidisciplinary knowledge (Yang, 2020).

Based on this, this study searches the relevant literatures on the construction of STEM courses, finding out the current status and trend of the development of STEM courses in China. Through the analysis on the differences between domestic and foreign studies, puts forward the future research directions of STEM Courses in China.

## 2. RESEARCH DESIGN

This study focus on CNKI and Web of Science, using keywords STEM and STEM Curriculum, by artificial removal of relevance to the theme of literature, selected 30 papers as the research samples. It aims to answer the following questions:

- (1) What is the current situation of STEM curriculum research in China?
- (2) What are the differences in STEM curriculum development at home and abroad?
- (3) What are the suggestions for the future development of Chinese STEM curriculum?

## 3. FINDINGS

China's basic education has been dominated by "exam-oriented education" in form for a long time. Traditional teaching only focuses on the impart of knowledge and skills while neglecting the cultivation of students' innovative ability. The introduction of STEM education concepts is a good to China to improve students' innovative ability. In recent years, the research on STEM courses in China shows an increasing trend, and STEM courses continue to attract the attention of more and more researchers and teachers. The development of STEM courses can be broadly divided into the following stages:

**Introduction stage.** Before 2014, researchers begin to pay attention to STEM course. In 2014, Shanghai rely on Shanghai STEM cloud center and Shanghai international research center for science education, developed a high-quality curriculum and good topics for students in learning, introducing and drawing lessons from foreign curriculum resources. Beginning the experimental of teacher training and implementation of the pilot programs, Beijing, Jiangsu and other provinces follow. (Feng Hua, 2016)

**Exploratory stage.** Around 2015, researches on STEM education focused more on the characteristics of STEM curriculum, considering the possibility and direction of its future development. Researchers put forward suggestions on the localization design of STEM curriculum on the basis of eliminating realistic obstacles.

**Booming development stage.** After 2018, researchers gradually focused on course design strategies and structural framework, and effectively improved students' computational thinking and innovation ability through course design.

At present, the related research of STEM curriculum mainly focuses on the primary and secondary education. Research topics tend to be mathematics courses

and engineering courses. In engineering, some researchers put forward the STEM course mode of "subject-engineering integration". In the early school years, STEM education focus on improving students' interest in learning and their ability to understanding. At the high school level, STEM education seeks application, integration of subject content and high-quality products. (Feng Hua, 2016)

#### 4. DIFFERENCES BETWEEN DOMESTIC AND FOREIGN STUDIES

Abroad STEM education focus on design thinking, students' core skills and STEM literacy are cultivated. The curriculum activity objectives correspond to educational standards. (Chen Peng, 2019) Compared with foreign countries, there is a lack of STEM curriculum standards in China, which leads to vague curriculum objectives and no standardized teaching materials. Using STEM courses to increase the capital of college entrance, it alienates the goal of developing students' core literacy. Compared with foreign research, there are more policy introduction and interpretation and theoretical discussion, and less practical research.

America has developed the STEM teacher qualification and teacher training program, it can guarantee the standardization of teaching STEM courses. China has a vague image of qualified STEM teachers, which leads to the lack of a systematic admission system for STEM teachers. In addition, domestic STEM teacher professional development trainings are unable to meet the number of teacher professional development needs. (McFadden, 2017)

Foreign curriculum design has a certain theoretical support, the theoretical basis for the formulation of teaching objectives and learning outcomes in different grades. Domestic STEM courses lack the guidance of teaching theories. Existing STEM courses in schools usually attach importance to the integration of teaching forms or the use of technical tools to produce works, which fails to truly realize the integration of students' interdisciplinary knowledge and the improvement of their real problem-solving ability.

#### 5. CONCLUSION

##### 5.1 Improve the Standard System of STEM Curriculum

The history of STEM education proves that the integrated STEM education with interdisciplinary and knowledge fusion will be an inevitable trend in the future. Schools need to build subject standard and appropriately combining with STEM course standard, build STEM teacher practical community, encourage teachers' cooperation. (Roehrig, 2021) Reasonably plan the development approach of subsubjects and comprehensive courses, and carry out STEM education in an orderly way based on the actual teaching situation.

##### 5.2 Strengthen the Theoretical Construction of STEM Curriculum Development

Drawing lessons from foreign STEM curriculum development theories, the STEM curriculum will be more

flexible and systematic throughout the whole teaching process. We should encourage innovating teaching activities, project-driven teaching, pay attention to the creation of a learning community. Focusing on the cognitive and behavioral changes of students in the learning process, formative evaluation and summative evaluation are used to design the next step of learning to adapt to the development of students. (Hasani, 2021)

##### 5.3 Innovate the Teaching Mode of STEM Curriculum

Engineering education should be integrated into the "gap" of school curriculum in a reasonable way, so as to provide engineering learning context and opportunities for the integration of different courses. In the process of systematic knowledge imparting, focus on the individual learning needs of different students. With the support of various forces inside and outside the classroom, we integrate resources inside and outside the school to form an open, balanced and sustainable STEM learning ecosystem. (Gale, 2020)

#### 6. REFERENCES

- Chen, P. (2019) Research and Enlightenment of Innovative STEM Education Curriculum Based on Design Thinking -- A Case Study of Stanford University's D.Loft STEM Curriculum. *China Audio-visual Education*, 2019(08):82-90.
- Feng, H. (2016). Comprehensive curriculum construction from the perspective of STEM education. *Management of Primary and Secondary Schools*, (05):14-16.
- Gale, J., Alemdar, M., Lingle, J., & Newton, S. (2020). Exploring critical components of an integrated STEM curriculum: an application of the innovation implementation framework. *International Journal of STEM Education*, 7(1), 5.
- Hasani, A., Juansah, D. E., Sari, I. J., Islami, E., & Zaky, R. A. (2021). Conceptual Frameworks on How to Teach STEM Concepts in Bahasa Indonesia Subject as Integrated Learning in Grades 1–3 at Elementary School in the Curriculum 2013 to Contribute to Sustainability Education. *Sustainability*, 13(1), 173.
- McFadden, J. R., & Roehrig, G. H. (2017). Exploring teacher design team endeavors while creating an elementary-focused STEM-integrated curriculum. *International Journal of STEM Education*, 4(1), 1-22.
- Roehrig, G. H., Dare, E. A., Ring-Whalen, E., & Wieselmann, J. R. (2021). Understanding coherence and integration in integrated STEM curriculum. *International Journal of STEM Education*, 8(1), 1-21.

# **STEM Teacher Education and Professional Development**

# Teacher Sensemaking on Computational Thinking in a Community of Math Teachers

Chung Yiu SIU<sup>1</sup>, Mi Song KIM<sup>2\*</sup>, Wendy HUANG<sup>3</sup>, Chee-Kit LOOI<sup>4</sup>

<sup>1,2</sup>Curriculum Studies, Faculty of Education, Western University, Canada

<sup>3,4</sup>National Institute of Education, Singapore

syiu24@uwo.ca, mkim574@uwo.ca, wendy.huang@nie.edu.sg, cheekit.looi@nie.edu.sg

## ABSTRACT

For pedagogical innovation in innovative curriculum design, much attention has been paid to the importance of teachers' attitudes and beliefs about teaching and learning. However, little research focuses on elucidating the thorough process of teachers' sensemaking of pedagogical innovation such as integrating computational thinking (CT) into the school curriculum. Therefore, the aim of this study is to explore how mathematics teachers make sense of integrating CT into the mathematics curriculum. This study employed a case-study design with 9 teachers during the 2019-2020 school year using observations of teacher professional development meetings, semi-structured interviews with the teachers, and teacher artifacts. Using Weick's (1995) properties of sensemaking, our findings indicate that the most prevalent properties of sensemaking for the teachers in this study were *social*, *ongoing*, *driven by plausibility rather than accuracy*, and *retrospective*. These findings are important to support continuing professional development.

## KEYWORDS

Teacher Sensemaking, Sensemaking Properties, Teacher-Led Curriculum Innovation

## 1. INTRODUCTION

To respond to educational reforms, teachers are expected to support the increasingly sophisticated skills students need to learn for preparing further education and success at work in the 21<sup>st</sup> century. This is facilitated by continuing professional development where teachers learn and adjust the pedagogies needed to teach these skills (Darling-Hammond et al., 2017). For innovative curriculum design, much attention has been paid to the importance of teachers' attitudes and beliefs about teaching and learning. However, little research focuses on elucidating the thorough process of teachers' sensemaking of pedagogical innovation such as integrating computational thinking (CT) into the school curriculum. Therefore, the aim of this study is to explore how mathematics teachers make sense of integrating CT into the mathematics curriculum. Our research questions are: What are the patterns of teacher sensemaking? and How does the sensemaking perspective describe teacher-led curriculum innovation?

## 2. LITERATURE REVIEW

Sensemaking has been frequently used in non-educational fields such as organizational research (Dervin, 1983). Although recently it has become a growing topic of science education research, teacher sensemaking is relatively new in teacher education, in particular in CT. Therefore, in this study, we make use on the notion of sensemaking from organizational studies to elucidate the process of teacher sensemaking on Computational

Thinking in math lessons. Sensemaking occurs “when the discrepancy between what one expects and what one experiences is great enough, and important enough, to cause individuals or groups to ask what is going on, and what they should do next” (Maitlis & Christianson, 2014, p. 70). Sensemaking depicts the path that people as actors “structure the unknown” (Waterman, 1990, p. 41) and how people establish coherence, clarify situations, frame problems, make decisions, take actions, and justify their choices within organizational settings. For many teachers, the pedagogical innovations can be seen as something “unknown” as they are new to them. According to Coburn (2001), schools are also considered as organizational settings. Weick (1995) proposed seven properties of sensemaking: (a) *grounded in identity construction*, (b) *retrospective*, (c) *enactive of sensible environments*, (d) *social*, (e) *ongoing*, (f) *focused on and by extracted cues*, and (g) *driven by plausibility rather than accuracy*. The property of *grounded in identity construction* means that individuals make sense of a circumstance through their exceptional senses and self-identity and with their understanding of the influence of the circumstance on them. The “*retrospective*” refers to individuals' reflection on the past events which affect their sense making of present events. For example, in our study, we consider the past events as the past CT experiences of teachers. The *enactive of sensible environment* means that sensemakers contribute to the environment and the affected environment returns a kind of influence to sensemakers. The *social* property relates to the experience of shared collective action (Czarniawska-Joerges, 1992) beyond simply achieving shared meaning. The *ongoing* signifies that sensemakers perceive something as a disruption to their existing frame, seek to connect it to past experiences, and feel them. The *focused on and by extracted cues* denotes that a characteristic that the sensemakers recognizes as a crucial and representative trait of the entire phenomenon. The *driven by plausibility rather than accuracy* entails that sensemaking is about taking “a relative approach to truth, predicting that people will believe what can account for sensory experience but what is also interesting, emotionally appealing, and goal relevant” (Weick, 1995, p. 879). These sensemaking properties provide an effective way to examine how mathematics teachers make sense of integrating CT into the mathematics curriculum.

## 3. THE STUDY

The study employed a qualitative case study to understand how teachers make sense of Computational Thinking by designing and implementing math lessons. In this study as part of a larger design-based research project, our participants were secondary mathematics teachers in Singapore who intended to integrate CT in teaching



mathematics Grade 7 to 10 students. Since 2018, a group of mathematics teachers has been co-planning and implementing CT lessons. Our research team supported the participant teachers' CT concepts for designing and implementing math lessons in classrooms. They observed or learned about a series of Math+CT lesson guided by a mathematics educator who is familiar with CT. The demo lessons have considerable curriculum-relevant teaching materials and activities such as how to use spreadsheet software to understand the computing concept of recursion. Our case study with 5 teachers carried out during the 2019-2020 school year using observations of teacher professional development meetings, semi-structured interviews with the teachers, and teacher artifacts. For this study, extensive amount of qualitative data from the audio- and video-recordings of regular teacher meetings were transcribed in verbatim.

#### 4. FINDINGS

Our findings indicate that the properties of sensemaking for the teachers were *social, driven by plausibility rather than accuracy, retrospective, and ongoing*. For the teachers in this study, *social* appeared as the most salient property when teachers made sense of CT. Teachers held meetings twice a month to share their collaborative lesson planning experiences and understanding of CT and practices of CT lessons. The meetings enabled them to acquire more sources to make sense of CT as their interactions in meetings. For example, Teacher A was new to CT and did not use any CT terminologies at first. However, he used more CT terminologies later such as the four-pillar cognitive processes of CT: decomposition, pattern recognition, abstraction and algorithmic and became one of the most enthusiastic members in generating ideas of planning CT lessons in the group. Through their *social* collaboration during teacher meetings, the experienced often shared their previous CT lesson plans and experiences with the teachers, in particular new teachers who joined the project later. Experienced teacher B shared his difficulties and struggling on designing the part of pattern recognition for worksheets. As a result, the new teachers could make sense of CT with other experienced teachers. When they were making sense of CT, they were based on their status and situation to design and implement a CT lesson plan, *driven by plausibility*. As facing constant educational reform and extensive administrative work, the actions of teachers were often time-sensitive, subject to the speed/accuracy trade-off and responsive to new innovations. The scarcity of time led them to make plausible and sensible decisions for CT lessons rather than an accurate and comprehensive resolution. During the regular meetings, the experienced teacher participants often mention their retrospective experience in designing and implementing CT. Experienced teachers realized that the structure of the worksheets that ordered the four CT pillars did not encounter challenges. Other teachers were comfortable and agreeable to use previous teaching materials that were established by leaders in this project and successfully adopted them in lessons. Teachers made

sense based on their "feeling." Normally, this property allows people to make sense shortly and briefly as feeling is unstable and in a moment.

#### 5. DISCUSSION AND IMPLICATIONS

We argue that teacher sensemaking properties contribute a practical framework to analyze how mathematics teachers make sense of integrating CT into the mathematics curriculum. These sensemaking properties provide an effective way to examine how math teachers contended with matter of coherence and dealt with the meaning from mostly conflicting messages they confronted in their local environment. Our findings showed that teachers made sense of CT through *social, driven by plausibility rather than accuracy, retrospective and ongoing*. This reveals that a collaborative group is important for teachers to make sense of teacher-led curriculum innovation. The collaborative group, however, could not benefit teachers without regular meetings and teachers' initiative. In our case, as CT integration was new to the participant teachers in this study, their experience and knowledge were insufficient initially. So, a group of proactive teachers and a platform were of utmost importance for them to share understanding of innovation and practices with one another. This also made them more confidence in implementing the innovation. Further, teacher-led curriculum innovation from teachers happened based on their feeling. As teachers were not familiar with the innovation, they tended to relate it to what they felt rather than what it is. Since innovation comes endlessly, feeling it motivated them to take a path to comprehend new things. Last, given that retrospective relates to past experience, the experience regards to teacher-led curriculum innovation is crucial. The properties of sensemaking have shown that teachers need sources to make sense of innovation. Thus, it is important to establish teachers' on-going professional development innovation that will provide references for them to make sense of it.

#### 6. REFERENCES

- Coburn, C. E. (2001). Collective sensemaking about reading: How teachers mediate reading policy in their professional communities. *Educational Evaluation and Policy Analysis*, 23(2), 145-170.
- Czarniawska-Joerges, B. (1992). *Exploring complex organizations*. Sage Publications.
- Darling-Hammond, L., Hyster, M. E., & Gardner, M. (2017). Effective teacher professional development. *Learning Policy Institute*.
- Dervin, B. (1983). An overview of sense-making research: Concepts, methods, and results to date. *International Communication Association Annual Meeting*, 1-13.
- Maitlis, S., & Christianson, M. (2014). Sensemaking in organizations: Taking stock and moving forward. *Academy of Management Annals*, 8(1), 57-125.
- Waterman, R. H. (1990). *Adhocracy: The power to change*. Memphis, TN: Whittle Direct Books.
- Weick, K. E. (1995). *Sensemaking in organizations*. Sage

# A Systematic Review of Teachers' Preparedness towards Computational Thinking Integration in Mathematics

Shiau-Wei CHAN<sup>1</sup>, Chee-Kit LOOI<sup>2\*</sup>, Shivani MAHEDIRATA<sup>3</sup>, Mi Song KIM<sup>4</sup>

<sup>1,2</sup>National Institute of Education, Nanyang Technological University, Singapore

<sup>3,4</sup>University of Western Ontario, Canada

shiauwei5634@gmail.com, cheekit.looi@nie.edu.sg, smahedir@uwo.ca, mkim574@uwo.ca

## ABSTRACT

As earlier studies highlighted the importance of teachers' preparedness to develop computational thinking (CT) for students in school education, this study aims to explore the teaching areas involved in the mathematics teachers' preparedness to integrate CT in classrooms, as well as to investigate the considerations for effective training or professional development activities to prepare mathematics teachers in teaching CT. A total of 16 journal articles from 2015 to 2020 were reviewed in this study. The findings indicated that not all the teaching areas (i.e. classroom management, teaching methods, subject knowledge, technology, planned curriculum, assessing students, and choosing teaching materials) were involved in the teachers' preparedness for each study. Several considerations for effective training or professional development had been proposed. The results can be utilized to inform initial teacher education plans and ongoing professional development opportunities to better prepare the teacher to teach CT in the mathematics classrooms.

## KEYWORDS

Systematic review, teachers' preparedness, computational thinking, mathematics

## 1. INTRODUCTION

Teachers from all levels require educational experience to prepare them to teach CT concepts effectively (Rich, Yadav, & Schwarz, 2019). Chalmers (2018) findings maintain for teachers to be able to successfully integrate and teach CT in classrooms, they need to have increased knowledge and awareness of the subject and its concepts, only when the teachers are confident can they deliver meaningful knowledge to the students. This further highlights the importance of the preparedness of teachers. Thus, this study intends to conduct a systematic review of teachers' preparedness towards CT integration in mathematics. Two following research questions guide this systematic review:

- What are the teaching areas involved in the mathematics teachers' preparedness to integrate CT in classrooms?
- What are the considerations for effective training or professional development activities to prepare mathematics teachers in teaching CT?

## 2. LITERATURE REVIEW

Teachers' preparedness was defined by Gonzales (2018) as "[t]he state of 'being ready for some purpose, use or activity' (p. 15) before having to accomplish an activity. Ondimu (2018) described teachers' preparedness as

individual and collective knowledge, ability, skills, perceptions, and attitudes of teachers to support the enactment of curricula. The teacher's level of preparation is measured according to the teacher's views on the following seven teaching areas: (1) classroom management, (2) teaching methods, (3) subject knowledge, (4) technology, (5) planned curriculum, (6) assessing students and (7) choosing teaching materials (Lu, 2005).

Courses or training are implemented to meet the need for teacher preparation. Earlier studies (e.g. Angeli and Jaipal-Jamani, 2018) revealed that the training given to the pre-service teachers was able to develop pre-service teachers' CT skills and better prepare them to teach CT in the classrooms. Besides the teacher education courses or training, the CT professional development courses were also implemented for in-service teachers. For example, Yadav, Gretter, Good, and McLean (2017) executed a study with 76 in-service teachers in a program that included two 39-hour courses. The findings revealed that participants have a better understanding of CT concepts and practices, and have made improvements in three of the four knowledge-related dimensions related to technical knowledge content.

## 3. METHOD

The method utilized in this systematic review was based on the method of performing systematic reviews in the social sciences by Petticrew and Roberts (2006). Five scientific databases were employed to execute systematic review, namely Scopus, Web of Science, Science Direct, LearnTechLib, and ProQuest Education database. We used several combinations of search terms to find the relevant articles for this systematic review, i.e. "computational thinking" AND ("math" OR "mathematics") AND ("teacher"). The initial search resulted in a total of 156 articles.

The inclusion criteria for this systematic review were including (a) The article published in the last five years, i.e. between 1st January 2015 and 31st December 2020 as the field of CT in the mathematics teacher education was only being developed in recent years; (b) The article published in the peer-reviewed journals; (c) The article reported on the empirical evidence of the research, involving qualitative or quantitative, and mixed-method; (d) The article presented the CT in the mathematics teacher education; (e) The participants must be mathematics in-service teachers or pre-service teachers; and (f) The article published in the English language. Meanwhile, the exclusion criteria were including (a) The article published

in the book chapter, book series, and conference proceedings; (b) The article that only reported on the literature review, opinion, and framework or model; and (c) The article did not relate CT in the mathematics in-service teachers or pre-service teacher education. Using the above inclusion and exclusion criteria, 16 articles were included in this systematic review.

## 4. FINDINGS

### 4.1 Teacher Preparation

To review the math teachers' preparation to integrate CT in classrooms, we adapted Lu's (2005) seven teaching areas. It includes (1) classroom management, (2) teaching methods, (3) subject knowledge, (4) technology, (5) planned curriculum, (6) assessing students, and (7) choosing teaching materials (see Table 1).

*Table 1.* Teacher preparedness in seven teaching areas in the reviewed articles

No	Authors & Year	1	2	3	4	5	6
1	Li (2020)					/	
2	Piedade, Dorotea, Pedro, & Matos (2020)	/	/				
3	Reichert, Barone, & Kist (2020)		/	/			
4	Araujo, Floyd, & Gadanidis (2019)	/	/	/	/		
5	Papadakis & Kalogiannakis (2019)		/	/			/
6	Masfingat, & Maharani (2019)						/
7	Rich, Yadav, & Schwarz (2019)		/				
8	Tuhkala, Wagner, Iversen, & Kärkkäinen (2019)				/		
9	Yuan, Kim, Hill, & Kim (2019)			/	/		
10	Chalmers (2018)	/	/	/	/		/
11	Günbatar, & Bakırcı (2018)			/			
12	Valentine (2018)	/	/	/	/		
13	Wang, Utemov, Krivonozhkina, Liu, & Galushkin (2018)	/					
14	Gadanidis (2017)		/	/	/	/	/
15	Gadanidis, Cendros, Floyd, & Namukasa (2017)		/	/	/	/	/
16	Leonard et al. (2017)	/	/	/			

\*(1) classroom management, (2) teaching methods, (3) subject knowledge, (4) technology, (5) planned curriculum, (6) assessing students and (7) choosing teaching materials

### 4.2 Teacher Training and Professional Development

Yadav et al. (2017) concluded that teacher training and professional development activities are vital as it was observed that teachers only had a basic understanding and knowledge of CT. They found that the current training being provided to teachers is not enough, so 'training needs to begin early on in the teacher preparation programs to allow pre-service teachers to understand how computational thinking ideas are related to their content areas' (p. 217).

According to Chalmers (2018), a big part of the professional development practices should be, 'a greater awareness of computational thinking concepts, practices, and perspectives would increase teachers' understanding and confidence to embed computational thinking and robotics into primary school classrooms' (p. 97). Wang et al. (2017) shed light on access methodological resources like flipped classrooms, as a driving force to increase the teachers' motivation levels.

Valentine (2018) discussed how increasing chances for pre-service teachers to experience and interact with concepts and tools of math and CT and viewing them as doers or makers is an important consideration for professional development training. She adds that this lays a strong foundation and cultivates a habit of active thinking with respect to what to teach and how to teach those math and CT concepts in the classrooms. 'Future work might consider creating opportunities for pre-service teachers to plan their own constructivist-oriented mathematics lessons and try these out with classmates and in their field placements' (p. 16). Pre-service teachers would benefit significantly from STEM content courses taught in an integrated way since pre-service teachers tend to apply an integrated method to STEM teaching after they have been taught in such a way.

## 5. CONCLUSION

Research question one explored the level of mathematics teachers' preparedness to integrate CT in classrooms. The results revealed not all the seven teaching areas were covered for teachers' preparedness in each study. Most of the studies (11 studies) investigated the use of technology, followed by subject knowledge (8 studies), planned curriculum (6 studies), teaching methods (5 studies), assessing students (5 studies), classroom management (1 study), and choosing teaching materials (1 study).

Research question two investigated the considerations for effective training or professional development activities to prepare mathematics teachers in teaching CT. Several considerations for effective training or professional development activities were including the importance of introducing the teacher preparation programs early, imbue in a greater awareness of CT concepts, practices, and perspectives, access methodological resources, as well as experience and interact with concepts and tools of math and CT.

There is a need for teacher professional development and ongoing training for the pre-service and in-service teachers who integrate CT in their mathematics classrooms. This systematic review can be useful for teachers, educators, and researchers seeking to greatly improve the quality of training or professional development programs to enhance the teachers' preparedness of teaching CT in mathematics lessons.

## 6. REFERENCES

- Angeli, C., & Jaipal-Jamani, K. (2018). Preparing pre-service teachers to promote computational thinking in school Classrooms. In Khine M. (Eds). *Computational Thinking in the STEM Disciplines*. Springer, Cham.
- Araujo, R. C., Floyd, L., & Gadanidis, G. (2019). Teacher candidates' key understandings about computational thinking in mathematics and science education. *Journal of Computers in Mathematics and Science Teaching*, 38(3), 205-229.
- Chalmers, C. (2018). Robotics and computational thinking in primary school. *International Journal of Child-Computer Interaction*, 17, 93-100.
- Gadanidis, G. (2017). Five Affordances of Computational thinking to support elementary mathematics education. *Journal of Computers in Mathematics and Science Teaching*, 36(2), 143-151.
- Gadanidis, G., Cendros, R., Floyd, L., & Namukasa, I. (2017). Computational thinking in mathematics teacher education. *Contemporary Issues in Technology and Teacher Education*, 17(4), 458-477.
- Gonzales, K. K. (2018). *Teachers' confidence and preparedness for teaching mobile learners*. Dissertations, The University of Southern Mississippi.
- Günbatar, M. S., & Bakırcı, H. (2018). STEM teaching intention and computational thinking skills of pre-service teachers. *Education and Information Technologies*, 24(2), 1615-1629.
- Leonard, J., Mitchell, M., Barnes-Johnson, J., Unertl, A., Outka-Hill, J., Robinson, R., & Hester-Croff, C. (2017). Preparing teachers to engage rural students in computational thinking through robotics, game design, and culturally responsive teaching. *Journal of Teacher Education*, 69(4), 386-407.
- Li, Q. (2020). Computational thinking and teacher education: An expert interview study. *Human Behavior and Emerging Technologies*, 1 – 15.
- Lu, X. (2005). Teacher quality and teacher preparedness in public secondary schools: Evidence from SASS 1999-2000. *Dissertations*. 1044.
- Masfingatın, T., & Maharani, S. (2019). Computational thinking: Students on proving geometry theorem. *International Journal of Scientific & Technology Research*, 8(9), 2216 – 2223.
- Ondimu, S. M. (2018). *Teachers' preparedness for implementation of the competency based curriculum in private pre-schools in Dagoretti North Sub-county, Nairobi City County*. Master Thesis, University of Nairobi.
- Papadakis, S., & Kalogiannakis, M. (2019). Evaluating a course for teaching introductory programming with Scratch to pre-service kindergarten teachers. *International Journal of Technology Enhanced Learning*, 11(3), 231-246.
- Petticrew, M., & Roberts, H. (2006). *Systematic reviews in the social sciences: A practical guide*. Oxford, England: Blackwell.
- Piedade, J., Dorotea, N., Pedro, A., & Matos, J. F. (2020). On teaching programming fundamentals and computational thinking with educational robotics: A didactic experience with pre-service teachers. *Education Sciences*, 10(9), 214.
- Reichert, J. T., Barone, D. A. C., & Kist, M. (2020). Computational Thinking in K-12: An analysis with Mathematics Teachers. *EURASIA Journal of Mathematics, Science and Technology Education*, 2020, 16(6), em1847.
- Rich, K. M., Yadav, A., & Schwarz, C. V. (2019). Computational thinking, mathematics, and science: Elementary teachers' perspectives on integration. *Journal of Technology and Teacher Education*, 27(2), 165-205.
- Tuhkala, A., Wagner, M.-L., Iversen, O. S., & Kärkkäinen, T. (2019). Technology comprehension — Combining computing, design, and societal reflection as a national subject. *International Journal of Child-Computer Interaction*, 20, 54-63.
- Valentine, K. D. (2018). Tinkering with logo in an elementary mathematics methods course. *Interdisciplinary Journal of Problem-based Learning*, 12(2).
- Wang, Z., Utemov, V. V., Krivonozhkina, E. G., Liu, G., & Galushkin, A. A. (2018). Pedagogical readiness of mathematics teachers to implement innovative forms of educational activities. *Eurasia Journal of Mathematics, Science and Technology Education*, 14(1), 543-552.
- Yadav, A., Gretter, S., Good, J., & McLean, T. (2017). Computational thinking in teacher education. In P. Rich & C. B. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 205–220). Springer Publishing Company.
- Yuan, J., Kim, C., Hill, R., & Kim, D. (2019). Robotics integration for learning with technology. *Contemporary Issues in Technology and Teacher Education*, 19(4), 708-735.



# CTE-STEM

5TH APSCE INTERNATIONAL  
CONFERENCE ON COMPUTATIONAL  
THINKING AND STEM EDUCATION

# 2021

<https://cte-stem2021.nie.edu.sg/>  
[cte.stem2021@nie.edu.sg](mailto:cte.stem2021@nie.edu.sg)

