



Faculty of Computer Science and Information Technology

**Effectiveness of Combined Unplugged and Block-Based Programming
Approaches on Computational Thinking Skills and Programming
Attitudes Across Gender**

Badruliman Batni

**Master of Science
2024**

Effectiveness of Combined Unplugged and Block-Based Programming
Approaches on Computational Thinking Skills and Programming Attitudes
Across Gender

Badruliman Batni

A thesis submitted

In fulfillment of the requirements for the degree of Master of Science

(Human Computer Interaction)

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2024

DECLARATION

I declare that the work in this thesis was carried out in accordance with the regulations of Universiti Malaysia Sarawak. Except where due acknowledgements have been made, the work is that of the author alone. The thesis has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.



Name: Badruliman Bin Batni

Matric No.: 22020154

Faculty of Computer Science and Information Technology

Universiti Malaysia Sarawak

Date: 14 October 2024

ACKNOWLEDGEMENT

Alhamdulillah, thank Allah for all the blessings you have given me. The ability, courage, perseverance, and patience you grant me in carrying out this research. First, I would like to express my sincere gratitude to Ministry of Education for letting me be part of *Hadiah Latihan Persekutuan* (HLP) scholarship's programme and the funding of my post-graduate education. Next, my deepest gratitude goes to my primary supervisor, Ts. Syahrul Nizam Junaini, whose insight and knowledge on the subject guided me through this research. His dedicated supervision and constant encouragement to complete this thesis encouraged me to do my best. I would also like to thank the schools, principals, teachers, and students who contributed and participated in data collection for my research.

I owe my loving gratitude to my supportive, encouraging, and patient wife Murdiana Musa and my children Aisya Salsabila, Ammara Sofia and Ammar Sidqi and my greatest gratitude goes to my parents Nuraini Din and Batni Saidon who have provided me with unconditional support and continuous encouragement throughout my years of study and during the process of researching and writing this thesis. Without all of them, this success would not have been possible. Thank you very much.

ABSTRACT

This study addresses the challenge of effectively developing computational thinking (CT) skills and fostering positive attitudes towards programming among secondary students in fully boarding schools in Kedah, Malaysia. The primary objective is to evaluate the impact of integrating Unplugged Activities (UA) with Block-Based Programming (BBP) (UA+BBP) compared to using BBP alone. The proposed solution combines tactile, hands-on activities with digital programming tools to create a more engaging and comprehensive learning experience. Using a quasi-experimental design with mixed-methods analysis, the study involved two groups: one using only BBP and the other using UA+BBP. Quantitative analysis, including two-way ANCOVA and Welch's ANOVA, was used to assess the effects of teaching methods and gender on CT skills and programming attitudes. Results showed that while both groups improved in CT skills, there were no statistically significant differences in post-test scores between the two teaching methods. Gender did not significantly impact CT skill development. However, the UA+BBP group exhibited significantly higher attitudes towards programming compared to the BBP-only group (mean difference = 0.65741, $p < 0.001$). Additionally, the integrated approach balanced gender differences in attitudes, unlike the BBP-only group where males scored significantly higher than females. Qualitative data from student interviews provided deeper insights, revealing that fun and challenging UA+BBP activities enhanced their understanding of CT and made programming more accessible and enjoyable. The findings suggest that integrating unplugged activities with BBP not only enriches students' learning experiences but also cultivates a more positive and inclusive attitude towards programming. This study underscores the potential of blended teaching strategies in advancing CT education in secondary schools.

Keywords: Computational thinking, secondary school, unplugged activity, block-based programming

Keberkesanan Pendekatan Gabungan Aktiviti Tanpa Komputer dan Pengaturcaraan Berasaskan Blok terhadap Kemahiran Pemikiran Komputasional dan Sikap Terhadap Pengaturcaraan Merentas Jantina

ABSTRAK

Pengaturcaraan Berasaskan Blok (PBB) dalam meningkatkan kemahiran pemikiran komputasional (PK) dan sikap terhadap pengaturcaraan di kalangan pelajar sekolah menengah di sekolah berasrama penuh di Kedah, Malaysia. Penyelidikan ini menangani masalah keberkesanan dalam membangunkan kemahiran PK dan memupuk sikap positif terhadap pengaturcaraan dalam persekitaran sekolah. Menggunakan reka bentuk kuasi-eksperimen dan analisis kaedah campuran, kajian ini membandingkan dua kumpulan: satu menggunakan hanya PBB dan satu lagi menggabungkan ATK dengan PBB (ATK+PBB). Kaedah kuantitatif termasuk ANCOVA dua hala untuk menganalisis kesan kaedah pengajaran dan perbezaan jantina terhadap kemahiran PK dan ANOVA Welch untuk menilai sikap pelajar terhadap pengaturcaraan. Keputusan menunjukkan bahawa walaupun kemahiran PK meningkat dalam kedua-dua kumpulan, perbezaan skor PK pasca-ujian antara kumpulan PBB dan ATK+PBB tidak signifikan secara statistik. Jantina tidak mempengaruhi perkembangan kemahiran PK secara signifikan. Walau bagaimanapun, kumpulan ATK+PBB menunjukkan skor sikap yang lebih tinggi secara signifikan terhadap pengaturcaraan berbanding kumpulan PBB sahaja (perbezaan min = 0.65741, $p < 0.001$). Perbezaan jantina dalam sikap adalah signifikan dalam kumpulan PBB, dengan lelaki menunjukkan skor yang lebih tinggi (perbezaan min = 0.42045, $p = 0.016$), manakala kumpulan ATK+PBB mengimbangi sikap ini merentas jantina (perbezaan min = -0.039, $p = 0.976$). Analisis kualitatif temubual pelajar dari kumpulan ATK+PBB memberikan gambaran yang lebih mendalam mengenai pengalaman mereka, mendedahkan bahawa penyertaan dalam aktiviti yang menyeronokkan dan mencabar secara signifikan

meningkatkan pemahaman mereka tentang PK dan menjadikan pengaturcaraan lebih mudah diakses dan menyeronokkan. Penemuan ini menyoroti manfaat pendekatan bersepadu yang menggabungkan pengalaman pembelajaran berasaskan sentuhan dengan teknologi digital. Pendekatan ini bukan sahaja meningkatkan pemahaman pelajar mengenai konsep PK tetapi juga memupuk sikap positif dan penglibatan yang lebih besar dalam pengaturcaraan.

Kata kunci: *Pemikiran komputasional, sekolah menengah, aktiviti tanpa komputer, pengaturcaraan berasaskan blok*

TABLE OF CONTENTS

	Page
DECLARATION	i
ACKNOWLEDGEMENT	ii
ABSTRACT	iii
<i>ABSTRAK</i>	v
TABLE OF CONTENTS	vii
LIST OF TABLES	xiv
LIST OF FIGURES	xv
LIST OF ABBREVIATIONS	xvi
CHAPTER 1 INTRODUCTION	1
1.1 Overview	1
1.2 Study Background	1
1.3 Problem Statement	3
1.4 Objectives	7
1.5 Research Questions	7
1.6 Definition of Terms	9
1.7 Significance of the Study	10
1.8 Scope of the Study	12

1.9	Structure of the Thesis	14
CHAPTER 2 LITERATURE REVIEW		16
2.1	Overview	16
2.2	Computational Thinking and Programming	16
2.3	Trends and Developments in Computational Thinking Education	20
2.4	Computational Thinking for Problem Solving	22
2.5	Gender Different in Computational Thinking	24
2.6	Computational Thinking Landscape	26
2.6.1	Computational Thinking in K12	26
2.6.2	Integrated Computational Thinking Learning in Other Subjects	30
2.6.3	Pedagogical Strategies for Teaching Computational Thinking	34
2.6.4	Computational Thinking Teaching Tool	39
2.7	The Relationship Between Computational Thinking, Unplugged Activities, and Block-Based Programming	43
2.7.1	Unplugged Activities in Computational Thinking Education	43
2.7.2	The Role of Block-Based Programming (BBP) in Developing Computational Thinking Skills	47
2.7.3	Rationale for the Current Research Focus	50
2.8	Literature Gap	51
2.8.1	Gender Differences in Computational Thinking	52
2.8.2	Context-Specific Studies	52

2.8.3	Hypothesis Development	53
2.9	Theoretical Framework	55
2.10	Summary	57
CHAPTER 3 RESEARCH METHODOLOGY		58
3.1	Overview	58
3.2	Research Design	58
3.3	Variables and Instrument Development	59
3.3.1	Dependent and Independent Variables	59
3.3.2	Computational Thinking Test	61
3.3.3	Programming Attitude Survey	62
3.3.4	Semi-Structured Interview Protocol	64
3.4	Revisions of Research Instruments after Pilot Test	67
3.5	Selection of Respondents	68
3.5.1	Sampling Strategy	68
3.5.2	Sample Size and Composition	69
3.5.3	Inclusion Criteria	71
3.6	Data Collection Methods	71
3.7	Data Analysis	73
3.7.1	Quantitative Data Analysis	73
3.7.2	Qualitative Data Analysis	74

3.8	Classroom Intervention Implementation	75
3.8.1	Teacher Training	76
3.8.2	Intervention Session Structure	76
3.8.3	Teacher Input and Lesson Plan Review	77
3.8.4	Participants for Pilot Test	78
3.9	Intervention Implementation Rationale	78
3.9.1	Rationale for Scratch as a Teaching Tool	78
3.9.2	Selection and Application of UA	80
3.9.3	Summary	81
	CHAPTER 4 RESULTS	83
4.1	Overview	83
4.1.1	Data Screening, Missing Data, Outliers, and Normality Analysis	83
4.1.2	Data Screening and Handling Missing Data	84
4.1.3	Outliers Detection and Management	84
4.1.4	Testing the Assumptions of Normality and Homogeneity	85
4.2	Testing the Assumptions of Normality and Homogeneity	85
4.2.1	Normality of CT Test Score	86
4.2.2	Normality of Attitudinal Survey Scores	88
4.2.3	Levene's Test and Further Analysis	89
4.3	Rationale for Using ANCOVA and Welch's ANOVA	89

4.4	Justification for Using ANCOVA	90
4.5	Justification for Using Welch's ANOVA	90
4.6	Research Findings	91
4.6.1	Two-Way ANCOVA	91
4.6.2	RQ1: Impact of Teaching Method on CT Skills	93
4.6.3	RQ2: Gender Differences in CT Skills Across Teaching Methods	94
4.6.4	Welch's ANOVA	95
4.6.5	RQ3: Influence of UA+BBP on Attitudes Towards Programming	97
4.6.6	RQ4: Gender Differences in Attitudes Towards Programming	98
4.7	Qualitative Analysis of Student Interviews	99
4.8	Summary	103
	CHAPTER 5 DISCUSSION	104
5.1	Overview	104
5.2	Effectiveness of Integrating UA with BBP in Enhancing CT Skills	104
5.3	Gender Differences in CT Development with UA+BBP and BBP Only Approaches	106
5.4	Impact of UA+BBP versus BBP Only on Student Attitudes Toward Programming	109
5.5	Gender-Based Attitudinal Differences in Response to UA+BBP and BBP Only Methods	111
5.6	Thematic-based Findings	112

5.6.1	Theme 1: Improving Programming Through Fun and Challenging Activities	113
5.6.2	Theme 2: Simplifying Complex Programming Concepts	113
5.6.3	Theme 3: Foundation for Programming	114
5.6.4	Theme 4: Changes in Attitude Towards Programming	114
5.7	Implications	115
5.7.1	Theoretical Implications	115
5.7.2	Practical Implications	116
5.8	Limitation	118
5.8.1	Sample Specificity	118
5.8.2	Measurement Tools	119
5.8.3	Data Collection and Analysis	119
5.8.4	Implementation Factors	120
5.9	Summary	120
CHAPTER 6 CONCLUSION AND FUTURE WORK		121
6.1	Overview	121
6.2	Conclusion	121
6.2.1	Objective 1	121
6.2.2	Objective 2	122
6.2.3	Objective 3	122
6.2.4	Objective 4	122

6.3	Future Work	123
6.3.1	Longitudinal Studies on CT Skills and Academic Outcomes	123
6.3.2	Expanding Research to Diverse Educational Contexts	123
6.3.3	Evaluating Teacher Efficacy and Attitude Towards UA+BBP	124
6.3.4	Developing Multidisciplinary CT Curriculum Designs	124
6.3.5	Gender-Specific Responses and Inclusivity in STEM Education	124
	REFERENCES	126
	APPENDICES	159

LIST OF TABLES

	Page
Table 1.1: Defination of Terms Used	9
Table 2.1: Mapping CT Elements to Programming	19
Table 3.1: Distribution of Students by Group, School, and Gender	70
Table 4.1: Normality Test for CT Test Score by Group	87
Table 4.2: Normality Test for the CT Test Score by Gender	87
Table 4.3: Normality Test for Attitudinal Survey Scores by Group	88
Table 4.4: Normality Test for Attitudinal Survey Scores by Gender	89
Table 4.5: Descriptive Statistics for Post-test CT Test Scores	92
Table 4.6: Two-way Analysis of Covariance (ANCOVA) Result	93
Table 4.7: Descriptive Statistics for Attitude Toward Programming Survey	95
Table 4.8: Post Hoc Test Analysis for Differences in Attitudes Toward Programming	96
Table 4.9: Quotes from Student Interview Responses	99

LIST OF FIGURES

	Page
Figure 2.1: Theoretical Framework	56

LIST OF ABBREVIATIONS

CGS	Centre for Graduate Studies
UNIMAS	Universiti Malaysia Sarawak
CT	Computational Thinking
UA	Unplugged Activity
BBP	Block Based Programming
ICT	Information and Communication Technology
PISA	Program for International Student Assessment
ISTE	International Society for Technology in Education
CSTA	Computer Science Teachers Association
STEM	Science Technology Engineering and Mathematic
STEAM	Science Technology Engineering Art and Mathematic
MIT	Massachusetts Institute of Technology
ADRI	Approach Deployment Result Improvement
CT-S	Computational Thinking for Science
KCK	Kniwwelino Classroom Kit
5E	Engagement Exploration Explanation Elaboration Evaluation
IoT	Internet of Things
AI	Artificial Intelligence
TPACK	Technological Pedagogical Content Knowledge
PMA	Primary Mental Abilities

CHAPTER 1

INTRODUCTION

1.1 Overview

This chapter presents the background of the study, problem statement, and research objectives, as well as the research questions. It also highlights the conceptual framework and provides clear definitions of the key terms used throughout the research. Additionally, it outlines the significance and scope of the study, explaining the importance of the research in the broader context of computational thinking (CT) education.

1.2 Study Background

The world stands on the precipice of the fourth industrial revolution, also known as Industry 4.0, where the labour market is evolving at an unprecedented pace. According to the World Economic Forum (2020), while automation may displace up to 85 million jobs by 2025, it is poised to generate around 97 million new roles, demanding advanced technological skills. To keep pace with this seismic shift, individuals and organizations alike need to stay attuned to technological trends and cultivate relevant skills in technology and programming (OECD, 2019). Ayuso et al. (2022) emphasize that 70% of STEM jobs today require programming expertise, highlighting an urgent need for a skilled workforce.

In response to the digital transformation sweeping through our educational landscapes, CT has emerged as a pivotal cornerstone in education. Originally conceptualized by Papert (1980), CT transcends the traditional boundaries of computer science. It's not just about programming, but a powerful cognitive framework that empowers individuals to deconstruct complex problems into manageable components, find patterns, and create algorithmic solutions that both humans and computers can understand (Aho, 2012; Yadav et

al., 2018). Wing (2006) assertion that CT is a fundamental skill for all not just for computer scientists. It is more relevant today than ever and highlights the importance of integrating CT at all levels of education.

Educators worldwide are now challenged to not only cultivate innovation and creativity but also to impart the technical skills necessary for solving increasingly complex problems. Studies indicate that CT skills bolster analytical capabilities, enhance critical thinking, and improve decision making across various student demographics (Cheng et al., 2023; Orhan and Çeviker Ay, 2023; Tiruneh et al., 2017). As technology's role in our lives continues to grow, there is a pressing need for education systems to evolve and weave CT effectively into curricula. This requires a holistic approach that marries programming knowledge with essential digital skills such as creativity and problem-solving (Juškevičienė et al., 2021; Shute et al., 2017). With students at risk of falling behind if they lack these skills, educators are tasked with finding the most effective ways to nurture CT in classrooms.

One such approach is Block-Based Programming (BBP), which has proven to be a popular and effective way to teach CT. Tools like Scratch and Blockly offer structured learning environments where students manipulate blocks of code to create programs, giving them a taste of real-world programming while maintaining a fun and engaging atmosphere (Alp and Bulunuz, 2023; Jiang and Li, 2021; Kwon and Cheon, 2019). Weintrop and Wilensky (2017) found that BBP not only imparts technical skills but also fosters long-term interest in computer courses. However, challenges persist, such as teacher training requirements and the tendency of students to focus too heavily on the visual aspects rather than the underlying logic of programming.

In contrast, unplugged activities (UA) provide an alternative pathway to learning CT by using physical games and activities that require no computers. These activities help

students build foundational concepts through hands-on learning, making abstract ideas more concrete (Aivaloglou and Hermans, 2016; del Rey et al., 2021). Research shows that UA can improve student motivation and engagement, especially for those who have had limited exposure to programming (Yin et al., 2020). Moreover, recent studies by Parham-Mocello et al. (2023) discovered that game-based, non-coding curricula are effective in helping middle school students understand complex concepts such as algorithms and conditions.

Building on this foundation, integrating UA with BBP could create a more comprehensive educational strategy. While BBP provides structured and technical instruction, UA introduces a tactile, game-like element that enhances comprehension and engagement, suggesting that a synergistic combination of these approaches could forge a robust framework for developing CT skills across diverse learning environments. By examining the instructional sequencing of these approaches, this study seeks to understand how their integration can improve CT concepts and practices among secondary school students, while also exploring how these methods influence student attitudes toward programming.

1.3 Problem Statement

The global Programme for International Student Assessment (PISA) provides a clear benchmark for the effectiveness of education systems in cultivating these essential skills. The results of the 2022 PISA assessment highlight a concerning trend: Malaysian students performed well below the OECD averages in mathematics (41% proficiency), reading (42%), and science (52%)—all core areas linked to CT (OECD, 2023). These deficiencies suggest a deeper issue within Malaysia's education system, which fails to effectively integrate CT concepts into the curriculum (Hashim and Husnin, 2019). Moreover, the increasing demands for digital literacy and problem-solving in various sectors have

magnified the urgency to improve CT education at the secondary school level (Angeli et al., 2016).

These findings point to a general problem in Malaysia's secondary school curriculum, which does not adequately prepare students with the essential problem-solving and technical skills required for success in both higher education and the workforce (Jawawi et al., 2022). More broadly, this reflects a global issue in CT education, where a focus on coding and programming often overshadows the development of broader problem-solving skills (Lye and Koh, 2014). Without the right strategies to foster these abilities, students are ill-equipped to meet the challenges posed by modern technological advancements.

One of the primary specific problems in CT education is the reliance on programming-focused curricula that overlook the multifaceted nature of CT. Current educational models often prioritize teaching students to code, rather than emphasizing the broader cognitive skills necessary to solve real-world problems (Kafai and Proctor, 2022). This programming-centric approach not only narrows the scope of CT education but also alienates students who might struggle with coding but possess strong problem-solving potential in other aspects of CT (Grover and Pea, 2013).

For instance, while programming teaches algorithmic thinking, it is only one component of CT. Other equally important skills, such as abstraction, pattern recognition, and generalization, are often neglected (Wing, 2008). As a result, students are exiting these programs without the ability to effectively decompose and analyse complex problems, limiting their ability to apply CT in real-world contexts (Brennan and Resnick, 2012).

Moreover, research shows that students frequently hold misconceptions about what CT entails, leading to disengagement and negative attitudes towards the subject (Kite and

Park, 2023). This problem is exacerbated by outdated teaching methods that rely heavily on individual work rather than promoting active, collaborative learning environments, which have been proven to enhance CT skills (García-Peñalvo et al., 2020). Collaborative learning fosters deeper understanding and engagement, yet many educators are either unaware of or unequipped to implement these strategies effectively (Ye et al., 2023).

Another specific problem is the neglect of gender differences in the way CT is taught and experienced by students. Studies have shown that male and female students often approach CT learning differently, with male students typically more inclined towards competitive learning environments and female students benefiting more from collaborative, hands-on experiences (Threekunprapa and Yasri, 2020). Despite this, many current CT teaching models fail to account for these differences, leading to a gender gap in engagement and achievement in CT-related fields (Gao et al., 2023). Addressing these discrepancies is critical to ensuring equitable access to CT education for all students, regardless of gender.

Furthermore, an over-reliance on technology-based solutions for CT education presents another significant challenge. While technology tools such as robot kit or computer based activity offer valuable opportunities for students to practice programming, they can unintentionally exacerbate resource and financial constraints, particularly in underdeveloped or rural areas where access to technology is limited (Ogegbo and Ramnarain, 2022). These inequalities further widen the digital divide and hinder efforts to democratize CT education. In such contexts, teaching strategies that rely heavily on technology fail to provide equitable learning experiences, leaving many students without the necessary resources to engage fully with the curriculum (Ung et al., 2018)

This study responds to these educational deficiencies by investigating how the integration of UA with BBP can provide a more comprehensive and equitable approach to

CT education. UA offer a low-tech, hands-on approach to teaching CT concepts, allowing students to engage with abstract ideas through physical, tangible means (Munasinghe et al., 2023). By integrating UA with BBP, students can first develop a conceptual understanding of CT without the barriers posed by technology, and then apply these concepts in a BBP environment to solidify their learning. This dual approach not only reinforces problem-solving skills but also addresses resource disparities by providing low-tech alternatives for learning CT (Habib et al., 2021).

In addition, this study will evaluate how integrating UA with BBP can create more inclusive learning environments that cater to the diverse needs of students, particularly in terms of gender differences. The UA+BBP method emphasizes active, collaborative learning, which research shows is especially effective for female students who may not engage as strongly with competitive, coding-focused curricula (Hiley et al., 2023). By fostering an environment where students of all genders can thrive, this approach aims to reduce the gender gap in CT education.

Finally, the deficiencies in Malaysia's CT education, as highlighted by poor PISA results and widespread reports of disengagement among students, underscore the need for innovative teaching strategies that go beyond programming and foster critical problem-solving skills. This study will provide crucial insights into how the integration of UA and BBP can bridge these gaps and create more inclusive, equitable, and effective CT education at the secondary school level.

1.4 Objectives

The primary aim of this study is to explore the effectiveness of integrating unplugged activities and block-based programming in enhancing computational thinking skills among secondary school students. To achieve this aim, the study is organized around the following objectives:

- i. To measure the effectiveness of integrating unplugged activities with block-based programming on students' computational thinking concepts and practices.
- ii. To analyse the influence of gender on students' computational thinking concepts and practices when comparing the integrated unplugged activities + block-based programming teaching method to block-based programming alone.
- iii. To evaluate the impact of different teaching methods (unplugged activities + block-based programming versus block-based programming alone) on students' attitudes toward programming.
- iv. To compare gender differences in students' attitudes toward programming when exposed to unplugged activities + block-based programming versus block-based programming alone teaching methods.

1.5 Research Questions

The following research questions will be addressed to understand the impact and implications of integrating unplugged activities with block-based programming on computational thinking:

- i. RQ1: Does the teaching method (unplugged activities + block-based programming vs. block-based programming alone) significantly affect students' computational thinking skills?
- ii. RQ2: Does gender influence students' computational thinking skills across the different teaching methods (unplugged activities + block-based programming vs. block-based programming alone)?
- iii. RQ3: Does the teaching method (unplugged activities + block-based programming vs. block-based programming alone) affect students' attitudes toward programming?
- iv. RQ4: Are there gender differences in students' attitudes toward programming when comparing unplugged activities + block-based programming and block-based programming alone?

According to such research objective and research question, the study developed the following null hypotheses:

- i. H₀₁: There is no significant difference in computational thinking skills between students taught using unplugged activities + block-based programming and those taught using BBP alone.
- ii. H₀₂: Gender does not significantly influence computational thinking skills across different teaching methods (unplugged activities + block-based programming vs. block-based programming alone).
- iii. H₀₃: The integration of unplugged activities + block-based programming does not significantly affect students' attitudes toward programming compared to block-based programming alone.

- iv. H₀4: There are no significant gender differences in students' attitudes toward programming between unplugged activities + block-based programming and block-based programming alone.

1.6 Definition of Terms

The table 1.1 provides both conceptual and operational definitions for key terms used throughout this study. These definitions are crucial for understanding the scope and context of the research, ensuring clarity and consistency in how terms are interpreted and applied.

Table 1.1: Defination of Terms Used

Terms	Conceptual Definition	Operational Definition
Computational Thinking	A problem-solving approach involving identifying a problem, breaking it down, recognizing patterns, and creating steps (algorithms) to solve it (Tsarava et al., 2022; Wing, 2011).	A method of solving problems by breaking them into smaller parts, finding patterns, and creating algorithms.
Programming	Writing code to create instructions for a computer, helping students learn math concepts and problem-solving skills in K-12 education (Reynolds et al., 2010; Tehlan et al., 2020).	Designing and running algorithms in a BBP environment to understand CT principles like abstraction and loops.

Unplugged Activity	Teaching CT and programming without computers, using hands-on tasks that simulate real programming scenarios (Caeli and Yadav, 2020; Peel et al., 2022).	Hands-on tasks that teach programming concepts without computers, focusing on logical structuring and algorithmic thinking.
Block-Based Programming	Using visual code blocks to create correct code by preventing errors and guiding users (Weintrop and Wilensky, 2015).	Implemented with Scratch, using projects like drawing shapes and creating a math race game to teach CT concepts.
Programming Perception	Students' attitudes and beliefs about programming, including difficulty, interest, and ability (Kong and Wang, 2019; Schut et al., 2018).	Surveying students on their enjoyment, interest, and confidence in programming.
K-12 Education	Education from kindergarten through 12th grade, providing essential knowledge and skills for higher education and careers (National Research Council, 2012).	The school setting from kindergarten, primary school, and secondary students. In this study, the focus is specifically on secondary school students aged 13 years old.

1.7 Significance of the Study

The significance of this study is manifold. Firstly, it addresses the recognized need to transition from traditional teacher-centred educational models to more dynamic and

interactive student-centred learning approaches. These student-centred environments are known to be more engaging, competitive, entertaining, and content-rich, which significantly enhances students' overall learning experiences (Srinivasan and Centea, 2019). Furthermore, previous research underscores the importance of integrating UA, especially for students who may have limited computer skills. Studies suggest that combining modern technology with student-centred learning strategies significantly aids in the acquisition of complex and novel skills (Battal et al., 2021; Sitthiworachart et al., 2022).

This synergy between technology and pedagogical strategies not only enriches the learning environment but also empowers students to develop essential computational thinking and programming skills more effectively. However, this study offers a critical perspective by focusing on the often-overlooked aspect of blending traditional and modern educational techniques. Unlike previous research that typically emphasizes technology-based learning, this study underscores the significance of UA in enhancing CT and programming perceptions.

Second, applying cognitive science to educational strategies offers profound insights into improving learning outcomes. Recent research by Xu et al. (2021) explores the brain's constraints when engaging in complex thinking tasks, underscoring the necessity for teaching strategies that are cognitively optimized. Cognitively optimized strategies often involve dual coding using both verbal and visual materials. This approach helps create more robust memory traces by providing multiple ways to retrieve the same information.

By structuring and supporting the learning process, and gradually removing these aids as learners become more proficient, this study aims to build understanding incrementally without overwhelming students. This method aligns with cognitive load theory, ensuring that students can manage the complexity of learning tasks more effectively.

The insights gained from this study can inform the development of teaching methods that enhance cognitive processing, leading to improved educational outcomes and a deeper understanding of CT concepts and practices.

Third, the psychological aspects of learning, such as the joy and fulfilment that come from problem solving, play a crucial role in the educational environment. Students feel a sense of achievement when they overcome challenges, which increases their motivation and engagement in learning (Grace et al., 2020). Furthermore, the significant distraction caused by the over-reliance on technological devices such as laptops or computers in classrooms highlights the need for educational models that promote active participation over passive consumption (Attia et al., 2017; Pérez-Juárez et al., 2023). This study supports a pedagogical shift toward incorporating more engaging, competitive, and entertaining elements into the classroom, which fits well with our cognitive predisposition for active learning.

In addition, the diversity of students' learning styles requires a pedagogical approach that considers different sensory preferences. The integration of UA and BBP not only supports kinaesthetic and visual learners by providing hands-on and visual learning experiences, but also increases the inclusivity and effectiveness of the learning environment and helps students better understand abstract CT concepts and practices (del Olmo-Muñoz et al., 2020; Zhang et al., 2021). Drawing on insights from existing literature, the utilization of UA and BBP improve students' attitudes toward programming, and reduce cognitive load, fostering a more comprehensive and effective learning environment.

1.8 Scope of the Study

The main aim of the present study is to investigate the effectiveness of integrating UA+BBP in developing CT skills in secondary school students. Specifically, the study aims to evaluate the impact of the intervention on students' conceptual understanding of CT and

their ability to apply CT practices. A sample of 106 students aged 13 years from three fully boarding schools (*Sekolah Berasrama Penuh*) in Kedah were selected to participate in the study.

To assess the effectiveness of the intervention, the study collected quantitative data limited to CT tests and attitudes toward programming surveys. Rigorous statistical analyses were performed to ensure the validity and reliability of the data collected. The CT tests are used to assess CT skills and ensure compliance with recognized standards. Furthermore, the attitudes toward programming surveys, aim to capture students' perspectives on programming, particularly their interest, enjoyment, perceived challenges, and confidence. These surveys provide deeper insights into the affective dimensions of learning and complement competency assessment with a more comprehensive understanding of student engagement and attitudes.

In addition, as part of the research, qualitative data were collected through interviews with the participants limited to only UA+BBP group. These qualitative methods aim to examine students' perceptions of the effectiveness of the intervention and their level of engagement. Carefully crafted interview questions address the students' experiences, viewpoints, and evaluations of the planning, implementation, and outcomes of the intervention. To gain deeper insights into the impact of the intervention, a comprehensive qualitative analysis is carried out. The study specifically examines the inclusion of UA and BBP in improving students' CT skills, with a focus limited to CT concepts and practices. The BBP language Scratch was the only programming language used throughout the study. Previous research had demonstrated Scratch's effectiveness in enhancing CT skills (Brennan and Resnick, 2012; Hsu et., 2021). The UA are carefully selected and tailored to the teaching objectives to ensure their relevance and effectiveness.

It is important to note that this study specifically focuses only on students in fully boarding schools, given the unique and supportive learning environment they provide (Bass, 2014; Shi, 2020). The controlled and structured nature of boarding schools offers advantages in terms of the validity and precision of the study's results, providing a solid foundation for future research in the field. However, caution should be exercised when generalizing the findings to other educational contexts. In summary, the scope of the study includes the use of CT tests and attitudinal surveys, with a focus on UA and BBP scratch. The study focuses specifically on students in fully boarding schools. Findings have the potential to support the development of innovative pedagogical interventions aimed at improving CT skills in secondary school students.

1.9 Structure of the Thesis

This thesis is organized into six chapters to provide a comprehensive overview of research examining the effectiveness of combining UA and BBP in developing CT skills in secondary school students. **Chapter one** introduces the research topic by providing background study, defining the problem, outlining the research objective and questions, and presenting the conceptual framework. The definition of terms, significance of the study and its scope are also discussed.

Chapter two provides a comprehensive review of the literature on CT education and programming. It examines recent trends and developments in the field, examines the application of CT in problem-solving contexts, and discusses the integration of CT into K-12 education. Additionally, this chapter addresses the role of gender in CT education and provides an overview of the broader CT educational landscape. The literature gap and theoretical framework underlying the study is also described in detail in this chapter.

Chapter three describes the research methodology and provides details of the research design, sampling methods, data collection techniques, intervention implementation and data analysis procedures. **Chapter four** presents the results for each research question. For RQ1 to RQ3, the chapter offers detailed comparisons of CT test scores between the groups and by gender, highlighting any significant interactions between teaching approaches and gender. For RQ4 to RQ6, it examines the differences in scores on the attitudinal survey based on group and gender, as well as the interaction between them.

Chapter five discusses the results for each research question in more detail, explains the implications for educational practice, and highlights the limitations of this study. **Chapter six** concludes the work by summarizing the research findings. In addition, recommendations for future research are provided and opportunities for further investigation are identified.

CHAPTER 2

LITERATURE REVIEW

2.1 Overview

This chapter reviews previous and current work related to this study, addressing fundamental of CT and programming, its development, and its impact across different educational levels. Additionally, the review explores CT's role in problem-solving, investigates gender differences, and examines its integration into educational environments. The synergistic effects of combining UA and BBP, along with various aspects of CT education, are analysed. This analysis provides a comprehensive understanding of trends, challenges, and advancements in the field. The theoretical framework of the current study is discussed at the end of the chapter.

2.2 Computational Thinking and Programming

The inherent digital literacy of the new generation has driven a shift toward dynamic, technology-enabled teaching methods aligned with the educational needs of the 21st century (Leavy et al., 2023). This shift has led to a growing body of research focused on the integration and assessment of CT in educational systems worldwide. Hutchins et al. (2020) highlight this trend by highlighting an increasing focus on the practical implementation and assessment of CT skills in educational settings. This represents an important turning point and emphasizes that CT is a critical priority.

In the mid-2000s, Jeannette Wing emerged as a leading advocate of CT, pushing for its integration into education alongside reading, writing, and arithmetic. In her seminal 2006 paper, Wing introduced CT as a critical skill set for all individuals, not just computer scientists. She underscored that CT involves problem-solving techniques derived from

computer science, such as abstraction, algorithm design, and automation. Wing advocated for the inclusion of CT in K-12 education, emphasizing that these skills are essential for developing students' ability to think critically and systematically (Wing, 2006). In 2008, Wing elaborated on her initial conceptualization of CT, delineating its core components more explicitly. She identified key CT practices, including decomposition (breaking down complex problems into manageable parts) and pattern recognition. This expanded definition aimed to provide educators with a clearer framework for incorporating CT into their teaching strategies, thereby fostering students' ability to approach problems with a computational mindset (Wing, 2008).

By 2011, Wing's advocacy had further evolved to emphasize the necessity of CT across various educational and professional fields. She categorized CT competencies into four main areas: decomposition, pattern recognition, pattern generalization and abstraction, and algorithm design (Wing, 2011). This categorization was designed to help educators develop curricula that systematically build these skills in students. Wing's continued efforts highlighted the importance of CT in preparing students to navigate and succeed in an increasingly technology-driven world. The advocacy helped CT expand beyond its niche in computer science and make it a fundamental part of academic curricula (Hurt et al., 2023; Santo et al., 2022). This era marked a change in which CT became indispensable for students of various disciplines.

Aho (2012) described CT as the mental process required to develop problem-solving skills, like how computers work. Over time, CT has evolved beyond its computer-centric roots to become a critical cognitive skill for understanding the world through information processing and system design principles (Denning and Tedre, 2021). CT is no longer limited

to computer scientists but is now recognized as a universal strategy for innovatively addressing challenges (Barr and Stephenson, 2011; Moreno-Leon and Robles, 2016).

The growing body of CT research underscores its pivotal role in education and lifelong learning. Researchers like Erümit (2020) and Karaahmetoğlu and Korkmaz (2019) emphasize that CT's core elements such as decomposition, pattern recognition, and programming are versatile tools applicable across disciplines. They are crucial in programming, robotics, and STEM education, helping learners grasp complex concepts, innovate, and refine their analytical thinking. These tools provide the foundation for creating systematic strategies that can tackle challenges in data analysis, engineering, scientific research, and project management.

As CT gains prominence in various educational contexts, its principles are increasingly influencing curricula at all levels. The emphasis on computer literacy helps students adapt to an evolving technology landscape and navigate a data-driven world. By using CT in education, institutions prepare learners to address challenges in various areas creatively and effectively, fostering skills essential for the future. Recognizing the need for a structured educational framework, the International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) collaborated in 2011 to define and refine CT's essential competencies. They emphasized abstraction, algorithmic thinking, and automation as foundational components. Algorithms break tasks into manageable steps, abstraction promotes pattern recognition, and automation increases efficiency through digital tools (ISTE and CSTA, 2011).

While definitions may evolve as technology and computing knowledge advance, this framework remains valuable to researchers and practitioners developing interventions and assessments for CT (Barr and Stephenson, 2011). Despite no consensus on precise

components, problem decomposition, abstraction, algorithmic thinking, conditional logic, iterative and recursive thinking, and debugging are generally accepted as the primary elements (Grover and Pea, 2013). Brennan and Resnick (2012) emphasize that CT principles such as decomposition, pattern recognition, abstraction and algorithmic thinking are crucial for solving various programming challenges.

These principles form the basis of effective programming practices and help beginners easily understand key CT features such as problem solving and algorithmic thinking (Flórez et al., 2017). Decomposition allows complex programming problems to be broken down into manageable tasks, pattern recognition identifies recurring elements and themes that enable efficient code reuse, abstraction simplifies problem solving by hiding unnecessary details, and algorithmic thinking provides clear, systematic, and efficient logical sequence programs.

Mapping CT into programming, loops enhance efficiency in repetitive tasks, embodying the concept of pattern recognition (Grover and Pea, 2013). Strategic function calls streamline multiple operations through abstraction, and algorithms, as described by Aho (2012), are detailed plans that solve problems step-by-step. Together, these principles form a cohesive framework that programmers rely on to build innovative, scalable solutions, emphasizing how CT provides practical strategies to address a wide range of programming challenges. Table 2.1 illustrates how CT elements map to programming notations, demonstrating the application of theoretical concepts in programming.

Table 2.1: Mapping CT Elements to Programming

CT elements	Programming Notation
Decomposition	Keywords, assignments
Pattern Recognition	Loops, comparisons, Conditions, function definitions

Table 2.1: Continued

Abstraction	Function calls
Algorithmic	A correct sequence of statements that capable of produce output, defining a function, loop, statements.

The relationship between CT concepts and programming techniques offers practical frameworks that enhance problem-solving capabilities. This synergy between CT and programming can significantly improve students' analytical skills, empowering them to address complex challenges creatively and systematically. Additionally, the integration of CT and programming encourages students to embrace a growth mindset, as they can learn iteratively through practical exercises and refine their approaches over time.

2.3 Trends and Developments in Computational Thinking Education

The field of CT in educational research has experienced a remarkable revival, reflecting a broader academic effort to combine theoretical models with practical educational tactics. Pollak and Ebner (2019) highlight a committed movement to refine teaching methods and more fully integrate computers into educational systems. Following this development, Pelánek and Effenberger (2023) observe a significant shift towards practical application and thorough assessment of CT skills in educational contexts. This shift marks a significant development from purely theoretical discussions about CT towards its concrete integration and effectiveness in teaching and learning environments.

To further advance the theoretical landscape, Tikva and Tambouris (2021) developed a specific conceptual framework for CT education aimed at K-12 programming. Their comprehensive analysis of 101 scholarly papers identified six key elements essential to a solid CT education: foundational knowledge, teaching strategies, assessment tools,

educational resources, influencing factors, and institutional capabilities. This framework not only describes the essential components for effective CT teaching, but also establishes guidelines for creating CT curricula that are both comprehensive and flexible and tailored to different educational environments.

A crucial aspect of CT research focuses on the assessment of CT capabilities, which is proposed by Tang et al. (2020) were carefully examined. Their review of 96 journal articles examines how CT assessments are conducted in educational settings, paying particular attention to the reliability and type of assessments used. This study highlights the urgent need for wider adoption of CT assessments in secondary schools, colleges and teacher training programs and advocates for methodological approaches to measuring and strengthening CT competencies. Such assessments are critical to validating the success of CT educational strategies and ensuring that they meet the pedagogical goals of promoting problem-solving skills in students.

Research on CT education has shown significant growth and evolution in recent years. Studies show that CT is essential for adapting to the future. Educational experiments have demonstrated the effectiveness of teaching CT to young children through various media (Akiba, 2022; Bers et al., 2014; Critten et al., 2022; Kalelioglu and Sentance, 2020; Sun et al., 2021). The integration of CT into K-12 science standards highlights the importance of teaching CT as a science discipline (Hurt et al., 2023; Peters-Burton et al, 2023; Sengupta et al., 2013).

Assessment tools for CT have been developed and validated for different education levels, emphasizing the need to evaluate students' learning progress (Román-González et al., 2017; Tsarava et al., 2022). Bibliometric mapping analyses have revealed that CT research is an emerging field with a focus on integrating CT into STEM education, assessing CT

skills, and defining CT and its skills (Tekdal, 2021). Overall, these trends emphasize the increasing recognition of CT as a crucial skill set for students across different educational levels.

2.4 Computational Thinking for Problem Solving

CT has become a basic skill that includes problem solving, systematic thinking and algorithmic thinking. These skills are now recognized not only as academic exercises but also as crucial competencies in various professional and academic fields. Hwang and Tu (2021) and Ogegbo and Ramnarain (2022) conducted a comprehensive study of the fundamental concepts and techniques of CT, ranging from problem representation to prediction. Their study highlighted the importance of CT not only in computer science, but also as an essential tool for tackling complex challenges across a wide range of scientific and mathematical disciplines.

Contrary to popular belief, coding is not just a method for teaching CT; it embodies a diverse set of skills that transcend the fundamentals of programming, delving into more nuanced cognitive processes. This includes breaking down complex problems into manageable components, identifying patterns, and employing logical reasoning to devise efficient solutions (Shute et al., 2017). Thus, CT plays a pivotal role in problem-solving across various domains, offering individuals the tools and mindset needed to navigate an increasingly data-driven and technologically complex world.

CT transcends the traditional boundaries of computer science and represents a versatile problem-solving framework that is relevant in numerous areas (Weintrop et al., 2016). Recent pedagogical trends have shown a shift from memorization to deeper engagement with core CT concepts, with an emphasis on critical thinking (Castle, 2023; del Rey et al., 2021). This shift highlights CT's role in applying algorithmic thinking to solve

real-world problems, making it an indispensable part of today's problem-solving toolkit. Integrating specific CT constructs into students' problem-solving skills is critical to their cognitive development. Research in this area highlights the importance of elements such as loops, events, simulations, conditions, sequences, and variables. For example, Pellas et al. (2018) explored how simulation games could impact computational problem-solving strategies, revealing that such interactive experiences significantly enhance students' problem-solving skills by immersing them in CT practices.

Rottenhofer et al. (2022) further investigated the scope of CT applicability examined how CT skills can facilitate language learning and more complex forms of problem solving. Their results showed that students' progress through advanced computational practices such as abstraction, pattern generalization, iterative looping, error correction, debugging, and iterative refinement. These processes require complicated pattern recognition, decomposition, and algorithmic thinking skills that are fundamental to effective problem solving. In a similar vein, Çakiroğlu et al. (2022) found that successful problem-solving often requires the ability to recognize patterns, generalize these findings, and focus on essential elements while eliminating unnecessary details.

The value of a systematic approach to tackling complex challenges cannot be overstated. Findings from Cheng et al. (2023) and Ezeamuzie et al. (2022) highlight how systematic thinking helps manage complex systems and processes. Relkin et al. (2020) and Saxena et al. (2020) emphasize the crucial role of logical, step-by-step thinking in dealing with complicated scenarios. In addition, detailed discussions on algorithmic thinking by Jody Clarke et al. (2021) and Relkin (2021) illustrate its importance in developing structured problem-solving strategies, which are crucial in various scientific and technical fields.

Palts and Pedaste (2020) propose a new model for developing CT skills in three stages: defining the problem, solving the problem, and analysing the solution. This model identifies ten specific CT skills and offers a structured approach to improve CT education. Their cyclic approach to CT skill development encourages continuous refinement: after analysing a solution, the process starts over for further improvement. This iterative process emphasizes the importance of continuous growth and practical application of CT skills in solving problems. CT is not only about mastering coding or computer systems; it involves a broad spectrum of analytical, systematic, and creative skills that equip individuals to tackle complex problems in various fields. This multidisciplinary approach ensures that students are not only proficient in technical skills but also develop a profound understanding of how to apply these concepts in practical, real-world situations.

2.5 Gender Different in Computational Thinking

Research into gender differences in programming and CT education has revealed nuanced findings that highlight different engagement strategies between genders. A study conducted by Ardito et al. (2020) discovered different approaches to collaborative CT development among sixth graders: boys often gravitated toward hands-on building and programming tasks, while girls were more committed to improving group dynamics. This finding suggests that differentiated teaching strategies are needed that address these different interests and strengths and potentially maximize the engagement and effectiveness of CT education for both genders.

A study by Bati (2022) refuted the popular belief that gender has a direct influence on programming and CT skills. It turned out that given the same opportunities and resources, both girls and boys achieve similar results in these areas. Additionally, Bati's research found that UA that do not rely on computers have a slight advantage in promoting CT skills because

they are tangible, interactive, and appeal to a broader range of learning styles. In a separate study, Malik and Coldwell-Neilson (2018) presented a new teaching and learning approach based on the Approach, Deployment, Result, Improvement (ADRI) model for an introductory programming course. This approach aimed to improve student learning outcomes and engagement. Their analysis of gender differences revealed that female students performed particularly well in the high-achieving category, while male students performed better in the medium and low-achieving categories. They conducted a survey to gather student feedback on this new teaching approach and found that male students were more satisfied than female students. This finding contributes to understanding students' perceptions of different teaching methods.

Yildiz Durak et al. (2023) further examined the affective dimensions of CT education and emphasized that gender can significantly influence students' attitudes and interests toward programming. The finding also suggests that while academic skills may be similar, motivating factors vary, potentially impacting long-term engagement and career choice in technology-related fields. Ezeamuzie (2023) pointed out that acquiring programming skills does not necessarily correlate with improved CT skills, emphasizing that effective CT training involves conscious integration of programming into broader problem-solving and reasoning exercises without significant gender differences requires the transferability of skills.

Addressing broader impacts, studies by Jun Liu et al. (2023) found that factors such as family background and socioeconomic status significantly influence programming confidence more than gender alone. This underscores the importance of creating equitable learning environments that provide all students with the resources and support they need to succeed. At the primary education level, research Herrero-Álvarez et al. (2023) supported

the notion that CT skill acquisition is generally independent of gender, suggesting that early educational interventions can serve as equalizers in tech education.

Complementing this, Kazimoglu (2020), along with Garneli and Chorianopoulos (2018), showed that game-based learning and video-game construction effectively enhance CT skills across genders, advocating for educational strategies that leverage these tools. Zdawczyk and Varma (2023) emphasized the appeal of BBP languages like Scratch, which could play a critical role in attracting more females to the field of computer science. This suggests that engaging, visually intuitive programming environments can help diminish traditional gender disparities in tech.

Collectively, these findings suggest that while gender differences in attitudes and interests towards programming exist, the capacity for both genders to develop CT and programming skills is similar given equitable educational conditions. This underscores the need for educators to continuously refine their approaches, ensuring that they cater to the diverse needs of their students and leverage the unique strengths and interests of each gender.

2.6 Computational Thinking Landscape

2.6.1 Computational Thinking in K12

Since the early 2000s, the integration of CT into educational environments has undergone a remarkable transformation. The journey began with the innovative efforts of Bers et al. (2014) who introduced young learners to programming and robotics through the Tangible Robotics Program. Their pioneering research indicated that even kindergarten-aged children could engage effectively with complex technological concepts if these concepts were presented in a manner suited to their developmental stages, significantly boosting their learning outcomes.

A groundbreaking development was introduced by Gresse von Wangenheim et al. (2021) who showed that young children could comprehend advanced concepts like machine learning when introduced in non-formal, collaborative learning environments. This challenges traditional views on what is age-appropriate and advocates for the early integration of sophisticated technological education, expanding the boundaries of what can be taught in early schooling. Building on this foundational work, subsequent studies, such as those by Yang et al. (2022) have emphasized the benefits of embedding basic technological knowledge from an early age. These researchers supported the early adoption of CT concepts and drew attention to the importance of developing teaching methods that consider diverse factors. Their work also highlighted the need for teaching approaches that consider gender differences and the dynamics of collaborative learning, aiming to create educational experiences that resonate with the diverse needs of young learners.

Bati (2022) also emphasizes the importance of age in how effectively children acquire CT skills, noting that younger learners may require different teaching approaches compared to their older peers. The study offers valuable insights into how both plugged-in (using computers and technology) and unplugged (without technology) activities can improve children's CT skills. Notably, there is a slight preference for UA, as they provide concrete learning experiences. Despite educators increasingly recognizing the significance of CT, a considerable gap remains in effective implementation UA strategies. This gap presents a promising opportunity for further research to identify and promote best practices in CT education.

Meanwhile, several studies have underscored CT's role in fostering early education in problem-solving skills. For instance, Avcı and Deniz (2022) showed that web-based parent education programs and smartphone apps can significantly enhance children's CT

skills, revealing the potential of technology-based interventions in early education. Likewise, initiatives using educational ground robots have demonstrated that structured activities designed to improve spatial organization and self-assessment can help children as young as five develop CT skills (Marta Terroba et al., 2022). Comparative research has shown that programming robots can greatly improve sequencing abilities and CT, especially among children with lower self-regulation skills (Yang et al., 2022).

In recent years, the integration of CT into primary and secondary education has proven transformative, bringing significant benefits to a range of disciplines. The journey began with research showing that incorporating CT into math and science curricula improves students' problem-solving skills and their understanding of fundamental concepts (Atun and Usta, 2019; Weintrop et al., 2016). Cheng et al. (2023) highlighted these implications for STEM education and demonstrated how incorporating CT leads to notable improvements in student learning outcomes with significant overall effect sizes.

Ye et al. (2023) looked in more detail at the interaction between CT and mathematics learning and demonstrated a strong synergy. Their research showed that students who use math to create math projects, predict outcomes, and explore new concepts through math activities simultaneously strengthen their CT and math skills. But despite these advances, the differences between computational and mathematical thinking often pose a challenge for students trying to grasp key concepts. This indicates that there is a clear need for educators to develop innovative strategies to bridge these gaps.

Cui et al. (2023) addressed this challenge by designing a series of programming-based learning tasks for middle school students to develop both CT and mathematical thinking. Various CT concepts and practices articulating the designated mathematical problems were involved in the tasks. Their research highlighted the interconnected growth

of CT and mathematical concepts across four mathematical domains. They identified two main types of interactions: using mathematical knowledge to build CT artifacts and generating new mathematical knowledge through CT practices.

Similarly in science learning, Hurt et al. (2023) introduces CT for Science (CT-S) framework, a toolkit that provides a roadmap for seamlessly integrating CT into science education. This framework allows researchers to identify learning experiences that promote CT while providing strategies for measuring student progress. Although this approach is promising, it requires further validation in different educational settings to ensure that it can be effectively transferred to different classrooms. Noh and Lee (2020) explored the role of robotics programming in enhancing CT and creativity among elementary school students. Their research demonstrated that using robots to teach programming significantly improves both skills, showcasing the untapped potential of hands-on technology in education. However, challenges remain in ensuring equitable access to such innovative tools, as schools with fewer resources might struggle to implement robotics programs effectively.

Similarly, Marín-Marín et al. (2024) examined the use of Arduino and Scratch in K-12 CT education, further highlighting the potential and challenges of integrating technology into the classroom. While Arduino emerged as a primary technology for CT education, and Scratch as the most widely adopted software for learning CT and programming languages, their study revealed mixed results. Some projects proved difficult for students, but overall, these tools led to increased motivation and improved soft skills like problem-solving. This underscores the critical need for creating accessible content that effectively engages students without overwhelming them and highlights the ongoing challenge of ensuring equitable access to educational technologies.

To further enrich the classroom experience, Schwartz et al. (2024) introduced the *Kniwwelino* Classroom Kit (KCK), a creative environment designed to introduce children to programming and electronics. Teachers praised KCK for its novelty, stimulation, and attractiveness, while suggesting improvements to make it easier, more predictable, and faster. Despite these recommendations, the overall feedback was overwhelmingly positive, with students comparing KCK favourably to tools like Scratch. In a similar vein, Costa Junior et al. (2024) investigated the usability of the ThinkCarpet, an interactive robotics tool built with alternative materials and Arduino, finding that it enhanced middle school students' understanding of algorithms. However, as with other educational technologies, a thorough analysis of its long-term impact is necessary to fully understand its potential.

Amidst this dynamic landscape, educators are finding innovative ways to integrate CT into elementary and secondary school curricula. However, without addressing gaps in access, teaching strategies, and long-term assessment, these efforts may not fully realize their transformative potential across different educational contexts. There remains a pressing need for further research that critically evaluates these tools and frameworks to ensure every student can benefit from a strong foundation in CT, regardless of their background or resources.

2.6.2 Integrated Computational Thinking Learning in Other Subjects

The increasing importance of CT has fundamentally changed education and led educators to rethink their curricula. Weintrop et al. (2016) introduced a comprehensive taxonomy that divides CT into four key areas: data practices, modelling, problem solving, and systems thinking. This framework provides a structured approach to embedding CT concepts into teaching, particularly in mathematics and science lessons. However,

challenges such as the lack of clear assessment standards and inconsistencies between current curricula remain, highlighting the need for a more strategic and coherent implementation plan.

Educational systems worldwide are integrating CT with particular attention to social cognitive factors such as gender, prior knowledge and motivation. Huang and Parker (2023), Lai et al. (2023), and Turchi et al. (2019) emphasize the need to adapt CT education to different learning profiles and enable personalized teaching. While this approach is promising, it requires extensive teacher training, curriculum adjustments, and policy-level support to be effective. Tailored CT education requires a delicate balance between personalized learning, maintaining academic rigor, and ensuring equitable access to resources. When these factors come together, students can maximize their learning outcomes and engagement with CT.

Integrating CT into core STEM subjects is critical to developing students' problem-solving skills and computer literacy. Cheng et al. (2023), Kite and Park (2024), and Yadav and Chakraborty (2023) emphasize the transformative impact of CT on STEM curricula, where it empowers students to analyse and solve real-world problems. These researchers also identify significant challenges, including limited teacher familiarity with CT concepts, varying regional adoption rates, and the need for interdisciplinary collaboration among educators. While CT improves analytical skills, creating a supportive learning environment that encourages experimentation, iteration and creativity is critical.

Interdisciplinary relevance remains an essential aspect of CT education. Barcelos et al. (2018), Hickmott et al. (2018), Ogegbo and Ramnarain (2022), Subramaniam et al. (2022) and Wang et al. (2021) show how CT strengthens students' understanding of scientific and technological concepts. By connecting disciplines, students can understand the context of

current challenges and develop holistic solutions. In physics class, Ntourou et al. (2021) examined how Arduino and BBP affect self-efficacy, motivation, and CT. Although their study found improved understanding of electricity and CT, inconclusive results on motivation suggest that educators need to refine their engagement strategies for this interdisciplinary approach to be more effective.

In parallel, Christensen and Lombardi (2020) present a novel approach by integrating CT into the teaching of biological evolution. They address misconceptions, emphasize scientific principles, and engage students through modelling. Their framework identifies a progression from simple too complex to facilitate teaching CT and evolution while specifying the teaching context, computational artifact, and process. However, adaptable teaching materials and sufficient teaching time are essential for effective implementation.

The synergy between CT and mathematics has garnered significant scholarly attention. Pellas et al. (2018) and Xie et al. (2023) emphasize that combining CT and mathematics enriches mathematical learning and fosters a dynamic, inquiry-driven approach to problem-solving. Despite acknowledging potential challenges like limited teacher computational literacy and rigid traditional teaching models, their work underscores the importance of promoting seamless interdisciplinary integration. Innovative approaches, such as those by Budinski et al. (2019) and Rodríguez-Martínez et al. (2020) combine programming with other activities to teach mathematical and coding concepts. Budinski et al. (2019) used creative origami to create fractals, while BBP platforms used. Both methods simultaneously promoted CT knowledge and mathematical understanding, highlighting the potential of creativity to improve interdisciplinary learning.

In English learning, Kabak and Korucu (2021) examined how integrating CT into digital game development affected students' academic performance and attitudes. Their

research suggested that incorporating CT concepts positively influenced student engagement and learning outcomes. Parsazadeh et al. (2021) examined language teaching through CT-based digital storytelling and concluded that this approach improves language knowledge and problem-solving skills while improving motivation and performance in second language acquisition. To effectively support the development of digital games and storytelling frameworks, adequate infrastructure and balanced screen time is critical to avoid potential negative impacts on student well-being.

Similarly, Peng et al. (2023) examined the effects of basic CT concepts on reducing writing anxiety and increasing motivation in English writing. Their results showed that CT principles significantly reduced writing anxiety and increased motivation compared to traditional learning methods. By promoting structured English writing practices, CT encouraged students to be more organized and motivated while reducing their writing anxiety. Effective teacher guidance is essential as educators must thoroughly understand and implement CT concepts to guide students through these novel learning strategies.

In music education, Bell and Bell (2018) combined CT with music theory using tools such as CS Unplugged and BBP languages. This integration emphasized teaching multiple subjects simultaneously to improve efficiency. However, without strategic planning and interdisciplinary coordination, these approaches risk overwhelming educators. Along similar lines, Petrie (2022) examined the integration of CT into technology through algorithmic music composition with Sonic Pi. The study highlights the challenges of code blocks, sound patterns and experiments and highlights practical aspects of interdisciplinary CT learning.

The cross-curricular integration of CT has changed the way students' approach and solve complex problems. Building interdisciplinary connections promotes a comprehensive understanding of CT concepts and enables students to creatively use these skills to address

the challenges of the digital age. However, educators and policymakers must address barriers such as teacher training, resource allocation, and curriculum flexibility to ensure widespread adoption. Instructional design for CT implementation should be carefully crafted to meet pedagogical expectations and goals and to ensure that students receive the maximum benefit from a CT-enriched curriculum.

2.6.3 Pedagogical Strategies for Teaching Computational Thinking

As the educational landscape continues to evolve, a variety of pedagogical strategies have emerged to effectively integrate CT into learning environments. Approaches such as design-based, model-based, project-based, game-based, inquiry-based, and problem-based learning aim to deepen CT skills by improving both conceptual understanding and practical application. However, despite their notable successes, implementing these strategies is challenging, particularly when it comes to providing adequate teacher training and equitable access to essential technological resources.

Hsu et al. (2018) in their review highlighted the importance of project-based, problem-based, and game-based learning as dominant strategies for CT teaching. These approaches are characterized by their ability to provide practical, engaging, and interdisciplinary learning experiences that resonate with students. Tikva and Tambouris (2021) also identified game-based, modelling and simulation-based, problem-solving, project-based, scaffolding, and collaborative learning as common techniques in CT education.

Current research highlights the growing interest in simulation-based education, particularly in higher education. This approach enhances learning through interactive and immersive environments and brings benefits such as improved critical thinking, content retention and decision-making skills by immersing learners in realistic, real-world scenarios.

For example, Campos et al. (2020) highlight these advantages. In parallel, scaffolding remains a cornerstone in educational strategies. Studies by Turner et al. (2018) affirm its effectiveness in providing temporary support that helps students bridge the gap between independent skills and guided performance, thereby promoting deeper understanding and skill development. Additionally, collaborative learning remains an important method for promoting teamwork and problem-solving skills. Recent findings from Costley (2021) show that group work improves the ability to complete complex tasks and promotes better learning outcomes through shared knowledge and different perspectives.

These diverse methods are further enriched by complementary approaches such as hands-on activities, media creation, storytelling, and guided discovery. Through hands-on activities, students are engaged in practical problem-solving tasks that enable them to apply theoretical CT concepts in real-world contexts and consolidate their understanding through tangible results (Saxena et al., 2020). Media creation encourages creativity and critical thinking as students design projects that visualize their computational solutions.

Storytelling, commonly used in game-based learning, allows students to develop narratives around problem-solving challenges and weave CT principles into engaging storylines that increase emotional engagement and maintain interest (Palioura and Dimoulas, 2022). By crafting stories, students create a memorable framework that helps them retain concepts and apply them more naturally to different contexts. Guided discovery allows educators to balance structure with the freedom for students to explore and learn independently.

Together, these complementary approaches provide a comprehensive and engaging CT learning experience that accommodates diverse learning styles and abilities. By tailoring their strategies to students' individual needs, educators can develop a deeper understanding

of computing principles, fostering a learning environment where creativity, collaboration, and critical thinking are seamlessly integrated into the CT journey. Park and Kwon (2023) developed a program that integrates CT into technological problem-solving processes. The 12-hour program involved modifying a remote-control (RC) car using the Microbit development tool. Their findings showed that this approach significantly improved participants' CT skills while positively influencing their attitudes toward technology in education classes. This research highlights how problem-solving CT programs can greatly enhance student engagement and technological proficiency.

Similarly, Karaahmetoğlu and Korkmaz (2019) examined the effects of project-based Arduino educational robot applications on students' CT skills and their perception of STEM proficiency. Their study found that these applications significantly boosted students' CT skills, especially in problem-solving, while positively impacting their perceptions of STEM. This research illustrates how project-based learning can significantly improve students' CT skills and STEM perception.

The popularity of project-based and problem-based learning is driven by technological advancements and the increasing availability of resources for comprehensive CT courses. In a two-year study, Valls Pou et al. (2022) emphasized the importance of integrating CT and educational robotics into project-based learning. The study involved multiple sessions within a project-based learning framework where students used the BBP platform Scratch. As a result, students improved their CT skills and knowledge while learning to navigate educational robotics platforms.

Zapata-Cáceres et al. (2021) employed problem-solving strategies and a three-dimensional framework to create a game-based learning environment in both individual and collaborative modes. Their exploratory case study found that collaborative play fosters long-

lasting learning and serves as an effective assessment tool for CT skills. This aligns with the findings of Jawawi et al. (2022), who showed that game-based and problem-based learning frameworks effectively support cognitive development in computational contexts. However, they also highlighted that without sufficient scaffolding, such as step-by-step guidance or structured prompts, students can lose focus or struggle to grasp concepts fully, limiting their learning outcomes.

Hutchins et al. (2020) examined model and problem-based learning strategies to improve kinematic understanding in physics while applying CT to scientific investigations. Their research demonstrated the transformative potential of various pedagogical approaches for developing a comprehensive understanding of CT and advocated for agile instructional design to meet diverse student needs. Building on this, Gao and Hew (2019) emphasized the effectiveness of the 5E (engagement, exploration, explanation, elaboration, and evaluation) inquiry-based teaching model. They demonstrated how the 5E model encourages active participation and improves problem-solving skills. The results showed significant improvement in students' CT concepts and computational problem-solving skills.

Andriyani (2023) expanded the value of game-based learning by developing media to improve cognitive and psychomotor skills in arithmetic operations, especially multiplication. By integrating fundamental CT principles with interactive game-based learning, this framework stimulates students' cognitive and psychomotor skills and helps them overcome computational challenges. The study highlighted the importance of balancing screen time with hands-on activities and carefully monitoring student engagement to ensure learners remain focused and achieve desired learning outcomes.

Li et al. (2023) made a further contribution by introducing a design-based learning approach to improve CT in elementary school students using *LabPlus* electronic kits and

Scratch programming software. By allowing students to create digital projects with real-world applications, this approach encouraged hands-on experimentation and creativity. Consequently, students in the treatment group showed significantly higher CT skill gains than those in the control group.

Collaborative design activities also have gained prominence as powerful tools that not only facilitate the learning of CT concepts but also sharpen teamwork and problem-solving skills. These strategies leverage group dynamics and collaborative tasks to create vibrant learning environments where CT principles are thoroughly explored. As Arık and Topçu (2022) found, such an approach significantly enhances student engagement while deepening their understanding of CT concepts. The dual benefits of collaborative activities are evident: students learn complex computational ideas while also developing essential collaboration skills, preparing them for the challenges of a technology-driven world.

The importance of collaboration is particularly evident in UA. Caeli and Yadav (2020) highlight how collaboration enriches students' understanding and application of computational concepts. Hsu and Liang (2021) further emphasize that fostering collaboration in CT education helps reduce misunderstandings, negative attitudes, and communication issues, creating focused research avenues on the impact of collaborative methods in K-12 education. Despite these successes, educators still face the challenge of achieving consistent and effective CT implementation. Solid teacher training, sufficient resources and well-structured curriculum guidelines are essential for the successful application of these methods. By using diverse learning approaches, educators can better prepare students to tackle complex problems and develop the skills needed for the digital age.

2.6.4 Computational Thinking Teaching Tool

A CT teaching tool is a resource or method designed to help learners develop CT skills. A study by Chan et al. (2023) identified various types of CT teaching tool, including digital tangible, apps and games, programming languages, and other technological tool. The CT competencies most assessed using these tools include algorithms and algorithmic thinking, abstraction, testing and debugging, loops, and sequences, all of which emphasize fundamental CT skills. These foundational skills help build a solid programming foundation that can then be built upon for more advanced problem-solving and specialized programming tasks.

Digital tangibles are physical objects or devices that bridge the digital and physical worlds, providing learners with a tactile way to interact with abstract computational concepts and practices (De Jong, 2023). These tools make CT concepts and practices more accessible and engaging, particularly for younger students. In CT education, digital tangibles often complement programming, creating an immersive learning experience that reinforces computational concepts in a fun and practical manner.

Examples of digital tangibles include programmable robots, microcontrollers or DIY kits, and interactive boards or blocks. Programmable robots offer students practical, hands-on experience with coding and CT concepts. A study by Souza et al. (2019) revealed that programmable robots like LEGO Mindstorms, one of the many robotic kits available, achieved excellent results across multiple criteria, such as modularity level, hardware quality, curriculum alignment, and cost-effectiveness. The modular design allows students to configure their robots creatively and adapt them to specific tasks.

A study by Ardito et al. (2020) and Zhou (2023) also found that the LEGO Mindstorms kit enables students to write code through a visual programming interface,

which guides their robots to execute precise tasks. This offers a tangible representation of programming concepts like loops, conditionals, and variables. By using the intuitive drag-and-drop interface, students can easily arrange programming blocks into logical sequences that control the robot's movements. This hands-on application helps students deepen their understanding of programming because they can immediately observe how their code affects the robot's actions, making abstract concepts more concrete and fostering a deeper engagement with CT.

Another study by Seckel et al. (2023) found that the Bee-Bot robot serves as an excellent resource for developing CT skills in younger children. Bee-Bot is a simple, child-friendly robot that encourages students to sequence commands through directional buttons, guiding the robot across grids to reach predetermined goals. This interactive and playful approach allows children to practice CT concepts like sequencing, problem-solving, and debugging, all in a low-stress, gamified environment.

This is consistent with the work of Diago et al. (2022), who found that students who participated in an educational robotics intervention using Bee-Bot had higher CT scores than those who received traditional instruction. Even after controlling for initial skill levels, the intervention group outperformed their peers, suggesting that Bee-Bot is effective in promoting important CT skills. The simplicity of the Bee-Bot interface allows students to focus on logical command flow and algorithmic thinking, building foundational skills while maintaining engagement and interest through playful learning.

Arduino kits are an example of microcontrollers that provide a versatile platform for exploring CT. These kits include open-source hardware and software that allow students to programmatically control various electronic components such as LEDs, sensors, and motors (Yilmaz Ince and Koc, 2021). This hands-on experience gives learners a concrete

understanding of important programming concepts such as loops, conditionals, and functions (Hsiao et al., 2022; Yin et al., 2020).

The flexibility of Arduino supports a wide range of projects, from basic circuits to advanced robotics and Internet of Things (IoT) applications. Because of the extensive online community and wealth of resources, students and educators have access to tutorials and forums to help them troubleshoot problems and expand their knowledge Fidai et al. (2020). However, their high costs can be prohibitive for some schools and limit equal access. Additionally, their reliance on specific hardware ecosystems may result in students acquiring limited, brand-specific skills rather than broader, transferable programming skills.

Interactive boards or blocks, such as *Cubetto* or Osmo Coding, are systems that students use to arrange blocks or physical boards into coding sequences (Madariaga et al., 2023). These systems allow students to visually represent abstract programming concepts, which fosters problem-solving skills in an intuitive, hands-on manner. The tactile nature of these tools makes programming accessible, even to very young learners. Soleimani et al. (2019) presented a research-through-design example of *CyberPLAYce*, a tangible interactive learning kit designed for children to support storytelling and CT. Their study showed that *CyberPLAYce*'s activities promoted engaging storytelling and CT practices.

Kuo and Hsu (2020) examined the board game Robot City, which was used as an educational material to teach seventh-grade students CT concepts such as sequential, conditional, and repetitive structures and function calls. The study examined the effects on students' learning outcomes and behaviour patterns while playing. Results showed that Robot City improved students' CT skills by providing structured tasks that promoted teamwork, focused problem solving, and iterative collaboration. However, the board game's

limited scenarios might not fully encompass real-world programming challenges, potentially limiting the applicability of the acquired skills.

The types of programming languages commonly associated with CT include text-based programming and visual BBP. Text-based programming such as Python enable students to write explicit code, fostering an understanding of syntax, logic, and computational concepts (Ezeamuzie et al., 2022; Zitouniatis et al., 2023). Visual BBP like Scratch provide drag-and-drop interfaces, minimizing syntax errors and emphasizing logical flow, making them accessible for beginners but its simplified syntax might leave students unprepared for transitioning to text-based coding environments (Armoni et al., 2015). Python, while more representative of professional programming, can be intimidating for beginners, especially without adequate instructional support (Rastovac et al., 2021).

These languages are frequently utilized in educational settings to enhance CT skills among students, particularly at secondary and higher education levels, where computer science, mathematics, and engineering are major focuses Fagerlund et al. (2021). In secondary education, students use programming languages to design algorithms, understand data structures, and implement software solutions aligned with STEM curricula. Higher education courses incorporate programming languages to tackle complex problems in fields like artificial intelligence, software engineering, and data analysis (Tran et al., 2019).

The project-based learning environment in Scratch BBP encourages creativity by allowing users to create interactive games, animations, and stories. This approach makes programming fun and directly engages students, fostering a strong interest in computer science (Moreno-LeÓN et al., 2020; Valls Pou et al., 2022). Scratch also has a built-in online community where learners can share projects, collaborate, and explore others' work. This creates a social aspect that supports learning among like-minded people and inspires new

ideas (Moreno-Leon and Robles, 2016). Although simple, Scratch BBP introduces basic programming concepts such as loops, conditionals, variables, and event-driven programming, providing learners with a solid conceptual foundation that will benefit them in other programming languages (Fields et al., 2017).

Additionally, Scratch BBP is supported by extensive resources including tutorials, guides, and a supportive community to support learners and educators (Da Silva Pereira and Lopes, 2020). Educators can integrate it with other subjects such as mathematics and language arts, increasing its educational value. Scratch BBP provides an engaging and supportive environment that encourages CT and creativity, making it an excellent choice for introducing programming concepts to beginners.

The success of CT teaching tools relies on student-centered approaches that emphasize creativity, engagement, and problem-solving. By embedding these tools within a comprehensive curriculum that supports diverse learners, educators can establish a strong foundation for students to excel in an increasingly complex digital landscape. Moreover, ensuring equitable access to these tools across various school environments and settings is crucial for empowering all students with the essential CT skills needed to succeed.

2.7 The Relationship Between Computational Thinking, Unplugged Activities, and Block-Based Programming

2.7.1 Unplugged Activities in Computational Thinking Education

UA have gained widespread recognition as a novel and effective pedagogical tool for introducing CT concepts through hands-on, non-digital methods. These activities employ everyday materials and real-world scenarios to translate abstract computational ideas into tangible learning experiences, thereby enhancing comprehension and student engagement without the reliance on digital devices (Curzon, 2013). By incorporating interactive

exercises, UA facilitates students' visualization and internalization of complex concepts, creating a more concrete and accessible learning environment that caters to diverse learners.

The pedagogical value of UA is further substantiated by the work of Kafai and Proctor (2022), who advocate for the development of collaborative learning spaces that integrate both digital and traditional methodologies to foster an innovative educational setting. Empirical studies have consistently validated UA's effectiveness in building foundational CT skills, such as pattern recognition, decomposition, and abstraction (Bell and Vahrenhold, 2018). These activities encourage active learning and collaboration, enabling students to directly engage with CT concepts, thereby enhancing their problem-solving abilities and deepening their conceptual understanding.

A notable advantage of UA is its potential to democratize access to CT education. Yadav et al. (2016) emphasize that UA enables students, particularly those with limited access to technology, to engage with core computational concepts without the need for computers. This approach levels the educational playing field, especially in under-resourced environments, by leveraging familiar classroom resources like cards, paper, or everyday items to make lessons both relatable and engaging. By providing a practical and accessible method for teaching CT, UA not only stimulates interest but also allows students to grasp complex computational concepts effectively.

The flexibility of UA is evident in its broad range of activities, which cater to various learning styles. For instance, algorithmic thinking exercises guide students through solving logical puzzles and sequencing tasks, helping them recognize patterns and develop structured solutions (Peel et al., 2022). Narrative-driven science investigations integrate CT concepts within storytelling, encouraging students to solve problems through engaging scenarios (Kite and Park, 2023). Additionally, practical exercises like mind mapping help

students comprehend programming logic by breaking down real-world tasks into systematic steps and rules (Metin, 2022).

Educators acknowledge the importance of creating meaningful and inclusive learning experiences through UA. Tripon (2022) highlights that these methods are essential for making CT education accessible to diverse learners, including those with limited access to technology or formal programming instruction. By offering hands-on activities that connect CT concepts to real-world situations, UA bridges educational gaps and ensures that all students can participate meaningfully in CT learning. Furthermore, Lindstrom et al. (2019) argue that UA activities are resource-efficient, enabling schools with limited budgets to provide quality CT education without significant technological investments. For example, simple materials like cards, blocks, or paper can effectively convey CT concepts such as pattern recognition and sorting algorithms, making UA a viable option for a wide range of educational settings.

However, despite these advantages, some scholars caution against the potential for oversimplification in UA, which could lead to misconceptions about complex computational concepts (Bell and Vahrenhold, 2018). To mitigate this risk, it is crucial to implement UA activities with appropriate scaffolding and progression, ensuring that foundational concepts are accurately represented before advancing to more complex tasks. Merino-Armero et al. (2022) illustrates how UA's unique educational pathways promote CT skills through practical problem-solving, collaborative exercises, and creative thinking, distinct from traditional academic metrics. These studies affirm that UA can enhance computational competencies while offering an inclusive, engaging, and practical approach to CT learning.

In addition to traditional UA methods, non-digital tools, such as Smart Games, have demonstrated how UA can complement computer-based programming instruction

effectively (Fanchamps et al., 2024). These tools use puzzles and logical challenges to simulate computational problems, helping students understand core programming concepts without relying on digital devices. Bell and Vahrenhold (2018) advocate for the integration of UA with programming instruction, emphasizing that a balanced combination of non-digital activities and computer-based programming can sustain student motivation while fostering critical thinking and problem-solving skills.

Sun et al. (2021b) show that UA can enhance students' CT skills regardless of their mathematical abilities or prior programming experience. These activities break down complex CT concepts into simple, approachable exercises that nurture problem-solving skills. Creative tools like mind mapping further bolster programming skills by encouraging students to visualize problems and explore solutions through hands-on approaches (Sun et al., 2022). Chun et al. (2021) highlights the importance of integrating UA with Scratch BBP, noting that hands-on activities can deepen students' engagement with abstract programming principles while promoting collaborative exercises.

Overall, the literature affirms the growing importance of UA in CT education. These tools are versatile and comprehensive, significantly enhancing students' CT skills by making abstract concepts concrete and accessible. However, combining UA with programming lessons remains crucial for providing a comprehensive learning experience. As educators increasingly recognize the value of UA, they should continue to develop and implement these methods thoughtfully to ensure students receive a balanced education that promotes collaboration, creativity, problem-solving, and CT skills.

2.7.2 The Role of Block-Based Programming (BBP) in Developing Computational Thinking Skills

BBP environments, such as Scratch, Blockly, and Alice, have revolutionized introductory programming education by providing an accessible platform for exploring programming concepts visually and intuitively. By representing code as interlocking graphical blocks, these tools eliminate the cognitive barriers associated with text-based coding, allowing learners to focus on the logical structuring of their programs without being hindered by syntactic complexities (Bau et al., 2017). This approach mitigates the intimidation often associated with learning coding syntax, emphasizing instead the logical and problem-solving aspects of programming (Çakiroğlu et al., 2022).

BBP tools simplify abstract programming concepts into manipulable elements, making the learning process more approachable and engaging for beginners. They foster a deeper interest in computer science by demystifying programming intricacies, enabling students, particularly those with limited programming experience, to grasp foundational programming concepts and develop essential CT skills (Weintrop et al., 2016). Activities that involve abstraction, pattern recognition, and problem-solving engage students in an intuitive and effective manner, facilitating the development of robust CT competencies.

Educational research suggests that the impact of BBP tools is maximized through carefully designed activities that promote critical thinking and problem-solving. For instance, Yildiz Durak (2020) found that students engaging in game design and mathematical simulations within BBP environments significantly enhanced their algorithmic thinking. These activities provide practical contexts for applying logical reasoning, thereby bolstering analytical capabilities in an engaging way. Additionally, structured activities positively

influence students' attitudes towards technology and learning, fostering a proactive approach to technological education (Erümit, 2020).

Calder (2018) discovered that creating digital learning objects within BBP environments promoted mathematical thinking and enhanced students' understanding of both mathematical and CT concepts. Students worked collaboratively on programming tasks to build digital artifacts, applying logical reasoning and mathematical knowledge. Similarly, Estevez et al. (2019) integrated CT with BBP to introduce artificial intelligence (AI) concepts. Through project-based activities using Scratch, students engaged with coding puzzles and challenges, gaining a deeper understanding of AI's practical applications, ethical considerations, and its potential for solving real-world problems.

Beyond individual learning, BBP tools are integral to broader educational frameworks. Incorporating them within the Technological Pedagogical Content Knowledge (TPACK) framework enables educators to integrate CT across various subjects, demonstrating the relevance of computational skills beyond computer science (Atun and Usta, 2019). BBP tools enhance student engagement through exploratory learning, providing immediate feedback that fosters experimentation and iterative learning. Studies by Coşkunserçe (2023) and Geng (2023) underscore how BBP tools increase students' programming confidence and inspire them to tackle real-world problems through project-based learning.

Despite their benefits, transitioning from BBP environments to advanced, text-based programming can be challenging. Brennan and Resnick (2012) observed that students working with BBP environments like Scratch often struggled to explain their code, frequently copying programming constructs without fully understanding them. This highlights a potential pitfall of BBP tools, where the visual simplicity may lead to a

superficial understanding, prompting students to rely on copying and pasting rather than developing their problem-solving skills.

To mitigate this issue, Fagerlund et al. (2021) emphasize the importance of creating educational pathways that begin with simplified environments but gradually introduce more complex programming challenges. They advocate for a scaffolded approach that starts with BBP tools to familiarize students with basic programming concepts such as loops and conditionals before moving on to text-based programming. This gradual progression ensures that students can translate their understanding from BBP to text-based coding, gaining proficiency in syntax and debugging while enhancing their logical reasoning skills.

While BBP tools are invaluable for introducing computational concepts, their full potential can only be realized through meticulous curriculum design. For example, blending Scratch with Arduino kits offers practical, real-world applications that enhance students' understanding of both programming and mechanical functionalities (Fidai et al., 2020; Ntourou et al., 2021). Introducing robotics into BBP helps students understand the relationship between software and hardware, deepening their comprehension of how coding commands translate into mechanical actions (Çalışkan, 2020).

The integration of UA with BBP offers a comprehensive framework that simplifies programming concepts while boosting motivation and engagement. UA provides a flexible, low-cost alternative that allows educators to teach core computational concepts using accessible materials (Caeli and Yadav, 2020). Combining UA and BBP not only enriches the learning experience but also aligns with real-world scenarios, ensuring that CT education remains relevant and practical (Kirçali and Özdener, 2022).

Nevertheless, seamlessly integrating UA and BBP into the curriculum requires educators to craft activities that captivate students while developing advanced CT skills. Transitioning from block-based to text-based environments should be smooth, preparing students for more sophisticated challenges in technology-driven careers (Marwan and Price, 2023). This strategy demands continuous adaptation and evaluation to ensure that CT education aligns with evolving standards and technological advancements.

2.7.3 Rationale for the Current Research Focus

The evolution of CT education research reflects a growing recognition of the need for pedagogical approaches that effectively transcend traditional subject boundaries. Early efforts primarily focused on text-based programming languages such as Python and Java, which, although powerful, often proved too challenging for beginners, particularly young students (Li et al., 2023). This led to the exploration of alternative methods that could make CT concepts more accessible and engaging for novice learners.

The development of UA was a response to these challenges, offering a non-digital approach that effectively introduces CT concepts through hands-on, interactive exercises (Relkin et al., 2021). These activities have been particularly valuable in resource-constrained educational settings, making CT education more inclusive. However, as students advanced, the limitations of UA became apparent, necessitating a more interactive and engaging approach to deepen their understanding of complex problem-solving tasks (Peel et al., 2022).

Concurrently, the advent of BBP tools like Scratch provided an intuitive platform for teaching programming. BBP removed the syntactic barriers of text-based coding, enabling learners to focus on the logic and structure of programming. While BBP made programming more approachable, it did not fully address the need for a structured, scaffolded learning progression (Grover et al., 2017).

The integration of UA and BBP emerged as a promising solution, leveraging the strengths of both approaches to create a comprehensive learning pathway. By starting with concrete, hands-on activities and gradually introducing more abstract, digital concepts, this integrated approach supports a smoother transition for students, enhancing their confidence and competence in CT skills (Habib et al., 2021). This approach aligns with constructivist and experiential learning theories, which emphasize the importance of building knowledge through active engagement and real-world experiences.

This research focuses on exploring the combined use of UA and BBP to address the identified gaps in CT education. By leveraging the strengths of both methods, the study aims to develop a comprehensive framework that supports a progressive learning experience from tangible, hands-on activities to complex, digital problem-solving tasks. This integrated approach is expected to enhance students' engagement, confidence, and proficiency in CT, ultimately preparing them for more advanced computational challenges and fostering a deeper understanding of the principles underlying computational thinking.

2.8 Literature Gap

Despite considerable progress in understanding CT and the effectiveness of various pedagogical approaches, several gaps remain in the existing literature. These gaps underscore the need for a more detailed examination of key variables, such as gender and teaching methods, that may influence CT skills and attitudes towards programming. This study aims to address these gaps by exploring the interactions between these variables in a context-specific setting.

2.8.1 Gender Differences in Computational Thinking

Gender differences in CT skills have been the subject of extensive research, yet findings remain inconclusive. A meta-analysis by Hu (2024) synthesized results from 30 studies, covering 51 independent effect sizes and a sample of 9,181 individuals (4,254 males and 4,927 females) across various countries. The analysis revealed statistically significant, though modest, gender differences in CT, with males generally outperforming females. Importantly, these differences tend to become more pronounced during high school, suggesting that educational interventions may need to be tailored differently for boys and girls as they advance in their academic careers.

Regional variations further complicate the picture of gender differences in CT. Hu (2024) found that in the "East Asia and Pacific" region, gender disparities in CT were negligible, whereas in "North America" and "Europe," males tended to outperform females. In contrast, findings from "Europe and Central Asia" were inconsistent, with some studies showing females outperforming males. These regional differences highlight the influence of cultural, societal, and educational factors on gender disparities in CT skills, indicating a need for more context-specific investigations. Understanding how these factors interact with teaching methods can inform the development of more inclusive educational strategies that address the needs of diverse student populations.

2.8.2 Context-Specific Studies

Most research on CT skills and programming attitudes has been conducted in diverse educational contexts, which makes generalization difficult. Most studies focus on mixed or undefined educational settings, overlooking the specific needs of student groups (Sun and Zhou, 2023; Tang et al., 2020). This study targets secondary school students in a fully boarding school in Kedah, offering insights specific to this educational context. Boarding

schools provide a unique learning environment characterized by extended learning hours, structured routines, and a close-knit community, all of which may significantly impact students' CT skill development and attitudes towards programming. By focusing on this demographic, the study seeks to generate findings that can inform the design and implementation of CT and programming instruction tailored to the boarding school environment, which may differ from more general educational settings.

2.8.3 Hypothesis Development

Building on the literature gaps identified, the following hypotheses have been formulated to investigate the impact of teaching methods and gender on CT skills and attitudes towards programming. These hypotheses are aligned with the study's objective to explore whether integrating UA with BBP offers added benefits over using BBP alone, and whether gender plays a significant role in these educational outcomes.

2.8.3.1 H₀₁: There is no significant difference in the improvement of CT skills between students in the Unplugged Activity + Block-Based Programming group and those in the Block-Based Programming-only group.

This hypothesis addresses the core research question regarding the effectiveness of combining UA with BBP compared to using BBP alone in enhancing CT skills. Previous studies have shown that while BBP is effective in developing CT skills, the integration of UA can provide a more comprehensive learning experience by introducing concepts through hands-on, non-digital methods before transitioning to digital environments (Amnouychokanant et al., 2021; Yeni et al., 2024). However, it remains to be empirically tested whether this combined approach leads to a statistically significant improvement in CT skills compared to BBP alone.

- 2.8.3.2 H₀₂: Gender does not influence the improvement in CT skills in students from either the Unplugged Activity + Block-Based Programming group or the Block-Based Programming-only group.

This hypothesis examines whether gender plays a role in the improvement of CT skills when different teaching methods are applied. While some research indicates that male and female students perform equally well in CT tasks when given the same opportunities (Yang and Bers, 2023), other studies suggest that the choice of teaching method can differentially affect gender-based performance in CT (Sun and Liu, 2024). This hypothesis aims to explore whether gender remains a neutral factor in CT skill improvement regardless of the teaching method used.

- 2.8.3.3 H₀₃: There is no significant difference in the attitudes toward programming between students in the Unplugged Activity + Block-Based Programming group and those in the Block-Based Programming-only group.

This hypothesis addresses the question of whether integrating UA with BBP influences students' attitudes toward programming more significantly than using BBP alone. Research has shown that hands-on, interactive teaching methods, such as those involving robotics or physical computing devices, can positively shape students' attitudes towards programming by making learning more engaging and relatable (Paucar-Curasma et al., 2023). However, it is necessary to test whether this effect is significantly enhanced by adding UA to BBP, or if BBP alone suffices to foster positive attitudes towards programming.

- 2.8.3.4 H₀₃: There are no significant gender differences in attitudes toward programming between the Unplugged Activity + Block-Based Programming group and the Block-Based Programming-only group.

This hypothesis investigates whether gender influences students' attitudes towards programming when different teaching methods are used. Some studies have reported that females may respond more positively to collaborative and interactive learning environments,

potentially reducing gender disparities in programming attitudes (Merayo and Ayuso, 2023). Conversely, other research suggests that effective instructional strategies can mitigate these gender differences, regardless of the teaching method employed. This hypothesis seeks to determine whether integrating UA with BBP can create an educational environment that equally benefits male and female students in terms of their attitudes towards programming.

2.9 Theoretical Framework

The theoretical framework of this study integrates key learning theories with specific teaching methodologies to explore their impact on educational outcomes, specifically using UA and BBP. Constructivism, as articulated by Lev Vygotsky, posits that learning is profoundly social, occurring through interaction within a cultural context. This perspective challenges traditional teacher-centred educational models by emphasizing that knowledge is co-constructed through social engagement and the use of cultural tools (Vygotsky and Cole, 1978).

Building on these ideas, Piaget (1976) cognitive development theory asserts that children actively construct their understanding through direct interaction with the environment. Extending this concept, Seymour Papert's constructionism advocates for experiential learning through creation. Influenced by Piaget, Papert's approach focuses on the educational value of students making tangible objects. He argues that this type of active participation not only makes learning personal but also deeply ingrains knowledge as students design, build, and problem-solve. This is especially pertinent in technology and computer-based education, where students benefit significantly from engaging in hands-on, creative projects that enhance their understanding and retention of computational concepts (Papert and Harel, 1991).

Together, these theories underscore a shift towards more dynamic and interactive learning environments where students not only absorb information but also apply it creatively to solve real-world problems. This framework supports the integration of UA and BBP to promote a holistic development of CT skills, aligning educational practices with the cognitive and developmental needs of learners. Figure 2.1 summarises the research theoretical framework in this study.

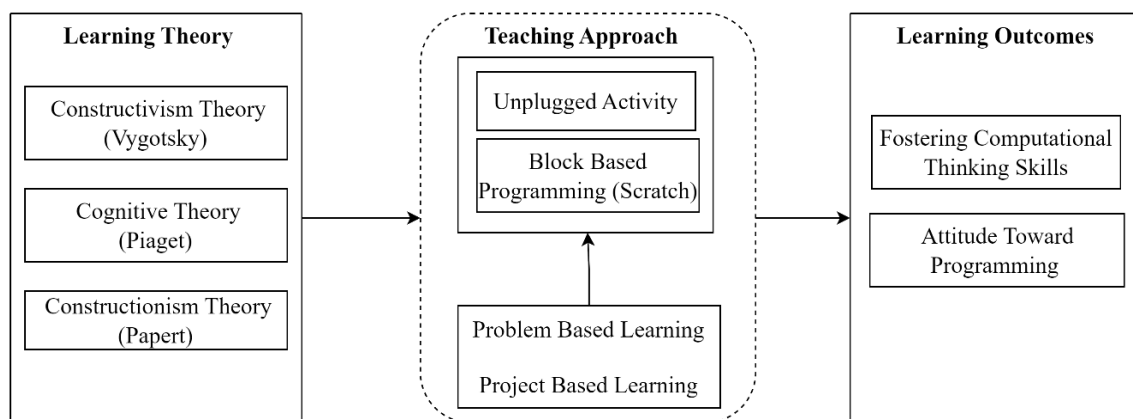


Figure 2.1: Research Theoretical Framework

The framework emphasizes UA and BBP as interactive teaching methods that promote active learning and knowledge construction. These approaches, connected to problem-based and project-based learning, bring to life the principles of UA and BBP in educational settings. The desired outcomes enhanced CT knowledge and more positive attitudes towards programming are the ultimate goals of this intervention, aiming to improve students' skills and positively shift their perceptions towards CT and programming.

In practical terms, students engage in activities like conditional card games, tower building, and stacking cup activities under UA, and use Scratch for BBP to develop simple math racing games and draw geometric shape designs. These activities promote

collaborative learning and enhance problem-solving skills, as supported by Garcia (2021) and Raviv et al. (2019), who note the benefits of cooperative learning for academic success and collaborative problem-solving.

Visual BBP environments encourage students to actively participate in project development, enhancing their understanding and practical skills through real-world applications. This integration of UA fosters an engaging and effective introduction to CT and programming concepts, suitable for beginners and enriching the educational experience through project completion. Project-based learning, especially when using Scratch, allows students to blend theory with practice, fostering active participation and collaborative learning, as noted by Plaza Merino et al. (2019).

This method has been shown to be effective in developing problem-solving skills. The framework comprehensively describes the theoretical foundations of the research and links selected pedagogical theories with teaching methods and expected outcomes. It is critical to laying the groundwork for the study's interventions and to clarifying how and why these interventions might influence students' learning experiences and outcomes.

2.10 Summary

This chapter reveals that BBP tools are widely used in K-12 education to facilitate the teaching of basic programming and CT. This study employs Scratch BBP, adhering to recognized educational practices that are known to boost engagement and learning outcomes in CT. It also addresses significant gaps noted in the literature regarding the integration of digital and UA.

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Overview

This chapter presents the methodology used to evaluate the impact of integrating UA with BBP on the development of CT skills and attitudes toward programming among secondary school students. A quasi-experimental design is utilized, comprising two groups: one receiving a combined UA and BBP intervention, and the other serving as the control group, receiving only BBP. The study adopts a mixed-methods approach, collecting both quantitative and qualitative data to provide a comprehensive understanding of the research problem. This chapter elaborates on the research design, the development and validation of research instruments, revisions made following the pilot test, the selection of respondents, data collection methods, and the data analysis techniques used.

3.2 Research Design

The study employs a quasi-experimental design that enables the comparison of different treatment groups without the need for random assignment, making it practical for educational settings where pre-existing groups are involved. The mixed-methods approach enhances the study by integrating the strengths of both quantitative and qualitative data, allowing for a more nuanced understanding of the effects of the interventions on students' CT skills and programming attitudes.

The research design is organized into two distinct phases. The quantitative phase involves the collection of numerical data through pre-tests and post-tests administered before and after the intervention, respectively, to establish baseline data and measure changes in students' CT skills and attitudes towards programming. Additionally, a programming attitude

survey is conducted to assess shifts in students' perceptions and attitudes towards programming.

The qualitative phase comprises semi-structured interviews with a purposive sample of students from the UA+BBP group. These interviews delve deeper into students' experiences and perceptions of the interventions, providing rich, contextual insights that complement the quantitative findings. The integration of data from both phases allows for a comprehensive analysis, wherein the qualitative data contextualizes and explains patterns observed in the quantitative data.

The integration of data from both phases allows for a comprehensive analysis, wherein the qualitative data contextualizes and explains patterns observed in the quantitative data. This dual approach not only validates the study's findings but also enhances the reliability and depth of understanding of the research problem, offering a well-rounded view of the effectiveness of the interventions on enhancing students' CT skills and attitudes toward programming.

3.3 Variables and Instrument Development

This section provides a detailed explanation of the variables under investigation and the development and validation process for the research instruments used to measure CT skills and programming attitudes. The instruments were carefully designed, validated, and refined to ensure they effectively captured the constructs of interest and were appropriate for the study's target population and objectives.

3.3.1 Dependent and Independent Variables

The study focuses on two main categories of variables: independent and dependent. The independent variables include the type of intervention and gender, while the dependent

variables are the outcomes being measured, specifically CT skills and programming attitudes.

The primary independent variable is the type of intervention, which is categorized into two distinct approaches. The first is the UA+BBP intervention, which integrates UA with BBP using Scratch. This combined approach is designed to offer a comprehensive learning experience by introducing fundamental CT concepts through hands-on, non-digital activities, which are subsequently reinforced and applied through digital programming tasks in Scratch. The control group is the BBP-only approach, which focuses exclusively on digital programming using Scratch, without the preliminary scaffolding provided by UA. This control group serves to evaluate the effectiveness of direct programming instruction in developing CT skills and programming attitudes.

Additionally, the study examines gender as an independent variable to explore potential differences in how male and female students respond to the interventions, providing insights into whether gender influences the effectiveness of the educational approaches. The dependent variables in this study are CT skills and programming attitudes. CT skills are measured using the CT test, which assesses key competencies such as problem decomposition, pattern recognition, abstraction, and algorithmic thinking.

These competencies are essential for problem-solving and programming, making them critical for evaluating the impact of the interventions. Programming attitudes are evaluated through a specially designed survey that measures students' interest, confidence, and perceived value of programming. This survey aims to capture how students perceive and engage with programming as a subject, and how these perceptions may evolve due to their participation in the interventions.

3.3.2 Computational Thinking Test

The CT test was used to evaluate students' CT skills, with focus on key competencies such as problem decomposition, pattern recognition, abstraction, and algorithmic thinking. The CT test developed by Roman-Gonzalez and colleagues was selected for the study because of its carefully examined validity and reliability. The test initially consisted of 40 multiple-choice questions and was refined through a rigorous content validation process involving a panel of experts. The result was a final version with 28 questions, each with four options and only one correct answer. This final iteration was conducted on a diverse group of 1,251 students aged 10 to 16 years (5th to 10th grade), 58% of whom were male. The results showed that the distribution of points was close to normal, with an overall medium level of difficulty ($p = 0.59$) and a range from easy ($p = 0.16$) to very easy ($p = 0.96$).

The internal consistency of the test, measured by Cronbach's alpha, was robust ($\alpha = 0.793$), confirming its reliability. Further validation was achieved through significant positive correlations with established psychological tests such as the Primary Mental Abilities (PMA) battery, which measures verbal, spatial, numerical, and reasoning abilities, and the RP30 problem-solving test. In particular, the CT test showed strong correlations with spatial ability ($r = 0.439$, $p < 0.01$), reasoning ($r = 0.442$, $p < 0.01$) and verbal ability ($r = 0.273$, $p < 0.01$) on the PMA as well as a significant correlation with the RP30 problem-solving test ($r = .669$, $p < .01$). These factors collectively underscore the CT test's suitability for assessing CT in a broad demographic, justifying its selection for this research study (Refer to Appendix K).

To ensure that the test effectively measures the targeted CT competencies, each item was mapped onto a specific construct:

- **Problem Decomposition:** Questions under this category evaluate students' ability to break down complex problems into smaller, more manageable components, a fundamental skill in programming and problem-solving.
- **Pattern Recognition:** These items assess students' capacity to identify patterns or similarities across different problem scenarios, enabling them to apply previously learned solutions to new situations.
- **Abstraction:** This construct involves questions designed to measure students' ability to filter out irrelevant information and focus on the essential aspects of a problem, thereby facilitating the generalization of solutions.
- **Algorithmic Thinking:** Questions in this category require students to devise step-by-step solutions to problems, reflecting their ability to think algorithmically and to apply logical sequences in problem-solving.

The careful alignment of test items with these constructs ensures that the CT test provides a comprehensive measure of students' CT skills, making it an integral part of the research methodology.

3.3.3 Programming Attitude Survey

To assess students' attitudes towards programming, including their interest, confidence, and perceived value of programming, a 17-item survey using a 5-point Likert scale was employed. The survey items were adapted from the science subscale of the Middle/High School Student Attitudes into STEM (S-STEM) Survey, originally developed by Unfried et al. (2015). The original survey contained only 8 items, but an additional 9 items were developed and integrated to better capture the full range of attitudes toward programming.

This adaptation was selected for its established validity and relevance to the educational context. The revised survey underwent a rigorous peer-review process to ensure content validity, and its reliability was further tested through a pilot study. To ensure the construct and content validity of the survey for this study, a panel of five subject matter experts evaluated each item, resulting in revisions that clarified language and reduced potential ambiguities.

The pilot study involved a sample of 106 students comprising 50 males and 56 females aged 13 years. The analysis of the pilot data revealed a strong internal consistency, with a Cronbach's alpha coefficient of 0.896, well above the generally accepted reliability threshold of 0.70. This high level of internal consistency confirms that the survey is a reliable instrument for measuring students' attitudes toward programming within the context of this research (Refer to Appendix L).

The survey items were carefully mapped to specific constructs to ensure a thorough evaluation of students' attitudes:

- **Interest in Programming:** This construct includes items designed to gauge students' curiosity, enjoyment, and enthusiasm for learning programming. These questions assess intrinsic motivation and interest in engaging with programming activities both inside and outside the classroom.
- **Confidence in Programming:** Items under this construct focus on self-efficacy and students' belief in their own abilities to learn and apply programming skills. Questions in this category measure students' confidence in tackling programming challenges and their perception of their capability to succeed in programming tasks.

- **Perceived Value of Programming:** This construct assesses students' views on the importance and applicability of programming skills in real-world scenarios. Items here evaluate how students perceive the relevance of programming to their future careers, academic pursuits, and its utility in solving everyday problems.

The clear alignment of survey items with these constructs ensures that the instrument effectively captures the multi-faceted nature of students' attitudes towards programming. This structured approach provides valuable insights into how different aspects of programming attitudes are influenced by the educational interventions under study, contributing to a comprehensive understanding of the factors that impact students' engagement with programming.

3.3.4 Semi-Structured Interview Protocol

The semi-structured interview protocol was carefully designed to investigate students' experiences and perceptions of the UA and BBP interventions, focusing on their understanding of CT concepts, engagement with the activities, and attitudes toward programming. Comprising 8 open-ended questions, the protocol aimed to elicit comprehensive, reflective responses that would provide deeper insights into the educational impact of the interventions beyond quantitative measures. The development process of the interview protocol was rigorous, beginning with the formulation of questions that were directly aligned with the study's research objectives. Each question was crafted to probe specific aspects of the students' learning experiences, ensuring that the data collected would be both rich and relevant.

To validate the clarity and relevance of the interview questions, the protocol underwent a thorough review by a panel of four educational researchers with expertise in computational thinking and programming education. Their feedback was invaluable,

highlighting potential ambiguities in wording and suggesting ways to make the questions more accessible and age-appropriate for secondary school students. This peer-review process led to several refinements, ensuring that the questions were not only aligned with the study's aims but also understandable and engaging for the target population. The validation process thus enhanced the protocol's effectiveness in capturing nuanced data from the participants.

Following this validation, a pilot interview was conducted with two students who were not part of the main study sample. This pilot phase served as a practical test of the protocol's usability and effectiveness in a real interview setting. The responses from the pilot participants indicated that, while the overall structure of the interview was effective, there were minor issues with the clarity of some questions. Specifically, three questions were identified as needing rephrasing to avoid potential misinterpretations and to ensure that they were easily comprehensible to the students. Based on this feedback, these questions were revised, focusing on simplifying the language and providing additional context where necessary.

The final version of the interview protocol, incorporating these adjustments, was successfully used to capture rich, qualitative data from the main study participants. This data provided valuable insights into the students' experiences with both the UA and BBP components of the interventions, offering a comprehensive view of how these different instructional approaches influenced their learning. The structured yet flexible nature of the semi-structured interviews allowed students to freely express their thoughts and reflections, enabling the researcher to explore emergent themes and patterns in their responses.

Mapping of Interview Questions to Constructs: To systematically capture the qualitative data, each interview question was strategically mapped to one of the three primary constructs:

Understanding of CT Concepts:

Interview Questions:

- "Can you describe a specific activity or lesson that you liked? How do you think it impacted your understanding of CT?"
- "How did the lesson enhance your understanding of complex programming, if at all? Can you recall a specific instance that stands out?"

Rationale: These questions were designed to explore students' comprehension of core CT concepts such as problem decomposition, pattern recognition, abstraction, and algorithmic thinking.

Engagement with Activities:

Interview Questions:

- "Can you describe a specific activity or lesson that you found enjoyable or challenging?"
- "What was your experience like when working with your classmates on the Scratch project?"

Rationale: These questions aimed to assess students' levels of interest, participation, and collaboration during the interventions. By focusing on their enjoyment and perceived challenges, the study sought to understand how the engaging nature of the activities contributed to their learning experiences.

Attitudes Toward Programming:

Interview Questions:

- "What was your attitude towards programming before and after you started engaging in these lessons?"
- "Do you feel that the activities without the computer helped you understand CT and programming better? Why or why not?"

Rationale: These questions explored any changes in students' perceptions and attitudes toward programming as a result of the interventions. The aim was to capture shifts in their initial apprehensions or misconceptions about programming. For instance, comments like "I thought coding would be hard and boring, but when I started learning how to play the conditional card game, I found out that coding is actually easy and fun!" illustrated a transformation in students' attitudes, showing that the interventions made programming more approachable and enjoyable.

This structured approach to mapping questions to constructs ensured that the interview data would provide a holistic view of the students' experiences, complementing the quantitative findings and contributing to a deeper understanding of the educational impact of the interventions.

3.4 Revisions of Research Instruments after Pilot Test

Following the pilot test, the research instruments underwent thorough revisions to address identified issues and enhance their validity and reliability, ensuring their suitability for the target population and alignment with the study's objectives. For the CT test, no modifications were necessary as the test was used in its original form, developed by Román-González and colleagues. Permission was obtained to employ this validated and reliable instrument, which had been extensively tested and confirmed to accurately assess CT skills across various educational contexts.

The Programming Attitude Survey required several adjustments based on pilot test feedback and statistical analysis. Four items were reworded to eliminate ambiguity and improve clarity, ensuring that all respondents interpreted the questions consistently. These changes were aimed at refining the language to be more comprehensible and relevant to secondary school students. Moreover, three items were removed because their inclusion lowered the overall reliability of the survey, as evidenced by their poor alignment with other items. The removal of these items was based on statistical analysis, which indicated that they did not contribute to the survey's internal consistency, thereby justifying their exclusion to enhance the instrument's robustness and coherence.

The semi-structured interview protocol also underwent refinement. Three question was rephrased to simplify the language and ensure that it was accessible to students, thereby reducing the likelihood of misinterpretation and enabling more accurate and in-depth responses. This change was based on feedback from the pilot interviews, which revealed that the original phrasing was somewhat complex for the participants. These revisions were essential for ensuring that the research instruments were not only valid and reliable but also effective in capturing the nuanced data required for a comprehensive analysis of the interventions' impact on students' CT skills and programming attitudes. This meticulous refinement process underscores the commitment to maintaining the rigor and integrity of the study's data collection methods.

3.5 Selection of Respondents

3.5.1 Sampling Strategy

The study employs a purposive sampling strategy, recruiting a total of 106 students from three different boarding secondary schools in Kedah. These schools were selected for their existing basic computer science programs, which align closely with the study's focus

on developing CT skills and programming attitudes. The study also involves three ICT teachers, each responsible for instructing their respective classes. The researcher initiated the process by introducing himself to both the teachers and students, providing a comprehensive explanation of the study's objectives and emphasizing the significance of their participation in evaluating the effectiveness of the educational interventions.

To ensure ethical compliance and maintain transparency, paper copies of the Parental Consent Form (Appendix D) and the Student Consent Form (Appendix E) were distributed to all potential participants. Written consent was obtained from both parents and students who agreed to take part in the study, ensuring that participation was voluntary and that all ethical guidelines were strictly followed. The researcher takes care to emphasize that participation is entirely voluntary and assures students that their involvement, or lack thereof, will have no impact on their academic grades or standing within the school.

It is also highlighted that while all students will engage in the same classroom activities during the intervention period, data will only be collected from those who have provided informed consent. This approach ensures that the ethical standards set by the University of Malaysia Sarawak and the Research and Development Department of the Ministry of Education are strictly adhered to. Furthermore, students are informed that the data collected, including their test results and feedback, will be used solely for research purposes and will remain confidential, not being shared with their teachers or used in any evaluative capacity.

3.5.2 Sample Size and Composition

The study comprises two distinct groups, each consisting of 53 students. The first group, referred to as the UA+BBP Group, engages in a combined intervention that integrates both UA and Scratch-BBP. This group is designed to assess the synergistic effect of both

educational approaches on enhancing CT skills and programming attitudes. The second group, termed the BBP-Only Group, participates solely in digital programming tasks using Scratch, allowing for a comparison between the effects of integrated versus singular teaching methods.

To gain deeper qualitative insights, a purposive sub-sample of 12 students from the UA+BBP group is selected for semi-structured interviews. These students are chosen based on their varying levels of engagement during the interventions and their willingness to share their experiences. This targeted selection aims to capture a diverse range of perspectives, providing a more nuanced understanding of how the combined interventions impact student attitudes and learning processes. Table 3.2 below provides a detailed breakdown of the number of participants in each group, along with the types of interventions they receive and their gender distribution.

Table 3.1: Distribution of Students by Group, School, and Gender

Group	School	Gender		Total
		Male	Female	
BBP-Only Group	A	14	12	26
	B	14	13	27
UA+BBP Group	B	13	13	26
	C	14	13	27
Total		55	51	106

3.5.3 Inclusion Criteria

The selection of participants is based on their enrolment in basic computer science classes, as these courses provide an appropriate foundational context for the study's interventions. Students are also chosen based on their availability to participate throughout the intervention period, ensuring consistency and completeness of data collection. Importantly, the study adheres to stringent ethical guidelines by obtaining both parental consent and student assent for participation in the research.

This dual consent process ensures that both parents and students are fully informed about the nature and purpose of the study, the activities involved, and the data collection procedures. Only those who voluntarily agree to participate are included in both the quantitative and qualitative components of the research, ensuring the integrity and ethical standards of the study.

3.6 Data Collection Methods

Data collection in this study utilizes a combination of pre- and post-tests, surveys, and interviews to provide a comprehensive assessment of students' CT skills and their attitudes toward programming. This mixed-methods approach allows for a robust evaluation of the interventions' impact from both quantitative and qualitative perspectives.

The quantitative data collection begins with the administration of a pre-test and an attitude survey one day before the interventions commence. The pre-test, a standardized multiple-choice assessment, establishes baseline data on students' CT skills, evaluating competencies such as problem decomposition, pattern recognition, abstraction, and algorithmic thinking. Simultaneously, the attitude survey, adapted from the S-STEM Survey, measures students' initial perceptions, confidence, and interest in programming. These

baseline measurements are crucial for understanding the starting point of each student and enable a direct comparison with post-intervention results.

The intervention phase spans four weeks, with students participating in two 60-minute sessions per week. The UA+BBP group engages in a combination of unplugged activities designed to build foundational CT concepts and Scratch programming tasks that allow students to apply these concepts in a digital context. In contrast, the BBP-only group focuses solely on Scratch programming, exploring digital coding practices without the additional unplugged activities. This structured intervention period is designed to provide diverse learning experiences and expose students to different approaches in developing their CT skills and programming attitudes.

Immediately following the intervention period, a post-test and a follow-up attitude survey are administered to assess changes in students' CT skills and programming attitudes. The post-test, identical in format to the pre-test, enables a direct comparison of pre- and post-intervention scores, highlighting any gains in CT skills. The follow-up attitude survey reassesses students' perceptions, confidence, and interest in programming, providing insight into how their attitudes may have evolved because of their participation in the interventions.

Complementing the quantitative data, qualitative data is gathered through semi-structured interviews with a purposive sample of 12 students from the UA+BBP group. These interviews aim to delve deeper into students' experiences with the integrated intervention. The interview protocol includes open-ended questions designed to explore students' engagement with the unplugged activities and Scratch programming, their understanding of CT concepts, and their attitudes towards programming. The interviews are recorded with the students' consent, transcribed verbatim, and analysed using thematic analysis to identify recurring themes and patterns in their responses. This qualitative

component provides rich, contextual insights that add depth to the quantitative findings, offering a more nuanced understanding of the educational impact of the interventions.

3.7 Data Analysis

The data analysis process is meticulously structured to ensure a comprehensive examination and accurate interpretation of both quantitative and qualitative data. This approach is designed to rigorously evaluate the effectiveness of the interventions on students' CT skills and programming attitudes, while also addressing potential confounding variables and enhancing the reliability of the findings.

3.7.1 Quantitative Data Analysis

Quantitative data analysis begins with the application of descriptive statistics to provide a comprehensive overview of the dataset. Using IBM SPSS software, key statistical measures such as means, standard deviations, and frequency distributions are calculated to summarize the central tendencies and variability of pre- and post-test scores, as well as responses from the programming attitude surveys. This initial analysis offers a clear snapshot of the overall data trends, patterns, and any outliers present in the sample. Establishing this baseline understanding is essential for guiding further analysis and interpreting the impact of the interventions on students' CT skills and programming attitudes.

For inferential statistics, several advanced techniques are employed to draw conclusions about the impact of the interventions. Paired-sample t-tests are conducted to compare pre- and post-test scores within each group (UA+BBP and BBP-only). This analysis assesses whether the observed changes in CT skills and programming attitudes are statistically significant within each group over time. To further refine these comparisons, an Analysis of Covariance (ANCOVA) is utilized to control for any initial differences in pre-

test scores. This technique allows for a more accurate assessment of the interventions' effects on post-test outcomes by statistically adjusting for baseline scores, thereby isolating the impact of the interventions themselves.

Additionally, effect size calculations using Cohen's d are performed to evaluate the practical significance of the observed changes, beyond mere statistical significance. This metric provides a measure of the magnitude of the intervention effects, offering valuable insight into their real-world applicability and importance. To address assumptions and validate the appropriateness of the statistical tests, several validity and reliability checks are performed.

Normality tests, specifically the Shapiro-Wilk test, are conducted to assess whether the pre- and post-test score distributions deviate from a normal distribution. If significant deviations are found, appropriate non-parametric tests will be considered. Levene's test is employed to verify the homogeneity of variances between groups, ensuring that the assumptions of the ANCOVA and t-tests are not violated. For cases where the assumption of equal variances is not met, Welch's ANOVA is applied as a robust alternative to traditional ANOVA, providing reliable results even in the presence of unequal variances.

3.7.2 Qualitative Data Analysis

Qualitative data analysis is conducted using a systematic approach to explore the in-depth perspectives of students regarding their experiences with the UA+BBP intervention. Thematic analysis is employed to analyse the transcribed interviews, following a structured process of coding and theme development. NVivo software is utilized for this purpose, offering advanced tools for organizing, coding, and visualizing qualitative data, thereby enhancing the rigor and efficiency of the analysis. The process begins with familiarization,

where the researcher immerses themselves in the data by reading and re-reading the transcripts to gain a deep understanding.

Next, initial codes are generated in NVivo to identify significant phrases and patterns related to students' engagement, understanding of CT concepts, and attitudes toward programming. These codes are then organized into broader themes that encapsulate the core findings of the qualitative data. The final themes are reviewed and refined within the software to ensure they accurately represent the data and provide meaningful insights. This structured approach, supported by NVivo's analytical capabilities, ensures a thorough and systematic exploration of the qualitative data, yielding rich and nuanced insights into the students' experiences.

To enhance the credibility of the qualitative findings, several strategies are employed, including peer debriefing, where an external researcher reviews the coding process and thematic development to confirm the validity of the interpretations. Member checking is also conducted, with selected participants reviewing the initial findings to verify the accuracy and representativeness of their responses. This thorough analysis provides rich, contextual insights that complement the quantitative results, offering a more holistic understanding of the educational impact of the interventions.

3.8 Classroom Intervention Implementation

The implementation of the classroom interventions was meticulously planned to ensure the effective delivery of the educational content and to actively engage students in the development of their CT skills. The interventions were designed to integrate a combination of UA and BBP using Scratch, aimed at catering to the diverse learning needs of secondary school students.

3.8.1 Teacher Training

Prior to the commencement of the intervention, comprehensive training sessions were conducted for the participating teachers to ensure they were well-prepared to facilitate the activities effectively. These sessions focused on the objectives, content, and delivery methods of both the UA and BBP interventions, as well as on strategies for integrating these activities into the regular curriculum.

The training provided a detailed overview of the intervention structure, emphasizing the theoretical underpinnings of CT and the practical aspects of using Scratch for programming education. Teachers were also guided on how to provide ongoing support and encouragement to students throughout the intervention period. This preparatory phase was essential to ensure the consistency and fidelity of the interventions and to enhance the teachers' confidence and competence in delivering the new instructional content.

3.8.2 Intervention Session Structure

The intervention was conducted over a four-week period, with each group receiving two 60-minute sessions per week. The structure of the intervention sessions varied between the two groups to facilitate a comparative analysis of the different instructional approaches. The UA+BBP group engaged in both unplugged activities and Scratch programming. This group was introduced to unplugged activities during the first two weeks, which focused on developing foundational CT skills such as problem decomposition, sequencing, and logical thinking through hands-on, non-digital tasks.

In the subsequent two weeks, they transitioned to Scratch programming, where they applied the CT concepts learned from the unplugged activities in a digital context. This progressive approach was designed to scaffold students' learning, enabling them to build on

their understanding of CT concepts before engaging in more complex digital programming tasks.

In contrast, the BBP-only group focused exclusively on Scratch programming throughout the entire four weeks. This group engaged in structured programming exercises that emphasized core programming constructs such as loops, conditionals, and variables. The aim was to explore how the direct introduction of programming concepts, without the support of unplugged activities, impacted students' CT skills and attitudes towards programming. The distinct separation of the intervention types allowed for a direct comparison of the effectiveness of integrating unplugged activities with digital programming versus digital programming alone.

3.8.3 Teacher Input and Lesson Plan Review

Before the implementation of the interventions, all lesson plans and teaching materials underwent a rigorous review process conducted by four experienced CT educators. These educators were selected based on their extensive experience and expertise in teaching CT at the secondary school level. The review process focused on evaluating the pedagogical soundness, clarity, and appropriateness of the materials for the target age group.

Specific attention was given to the pacing of the lessons, the complexity of the content, and the incorporation of differentiated instruction to accommodate the varying abilities of students. The feedback provided by these educators led to significant refinements in the lesson plans, ensuring that the instructional content was both engaging and accessible to all students. This peer review process was instrumental in aligning the lesson plans with best practices in CT education and enhancing the overall quality of the intervention materials (Refer to Appendix F for detailed feedback).

3.8.4 Participants for Pilot Test

The pilot test was conducted with 32 students from full boarding schools, representing a diverse range of CT and programming skills. This sample size is consistent with best practices for pilot studies, which aim to manage resources effectively while collecting sufficient data to identify potential issues and refine the study instruments. During the pilot phase, students participated in both unplugged activities and Scratch BBP projects. Feedback forms were administered after each activity to assess the students' understanding of the objectives, their enjoyment of the activities, and their interest in future participation.

The feedback received was overwhelmingly positive, indicating that the activities were engaging and educationally valuable. This pilot phase also allowed the researcher to observe teacher-student interactions and gather additional insights into the efficacy of the instructional methods. Teachers overseeing the pilot test provided professional feedback, corroborating the positive student responses and highlighting the effectiveness of the teaching methods and materials used. This iterative refinement process was essential in ensuring that the main study would be both effective and impactful.

3.9 Intervention Implementation Rationale

3.9.1 Rationale for Scratch as a Teaching Tool

Scratch is a visual BBP language developed by the Massachusetts Institute of Technology (MIT) that allows students to create interactive stories, animations, and games. Scratch is an effective tool for developing CT skills and programming in middle and high school students, as several studies have shown. Additionally, Scratch is available both online and offline, making it suitable for classroom settings where internet connectivity is often

unreliable. The Scratch website provides extensive documentation for both students and educators, making it easy to learn and use.

3.9.1.1 Drawing Geometrical Shapes

The "Drawing Geometrical Shapes" project underscores essential coding principles such as sequencing and looping. Utilizing the Scratch pen tool, students apply these concepts in a visually engaging and interactive medium. This activity integrates coding with geometric principles by concentrating on the creation of snowflakes—naturally occurring fractal patterns. This integration not only bolsters students' understanding of polygons and symmetry but also demonstrates how programming can serve as a tool to explore and represent mathematical concepts.

Creating geometrical shape in Scratch allows students to immediately see the outcomes of their coding decisions, providing a tangible result that enhances both understanding and retention. This visual component is particularly effective in helping students comprehend abstract concepts by linking them to concrete visual patterns. Additionally, the creative process of designing and drawing snowflake patterns maintains student engagement and interest, vital for learning and applying new skills. The project strikes a balance between structure and creativity, encouraging students to explore coding in a playful yet educational context.

3.9.1.2 Math Race Game

The "Math Race Game" project effectively merges math skills with coding, offering a practical application of mathematics within a programming framework. By incorporating mathematical challenges into the game mechanics, students practice their math skills while appreciating the relevance of mathematics in real-world scenarios and digital environments. Designing a game necessitates thoughtful consideration of rules, mechanics, and user

experience. Students engage in logical thinking and problem-solving to create a game that is both functional and engaging, reflecting the developmental processes used by professional developers.

The incorporation of gamification serves as a potent educational tool, known to boost motivation and engagement. Developing a math-centric game allows students to experience the excitement of learning through play, which can positively influence their attitudes towards both mathematics and programming. This approach not only makes learning more enjoyable but also reinforces the applicative nature of programming skills in solving real-world problems.

3.9.2 Selection and Application of UA

UA are non-digital tasks designed to enhance CT skills, such as problem-solving, algorithmic thinking, and pattern recognition. These activities offer students a hands-on, interactive learning experience that fosters critical thinking and creativity, as noted by Yıldız and Karal (2021). To ensure that these activities align with students' learning goals and developmental stage, the study includes a series of three proven UA: the Building Tower Activity, the Stacking Cup Challenge, and the Conditional Card Game.

The selection was confirmed by experienced teachers for its pedagogical relevance and age appropriateness and was further validated through a pilot test that collected student feedback on its effectiveness. To enrich the study's examination of CT skills enhancement, the selection of UA was deliberate and methodical, aimed at embodying core CT principles. The activities chosen are the Building Tower Activity, Conditional Card Game, and Stacking Cup Challenge, each aligning with essential computational concepts.

3.9.2.1 Building Tower Activity

The building tower activity was chosen to highlight algorithmic thinking and sequencing (Smith and Caldwell, 2016). Students participate in a systematic construction process that is similar to the sequential logic inherent in programming (Bers et al., 2014). The activity inherently introduces the practice of debugging as students iterate on their design strategies as they encounter design limitations, fostering important CT problem-solving skills (Lee et al., 2011). Additionally, this activity is collaborative and simulates the dynamic team-based environments often found in computer workstations, improving communication skills essential to CT (Grover and Pea, 2013).

3.9.2.2 Stacking Cup Challenge

Chosen for its embodiment of sequencing and pattern recognition, the Stacking Cup Challenge introduces students to the notions of loops and iterations—a staple in programming workflows (Yadav et al., 2014). The activity also subtly integrates the principle of efficiency, prompting students to seek optimal stacking strategies, which metaphorically relates to the optimization processes in software development (Selby, 2015).

3.9.2.3 Conditional Card Game

The Conditional Card Game serves as an exemplary tool for understanding the basic concept of conditionals in CT. Through gameplay, students are encouraged to practice logical thinking while making decisions based on the conditional rules presented, which is analogous to the decision logic used in programming constructs (Brackmann et al., 2017). This engagement with a concrete representation of an abstract concept is crucial to understanding the intricacies of computer logic (Brennan and Resnick, 2012).

3.9.3 Summary

This chapter provides a detailed description of the research process, including the development, validation, and revision of research instruments, the selection of respondents,

data collection methods, and data analysis procedures. The mixed-methods design, incorporating both quantitative and qualitative data, ensures a comprehensive evaluation of the impact of integrating UA with BBP on students' CT skills and programming attitudes. The next chapter will present the results of the data analysis, offering insights into the effectiveness of the interventions and their implications for CT education.

CHAPTER 4

RESULTS

4.1 Overview

This chapter provides a comprehensive analysis of the effects of integrating UA and BBP on developing CT skills and attitudes toward programming among students. The chapter addresses key research questions, including whether the type of teaching method (BBP alone vs. UA+BBP) significantly affects CT skills among secondary school students and whether there are significant differences in CT skills between male and female students across different teaching methods. It also explores the interaction effects between teaching method and gender on CT skills. Additionally, it investigates whether the integration of UA+BBP influences students' attitudes toward programming differently than BBP alone and if there are gender differences in the effectiveness of UA+BBP and BBP-only methods on students' attitudes toward programming. To supplement the quantitative findings, the chapter includes a qualitative review through thematic analysis of interviews with students in the UA+BBP group.

4.1.1 Data Screening, Missing Data, Outliers, and Normality Analysis

Before conducting the primary statistical analyses, it was critical to undertake a series of preliminary data screening procedures. These included addressing missing data, detecting and handling outliers, and assessing the normality of the dataset. Such measures are essential to ensure the accuracy, validity, and reliability of the subsequent statistical analyses, thereby enhancing the robustness of the study's findings.

4.1.2 Data Screening and Handling Missing Data

The initial dataset comprised 106 students who provided complete data on both pre-test and post-test CT scores and responses to the programming attitude survey. A meticulous inspection of the dataset revealed a small proportion of missing responses (< 5%) across some variables. Given the minimal extent of missing data, mean imputation was employed to handle these gaps. This method involves replacing missing values with the mean of the available data for the affected variable, a strategy appropriate for datasets with low levels of missing data. Mean imputation is advantageous in preserving the sample size and maintaining the statistical power of the analysis without significantly altering the distribution of the data.

4.1.3 Outliers Detection and Management

Outliers were identified using both graphical and statistical methods. Initially, boxplots were examined for any visually conspicuous outliers. Subsequently, Z-scores were calculated for each observation, with values exceeding ± 3 flagged as potential outliers (Tabachnick et al., 2013). In total, four outliers were detected in the post-test CT scores and five in the attitudinal survey responses. To assess the influence of these outliers on the overall results, a sensitivity analysis was conducted. The analysis indicated that the presence of these outliers did not materially affect the results. Given the importance of preserving the sample size and the relatively minor impact of these outliers, it was decided to retain them in the dataset. This approach ensures the integrity of the dataset and avoids the potential biases that might result from arbitrary data exclusion.

4.1.4 Testing the Assumptions of Normality and Homogeneity

This analysis meticulously examines the relationship between two independent variables teaching approach (BBP vs. UA+BBP) and gender (male vs. female)—and two dependent variables, specifically students' CT test scores and attitudes towards programming survey result. Before selecting the appropriate statistical tests for data analysis, it was crucial to first evaluate the assumptions of normality and homogeneity of variances. When assessing these assumptions, specific statistical thresholds are applied. For the Shapiro-Wilk tests, which are used to assess normality, a p-value greater than 0.05 suggests that the data can be considered normally distributed, thus failing to reject the null hypothesis of normality. Conversely, if the p-value is less than or equal to 0.05, the data do not follow a normal distribution, leading us to reject the null hypothesis.

Regarding the assessment of variances, Levene's test is employed to check for homogeneity across different groups. If the p-value from Levene's test is greater than 0.05, it indicates that the variances are homogeneous, thus meeting the assumptions required for certain parametric tests such as ANOVA. A p-value of 0.05 or less, however, would indicate significant differences in variances, suggesting that the data do not meet the criteria for homogeneity of variances.

4.2 Testing the Assumptions of Normality and Homogeneity

This analysis meticulously examines the relationship between two independent variables teaching approach (BBP vs. UA+BBP) and gender (male vs. female)—and two dependent variables, specifically students' CT test scores and attitudes towards programming survey result. Before selecting the appropriate statistical tests for data analysis, it was crucial to first evaluate the assumptions of normality and homogeneity of variances. When assessing these assumptions, specific statistical thresholds are applied. For the

Shapiro-Wilk tests, which are used to assess normality, a p-value greater than 0.05 suggests that the data can be considered normally distributed, thus failing to reject the null hypothesis of normality. Conversely, if the p-value is less than or equal to 0.05, the data do not follow a normal distribution, leading us to reject the null hypothesis.

Regarding the assessment of variances, Levene's test is employed to check for homogeneity across different groups. If the p-value from Levene's test is greater than 0.05, it indicates that the variances are homogeneous, thus meeting the assumptions required for certain parametric tests such as ANOVA. A p-value of 0.05 or less, however, would indicate significant differences in variances, suggesting that the data do not meet the criteria for homogeneity of variances.

4.2.1 Normality of CT Test Score

The results of the normality tests for the CT test scores indicate that the data were normally distributed in both teaching groups. For the BBP group, the Shapiro-Wilk statistic was 0.960 with a p-value of 0.073, indicating normal distribution as the p-value is greater than the typical alpha level of 0.05. Similarly, the CT test scores for the UA+BBP group had a Shapiro-Wilk statistic of 0.960 with a p-value of 0.077, also demonstrating normality. These results suggest that both groups' data do not significantly deviate from a normal distribution, allowing for the use of parametric tests in further analyses. Table 4.1 shows the normality test results for the post-CT test scores by group.

Table 4.1: Normality Test for CT Test Score by Group

Group	Shapiro-Wilk			Skewness	Kurtosis
	Statistic	df	Sig.		
BBP	0.960	53	0.073	0.124	-1.058
UA+BBP	0.960	53	0.077	0.426	-0.274

On the other hand, the normality tests for the CT test scores revealed that the data were normally distributed for males but not for females. Specifically, the Shapiro-Wilk test for males yielded a statistic of 0.977 with a p-value of 0.360, confirming normal distribution as the p-value exceeds the conventional alpha level of 0.05. Conversely, the Shapiro-Wilk test for females returned a statistic of 0.929 with a p-value of 0.005, suggesting a deviation from normal distribution. However, despite this deviation, the values of skewness and kurtosis for females (skewness = -0.754; kurtosis = -0.121), which fall between -1.5 and +1.5, indicate that the distribution of means across samples can be assumed to satisfy the assumption of normality, as suggested by Tabachnick et al. (2013). Table 4.2 presents the normality test results for the post-CT test scores by gender.

Table 4.2: Normality Test for the CT Test Score by Gender

Gender	Shapiro-Wilk			Skewness	Kurtosis
	Statistic	df	Sig.		
Male	0.977	55	0.360	-0.219	-0.246
Female	0.929	51	0.005	-0.754	-0.121

4.2.2 Normality of Attitudinal Survey Scores

The results of the normality tests for the attitudinal survey scores indicate that the data were normally distributed for the UA+BBP group but not for the BBP group. For the BBP group, the Shapiro-Wilk statistic was 0.945 with a p-value of 0.016, indicating a deviation from normal distribution as the p-value is less than the typical alpha level of 0.05. However, for the UA+BBP group, the Shapiro-Wilk statistic was 0.981 with a p-value of 0.561, indicating normal distribution. The skewness and kurtosis values for both groups are within the acceptable range of -1.5 to +1.5, which supports the assumption of normality for further analyses. Table 4.3 shows the normality test results for the attitudinal survey scores by group.

Table 4.3: Normality Test for Attitudinal Survey Scores by Group

Gender	Shapiro-Wilk			Skewness	Kurtosis
	Statistic	df	Sig.		
BBP	0.945	53	0.016	-0.747	0.456
UA+BBP	0.981	53	0.561	-0.244	0.185

For gender-based analysis, the normality tests revealed that the data were normally distributed for males but not for females. Specifically, the Shapiro-Wilk test for males yielded a statistic of 0.973 with a p-value of 0.248, confirming normal distribution. Conversely, the Shapiro-Wilk test for females returned a statistic of 0.926 with a p-value of 0.003, suggesting a deviation from normal distribution. Nevertheless, the skewness and kurtosis values for females (skewness = -1.024; kurtosis = 0.985), along with those for males,

fall within the acceptable range, allowing for the assumption of normality. Table 4.4 presents the normality test results for the attitudinal survey scores by gender.

Table 4.4: Normality Test for Attitudinal Survey Scores by Gender

Gender	Shapiro-Wilk			Skewness	Kurtosis
	Statistic	df	Sig.		
Male	0.973	55	0.248	-0.285	-0.349
Female	0.926	51	0.003	-1.024	0.985

4.2.3 Levene's Test and Further Analysis

Levene's test was initially conducted to assess the homogeneity of variances for these variables. The results showed that variances in post-test CT scores were homogeneous across both groups and genders ($F = 1.337$, $p = 0.267$), enabling the use of parametric tests for further analysis. Subsequently, a two-way ANCOVA was performed, incorporating pretest CT scores as a covariate. However, significant differences in variances were observed in the post-attitude survey scores across both groups and genders ($F = 4.624$, $p = 0.004$), indicating the necessity for employing non-parametric methods or alternative analytical strategies that do not rely on the assumption of equal variances. To address this, Welch's ANOVA was utilized to assess students' attitudes toward programming. This approach allows for more accurate analysis under conditions where the assumption of homogeneity of variances is violated, ensuring the robustness of the findings regarding students' attitudes.

4.3 Rationale for Using ANCOVA and Welch's ANOVA

The selection of statistical tests was guided by the specific nature of the research questions and the characteristics of the data observed during the screening process. The use

of both Analysis of Covariance (ANCOVA) and Welch's ANOVA was justified based on several considerations:

4.4 Justification for Using ANCOVA

ANCOVA was chosen to evaluate the effects of the independent variables teaching method (UA+BBP vs. BBP alone) and gender on the dependent variable of post-test CT scores, while controlling for the covariate of pre-test CT scores. This approach was appropriate given that pre-test scores could significantly influence post-test outcomes, potentially confounding the results. By including pre-test scores as a covariate, ANCOVA adjusts for any initial differences in CT skills between groups, thereby isolating the effect of the teaching interventions on the post-test scores. This statistical adjustment enhances the validity and reliability of the findings, allowing for a more accurate estimation of the true impact of the instructional methods.

Furthermore, ANCOVA is robust to minor violations of the assumption of normality and homogeneity of variance, as indicated by the normality and Levene's test results. Although minor deviations were noted, the overall distributional properties of the data justified the use of ANCOVA for analysing the main effects and interaction effects of teaching methods and gender on CT scores.

4.5 Justification for Using Welch's ANOVA

Welch's ANOVA was selected for analysing students' attitudes toward programming due to the violation of the homogeneity of variance assumption, as indicated by Levene's test results ($F = 4.624$, $p = 0.004$). Unlike traditional ANOVA, which requires equal variances across groups, Welch's ANOVA is robust to heterogeneity of variances and provides more reliable results when this assumption is violated (Delacre et al., 2019). This

test is particularly useful for datasets with unequal group sizes or variances, ensuring accurate Type I error rates and increased statistical power in detecting significant effects (Moder, 2010).

Given the significant variance differences observed in the attitudinal survey scores across groups and genders, Welch's ANOVA was deemed the most suitable choice for analyzing the data. This decision was crucial for maintaining the integrity of the analysis and ensuring that the results reflect the true differences in students' attitudes toward programming across the different teaching methods and gender groups.

4.6 Research Findings

This section presents the results of a two-way ANCOVA and Welch's ANOVA conducted to assess the effects of teaching practices and gender differences on student outcomes. The analysis is divided into three sections. The first section examines the influence of teaching group (BBP versus UA+BBP) and gender on post-test CT results using a two-way ANCOVA. The second section utilizes Welch's ANOVA to evaluate students' attitudes toward programming. The third section involves a thematic analysis of student interviews, providing qualitative insights that complement the quantitative data.

4.6.1 Two-Way ANCOVA

A two-way ANCOVA analysis was conducted to examine the effects of instructional methods and gender on students' CT test scores. This analysis specifically addressed how these factors influence CT development while accounting for differences in students' initial proficiency levels as evidenced by pretest scores. This method is particularly effective in adjusting for initial differences in proficiency levels and allows for a clearer understanding of the true impact of instructional methods and gender on the outcome variable (Field, 2013;

Salkind and Frey, 2019). Using pretest scores as a covariate helps control for these initial differences, thereby reducing bias and increasing the validity of the results (Keith, 2019).

Table 4.5 shows descriptive Statistics for Post-test CT Test Scores

Table 4.5: Descriptive Statistics for Post-test CT Test Scores

Group	Gender	Mean	Std. Deviation	N
BBP	Male	19.21	3.542	28
	Female	16.40	3.452	25
	Total	17.89	3.745	53
UA+BBP	Male	19.41	3.713	27
	Female	17.38	2.954	26
	Total	18.42	3.483	53
Total	Male	19.31	3.595	55
	Female	16.90	3.214	51
	Total	18.15	3.609	106

These descriptive statistics indicate that the UA+BBP group generally had higher mean scores compared to the BBP group alone, and that males tended to score higher than females across both groups. Table 4.6 shows ANCOVA results for post-test CT test scores.

Table 4.6: Two-way Analysis of Covariance (ANCOVA) Result

Source	Type III Sum of Squares	df	Mean Square	F	Sig.	Partial Eta Squared
Corrected Model	660.167 ^a	4	165.042	23.563	<0.001	0.483
Intercept	203.687	1	203.687	29.081	<0.001	0.224
Pre-Test	493.968	1	493.968	70.525	<0.001	0.411
Group	18.113	1	18.113	2.586	0.111	0.025
Gender	8.015	1	8.015	1.144	0.287	0.011
Group * Gender	3.592	1	3.592	0.513	0.476	0.005
Error	707.418	101	7.004			
Total	36290.000	106				
Corrected Total	1367.585	105				

^a= R Squared = 0.483 (Adjusted R Squared = 0.462)

The corrected model explained approximately 48.3% of the variance in post-test CT scores (Partial Eta Squared = 0.483), demonstrating the significant overall impact of the factors considered in the model.

4.6.2 RQ1: Impact of Teaching Method on CT Skills

The descriptive statistics presented in Table 4.5 provide a preliminary view of the impact of different teaching methods on students' computational thinking (CT) skills. The mean post-test CT score for the UA+BBP group was slightly higher ($M = 18.42$, $SD = 3.48$) compared to the BBP group alone ($M = 17.89$, $SD = 3.75$). This suggests that integrating

unplugged activities with block-based programming might have a marginal positive effect on students' CT skills. However, these differences in mean scores need to be evaluated in the context of statistical significance to ascertain whether they are attributable to the teaching method or are merely the result of random variation.

To control for potential confounding variables, a two-way Analysis of Covariance (ANCOVA) was conducted, using the pre-test CT scores as a covariate. The results are presented in Table 4.6. The analysis revealed that the effect of the teaching method (Group) on the post-test CT scores was not statistically significant, $F(1, 101) = 2.586$, $p = 0.111$, $\eta^2 = 0.025$. The small effect size ($\eta^2 = 0.025$) indicates that the teaching method accounted for only 2.5% of the variance in post-test CT scores after controlling for pre-test scores. This suggests that, although the UA+BBP group demonstrated a higher mean score than the BBP group, this difference is not statistically significant when the pre-existing CT skill levels are considered.

4.6.3 RQ2: Gender Differences in CT Skills Across Teaching Methods

As shown in Table 4.5, males generally scored higher than females in both teaching groups. In the BBP group, male students had a mean score of 19.21 (SD = 3.54) compared to 16.40 (SD = 3.45) for female students. Similarly, in the UA+BBP group, the mean score for males was 19.41 (SD = 3.71) while that for females was 17.38 (SD = 2.95). These patterns suggest a trend of higher performance among male students across both teaching methods. However, these mean differences alone do not determine whether gender has a statistically significant effect on CT skills; hence, further statistical testing is required.

To assess whether gender and the interaction between gender and teaching method significantly influenced CT skills, a two-way ANCOVA was conducted. The results indicated that the main effect of gender was not statistically significant, $F(1, 101) = 1.144$, p

= 0.287, $\eta^2 = 0.011$, and the effect size was minimal ($\eta^2 = 0.011$). This suggests that gender alone accounted for only 1.1% of the variance in post-test CT scores, which is a negligible effect. Additionally, the interaction effect between group and gender was not significant, $F(1, 101) = 0.513$, $p = 0.476$, $\eta^2 = 0.005$, implying that the teaching methods affected male and female students similarly, contributing to only 0.5% of the variance in CT scores.

4.6.4 Welch's ANOVA

Welch's ANOVA was employed to analyse students' attitudes toward programming, given its robustness in handling unequal variances among groups. This approach is particularly suitable for our dataset where the assumption of equal variances across different groups and genders was not met. According to Delacre et al. (2019), Welch's ANOVA is more reliable than the traditional ANOVA when the assumption of homogeneity of variances is violated. Additionally, Moder (2010) highlighted that Welch's ANOVA is a robust alternative that provides accurate Type I error rates and greater statistical power under conditions of unequal variances. Descriptive statistics and post hoc comparisons are presented to explore differences in attitudes across groups and genders. Table 4.7 shows descriptive statistics for attitudes programming survey.

Table 4.7: Descriptive Statistics for Attitude Toward Programming Survey

Group	Gender	Mean	Std. Deviation	N
BBP	Male	3.8279	0.41641	28
	Female	3.4075	0.54550	25
	Total	3.6296	0.52179	53
UA+BBP	Male	4.0649	0.40517	27

Table 4.7: Continued

	Female	4.1040	0.27928	26
	Total	4.0841	0.34636	53
Total	Male	3.9443	0.42430	55
	Female	3.7626	0.55272	51
	Total	3.8569	0.49637	106

The sample size (N) indicates the number of students in each category. Females in the BBP group have a lower average score of 3.41, suggesting fewer positive attitudes compared to males. In the UA+BBP group, both males and females have higher average scores (4.06 and 4.10, respectively), indicating more positive attitudes. Overall, males tend to have higher attitude scores than females, but the UA+BBP group shows higher attitudes for both genders compared to the BBP group. Table 4.8 displays the results from pairwise comparisons between groups, indicating significant differences where applicable and providing confidence intervals to aid in interpreting the magnitude and reliability of these differences.

Table 4.8: Post Hoc Test Analysis for Differences in Attitudes Toward Programming

(I) Group	(J) Group	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Lower Bound	Upper Bound
BBP Male	BBP Female	0.42045*	0.13452	0.016	0.0615	0.7794

Table 4.8: Continued

	UA+BBP Male	-0.23696	0.11078	0.154	-0.5308	0.0569
	UA+BBP Female	-0.27609*	0.09588	0.029	-0.5314	-0.0208
BBP Female	BBP Male	-0.42045*	0.13452	0.016	-0.7794	-0.0615
	UA+BBP Male	-0.65741*	0.13410	<0.001	-1.0154	-0.2994
	UA+BBP Female	-0.69654*	0.12208	<0.001	-1.0256	-0.3675
UA+BBP Male	BBP Male	0.23696	0.11078	0.154	-0.0569	0.5308
	BBP Female	0.65741*	0.13410	<0.001	0.2994	1.0154
	UA+BBP Female	-0.03913	0.09529	0.976	-0.2931	0.2148
UA+BBP Female	BBP Male	0.27609*	0.09588	0.029	0.0208	0.5314
	BBP Female	0.69654*	0.12208	<0.001	0.3675	1.0256
	UA+BBP Male	0.03913	0.09529	0.976	-0.2148	0.2931

4.6.5 RQ3: Influence of UA+BBP on Attitudes Towards Programming

Table 4.7 presents the descriptive statistics for students' attitudes toward programming across the two teaching methods. The mean attitude scores suggest that students in the UA+BBP group exhibit more positive attitudes toward programming ($M = 4.08$, $SD = 0.35$) compared to those in the BBP group alone ($M = 3.63$, $SD = 0.52$). This trend is consistent across both male and female students, with males in the UA+BBP group having a mean score of 4.06 ($SD = 0.41$) and females having an even higher mean score of

4.10 (SD = 0.28). In contrast, the BBP group shows lower mean scores for both males (M = 3.83, SD = 0.42) and females (M = 3.41, SD = 0.55). These differences suggest that incorporating UA alongside BBP may foster more positive attitudes toward programming among students.

The Welch's ANOVA results are presented in Table 4.8 and indicate a statistically significant effect of the teaching method on students' attitudes toward programming, $F(1, 100.48) = 15.672$, $p < 0.001$. Post-hoc comparisons using the Games-Howell test, which is suitable for unequal variances and sample sizes, reveal that the UA+BBP group had significantly higher attitude scores compared to the BBP group alone. The confidence intervals for the mean differences between the groups do not include zero, confirming that the observed differences are statistically significant.

4.6.6 RQ4: Gender Differences in Attitudes Towards Programming

Table 4.7 also provides a breakdown of attitudes toward programming by gender within each teaching method. In the BBP group, male students have a higher mean attitude score (M = 3.83, SD = 0.42) compared to female students (M = 3.41, SD = 0.55). However, in the UA+BBP group, both male and female students report similarly high attitude scores, with males scoring 4.06 (SD = 0.41) and females scoring 4.10 (SD = 0.28). This suggests that the UA+BBP approach may help mitigate gender differences in attitudes toward programming, as evidenced by the more balanced and positive attitudes among both genders in this group.

The Welch's ANOVA results for gender differences in attitudes toward programming indicate no significant main effect of gender, $F(1, 102.72) = 1.347$, $p = 0.249$. Additionally, the interaction between gender and teaching method is not significant, $F(1, 101.56) = 0.512$, $p = 0.476$. These results suggest that while there are mean differences in

attitudes between genders, these differences are not statistically significant across the two teaching methods.

The post-hoc comparisons do not indicate significant differences between male and female students within the same teaching method group. Both genders in the UA+BBP group reported similar attitudes, indicating that the integrated approach effectively supports positive attitudes toward programming for both male and female students.

4.7 Qualitative Analysis of Student Interviews

The post-intervention interviews with students from the UA+BBP group yielded profound insights into their experiences with the integration. A thematic analysis of the students' responses revealed detailed perspectives on how these educational methods influenced their attitudes in programming. The students' feedback provides a rich, nuanced understanding of the intervention's impact, transcending mere quantitative measures. Their reflections and perceptions offer valuable evidence of the effectiveness of merging UA with BBP, particularly in enhancing engagement, comprehension, and enthusiasm for programming. Table below show some quotes from student interview responses.

Table 4.9: Quotes from Student Interview Responses

Participant responses	Keywords
Theme 1: Improving programming through fun and challenging activities. Can you describe a specific activity or lesson that you like? How do you think it impacted your understanding of CT?	
"I loved the tower project. It's such an awesome game. It was very challenging and fun!" [Participant 1]	Challenging

Table 4.9: Continued

• "I like cup stacking because it made me realize how important instructions are, just like when we are coding." [P2] • "We did a project with Scratch where we Draw The Frozen Fractals (Snowflake). It was awesome because I could see how each block made the sprite move and draw the complex pattern." [P3] • " I loved the one where you need to stack the cup on top of each other. I think it was really fun and it helped me with coding." [P8] • "Working on the tower project was awesome. We made the project together like teamwork and solved it together." [P12]	• Important Instructions • Seeing the block moving • Game condition • Help with coding • Teamwork
Theme 2: Simplifying complex programming concepts. How did the lesson enhance your understanding of complex programming, if at all? Can you recall a specific instance that stands out?	
• "During the Math Race project, I felt like magic because I was able to solve the problem very well." [P1] • "Working on the snowflake project in Scratch with my classmates was fun and challenging, and it helped me understand how coding works, almost like a secret language." [P6] • "In the Snowflake project, I made two snowflakes without any instructions. It was fun doing a project all by myself and it got easier over time." [P8] • "When we make the cup stacking activity, it made me and my friend think about the orders of the action and it was like planning a more simple strategy and it made my coding skill better." [P9] • "During the Cup Stacking activity, I learned on how to make a code simple enough for a computer to understand." [P10]	• Problem solving • Understanding • Simplicity • Command

Table 4.9: Continued

• "During the Scratch snowflake drawing project, arranging the block codes was really fun. I understood how the blocks made the cat move, it felt like creating a game." [P11]	
Theme 3: Foundation for programming In your opinion, do you believe that doing activities without computer help you understanding CT and programming? Why or why not?	
• "Obviously! Doing things without a computer made it less hard and less scary. It's like learning how to write without correction - perfection." [P1] • "In my opinion, it very helps. For example, cup stacking activity. It helps to learn the basics." [P2] • "For me, the cup stacking was really basics. It was hands-on, and it made how the computer follow instructions easier to understand afterward." [P3] • "Yeah, doing the things without a computer made it less scary. It's like	• Less hard and less scary • Learn the basic • Hands-on • Instructions easier • Basic and the stater
• learning to ride a bike without the pedals first." [P4] • "Yes. It is because the activity without computer are like the basic and the starters before getting in real programming." [P6]	
Theme 4: Changes in attitude towards programming What was your attitude towards programming before and after you started engaging in this lesson?	
• "I thought coding would be hard and boring. But when I start learning how to play conditional card game, I found out that coding is actually easy and fun to learn! Everything changes my mind after that." [P4] •	• boring • Easy • Fun • ideas • Confusing • Fun

Table 4.9: Continued

• "At first I thought coding is really hard, but after I'm doing the drawing snowflake project, I realize that it's not as hard as I thought." [P6]	
• "Creating things in math race made it feel like a real game. Seeing what I have created on the screen was unbelievable! Like it was my own idea it moves!" [P7] • " I was really confused. I thought coding was only for geniuses. But when we did the cup thing, it felt like anyone could do it!" [P8] • "The building tower changed all my negative expectation about programming. It was very fun and enjoy." [P9]	

The quotes from students in the UA+BBP group illustrate the transformative effect of this combined educational approach on their programming journey. Students highlighted that engaging in fun and challenging activities, such as the tower project and cup stacking, significantly enhanced their understanding of CT. They appreciated how these activities demonstrated the importance of instructions and allowed them to visualize the movement of blocks in coding. Projects like drawing fractals in Scratch made the learning experience enjoyable and facilitated teamwork. Additionally, lessons and activities like the Math Race and Snowflake project in Scratch helped simplify complex programming concepts, making coding feel like solving a puzzle or understanding a secret language. These projects improved students' problem-solving skills and taught them to simplify commands, boosting their overall coding abilities.

UA, such as cup stacking, were perceived as foundational for understanding CT and programming. Students expressed that these hands-on activities made learning less intimidating and provided a solid basis for grasping how computers follow instructions. They

likened this approach to learning fundamental skills before advancing to more complex tasks. Initially, many students found coding daunting and boring. However, through activities like the conditional card game and the Scratch snowflake project, their attitudes shifted. They realized that coding could be easy, fun, and accessible. These activities helped demystify programming, making it feel more achievable and enjoyable.

4.8 Summary

Overall, the study provides insightful findings on the impact of instructional interventions using both BBP and UA+BBP methods on secondary students' understanding of CT concepts and practices, as well as their attitudes toward programming. The two-way ANCOVA analysis for CT scores and Welch's ANOVA for the attitude survey revealed that the addition of UA to BBP did not significantly improve CT scores. However, the UA+BBP method was particularly effective in fostering positive attitudes toward programming, especially among female students.

CHAPTER 5

DISCUSSION

5.1 Overview

The results of the quantitative and qualitative measurements for each research question were reported in the previous chapter. This chapter attempts to discuss the results and provide explanations for the results. In addition, implications of the results for teaching practice and limitation of research are also discussed in this chapter.

5.2 Effectiveness of Integrating UA with BBP in Enhancing CT Skills

The integration of UA with BBP led to a marginal numerical improvement in CT scores, although this increase did not achieve statistical significance. This finding prompts a critical examination of how these teaching methods are synergized and implemented within an educational setting. Ferguson-Patrick et al. (2018) and Gale et al. (2020) highlight that the efficacy of such integrative educational strategies relies heavily on their contextual adaptation within the curriculum. Effective integration must resonate with and amplify the core objectives of the curriculum, producing a synergistic effect that elevates the educational value of each component (Vinnervik, 2022).

The role of educators in this integration process is pivotal. Fojcik et al. (2022) emphasize that the success of blending UA with BBP significantly hinges on the quality of teacher engagement. Effective education transcends mere content delivery; it involves actively engaging students in ways that align with and promote learning goals, fostering an environment conducive to deeper understanding and application of CT principles (Holmes et al., 2022). This dynamic interaction is essential for the successful adoption of innovative teaching methods aimed at enhancing CT skills. For many students, this was their first formal

exposure to CT concepts and practices in the classroom. Given this context, it is unrealistic to expect them to show a high level of understanding after just an eight-hour intervention.

Research supports the potential benefits of integrating UA with BBP. Studies by Brackmann et al. (2017), Metin (2022), and Merino-Armero et al. (2022) and show that UA can significantly improve motivation and CT skills. These studies highlight the importance of engaging, hands-on learning experiences that make abstract concepts more concrete and accessible, thereby enhancing students' understanding and application of CT principles. Additionally, Minamide et al. (2020) and del Olmo-Muñoz et al. (2020) demonstrated that UA, such as board games and coding worksheets, positively influence CT skills and learning motivation.

Furthermore, Tsarava et al. (2019) showed that gamified, UA could serve as effective low-threshold introductions to CT, making it easier for students to grasp foundational concepts. The study indicated that life-size board games and other interactive activities help demystify CT, making it more approachable for younger learners. These findings suggest that unplugged activities, when integrated with digital programming tools like BBP, can create a well-rounded educational experience that enhances student engagement and comprehension.

Contrastingly, some research indicates that the effectiveness of these integrative strategies may not always be straightforward. A meta-analysis by Li et al. (2022) found that while both UA and programming exercises are useful in cultivating CT skills, programming exercises tend to be more effective than unplugged activities, especially for older students. This suggests that while UA can be beneficial, their impact might vary based on the age group and specific context. Sun et al. (2021b) conducted a quasi-experimental study comparing UA with traditional instructor-directed methods and found that while UA had

positive outcomes, they did not significantly outperform traditional methods in all aspects of CT skill development. This suggests that while unplugged activities can be beneficial, their impact might vary based on implementation specifics and student demographics.

Moreover, Kim et al. (2021) found that the effectiveness of UA can be influenced by the specific context and duration of the intervention. Their large-scale study using a web-based platform for programming indicated that while students showed improvements, the gains were not uniformly significant across all metrics of CT skills. Similarly, a study by Xu et al. (2023) found that the duration and context of the intervention significantly impact the effectiveness of UA. Short-term interventions may not provide enough time for students to fully develop CT skills, whereas longer, more integrated approaches yield better results. This highlights the need for careful consideration of how and when to integrate unplugged activities with digital tools to maximize their educational value.

The findings of this research study align with the broader literature on the integration of UA and BBP. While the observed improvements in CT skills were not statistically significant, the evidence suggests potential benefits from combining these approaches. The effectiveness of such integrative strategies depends on several factors, including the quality of teacher engagement, the contextual adaptation of the curriculum, and the duration and implementation specifics of the intervention. Future research should continue to explore these variables to optimize the integration of UA and BBP in educational settings.

5.3 Gender Differences in CT Development with UA+BBP and BBP Only Approaches

The study examined whether gender differences influenced the effectiveness of BBP and UA+BBP on CT skills. Although males generally scored higher than females, these differences were not statistically significant. This finding aligns with broader research

indicating that gender alone does not predict CT success when educational strategies are appropriately applied (Yildiz Durak et al., 2023). These results suggest that both BBP and UA+BBP approaches can create an inclusive learning environment that benefits all students, regardless of gender.

The findings that both BBP and UA+BBP methods are equally effective for male and female students align with research by Bati (2022) and Malik and Coldwell-Neilson (2018), which supports the idea that gender has no significant impact on CT outcomes in education. Strategies are applied appropriately. This reinforces the idea that well-designed teaching practices can provide equal learning opportunities for all students. Furthermore, the pedagogical approach that physically contextualizes computing concepts, as discussed by Rubio et al. (2015), can equalize perceptions and learning outcomes across genders. By making programming concepts tangible and relatable, these strategies demystify the subject matter and make it accessible to a broader range of learners.

However, it is important to consider contrasting findings from other studies. Previous research indicates that gender can significantly influence students' confidence and self-efficacy toward CT skills, subsequently impacting their academic performance (Gao et al., 2023; Kemp et al., 2019; Wofford, 2021). Despite female students reportedly showing lower interest and self-efficacy in using computer tools (Stupurienė et al., 2022), the supportive structure of UA+BBP enabled them to achieve significant improvements. Studies emphasize the importance of integrating CT with interactive and gender-balanced activities to empower girls' learning effectively (Hiley et al., 2023; Simmonds et al., 2023). This highlights that, with the right support and opportunities, female students can overcome initial disadvantages and excel in CT skills.

Interestingly, the integration of UA with BBP seemed to foster a more inclusive learning environment, particularly benefiting female students. While female students had lower mean scores on both pre- and post-tests, they exhibited higher learning gains compared to male students, demonstrating a greater ability to improve when given the opportunity. This underscores the importance of providing supportive and encouraging learning environments to help all students realize their potential. Torres-Torres et al. (2019) found that female students showed significant gains in CT skills when engaged with UA+BBP, highlighting the potential of this integrated approach to bridge the gender gap in CT education.

Contrastingly, some studies present findings that do not align with this perspective. Sun et al. (2024) reported that unplugged programming tends to benefit males more in enhancing CT skills. Additionally, Kim et al. (2021) found that girls generally performed better than boys in using technology and solving coding problems in a large-scale assessment using a web-based platform, although boys performed exceptionally well on certain coding tasks. These different results suggest that the effectiveness of educational strategies may vary depending on the specific context and implementation. It is possible that certain factors in the learning environment or teaching methods have different effects on male and female students.

The lack of a significant interaction between gender and teaching method also suggests that other factors may influence students' performance in CT. For example, students' prior exposure to programming and CT concepts may have impacted their performance regardless of the teaching method used. Research by Bui et al. (2023) suggests that prior experience plays a crucial role in student outcomes. Additionally, fluctuations in instructional quality and teacher engagement can significantly influence student outcomes.

Holmes et al. (2022) and Fojcik et al. (2022) emphasize that effective education involves not only the delivery of content but also the active engagement of students in a way that is consistent with and promotes learning goals. Factors such as classroom environment, peer support, and access to resources may also have influenced students' learning experiences and outcomes.

5.4 Impact of UA+BBP versus BBP Only on Student Attitudes Toward Programming

The research findings indicate a significant positive impact of integrating UA with BBP on students' attitudes toward programming. The UA+BBP group achieved a significantly higher mean attitude than the BBP-only group. This difference suggests that combining UA with BBP promotes a more positive attitude toward programming. The positive attitude change can be attributed to several factors inherent in the UA+BBP approach. UA add a playful, interactive dimension to programming education and make abstract concepts more accessible. Grover et al. (2024) found that students who were involved in UA demonstrated significantly higher levels of engagement and enjoyment. This increased engagement is likely due to the tangible, hands-on nature of UA, which helps students grasp complex programming ideas more intuitively.

Physically manipulating concepts can demystify programming and reduce the intimidation factor often associated with learning to code. Moreover, UA contribute to reducing anxiety and improving understanding of programming concepts. Zhu et al. (2023) found that students who engaged in UA showed better conceptual understanding and experienced lower anxiety levels compared to their peers who only used digital tools. This reduction in anxiety is crucial, as it fosters a more relaxed and conducive learning environment, allowing students to explore and make mistakes without the fear of failure.

The UA+BBP approach also significantly boosts students' motivation. According to Jiang and Wong (2018), integrating UA with digital programming instruction led to a notable increase in students' enthusiasm and willingness to engage with programming. The variety of learning experiences provided by this combination not only makes learning more fun but also helps students see the relevance of what they are learning, which in turn fosters a more positive attitude towards programming. Given these positive impacts, it is evident that the UA+BBP approach creates a richer, more engaging learning environment. Rehmat et al. (2020) emphasized the importance of varied instructional strategies to promote CT and cultivate positive attitudes towards computer science. By incorporating UA into their programming curricula, educators can better support students in their journey to becoming proficient and confident programmers.

However, the research also highlights some contrasting findings. For instance, Rodriguez et al. (2016) found that while UA can improve conceptual understanding, it does not always lead to sustained improvements in students' attitudes toward programming. This suggests that initial enthusiasm for UA may wane over time, resulting in a neutral impact on long-term attitudes. Additionally, some students prefer the immediacy and interactivity of digital tools over UA. Research by Kanika et al. (2020) indicated that students in the BBP-only group reported higher levels of satisfaction due to the direct application of programming concepts in a digital format.

This preference underscores the need for educators to balance unplugged and digital activities to cater to different learning preferences. Practical challenges also arise in the implementation of UA. Huang and Looi (2021) highlighted issues such as the additional time required for setup and the need for specialized teacher training. These logistical hurdles can

affect the overall effectiveness of the UA+BBP approach, potentially leading to mixed results in student attitudes.

5.5 Gender-Based Attitudinal Differences in Response to UA+BBP and BBP Only Methods

In the BBP-only group, males demonstrated significantly higher attitudes toward programming compared to females. The gender differences in attitudes could be attributed to several factors, including societal stereotypes, differences in self-efficacy, and previous experiences with technology. Female students may find the BBP-only approach less appealing or more challenging, which could negatively impact their attitudes toward programming (Amnouychokanant et al., 2021).

The integration of UA with BBP revealed clear patterns in students' attitudes toward programming and revealed significant gender differences. For male students, the addition of UA did not significantly change their attitudes, consistent with previous research indicating stable engagement levels among men in traditional BBP environments (Polat and Yilmaz, 2022). However, the UA+BBP approach has significantly improved the attitudes of female students, making the learning process more interactive and less intimidating, thereby promoting a more inclusive environment (Li et al., 2024).

Interestingly, there was no significant difference in attitudes toward programming between male and female in the UA+BBP group. This lack of significant differences suggests that integrating UA with BBP creates a more equitable learning environment for both genders. The UA+BBP approach appears to be equally inclusive of both male and female students and promotes a positive attitude toward programming regardless of gender. This suggests that the interactive, hands-on nature of UA helps bridge the gender gap by providing an inclusive and supportive learning experience (Cheng et al., 2019).

These results suggest a significant interaction effect between teaching method and gender on attitudes toward programming. The UA+BBP teaching method significantly improves attitudes towards programming, especially among female students. This suggests that integrating UA and BBP is more effective in promoting positive attitudes toward programming, with the effect being more pronounced among female students. Supporting these findings, Paucar-Curasma et al. (2023) found that female students exhibited greater interest and confidence in programming when exposed to a blend of UA and BBP, compared to BBP alone.

This supports the notion that UA activities can demystify programming concepts and make them more accessible to female students. Contrastingly, a study by Hiley et al. (2023) noted that while UA can engage students initially, long-term interest and attitudes towards programming still require consistent support and reinforcement, indicating that UA alone might not sustain engagement without additional strategies. The research findings suggest that integrating UA with BBP can mitigate gender disparities in attitudes toward programming. This combined approach engages both male and female students more effectively, promoting positive attitudes and a more inclusive learning environment. Future research should explore long-term impacts and identify additional strategies to sustain engagement and positive attitudes.

5.6 Thematic-based Findings

The qualitative analysis of student interviews from the experimental group, which employed a combined instructional approach of UA and BBP, revealed key themes that resonate deeply with the quantitative data collected on CT test scores and attitudes toward programming.

5.6.1 Theme 1: Improving Programming Through Fun and Challenging Activities

The integration of UA and BBP played a crucial role in enhancing students' engagement and understanding of CT. Participant (P1)'s enthusiasm for the tower project, describing it as "an awesome game" that was "challenging and fun," underscores the importance of engaging students through enjoyable yet intellectually stimulating activities. This enjoyment and challenge were pivotal in heightening interest in CT, correlating with improved attitudes, especially among female students. Participant (P2)'s reflection on cup stacking, which emphasized the critical nature of following instructions akin to coding, illustrates how UA effectively bridged practical programming aspects with enjoyable learning experiences. Furthermore, Participant (P12)'s comment on teamwork in the tower project highlights the collaborative skills fostered through UA, essential for real-world programming environments. However, it is essential to critically consider whether such enjoyment translates to sustained interest in CT beyond the initial novelty of the activities (Bell and Vahrenhold, 2018).

5.6.2 Theme 2: Simplifying Complex Programming Concepts

UA and BBP significantly demystified complex programming concepts, making them more accessible and less abstract for students. For instance, Participant (P9) noted that cup stacking encouraged planning and simplified strategies, which directly improved their coding skills. This gradual introduction and scaffolding of programming principles allowed students to build their understanding incrementally, reducing the cognitive load and making learning less overwhelming (Fanchamps et al., 2024). Participant (P10) remark on creating simple, understandable code during the cup stacking activity reflects how UA can effectively simplify complex ideas. The significant improvement in attitudes, particularly among female students, as seen in the quantitative findings, suggests that UA and BBP together create a

more inclusive learning environment, though further investigation is needed to ensure these improvements are sustainable.

5.6.3 Theme 3: Foundation for Programming

The foundational approach provided by UA was instrumental in building students' confidence and understanding of programming basics. Participant (P1)'s comment on how activities without a computer made learning "less hard and less scary" points to the efficacy of UA in reducing intimidation associated with programming. This hands-on, tangible engagement with programming principles allowed students to grasp fundamental concepts like sequencing and conditional logic effectively. Participant (P3)'s observation that the cup stacking activity clarified how computers follow instructions further supports this. However, it is essential to consider whether this foundational understanding translates into a seamless transition to more complex programming tasks encountered in BBP and beyond (Li and Yang, 2023). The shift in attitudes towards programming, particularly the recognition of coding as an accessible and enjoyable activity, highlights a significant transformation in student perceptions, consistent with the improved attitudes observed in the UA+BBP group.

5.6.4 Theme 4: Changes in Attitude Towards Programming

The integration of UA and BBP led to a significant shift in students' attitudes towards programming. Initially, students described coding as hard, boring, and confusing. After engaging in UA and BBP activities, their perceptions shifted to viewing coding as fun, easy, and accessible. Participant (P4)'s transformation from finding coding hard and boring to easy and fun after participating in the Conditional Card Game highlights the impact of these activities. Similarly, Participant (P8)'s realization that coding was not just for geniuses but something anyone could do reflects a significant attitudinal shift. This change is consistent

with the improved attitudes observed in the UA+BBP group, suggesting that these activities effectively made coding accessible and enjoyable.

5.7 Implications

5.7.1 Theoretical Implications

5.7.1.1 Reinforcement of Constructivist Learning Theories

The study's approach reinforces constructivist theories, which suggest that learners construct knowledge through experiences (Parra and García, 2022). By combining UA (experiential learning) and BBP (interactive digital engagement), the study provides empirical support to the efficacy of hands-on and interactive learning in enhancing cognitive development. The findings could inspire further theoretical development on how digital and physical interactions influence learning processes, potentially leading to new models of integrated learning environments.

The integration of UA and BBP aligns with constructivist principles by emphasizing active, experiential learning. UA activities allow students to engage physically and conceptually with CT concepts, facilitating deeper understanding through direct manipulation and problem-solving. BBP complements this by offering an interactive digital platform where students can apply their learned concepts in a virtual environment, reinforcing and expanding their knowledge through practice and iteration.

5.7.1.2 Cognitive Load Theory

By analysing how students engage with combined UA and BBP, the study could provide new insights into cognitive load management in learning environments. Specifically, it could reveal how integrating different types of activities helps distribute cognitive resources more effectively, potentially enhancing learning outcomes without overwhelming students. This approach aligns with the concept of cognitive load theory, which emphasizes

the importance of balancing the mental effort required by different tasks to optimize learning (Koć-Januchta et al., 2022). According to Chookhampaeng et al. (2023) different activities, when combined thoughtfully can complement each other in a way that reduces overall cognitive load. For example, UA might introduce foundational concepts in a simpler, more tangible format, allowing students to grasp the basics without the additional complexity of a programming environment.

Once these concepts are understood, students can then apply them in a more abstract setting, like BBP, with greater ease and confidence. This staggered approach can prevent cognitive overload, wherein too much information or too many complex tasks are presented simultaneously, which can hinder the learning process (Kossen and Ooi, 2021). Therefore, by examining the engagement patterns and outcomes associated with the UA and BBP combination, the study may highlight effective strategies for designing instructional materials and activities that facilitate better learning experiences.

5.7.2 Practical Implications

5.7.2.1 Curriculum Design

Based on the study's findings, education policymakers and curriculum developers can design modular activities that seamlessly integrate UA and BBP. This could encourage broader adoption of CT across various subjects, making it a core component of general education. The study supports the development of teaching strategies that cater to diverse learning styles, advocating for personalized learning paths within the curriculum that allow students to engage with material in ways that best suit their learning preferences (Ahmed, 2019).

5.7.2.2 Teacher Training and Professional Development

The outcomes highlight the need for specialized training programs that equip teachers with the skills to effectively deliver integrated CT education (Çakir, 2021; Humble and Mozelius, 2023). Professional development should include practical workshops on managing and facilitating both UA and BBP. There's a need for creating comprehensive resource kits that teachers can use to implement the studied methods, including lesson plans, activity guides, and assessment tools.

5.7.2.3 Resource Allocation and Infrastructure

Schools may need to invest in technology that supports BBP, such as computers, software like Scratch, and reliable internet access. Simultaneously, resources for UA (physical materials for activities) must be made available. Establishing online platforms and professional learning communities where educators can share experiences, resources, and best practices for integrated CT education is crucial. Cleveland-Innes and Hawryluk (2022) highlights the value of creating a community of inquiry by detailing the three foundational presences (social, cognitive, and teaching) and emphasizing their significance in fostering a collaborative and constructivist learning environment essential for deep and meaningful learning.

5.7.2.4 Community and Parental Engagement

The research underscores the importance of community and parental involvement in educational processes. Schools could initiate awareness campaigns to educate parents and the community about the value of CT and the specific benefits of integrating UA with BBP. Programs that involve parents and community members in CT activities could enhance support for CT education and provide additional resources for learning. Increased community and parental engagement can lead to more robust support for CT education. When parents and community members understand the importance of CT, they are more

likely to advocate for necessary resources, such as updated technology and professional development for teachers (Madhu Govind et al., 2020). This support can help create a more enriching educational environment for students.

5.7.2.5 Evaluation and Continuous Improvement

The establishment of robust feedback mechanisms to continually assess the effectiveness of the integrated teaching approach. Regular surveys and interviews with students and teachers provide valuable insights into the strengths and areas for improvement of the UA+BBP approach (Habib et al., 2021). These tools help assess student engagement, understanding of CT concepts, and attitudes toward programming. Regular performance assessments, including coding tasks, problem-solving exercises, and quizzes, measure students' progress and understanding of CT skills (Ogegbo and Ramnarain, 2022; Emily Relkin et al., 2020). Tracking these metrics over time helps identify trends and inform instructional adjustments.

Schools should adopt adaptive teaching strategies based on feedback from surveys, interviews, and assessments. This responsiveness ensures that teaching methods can be tailored to address student needs and improve learning outcomes. Integrating the latest educational research and best practices into the curriculum helps keep CT education current and effective. This approach ensures that teaching methods evolve with new insights and technological advancements.

5.8 Limitation

5.8.1 Sample Specificity

The study's findings are based on a relatively narrow demographic: 13-year-old students from three fully boarding schools in Kedah, Malaysia. This specificity might not represent the broader student population across different regions or educational systems,

such as non-boarding schools, urban or rural schools, potentially affecting the applicability of the results in varied settings. Due to the unique characteristics of the sample, the generalizability of the findings to other student populations in different geographical locations or educational settings is limited. Given that the findings derive from a distinct group of boarding school students within a particular setting, they may not directly translate to different educational environments or more diverse student populations (Shemshack and Spector, 2020).

5.8.2 Measurement Tools

The CT test used might predominantly measure lower-level cognitive skills within the CT spectrum, such as recognition and understanding, rather than higher-order thinking skills like analysis and synthesis. The developers of the test, Roman-Gonzalez et al. (2017), pointed out that the CT test “might be measuring CT at its lower cognitive complexity levels (‘recognize’ and ‘understand’). Therefore, this study might not have assessed students’ CT concepts and practices at higher levels of complexity. The Likert-style survey may introduce biases such as social desirability or response bias. The format might also oversimplify complex student experiences, failing to capture the depth of students’ attitudes and perceptions effectively.

5.8.3 Data Collection and Analysis

The voluntary nature of participation in interviews might attract students who have more positive experiences with the intervention, potentially leading to biased qualitative data that does not represent all students’ views. The statistical and thematic methods used may not adequately address the complexities of the data or capture interactions between different variables, which could overlook significant insights or subtleties in student responses. A further methodological limitation of the study is the decision to conduct interviews

exclusively with students from the UA+BBP group. This choice lacks a robust justification and could introduce a bias in understanding the full impact of the instructional methods. By not interviewing students from the BBP only group, the study potentially overlooks comparative insights that could further elucidate the differences or similarities in student experiences across the two teaching methods (Costley, 2021).

5.8.4 Implementation Factors

Factors such as the instructor's proficiency familiarity and competence with the materials, students' prior exposure to similar content, and classroom dynamics. Variations in these factors could yield different results, highlighting the need for standardized implementation protocols. The absence of a control group that did not receive any intervention also makes it challenging to determine whether the observed improvements were directly attributable to the intervention or if they could be influenced by external factors or natural maturation. The study does not fully account for potential variations in teaching quality or how consistently the methods are applied in different classroom settings (Tichavsky et al., 2015).

5.9 Summary

This chapter discussed the results of each research question and provided explanations for the findings. The implications of these results were explored in both theoretical and practical contexts. Additionally, the limitations of the research were addressed, focusing on sample specificity, measurement tools, data collection and analysis, and implementation factors.

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Overview

This chapter presents the suggestions for future research and conclusions drawn from the study. The findings highlight the importance of integrating UA with BBP as an effective instructional method. This combined approach not only improves students' achievement in CT but also significantly enhances their attitudes toward programming. The chapter concludes by emphasizing the potential of UA+BBP to serve as a valuable alternative teaching strategy in the field of programming education.

6.2 Conclusion

This study investigated the effects of integrating UA with BBP on students' CT skills and attitudes toward programming, with a particular focus on gender differences. The research compared the BBP-only instructional method with the combined UA+BBP instructional method, examining how these approaches influenced students' CT skills and attitudes.

6.2.1 Objective 1

The study revealed that while the integration of UA with BBP resulted in a numerical improvement in CT test scores, this increase was not statistically significant when compared to the BBP-only method. This indicates that although UA activities provide valuable experiential learning opportunities, they may not significantly enhance CT skills beyond what is achieved through BBP alone within the limited timeframe of the study. This finding suggests that while UA activities are beneficial in creating an engaging and interactive

learning environment, their impact on CT skill development might require a more extended period or more frequent integration to show significant results.

6.2.2 Objective 2

The research indicated no significant differences in CT skills based on gender when comparing the UA+BBP method to the BBP-only method. Both male and female students exhibited similar levels of improvement in CT test scores, implying that gender did not significantly affect the effectiveness of integrating UA with BBP in enhancing CT skills. This finding suggests that the UA+BBP approach is equally effective for both genders in developing computational thinking skills, highlighting the inclusivity of this educational strategy.

6.2.3 Objective 3

The UA+BBP method significantly improved students' attitudes toward programming, especially among female students. Female students in the UA+BBP group reported higher levels of engagement, interest, and confidence in programming compared to those in the BBP-only group. This suggests that integrating unplugged, hands-on activities with BBP can make programming more appealing and less intimidating, particularly for girls who are often underrepresented in STEM fields. The positive impact on attitudes highlights the potential of the UA+BBP approach to foster a more inclusive and supportive learning environment that encourages all students to pursue programming.

6.2.4 Objective 4

The study highlighted significant gender differences in attitudes toward programming when comparing the UA+BBP method to the BBP-only method. Female students showed a more substantial improvement in their attitudes toward programming in

the UA+BBP group compared to the BBP-only group. The thematic analysis of student interviews further supports these findings. Students found the UA+BBP approach both enjoyable and educational, appreciating the interactive and collaborative nature of the activities, which helped them understand complex programming concepts more easily. This feedback underscores the value of incorporating diverse teaching methods to cater to different learning preferences and needs.

6.3 Future Work

6.3.1 Longitudinal Studies on CT Skills and Academic Outcomes

To deepen our understanding of the integration of UA with BBP, future research should include longitudinal studies to evaluate the long-term effects on students' CT skills. These studies are essential for determining whether the improvements observed during the intervention are maintained over time and how they influence students' future academic and career choices. Moreover, analysing longitudinal data could provide insights into how continuous exposure to UA and BBP shapes students' problem-solving behaviours and learning habits.

6.3.2 Expanding Research to Diverse Educational Contexts

Expanding the research to diverse cultural and geographic contexts is also critical. This expansion would help validate the effectiveness of the UA+BBP approach across various educational systems and cultural backgrounds. By investigating the approach's impact in different educational settings such as public and private schools, urban and rural areas, and across different age groups researchers can gain a comprehensive view of its applicability and identify necessary adjustments for different environments.

6.3.3 Evaluating Teacher Efficacy and Attitude Towards UA+BBP

An important aspect of future work involves evaluating teachers' self-efficacy and attitudes towards the UA+BBP pedagogy. Understanding these elements can pinpoint the support and resources teachers need to implement these innovative teaching methods effectively. Additionally, the design and testing of comprehensive teacher training programs that tackle the specific challenges and opportunities of integrating UA with BBP would provide valuable insights. Analysing the outcomes of these programs will offer strategies that could enhance teacher preparedness and overall educational impact.

6.3.4 Developing Multidisciplinary CT Curriculum Designs

Furthermore, developing and testing various curriculum designs that integrate UA and BBP in subjects beyond computing, like mathematics, science, and languages, would promote a multidisciplinary approach to CT education. Innovating and evaluating new assessment techniques that accurately measure the depth of CT skills, particularly focusing on higher-order cognitive skills, is equally important. Integrating cutting-edge technologies such as virtual reality (VR) and augmented reality (AR) within the UA+BBP framework could further enhance the learning experience and student engagement. Exploring adaptive learning technologies that tailor the educational content to the individual pace and needs of students could also increase the efficacy of these interventions. Such technological advancements promise to make learning more dynamic and responsive to student needs.

6.3.5 Gender-Specific Responses and Inclusivity in STEM Education

Given that female students demonstrated significant increases in positive attitudes toward programming, future studies should explore gender-specific responses to the UA+BBP integration. This research could tailor interventions to promote gender equity in

STEM fields. Additionally, examining how the UA+BBP integration supports learners with diverse educational needs, including those with learning disabilities, is crucial for ensuring inclusivity in CT education.

REFERENCES

- Ahmed, D. T. (2019). Which styles of teaching and learning are effective for students? – students’ perspective. *International Conference on Computational Science and Computational Intelligence (CSCI)*, 880–883.
- Aho, A. V. (2012). Computation and computational thinking. *Computer Journal*, 55(7), 833–835.
- Aivaloglou, E., & Hermans, F. (2016). How kids code and how we know: An exploratory study on the scratch repository. *ICER 2016 - Proceedings of the 2016 ACM Conference on International Computing Education Research*, 53–61.
- Akiba, D. (2022). Computational thinking and coding for young children: a hybrid approach to link unplugged and plugged activities. *Education Sciences*, 12(11).
- Alp, G., & Bulunuz, N. (2023). Effect of web-based collaborative learning method with scratch on critical thinking skills of 5th grade students. *Participatory Educational Research*, 10(2), 82–104.
- Amnouychokanant, V., Boonlue, S., Chuathong, S., & Thamwipat, K. (2021). A study of first-year students’ attitudes toward programming in the innovation in educational technology course. *Education Research International*, 2021, 9105342.
- Andriyani. (2023). Stimulation of cognitive and psychomotor capability by game-based learning with computational thinking core. *Mathematics Teaching-Research Journal*, 15(5), 76 – 104.
- Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016). A K-6 computational thinking curriculum framework: Implications for teacher

- knowledge. *Journal of Educational Technology & Society*, 19(3), 47–57.
- Ardito, G., Czerkowski, B., & Scollins, L. (2020). Learning computational thinking together: effects of gender differences in collaborative middle school robotics program. *TechTrends*, 64(3), 373–387.
- Armoni, M., Meerbaum-Salant, O., & Ben-Ari, M. (2015). From scratch to “Real” programming. *ACM Transactions on Computing Education*, 14(4), 1–13.
- Attia, N. A., Baig, L., Marzouk, Y. I., & Khan, A. (2017). The potential effect of technology and distractions on undergraduate students’ concentration. *Pakistan Journal of Medical Sciences*, 33(4), 860–865.
- Atun, H., & Usta, E. (2019). The effects of programming education planned with TPACK framework on learning outcomes. *Participatory Educational Research*, 6(2), 26–36.
- Avcı, C., & Deniz, M. N. (2022). Computational thinking: early childhood teachers’ and prospective teachers’ preconceptions and self-efficacy. *Education and Information Technologies*, 27(8), 11689–11713.
- Ayuso, A., Merayo, N., Ruiz, I., & Fernández, P. (2022). Challenges of STEM vocations in secondary education. *IEEE Transactions on Education*, 65(4), 713–724.
- Barcelos, T. S., Munoz, R., Villarroel, R., Merino, E., & Silveira, I. F. (2018). Mathematics learning through computational thinking activities: A systematic literature review. *Journal of Universal Computer Science*, 24(7), 815–845.
- 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.

- Bass, L. R. (2014). Boarding schools and capital benefits: implications for urban school reform. *The Journal of Educational Research*, 107(1), 16–35.
- Bati, K. (2022). A systematic literature review regarding computational thinking and programming in early childhood education. *Education and Information Technologies*, 27(2), 2059–2082.
- Battal, A., Adanır, G. A., & Gülbahar, Y. (2021). Computer science unplugged: a systematic literature review. *Journal of Educational Technology Systems*, 50(1), 24–47.
- Bau, D., Gray, J., Kelleher, C., Sheldon, J., & Turbak, F. (2017). Learnable programming: blocks and beyond. *Commun. ACM*, 60(6), 72–80.
- Bell, J., & Bell, T. (2018). Integrating computational thinking with a music education context. *Informatics in Education*, 17(2), 151–166.
- Bell, T., & Vahrenhold, J. (2018). CS unplugged—How is it used, and does it work? In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Vol. 11011 LNCS*. Springer International Publishing. https://doi.org/10.1007/978-3-319-98355-4_29
- Bers, M. U., Flannery, L., Kazakoff, E. R., Sullivan, A., Kazakoff Myers, E., & Sullivan, A. (2014). Computational thinking and tinkering: Exploration of an early childhood robotics curriculum. *Computers & Education*, 72, 145–157.
- Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. *Proceedings of the 12th Workshop on Primary and Secondary Computing Education*, 65–72.
- Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the

- development of computational thinking. *Proceedings of the 2012 Annual Meeting of the American Educational Research Association, Vancouver, Canada, 1*, 25.
- Budinski, N., Lavicza, Z., Fenyvesi, K., Novta, M., & Budinski, N. (2019). Mathematical and coding lessons based on creative origami activities. *Open Education Studies, 1*(1), 220–227.
- Bui, G., Sibia, N., Zavaleta Bernuy, A., Liut, M., & Petersen, A. (2023). Prior programming experience: a persistent performance gap in CS1 and CS2. *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, 889–895.
- Caeli, E. N., & Yadav, A. (2020). Unplugged Approaches to computational thinking: a historical perspective. *TechTrends, 64*(1), 29–36.
- Çakir, R. (2021). Computational thinking skills of turkish and indian teacher candidates: a comparative study. *International Journal of Psychology and Educational Studies, 8*(1), 24–37.
- Çakiroğlu, Ü., Çevik, İ. I., Çakiroğlu, Ü., & Çevik, İ. I. (2022). A framework for measuring abstraction as a sub-skill of computational thinking in block-based programming environments. *Education and Information Technologies, 27*(7), 9455–9484.
- Calder, N. (2018). Using Scratch to facilitate mathematical thinking. *Waikato Journal of Education, 18*(2).
- Çalışkan, E. (2020). The effects of robotics programming on secondary school students' problem-solving skills. *World Journal on Educational Technology: Current Issues, 12*(4), 217–230.
- Campos, N., Nogal, M., Caliz, C., & Juan, A. A. (2020). Simulation-based education involving online and on-campus models in different European universities.

- International Journal of Educational Technology in Higher Education*, 17(1), 8, 1-15.
- Castle, S. D. (2023). Leveraging computational science students' coding strengths for mathematics learning. *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1*, 263–269.
- Chan, S.-W., Looi, C.-K., Ho, W. K., & Kim, M. S. (2023). Tools and approaches for integrating computational thinking and mathematics: a scoping review of current empirical studies. *Journal of Educational Computing Research*, 60(8), 2036–2080.
- Cheng, L., Wang, X., & Ritzhaupt, A. D. (2023). The effects of computational thinking integration in STEM on students' learning performance in k-12 education: A Meta-analysis. *Journal of Educational Computing Research*, 61(2), 416–443.
- Cheng, S., Chauhan, B., & Chintala, S. (2019). The rise of programming and the stalled gender revolution. *Sociological Science*, 6(13), 321–351.
- Cheng, Y.-P., Lai, C.-F., Chen, Y.-T., Wang, W.-S., Huang, Y.-M., & Wu, T.-T. (2023). Enhancing student's computational thinking skills with student-generated questions strategy in a game-based learning platform. *Computers & Education*, 200, 104794.
- Chookhampaeng, C., Kamha, C., & Chookhampaeng, S. (2023). Problems and needs assessment to learning management of computational thinking of teachers at the lower secondary level. *Journal of Curriculum and Teaching*, 12(3), 172–178.
- Christensen, D., & Lombardi, D. (2020). Understanding biological evolution through computational thinking. *Science & Education*, 29(4), 1035–1077.
- Chun, S. J., Jo, Y., & Lee, S. (2021). The effect of programming classes with tangible scratch blocks on the programming interest of 6th grade elementary school students. *International Journal of Information and Education Technology*, 11(9), 405–409.

- Cleveland-Innes, M., & Hawryluk, J. (2022). Designing online learning communities. In *Handbook of Open, Distance and Digital Education* (pp. 1–17). Springer Nature Singapore.
- Coşkunserçe, O. (2023). Comparing the use of block-based and robot programming in introductory programming education: Effects on perceptions of programming self-efficacy. *Computer Applications in Engineering Education*, 31(5), 1234–1255.
- Costa Junior, A. de O., Guedes, E. B., Lima e Silva, J. P. F., & Rivera, J. A. (2024). Developing computational thinking in middle school with an educational robotics resource. *Journal of Intelligent & Robotic Systems*, 110(2), 49.
- Costley, J. (2021). How role-taking in a group-work setting affects the relationship between the amount of collaboration and germane cognitive load. *International Journal of Educational Technology in Higher Education*, 18(1), 18-24.
- Critten, V., Hagon, H., & Messer, D. (2022). Can Pre-school children learn programming and coding through guided play activities? A case study in computational thinking. *Early Childhood Education Journal*, 50(6), 969–981.
- Cui, Z., Ng, O., & Jong, M. S.-Y. (2023). Integration of computational thinking with mathematical problem-based learning: Insights on affordances for learning. *Educational Technology & Society*, 26(2), 131-146.
- Curzon, P. (2013). cs4fn and computational thinking unplugged. *Proceedings of the 8th Workshop in Primary and Secondary Computing Education*, 47–50.
- Curzon, P., & McOwan, P. W. (2008). Engaging with computer science through magic shows. *Proceedings of the 13th Annual Conference on Innovation and Technology in Computer Science Education*, 40(3), 179–183.

- Da Silva Pereira, E., & Lopes, L. A. (2020). Electronic game creation through scratch software: Creative and collaborative learning fostering STEAM practices. *Acta Scientiae*, 22(3), 28–46.
- DeJong, S. (2023). Making the Digital Tangible Through Analog Games: Design Retrospective of Digital Literacy Games. In X. Fang (Ed.), *HCI in Games* (pp. 66–80). Springer Nature Switzerland.
- del Olmo-Muñoz, J., Cózar-Gutiérrez, R., & González-Calero, J. A. (2020). Computational thinking through unplugged activities in early years of Primary Education. *Computers & Education*, 150(February), 103832.
- del Rey, Y. A., Cawanga Cambinda, I. N., Deco, C., Bender, C., Avello-Martínez, R., & Villalba-Condori, K. O. (2021). Developing computational thinking with a module of solved problems. *Computer Applications in Engineering Education*, 29(3), 506–516.
- Delacre, M., Leys, C., Mora, Y. L., & Lakens, D. (2019). Taking parametric assumptions seriously: Arguments for the use of Welch’s F-test instead of the classical F-test in one-way ANOVA. *International Review of Social Psychology*, 32(1), 13.
- Denning, P. J., & Tedre, M. (2021). Computational thinking: a disciplinary perspective. *Informatics in Education*, 20(3), 361–390.
- Diago, P. D., González-Calero, J. A., & Yáñez, D. F. (2022). Exploring the development of mental rotation and computational skills in elementary students through educational robotics. *International Journal of Child-Computer Interaction*, 32, 100388.
- Erümit, A. K. (2020). Effects of different teaching approaches on programming skills. *Education and Information Technologies*, 25(2), 1013–1037.
- Estevez, J., Garate, G., & Grana, M. (2019). Gentle introduction to artificial intelligence for

- high-school students using scratch. *IEEE Access*, 7, 179027–179036.
- Ezeamuzie, N. O. (2023). Abstractive-based programming approach to computational thinking: discover, extract, create, and assemble. *Journal of Educational Computing Research*, 61(3), 605–638.
- Ezeamuzie, N. O., Leung, J. S. C., Garcia, R. C. C., & Ting, F. S. T. (2022). Discovering computational thinking in everyday problem solving: A multiple case study of route planning. *Journal of Computer Assisted Learning*, 38(6), 1779–1796.
- Fagerlund, J., Häkkinen, P., Vesisenaho, M., & Viiri, J. (2021). Computational thinking in programming with Scratch in primary schools: A systematic review. *Computer Applications in Engineering Education*, 29(1), 12–28.
- Fanchamps, N., van Gool, E., Slangen, L., & Hennissen, P. (2024). The effect on computational thinking and identified learning aspects: Comparing unplugged smartGames with SRA-Programming with tangible or On-screen output. *Education and Information Technologies*, 29(3), 2999–3024.
- Ferguson-Patrick, K., Ruth, R., & Macqueen, S. (2018). Integrating curriculum: a case study of teaching Global Education. *European Journal of Teacher Education*, 41(2), 187–201.
- Fidai, A., Capraro, M. M., & Capraro, R. M. (2020). “Scratch”-ing computational thinking with Arduino: A meta-analysis. *Thinking Skills and Creativity*, 38, 100726.
- Field, A. (2013). *Discovering statistics using IBM SPSS statistics*. sage.
- Fields, D. A., Kafai, Y. B., & Giang, M. T. (2017). Youth computational participation in the wild: understanding experience and equity in participating and programming in the online scratch community. *ACM Trans. Comput. Educ.*, 17(3), 1–22.

- Flórez, F. B., 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), 834–860.
- Fojcik, M., Fojcik, M. K., Høyland, S.-O., & Hoem, J. Ø. (2022). Challenges in teaching programming. *International Conference on Education and New Developments*, 160–163.
- 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), 5,1-17.
- Gao, H., Hasenbein, L., Bozkir, E., Göllner, R., & Kasneci, E. (2023). Exploring gender differences in computational thinking learning in a VR classroom: Developing machine learning models using Eye-Tracking data and explaining the models. *International Journal of Artificial Intelligence in Education*, 33(4), 929–954.
- Gao, X., & Hew, K. F. (2019). Toward a 5E-Based flipped classroom model for teaching computational thinking in elementary school: Effects on student computational thinking and problem- solving performance. *Journal of Educational Computing Research*, 9(6), 260–272.
- García-Peñalvo, F. J., Conde, M. Á., Gonçalves, J., & Lima, J. (2020). Advances in Computational thinking and robotics in education. *ACM International Conference Proceeding Series*, 10–13.
- Garneli, V., & Chorianopoulos, K. (2018). Programming video games and simulations in science education: exploring computational thinking through code analysis. *Interactive Learning Environments*, 26(3), 386–401.

- Geng, Z. (2023). Environmental design as a component of block-based programming. *Computer Applications in Engineering Education*, 31(2), 408–420.
- Grace, M., Campagnolo, P. D. B., Alves, I. M. D. R., De Borba, G. S., & Seemiller, C. (2020). What makes learning enjoyable Perspectives of today s college students in the U.S. and Brazil. *Journal of Pedagogical Research*, 5(1), 1–17.
- Gresse von Wangenheim, C., Hauck, J. C. R., Pacheco, F. S., & Bertonceli Bueno, M. F. (2021). Visual tools for teaching machine learning in K-12: A ten-year systematic mapping. *Education and Information Technologies*, 26(5), 5733–5778.
- Grover, S., Basu, S., Bienkowski, M., Eagle, M., Diana, N., & 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), 1–25.
- Grover, S., Jean, D., Broll, B., Cateté, V., Gransbury, I., Ledeczi, A., & Barnes, T. (2024). Design of tools and learning environments for equitable computer Science+ Data Science Education. In *Improving Equity in Data Science* (pp. 57–91). Routledge.
- Grover, S., & Pea, R. (2013). Computational Thinking in K-12: A review of the state of the field. *Educational Researcher*, 42(1), 38–43.
- Habib, M. A., Raja-Yusof, R. J., Salim, S. S., Sani, A. A., Sofian, H., & Abu Bakar, A. (2021). Analyzing students’ experience in programming with computational thinking through competitive, physical, and tactile games: The quadrilateral method approach. *Turkish Journal of Electrical Engineering and Computer Sciences*, 25(9), 2280–2297.
- Hashim, N. H. N., & Husnin, H. (2019). Teacher’s perception on computational thinking concept. *International Journal of Academic Research in Business and Social Sciences*,

9(11), 1536–1546.

- Herrero-Álvarez, R., Miranda, G., León, C., & Segredo, E. (2023). Engaging Primary and Secondary School Students in Computer Science Through Computational Thinking Training. *IEEE Transactions on Emerging Topics in Computing*, 11(1), 56–69.
- Hickmott, D., Prieto-Rodriguez, E., & Holmes, K. (2018). A scoping review of studies on computational thinking in K–12 mathematics classrooms. *Digital Experiences in Mathematics Education*, 4(1), 48–69.
- Hiley, K., Cebolla, H., & Elshehaly, M. (2023). The impact of non-formal computer science outreach on computational thinking in young women. *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 2*, 642–653.
- Holmes, S. R., Reinke, W. M., Herman, K. C., & David, K. (2022). An examination of teacher engagement in intervention training and sustained intervention implementation. *School Mental Health*, 14(1), 63–72.
- Hsiao, H. S., Lin, Y. W., Lin, K. Y., Lin, C. Y., Chen, J. H., & Chen, J. C. (2022). Using robot-based practices to develop an activity that incorporated the 6E model to improve elementary school students' learning performances. *Interactive Learning Environments*, 30(1), 85–99.
- Hsu, T.-C., Abelson, H., Patton, E., Chen, S.-C., & Chang, H.-N. (2021). Self-efficacy and behavior patterns of learners using a real-time collaboration system developed for group programming. *International Journal of Computer-Supported Collaborative Learning*, 16(4), 559–582.
- 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 and Education*, 126(July), 296–310.
- Hu, L. (2024). Exploring gender differences in computational thinking among K-12 students: A meta-analysis investigating influential factors. *Journal of Educational Computing Research*, 63(4), 60-75.
- Huang, J., & Parker, M. C. (2023). Developing computational thinking collaboratively: the nexus of computational practices within small groups. *Computer Science Education*, 33(3), 342–374.
- Huang, W., & Looi, C.-K. (2021). A critical review of literature on “unplugged” pedagogies in K-12 computer science and computational thinking education. *Computer Science Education*, 31(1), 83–111.
- Humble, N., & Mozelius, P. (2023). Grades 7–12 teachers’ perception of computational thinking for mathematics and technology. *Frontiers in Education*, 8(March).
- Hurt, T., Greenwald, E., Allan, S., Cannady, M. A., Krakowski, A., Brodsky, L., Collins, M. A., Montgomery, R., & Dorph, R. (2023). The computational thinking for science (CT-S) framework: operationalizing CT-S for K–12 science education researchers and educators. *International Journal of STEM Education*, 10(1), 1-16.
- Hutchins, N. M., Biswas, G., Maróti, M., Ledeczi, A., Grover, S., Wolf, R., Blair, K. P., Chin, D., Conlin, L., Basu, S., McElhaney, K., Lédeczi, Á., Grover, S., Wolf, R., Blair, K. P., Chin, D., Conlin, L., Basu, S., & McElhaney, K. (2020). C2STEM: A system for synergistic learning of physics and computational thinking. *Journal of Science Education and Technology*, 29, 83–100.
- Hwang, G.-J., & Tu, Y.-F. (2021). Roles and research trends of artificial intelligence in

- mathematics education: A bibliometric mapping analysis and systematic review. *Mathematics*, 9(6), 584,1-19.
- ISTE, Is., & CSTA, C. (2011). Operational definition of computational thinking for K-12 education. National Science Foundation, Arlington, VA.
- Jawawi, D. N. A., Jamal, N. N., Halim, S. A., Sa'adon, N. A., Mamat, R., Isa, M. A., Mohamad, R., & Hamed, H. N. A. (2022). Nurturing secondary school student computational thinking through educational robotics. *International Journal of Emerging Technologies in Learning (IJET)*, 17(03), 117–128.
- Jiang, B., & Li, Z. (2021). Effect of Scratch on computational thinking skills of Chinese primary school students. *Journal of Computers in Education*, 8(4), 505–525.
- Jiang, S., & Wong, G. K. W. (2018). Are children more motivated with plugged or unplugged approach to computational thinking?. *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, 1094-1099.
- Jody Clarke-Midura Deborah Silvis, J. F. S. V. R. L., & Kozlowski, J. S. (2021). Developing a kindergarten computational thinking assessment using evidence-centered design: the case of algorithmic thinking. *Computer Science Education*, 31(2), 117–140.
- Juškevičienė, A., Stupurienė, G., & Jevsikova, T. (2021). Computational thinking development through physical computing activities in STEAM education. *Computer Applications in Engineering Education*, 29(1), 175–190.
- Kabak, K., & Korucu, A. T. (2021). The effect of students' developing their own digital games on their academic achievement and attitudes towards for english lessons. *Participatory Educational Research*, 8(2), 74–93.
- Kafai, Y. B., & Proctor, C. (2022). A revaluation of computational thinking in K–12

- education: Moving toward computational literacies. *Educational Researcher*, 51(2), 146–151.
- Kalelioglu, F., & Sentance, S. (2020). Teaching with physical computing in school: the case of the micro:bit. *Education and Information Technologies*, 25(4), 2577–2603.
- Kanika, Chakraverty, S., & Chakraborty, P. (2020). Tools and techniques for teaching computer programming: A review. *Journal of Educational Technology Systems*, 49(2), 170–198.
- Karaahmetoğlu, K., & Korkmaz, Ö. (2019). The effect of project-based arduino educational robot applications on students' computational thinking skills and their perception of basic stem skill levels. *Participatory Educational Research*, 6(2), 1–14.
- Kazimoglu, C. (2020). Enhancing confidence in using computational thinking skills via playing a serious game: A case study to increase motivation in learning computer programming. *IEEE Access*, 8, 221831–221851.
- Keith, T. Z. (2019). *Multiple regression and beyond: An introduction to multiple regression and structural equation modeling*. Routledge.
- Kemp, P. E. J., Wong, B., & Berry, M. G. (2019). Female performance and participation in computer science: A national picture. *ACM Trans. Comput. Educ.*, 20(1), 1–28.
- Kim, H. S. A. N. S., Kim, S., Na, W., & Lee, W. J. E. E. (2021). Extending computational thinking into information and communication technology literacy measurement: gender and grade issues. *ACM Trans. Comput. Educ.*, 21(1(5)), 1–25.
- Kırçali, A. Ç., & Özdenir, N. (2022). A comparison of plugged and unplugged tools in teaching algorithms at the K-12 level for computational thinking skills. *Technology, Knowledge and Learning*, 28, 1485–1513.

- Kite, V., & Park, S. (2023). What's computational thinking?: Secondary science teachers' conceptualizations of computational thinking (CT) and perceived barriers to ct integration. *Journal of Science Teacher Education*, 34(4), 391–414.
- Kite, V., & Park, S. (2024). Context matters: Secondary science teachers' integration of process-based, unplugged computational thinking into science curriculum. *Journal of Research in Science Teaching*, 61(1), 203–227.
- Koć-Januchta, M. M., Schönborn, K. J., Roehrig, C., Chaudhri, V. K., Tibell, L. A. E., & Heller, H. C. (2022). “Connecting concepts helps put main ideas together”: cognitive load and usability in learning biology with an AI-enriched textbook. *International Journal of Educational Technology in Higher Education*, 19(1), 11,1-22.
- Kong, S.-C., & Wang, Y.-Q. (2019). Assessing perceptions of programming education among P-12 school teachers and principals: A multigroup invariance analysis. *Journal of Psychoeducational Assessment*, 37(6), 718–729.
- Kossen, C., & Ooi, C.-Y. (2021). Trialling micro-learning design to increase engagement in online courses. *Asian Association of Open Universities Journal*, 16(3), 299–310.
- 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.
- Kwon, K., & Cheon, J. (2019). Exploring problem decomposition and program development through block-based programs. *International Journal of Computer Science Education in Schools*, 3(1), 3–16.
- Lai, X., Ye, J., & Wong, G. K. W. (2023). Effectiveness of collaboration in developing computational thinking skills: A systematic review of social cognitive factors. *Journal*

of Computer Assisted Learning, 39(5), 1418–1435.

Leavy, A., Dick, L., Meletiou-Mavrotheris, M., Paparistodemou, E., & Stylianou, E. (2023).

The prevalence and use of emerging technologies in STEAM education: A systematic review of the literature. *Journal of Computer Assisted Learning*, 39(4), 1061–1082.

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, 32–37.

Li, F., Wang, X., He, X., Cheng, L., & Wang, Y. (2022). The effectiveness of unplugged

activities and programming exercises in computational thinking education: A Meta-analysis. *Education and Information Technologies*, 27, 7993–8013.

Li, Q., Jiang, Q., Liang, J.-C., Xiong, W., & Zhao, W. (2024). Engagement predicts

computational thinking skills in unplugged activity: Analysis of gender differences. *Thinking Skills and Creativity*, 52, 101537.

Li, W., & Yang, W. (2023). Promoting children's computational thinking: A quasi-

experimental study of web-mediated parent education. *Journal of Computer Assisted Learning*, 39(5), 1564–1575.

Li, X., Xie, K., Vongkulluksn, V., Stein, D., & Zhang, Y. (2023). Developing and testing a

design-based learning approach to enhance elementary students' self-perceived computational thinking. *Journal of Research on Technology in Education*, 55(2), 344 – 368.

Lindstrom, D. L., Schmidt-Crawford, D. A., & Thompson, A. D. (2019). Computational

thinking in content areas and feminine craft. *Journal of Digital Learning in Teacher Education*, 35(3), 126–127.

- Liu, J., Li, Q., Sun, X., Zhu, Z., & Xu, Y. (2023). Factors influencing programming self-efficacy: an empirical study in the context of Mainland China. *Asia Pacific Journal of Education*, 43(3), 835–849.
- 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.
- Madariaga, L., Allendes, C., Nussbaum, M., Barrios, G., & Acevedo, N. (2023). Offline and online user experience of gamified robotics for introducing computational thinking: Comparing engagement, game mechanics and coding motivation. *Computers & Education*, 193, 104664.
- Madhu Govind, E. R., & Bers, M. U. (2020). Engaging children and parents to code together using the ScratchJr App. *Visitor Studies*, 23(1), 46–65.
- Malik, S. I., & Coldwell-Neilson, J. (2018). Gender differences in an introductory programming course: New teaching approach, students' learning outcomes, and perceptions. *Education and Information Technologies*, 23(6), 2453–2475.
- Marín-Marín, J.-A., García-Tudela, P. A., & Duo-Terrón, P. (2024). Computational thinking and programming with Arduino in education: A systematic review for secondary education. *Heliyon*, 10(8), e29177,1-15.
- Marta Terroba Juan Miguel Ribera, D. L., & Anguera, M. T. (2022). Observational analysis of the development of computational thinking in Early Childhood Education (5 years old) through an intervention proposal with a ground robot of programmed directionality. *European Early Childhood Education Research Journal*, 30(3), 437–455.

- Marwan, S., & Price, T. W. (2023). iSnap: Evolution and Evaluation of a Data-Driven Hint System for Block-Based Programming. *IEEE Transactions on Learning Technologies*, 16(3), 399–413.
- Merayo, N., & Ayuso, A. (2023). Analysis of barriers, supports and gender gap in the choice of STEM studies in secondary education. *International Journal of Technology and Design Education*, 33(4), 1471–1498.
- Merino-Armero, J. M., González-Calero, J. A., Cózar-Gutiérrez, R., & del Olmo-Muñoz, J. (2022). Unplugged Activities in Cross-Curricular Teaching: Effect on Sixth Graders' Computational Thinking and Learning Outcomes. *Multimodal Technologies and Interaction*, 6(2), 13-24.
- Metin, S. (2022). Activity-based unplugged coding during the preschool period. *International Journal of Technology and Design Education*, 32(1), 149–165.
- Minamide, A., Takemata, K., & Yamada, H. (2020). Development of computational thinking education system for elementary school class. *2020 IEEE 20th International Conference on Advanced Learning Technologies (ICALT)*, 22–23.
- Moder, K. (2010). Alternatives to F-test in one way ANOVA in case of heterogeneity of variances (a simulation study). *Psychological Test and Assessment Modeling*, 52(4), 343-354.
- Moreno-Leon, J., & Robles, G. (2016). Code to learn with Scratch ? *2016 IEEE Global Engineering Education Conference (EDUCON)*, April, 150–156.
- Moreno-LeÓN, J., Robles, G., & RomÁN-González, M. (2020). Towards Data-Driven Learning Paths to Develop Computational Thinking with Scratch. *IEEE Transactions on Emerging Topics in Computing*, 8(1), 193–205.

- Munasinghe, B., Bell, T., & Robins, A. (2023). Unplugged activities as a catalyst when teaching introductory programming. *Journal of Pedagogical Research*, 7(2), 56–71.
- National Research Council, Division of Behavioral, Education, B. S., & Committee on a Conceptual Framework for New K-12 Science Education Standards. (2012). *A framework for K-12 science education: Practices, crosscutting concepts, and core ideas*. National Academies Press.
- Noh, J., & Lee, J. (2020). Effects of robotics programming on the computational thinking and creativity of elementary school students. *Educational Technology Research and Development*, 68(1), 463–484.
- Ntourou, V., Kalogiannakis, M., & Psycharis, S. (2021). A Study of the impact of Arduino and Visual Programming in self-efficacy, motivation, computational thinking and 5th grade students' perceptions on electricity. *Eurasia Journal of Mathematics, Science and Technology Education*, 17(5), 1–11.
- OECD. (2023). *PISA 2022 Results factsheets Malaysia (Volume I) (Issue 2)*. https://www.oecd.org/publication/pisa-2022-results/country-notes/malaysia-1dbe2061/%0Ahttps://www.oecd-ilibrary.org/education/pisa-2022-results-volume-i_53f23881-en
- OECD, O. (2019). *Social Impact Investment 2019 The Impact Imperative for Sustainable Development*. OECD.
- Ogegbo, A. A., & Ramnarain, U. (2022). A systematic review of computational thinking in science classrooms. *Studies in Science Education*, 58(2), 203–230.
- Orhan, A., & Çeviker Ay, Ş. (2023). How to teach critical thinking: an experimental study with three different approaches. *Learning Environments Research*, 26(1), 199–217.

- Palioura, M., & Dimoulas, C. (2022). Digital storytelling in education: A transmedia integration approach for the non-developers. *Education Sciences*, 12(8),559-570..
- Palts, T., & Pedaste, M. (2020). A model for developing computational thinking skills. *Informatics in Education*, 19(1), 113–128.
- Papert, S. (1980). *Mindstorms: children, computers, and powerful ideas*. Basic Books, Inc.
- Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism*, 36(2), 1–11.
- Parham-Mocello, J., Erwig, M., Niess, M., Weber, J., Smith, M., & Berliner, G. (2023). Putting computing on the table: Using physical games to teach computer science. *SIGCSE 2023 - Proceedings of the 54th ACM Technical Symposium on Computer Science Education*, 1, 444–450.
- Park, W., & Kwon, H. (2023). Bringing computational thinking to technology education classrooms: Hacking car activity for middle schools in the republic of korea. *International Journal of Technology and Design Education*, 33(3), 863–881.
- Parra, X. C., & García, F. A. J. (2022). Integrated gamification model in a constructivist learning environment for the promotion of creative skills. *Creativity. Theories – Research - Applications*, 9(1), 1–25.
- Parsazadeh, N., Cheng, P.-Y. Y., Wu, T.-T. T., & Huang, Y.-M. M. (2021). Integrating computational thinking concept into digital storytelling to improve learners’ motivation and performance. *Journal of Educational Computing Research*, 59(3), 470–495.
- Paucar-Curasma, R., Cerna-Ruiz, L. P., Acra-Despradel, C., Villalba-Condori, K. O., Massa-Palacios, L. A., Olivera-Chura, A., & Esteban-Robladillo, I. (2023). Development of Computational Thinking through STEM activities for the promotion of gender equality. *Sustainability*, 15(16),1-19.

- Peel, A., Sadler, T. D., & Friedrichsen, P. (2022). Algorithmic explanations: an unplugged instructional approach to integrate science and computational thinking. *Journal of Science Education and Technology*, 31(4), 428–441.
- Pelánek, R., & Effenberger, T. (2023). The landscape of computational thinking problems for practice and assessment. *ACM Trans. Comput. Educ.*, 23(2), 1-29.
- Pellas, N., Vosinakis, S., & Spyridon, V. (2018). The effect of simulation games on learning computer programming: A comparative study on high school students' learning performance by assessing computational problem-solving strategies. *Education and Information Technologies*, 23(6), 2423–2452.
- Peng, H.-H., Murti, A. T., Silitonga, L. M., & Wu, T.-T. (2023). Effects of the fundamental concepts of computational thinking on students' anxiety and motivation toward K-12 english writing. *Sustainability (Switzerland)*, 15(7), 1-19.
- Pérez-Juárez, M. Á., González-Ortega, D., & Aguiar-Pérez, J. M. (2023). Digital distractions from the point of view of higher education students. *Sustainability*, 15(7), 1-22.
- Peters-Burton, E., Rich, P. J., Kitsantas, A., Stehle, S. M., & Laclede, L. (2023). High school biology teachers' integration of computational thinking into data practices to support student investigations. *Journal of Research in Science Teaching*, 60(6), 1353–1384.
- Petrie, C. (2022). Interdisciplinary computational thinking with music and programming: a case study on algorithmic music composition with Sonic Pi. *Computer Science Education*, 32(2), 260–282.
- Piaget, J. (1976). Piaget's Theory. In B. Inhelder, H. H. Chipman, & C. Zwingmann (Eds.), *Piaget and His School: A Reader in Developmental Psychology* (pp. 11–23). Springer Berlin Heidelberg.

- Polat, E., & Yilmaz, R. M. (2022). Unplugged versus plugged-in: examining basic programming achievement and computational thinking of 6th-grade students. *Education and Information Technologies*, 27(7), 9145–9179. <https://doi.org/10.1007/s10639-022-10992-y>
- Pollak, M., & Ebner, M. (2019). The Missing Link to Computational Thinking. *Future Internet*, 11(12), 1–13.
- Rastovac, D., Mandic, M., Majski, V., & Cvetkovic, D. (2021). Exploring the Achievement and Motivation of Learning a Text Programming Language among Elementary School Students in the Republic of Serbia. *Croatian Journal of Education-Hrvatski Casopis Za Odgoj i Obrazovanje*, 23(3), 819–850.
- Rehmat, A. P., Hoda, E., & Cardella, M. E. (2020). Instructional strategies to promote computational thinking for young learners. *Journal of Digital Learning in Teacher Education*, 36(1), 46–62.
- Relkin, E., de Ruiter, L., & Bers, M. U. (2020). TechCheck: Development and Validation of an Unplugged Assessment of computational thinking in early childhood education. *Journal of Science Education and Technology*, 29(4), 482–498.
- Relkin, E., de Ruiter, L. E., & Bers, M. U. (2021). Learning to code and the acquisition of computational thinking by young children. *Computers & Education*, 169, 104222.
- Reynolds, L. P., Borowicz, P. P., Caton, J. S., Vonnahme, K. A., Luther, J. S., Hammer, C. J., Maddock Carlin, K. R., Grazul-Bilska, A. T., & Redmer, D. A. (2010). Developmental programming: The concept, large animal models, and the key role of uteroplacental vascular development^{1,2}. *Journal of Animal Science*, 88(suppl_13), E61–E72.

- Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using Scratch: an experiment with sixth-grade students. *Interactive Learning Environments*, 28(3), 316–327.
- Rodriguez, B., Rader, C., & Camp, T. (2016). Using student performance to assess CS Unplugged Activities in a classroom environment. *Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education*, 95–100.
- Román-González, M., Pérez-González, J. C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior*, 72, 678–691.
- Rottenhofer, M., Kuka, L., Leitner, S., & Sabitzer, B. (2022). Using computational thinking to facilitate language learning: A survey of students' strategy use in Austrian Secondary Schools. *IAFOR Journal of Education*, 10(2), 51–70.
- Rubio, M. A., Romero-Zaliz, R., Mañoso, C., & de Madrid, A. P. (2015). Closing the gender gap in an introductory programming course. *Computers & Education*, 82, 409–420.
- Salkind, N. J., & Frey, B. B. (2019). *Statistics for people who (think they) hate statistics*. Sage Publications.
- Santo, A. De, Farah, J. C., Martínez, M. L., Moro, A., Bergram, K., Purohit, A. K., Felber, P., Gillet, D., Holzer, A., De Santo, A., Farah, J. C., Martínez, M. L., Moro, A., Bergram, K., Purohit, A. K., Felber, P., Gillet, D., & Holzer, A. (2022). Promoting Computational Thinking Skills in Non-Computer-Science students: Gamifying Computational Notebooks to increase student engagement. *IEEE Transactions on Learning Technologies*, 15(3), 392–405.
- 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. *The Asia-Pacific Education Researcher*, 29(1), 55–66.
- Schut, S., Driessen, E., van Tartwijk, J., van der Vleuten, C., & Heeneman, S. (2018). Stakes in the eye of the beholder: an international study of learners' perceptions within programmatic assessment. *Medical Education*, 52(6), 654–663.
- Schwartz, L., Maquil, V., Johannsen, L., Moll, C., & Hermen, J. (2024). Teaching computational thinking with a tangible development platform: An exploratory field study at school with Kniwwelino. *Education and Information Technologies*, 29(4), 4935–4967.
- Seckel, M. J., Salinas, C., Font, V., & Sala-Sebastià, G. (2023). Guidelines to develop computational thinking using the Bee-bot robot from the literature. *Education and Information Technologies*, 28(12), 16127–16151.
- Selby, C. (2015). Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy. *WiPSCE '15: Workshop in Primary and Secondary Computing*, 80–87.
- Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, 18(2), 351–380.
- Shemshack, A., & Spector, J. M. (2020). A systematic literature review of personalized learning terms. *Smart Learning Environments*, 7(1), 33,1-10.
- Shi, Y. (2020). Who benefits from selective education? Evidence from elite boarding school admissions. *Economics of Education Review*, 74, 101907.
- Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking.

Educational Research Review, 22, 142–158.

Simmonds, J., Diaz, C., & Estay, C. (2023). Introducing computational thinking at vocational high schools. *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1*, 68–74.

Sitthiworachart, J., Joy, M., King, E., Sinclair, J., & Foss, J. (2022). Technology-Supported Active Learning in a Flexible Teaching Space. *Education Sciences*, 12(9),1-12.

Smith, N., & Caldwell, H. (2016). Teaching Computing Unplugged in Primary Schools: exploring primary computing through practical activities away from the computer. In *Teaching Computing Unplugged in Primary Schools*. Learning Matters.

Soleimani, A., Herro, D., & Green, K. E. (2019). CyberPLAYce—A tangible, interactive learning tool fostering children’s computational thinking through storytelling. *International Journal of Child-Computer Interaction*, 20, 9–23.

Souza, I. M. L., Andrade, W. L., Sampaio, L. M. R., & Araujo, A. L. S. O. (2019). A systematic review on the use of LEGO® robotics in education. *Proceedings - Frontiers in Education Conference, FIE, 2018-Octob.*

Srinivasan, S., & Centea, D. (2019). An Active Learning Strategy for Programming Courses. In M. E. Auer & T. Tsiatsos (Eds.), *Mobile Technologies and Applications for the Internet of Things* (pp. 327–336). Springer International Publishing.

Stupurienė, G., Jevsikova, T., & Juškevičienė, A. (2022). Solving ecological problems through physical computing to ensure gender balance in STEM education. *Sustainability*, 14(9),1-16.

Subramaniam, S., Maat, S. M., & Mahmud, M. S. (2022). Computational thinking in mathematics education : A systematic review. *Journal of Educational Sciences*, 17(6),

2029–2044.

- Sun, L., Hu, L., & Zhou, D. (2021a). Which way of design programming activities is more effective to promote K-12 students' computational thinking skills? A meta-analysis. *Journal of Computer Assisted Learning*, 37(4), 1048–1062.
- Sun, L., Hu, L., & Zhou, D. (2021b). Improving 7th-graders' computational thinking skills through unplugged programming activities: A study on the influence of multiple factors. *Thinking Skills and Creativity*, 42, 100926.
- Sun, L., Hu, L., & Zhou, D. (2022). Single or Combined? A study on programming to promote junior high school students' computational thinking skills. *Journal of Educational Computing Research*, 60(2), 283–321.
- Sun, L., & Liu, J. (2024). Effects of Gamified Python Programming on primary school students' computational thinking skills: A differential analysis of gender. *Journal of Educational Computing Research*, 62(3), 846–874.
- Sun, L., Liu, J., & Liu, Y. (2024). Comparative experiment of the effects of unplugged and plugged-in programming on computational thinking in primary school students: A perspective of multiple influential factors. *Thinking Skills and Creativity*, 52, 101542.
- Sun, L., & Zhou, L. (2023). Does text-based programming improve K-12 students' CT skills? Evidence from a meta-analysis and synthesis of qualitative data in educational contexts. *Thinking Skills and Creativity*, 49, 101340.
- Tabachnick, B. G., Fidell, L. S., & Ullman, J. B. (2013). *Using multivariate statistics* (Vol. 6). pearson Boston, MA.
- Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education*, 148(December

2019), 103798.

- Tehlan, K., Chakraverty, S., Chakraborty, P., & Khapra, S. (2020). A genetic algorithm-based approach for making pairs and assigning exercises in a programming course. *Computer Applications in Engineering Education*, 28(6), 1708–1721.
- Tekdal, M. (2021). Trends and development in research on computational thinking. *Education and Information Technologies*, 26(5), 6499–6529.
- Threekunprapa, A., & Yasri, P. (2020). Unplugged coding using flowblocks for promoting computational thinking and programming among secondary school students. *International Journal of Instruction*, 13(3), 207–222.
- Tichavsky, L. P., Hunt, A. N., Driscoll, A., & Jicha, K. (2015). “It’s just nice having a real teacher”: Student perceptions of online versus face-to-face instruction. *International Journal for the Scholarship of Teaching and Learning*, 9(2), 1-8.
- Tikva, C., & Tambouris, E. (2021). Mapping computational thinking through programming in K-12 education: A conceptual model based on a systematic literature Review. *Computers and Education*, 162(November 2020), 104083.
- Tiruneh, D. T., De Cock, M., Weldeclassie, A. G., Elen, J., & Janssen, R. (2017). Measuring Critical Thinking in Physics: Development and Validation of a Critical Thinking Test in electricity and magnetism. *International Journal of Science and Mathematics Education*, 15(4), 663–682.
- Torres-Torres, Y.-D., Román-González, M., & Pérez-González, J.-C. (2019). Implementation of unplugged teaching activities to foster computational thinking skills in primary school from a gender perspective. *Proceedings of the Seventh International Conference on Technological Ecosystems for Enhancing Multiculturality*, 209–215.

- Tran, V. D., Nguyen, T. M. L., De, N. Van, Soryaly, C., & Doan, M. N. (2019). Does cooperative learning may enhance the use of students' learning strategies? *International Journal of Higher Education*, 8(4), 79–88.
- Tripon, C. (2022). Supporting future teachers to promote computational thinking skills in teaching STEM—A case study. *Sustainability*, 14(19), 1–17.
- Tsarava, K., Moeller, K., & Ninaus, M. (2019). Board games for training computational thinking. *Games and Learning Alliance: 7th International Conference, GALA 2018, Palermo, Italy, December 5--7, 2018, Proceedings 7*, 90–100.
- Tsarava, K., Moeller, K., Román-González, M., Golle, J., Leifheit, L., Butz, M. V., & Ninaus, M. (2022). A cognitive definition of computational thinking in primary education. *Computers & Education*, 179(November 2020), 104425.
- Turchi, T., Fogli, D., & Malizia, A. (2019). Fostering computational thinking through collaborative game-based learning. *Multimedia Tools and Applications*, 78(10), 13649–13673.
- Turner, P. E., Johnston, E., Kebritchi, M., Evans, S., & David A, H. (2018). Influence of online computer games on the academic achievement of nontraditional undergraduate students. *Cogent Education*, 5(1), 1437671.
- Ung, L. L., Tammie, C. S., Jane, L., & Norazila, A. A. (2018). Preliminary Investigation: Teachers' Perception on Computational Thinking Concepts. *Journal of Telecommunication, Electronic and Computer Engineering*, 9(2), 23–29.
- Valls Pou, A., Canaleta, X., Fonseca, D., Pou, A. V., Canaleta, X., & Fonseca, D. (2022). Computational Thinking and Educational Robotics integrated into Project-Based Learning. *Sensors*, 22(10), 1–21.

- Vinnervik, P. (2022). Implementing programming in school mathematics and technology: teachers' intrinsic and extrinsic challenges. *International Journal of Technology and Design Education*, 32(1), 213–242.
- Vygotsky, L. S., & Cole, M. (1978). *Mind in society: Development of higher psychological processes*. Harvard university press.
- Wang, C., Shen, J., & Chao, J. (2021). Integrating Computational Thinking in STEM education: A literature review. *International Journal of Science and Mathematics Education*, 20, 1949–1972.
- 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.
- Weintrop, D., & Wilensky, U. (2015). To block or not to block, that is the question: Students' perceptions of blocks-based programming. *Proceedings of IDC 2015: The 14th International Conference on Interaction Design and Children*, 199–208.
- 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. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33–35.
- 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.
- Wing, J. (2011). Research notebook: Computational thinking—What and why? *The Link Magazine*, June 23, 2015. <http://www.cs.cmu.edu/link/research-notebook->

computational-thinking-what-and-why

- Wofford, A. M. (2021). Modeling the pathways to self-confidence for graduate school in computing. *Research in Higher Education*, 62(3), 359–391.
- World Economic Forum, V. (2020). The future of jobs report 2020. In *Retrieved from Geneva*.
- Xie, Z., Wong, G. K. W., Chiu, D. K. W., & Lei, J. (2023). Bridging K–12 Mathematics and Computational Thinking in the Scratch Community: Implications drawn from a Creative Learning Context. *IT Professional*, 25(2), 64–70.
- Xu, E., Wang, W., & Wang, Q. (2023). A meta-analysis of the effectiveness of programming teaching in promoting K-12 students' computational thinking. *Education and Information Technologies*, 28(6), 6619–6644.
- Xu, Y., Liu, X., Cao, X., Huang, C., Liu, E., Qian, S., Liu, X., Wu, Y., Dong, F., Qiu, C.-W., Qiu, J., Hua, K., Su, W., Wu, J., Xu, H., Han, Y., Fu, C., Yin, Z., Liu, M., ... Zhang, J. (2021). Artificial intelligence: A powerful paradigm for scientific research. *The Innovation*, 2(4), 100179.
- Yadav, A., Hong, H., & Stephenson, C. (2016). Computational Thinking for All: pedagogical approaches to embedding 21st century Problem Solving in K-12 classrooms. *TechTrends*, 60(6), 565–568.
- Yadav, A., Krist, C., Good, J. C., & Nadire, E. (2018). Computational thinking in elementary classrooms: measuring teacher understanding of computational ideas for teaching science. *Computer Science Education*, 28(4), 371–400.
- 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(1–16).

- Yadav, S., & Chakraborty, P. (2023). Introducing schoolchildren to computational thinking using smartphone apps: A way to encourage enrollment in engineering education. *Computer Applications in Engineering Education*, 31(4), 831–849.
- Yang, W., Ng, D. T. K., & Gao, H. (2022). Robot programming versus block play in early childhood education: Effects on computational thinking, sequencing ability, and self-regulation. *British Journal of Educational Technology*, 53(December 2021), 1817–1841.
- Yang, Z., & Bers, M. (2023). Examining gender difference in the use of ScratchJr in a programming curriculum for First Graders. *Computer Science Education*, 1–22.
- Ye, H., Liang, B., Ng, O.-L., & Chai, C. S. (2023). Integration of computational thinking in K-12 mathematics education: a systematic review on CT-based mathematics instruction and student learning. *International Journal of STEM Education*, 10(1), 3.
- Yeni, S., Grgurina, N., Saeli, M., Hermans, F., Tolboom, J., & Barendsen, E. (2024). Interdisciplinary integration of computational thinking in K-12 education: A systematic review. *Informatics in Education*, 23(1), 223–278.
- Yildiz Durak, H. (2020). The effects of using different tools in programming teaching of secondary school students on engagement, computational thinking and reflective thinking skills for problem solving. *Technology, Knowledge and Learning*, 25(1), 179–195.
- Yildiz Durak, H., Saritepeci, M., & Durak, A. (2023). Modeling of relationship of personal and affective variables with computational thinking and programming. *Technology, Knowledge and Learning*, 28(1), 165–184.

- Yilmaz Ince, E., & Koc, M. (2021). The consequences of robotics programming education on computational thinking skills: An intervention of the Young Engineer's Workshop (YEW). *Computer Applications in Engineering Education*, 29(1), 191–208.
- Yin, Y., Hadad, R., Tang, X., & Lin, Q. (2020). Improving and assessing computational thinking in Maker Activities: the Integration with Physics and Engineering Learning. *Journal of Science Education and Technology*, 29(2), 189–214.
- Yıldız, M., & Karal, H. (2021). A Computer Science Unplugged Activity: CityMap. *International Journal of Computer Science Education in Schools*, 5, 14–27. <https://doi.org/10.21585/ijcses.v5i2.110>
- Zapata-Caceres, M., Martin-Barroso, E., Roman-Gonzalez, M., Zapata-Cáceres, M., & Martín-Barroso, Estefanía Román-González, M. (2021). Collaborative Game-Based Environment and assessment tool for Learning Computational Thinking in primary school: A case study. *IEEE Transactions on Learning Technologies*, 14(5), 576–589.
- Zdawczyk, C., & Varma, K. (2023). Engaging girls in computer science: gender differences in attitudes and beliefs about learning scratch and python. *Computer Science Education*, 33(4), 600–620.
- Zhang, Y., Luo, R., Zhu, Y., & Yin, Y. (2021). Educational Robots Improve K-12 Students' Computational Thinking and STEM Attitudes: Systematic Review. *Journal of Educational Computing Research*, 59(7), 1450–1481.
- Zhou, L. (2023). How to Develop 21st Century Skills in Students: The Role of LEGO Education. *Science Insights Education Frontiers*, 15(2), 2281–2283.
- Zhu, Z., Zhang, Y., Zhu, W., & Ma, J. (2023). Computational thinking and academic achievement: The mediator roles of problem-solving ability and learning anxiety. 2023

International Symposium on Educational Technology (ISET), 94–98.

Zitouniatis, A., Lazarinis, F., & Kanellopoulos, D. (2023). Teaching computational thinking using scenario-based learning tools. *Education and Information Technologies*, 28(4), 4017–4040.

APPENDICES

Appendix A: Journal Publications

Batni, B. & Junaini, S. N. (2024). Redefining Computational Thinking: Synergizing Unplugged Activities with Block-Based Programming. *Education and Information Technologies*.

Batni, B., Junaini, S. N., Sidi, J., Mustafa, W. A. and Ismail, Z. (2024). Current Research Trends of Scratch Block Based Programming for K12: A Systematic Review. *Journal of Advanced Research in Applied Sciences and Engineering Technology*.

Batni, B., Junaini, S. N., Sidi, J., and Mustafa, W. A. (2024). Current Research Trends of Scratch Block Based Programming for K12: A Systematic Review. *3rd International Conference on Computing, Information Science, and Engineering 2023 (ICISE 2023)*

Appendix B: KPM Approval



Ruj. Kami : KPM.600-3/2/3-eras(17372)
Tarikh : 8 September 2023

BADRULIMAN BIN BATNI@BATENI
NO. KP : 860223135093

NO1353, JALAN BPJ 2/25, BANDAR PUTERI JAYA, SUNGAI PETANI
8000 SUNGAI PETANI
KEDAH

Tuan,

KELULUSAN BERSYARAT UNTUK MENJALANKAN KAJIAN :
Fostering Computational Thinking and Programming in North Peninsula Malaysian Boarding Schools: A Study of the Synergistic Effects of Unplugged Activities and Block-Based Programming

Perkara di atas adalah dirujuk.

2. Sukacita dimaklumkan bahawa permohonan tuan untuk menjalankan kajian seperti di bawah telah diluluskan dengan syarat :

" KELULUSAN INI BERGANTUNG KEPADA KEBENARAN PENGARAH JPN DAN PERTIMBANGAN PENTADBIR SEKOLAH. PENGUTIPAN DATA MELIBATKAN MURID HENDAKLAH DILAKSANAKAN OLEH DENGAN BANTUAN GURU SEKOLAH BERKENAAN. "

3. Kelulusan adalah berdasarkan kepada kertas cadangan penyelidikan dan instrumen kajian yang dikemukakan oleh tuan kepada bahagian ini. Walau bagaimanapun kelulusan ini bergantung kepada kebenaran Jabatan Pendidikan Negeri dan Pengetua / Guru Besar yang berkenaan.

4. Surat kelulusan ini sah digunakan bermula dari **8 September 2023** hingga **16 Februari 2024**

5. Tuan dikehendaki menyerahkan senaskhah laporan akhir kajian dalam bentuk *hardcopy* bersama salinan *softcopy* berformat pdf dalam CD kepada Bahagian ini. Tuan juga diingatkan supaya mendapat kebenaran terlebih dahulu daripada Bahagian ini sekiranya sebahagian atau sepenuhnya dapatan kajian tersebut hendak diterbitkan di mana-mana forum, seminar atau diumumkan kepada media massa.

Sekian untuk makluman dan tindakan tuan selanjutnya. Terima kasih.

"BERKHIDMAT UNTUK NEGARA"

Saya yang menjalankan amanah,

Ketua Penolong Pengarah Kanan
Sektor Penyelidikan dan Penilaian Dasar
b.p. Pengarah
Bahagian Perancangan dan Penyelidikan Dasar Pendidikan
Kementerian Pendidikan Malaysia

salinan kepada:-

JABATAN PENDIDIKAN KEDAH
JABATAN PENDIDIKAN PULAU PINANG
JABATAN PENDIDIKAN PERLIS

* SURAT INI DIJANA OLEH KOMPUTER DAN TIADA TANDATANGAN DIPERLUKAN *

Appendix C: JPN Kedah Approval



KEMENTERIAN PENDIDIKAN
JABATAN PENDIDIKAN NEGERI KEDAH
KOMPLEKS PENDIDIKAN, JALAN STADIUM
05604 ALOR SETAR
KEDAH DARUL AMAN

Telefon : 04-740 4000
Faks : 04-740 4342
Laman Web : <https://jpnkedah.moe.gov.my>

Ruj.Kami : JPNK.600-11/1/1 Jld.26(7)
Tarikh : 20 September 2023

BADRULIMAN BIN BATNI@BATENI
NO. KP: 860223135093

Tuan / Puan,

**KEBENARAN UNTUK MENJALANKAN KAJIAN / PENYELIDIKAN DI SEKOLAH-SEKOLAH
DI NEGERI KEDAH**

Saya dengan hormatnya diarah merujuk kepada perkara tersebut di atas.

2. Dimaklumkan bahawa permohonan tuan / puan untuk menjalankan penyelidikan yang bertajuk **"FOSTERING COMPUTATIONAL THINKING AND PROGRAMMING IN NORTH PENINSULA MALAYSIAN BOARDING SCHOOLS: A STUDY OF THE SYNERGISTIC EFFECTS ON UNPLUGGED ACTIVITIES AND BLOCK-BASED PROGRAMMING"** telah **diluluskan** dengan syarat pengutipan data secara bersemuka.

3. Sehubungan itu, pihak Jabatan Pendidikan Negeri Kedah **tiada halangan** pihak tuan untuk mengadakan kajian/penyelidikan di sekolah berikut:

BIL	NAMA SEKOLAH
1.	Sekolah Menengah Sains Sultan Muhammad Jiwa
2.	Sekolah Menengah Sultan Abdul Halim
3.	Sekolah Menengah Sains Pokok Sena
4.	Sekolah Berasrama Penuh Integrasi Kubang Pasu
5.	Sekolah Menengah Sains Kubang Pasu

4. Pihak tuan adalah diminta berhubung dengan Pegawai Pendidikan Daerah dan Pengetua/Guru Besar sekolah-sekolah yang berkenaan. Pihak tuan juga dinasihatkan agar dapat mematuhi syarat-syarat seperti berikut:

- 4.1 Mendapat persetujuan daripada Pegawai Pendidikan Daerah dan Pengetua sekolah berkenaan;
- 4.2 Surat kebenaran / kelulusan ini tidak boleh digunapakai bagi maksud untuk mendapat sumbangan dan tajaan daripada mana-mana pihak berkenaan;
- 4.3 Pihak tuan bertanggungjawab menjaga keselamatan dan kebajikan murid dan guru yang terlibat dengan program ini;
- 4.4 murid-murid, warga sekolah mahupun pihak sekolah sendiri tidak boleh dipaksa atau diwajibkan terlibat dengan kajian/penyelidikan ini;

- 4.5 Penyertaan murid-murid hendaklah mendapat kebenaran bertulis daripada ibu bapa atau penjaga;
 - 4.6 Pihak tuan hendaklah menanggung semua kos kajian/penyelidikan pengangkutan, makan dan minum;
 - 4.7 Tidak akan memberi **implikasi kewangan** kepada Pejabat Pendidikan Daerah, Jabatan Pendidikan Negeri Kedah dan Kementerian Pendidikan Malaysia;
 - 4.8 Jabatan ini berhak menarik balik surat kelulusan/kebenaran sekiranya pihak penganjur gagal mematuhi syarat dan peraturan yang ditetapkan atau bercanggah dengan dasar KPM yang sedang berkuatkuasa serta berkuatkuasa serta tidak mematuhi garis panduan KPM; dan
 - 4.9 Digalakkan memakai pelitup muka pada setiap masa.
5. Kelulusan ini adalah berdasarkan kepada apa yang terkandung di dalam cadangan kajian/penyelidikan yang tuan / puan kemukakan ke Kementerian Pendidikan Malaysia. Tuan / puan dikehendaki **mengemukakan senaskah laporan akhir** setelah selesai kelak dan diingatkan supaya mendapat kebenaran terlebih dahulu daripada Jabatan ini sekiranya sebahagian atau sepenuhnya dapatan kajian tersebut hendak dibentangkan di mana-mana forum, seminar atau diumumkan kepada media.
6. Kebenaran ini adalah tertakluk kepada persetujuan Pengetua / Guru Besar sekolah berkenaan dan adalah **sah dari 8 September 2023 hingga 16 Februari 2024** sahaja.

Sekian, terima kasih.

"MALAYSIA MADANI"
"BERKHIDMAT UNTUK NEGARA"
"PENDIDIKAN CEMERLANG KEDAH TERBILANG"

Saya yang menjalankan amanah,



(HAJI ISMAIL BIN OTHMAN, BCK.)

Timbalan Pendaftar
Institusi Pendidikan dan Guru
Jabatan Pendidikan Negeri Kedah
b.p. Ketua Pendaftar Institusi Pendidikan dan Guru
Kementerian Pendidikan Malaysia

Appendix D: Parent/Guardian Consent Form

Informed Consent Form for Research

Title of Study: Fostering Computational Thinking and
Programming in North Peninsula Malaysian Boarding Schools: A Study
of The Synergistic Effects of Unplugged Activities and Block-Based
Programming.

Researcher: Badruliman Bin Batni@Bateni (UNIMAS)
University Malaysia Sarawak

Dear Parent/Guardian,

Your child has been invited to participate in a research study aimed at understanding how students can learn computational thinking and programming through a four-week activity that introduces unplugged activities and block-based programming (BBP).

Information:

If you allow your child to participate, we will collect recorded data from the learning activities, including pre/post test data, Scratch projects, and audio recordings of student interviews. The pre- and post-tests consist of a computational thinking test and an attitude survey, completed online. Coding and Scratch projects are digital files created by students, and audio recordings are transcribed with all identifying information removed, then immediately destroyed.

Activities:

The study involves normal class hours in the computer room. Students' complete tests and surveys, engage in learning activities, create programming examples, and retake the tests and surveys at the end to assess the effectiveness of the activity.

Risks:

Participation involves minimal risks associated with regular computer use.

Benefits:

By allowing your child to participate, you are contributing to research on how students learn computational thinking and programming concepts and practices.

Confidentiality:

Your child will be assigned a unique identifier to ensure privacy. No reports will link your child's name or identity to the study. No photos or videos will be taken. All data will be stored securely, and identifying information will be destroyed after data collection.

Contact:

If you have any questions, you can contact Badruliman Bin Batni, Researcher, at 013-7145363.

Participation:

Your child's participation is voluntary, and you can refuse without penalty. You or your child may withdraw at any time without consequences.

Consent:

"I have read and understood the above information. My child has permission to participate in this study with the understanding that they can withdraw at any time. My child may participate in this research, and data collected during the research may be used by the researcher for educational research purposes."

Student's Name: _____

Parent/Guardian's Name: _____

By signing my name below, I certify that I agree with the above statement:

Parent/Guardian's Signature: _____ Date: _____

Appendix E: Student Assent Form

Student Informed Assent Form for Research

Title of Study: Fostering Computational Thinking and Programming in North Peninsula Malaysian Boarding Schools: A Study of The Synergistic Effects of Unplugged Activities and Block-Based Programming.

Researcher: Badruliman Bin Batni@Bateni (UNIMAS)
University Malaysia Sarawak

Dear Student,

You're invited to join a study to learn about computational thinking, an important skill for understanding how computers work. This study will happen during your regular computer science class over eight sessions. You'll answer questions, do activities, and create computer programs using Scratch and block-based programming (BBP).

What will happen in this study?

You'll start by answering questions about computational thinking and your attitude towards programming. Then, you'll learn about computational thinking through fun activities like unplugged games and creating computer programs. At the end, you'll answer the same questions again. This helps the researchers see how well the class is working. Don't worry, your grades won't be affected by this study.

After the study, you'll know how to write computer programs in Scratch. We'll collect data through online questions, digital files of your programs, and audio recordings of interviews. The audio recordings will be transcribed, and all names will be removed. We'll delete the recordings right after.

Participating in this study is safe, just like using a computer normally. If you can't join or your parents don't give permission, you'll still do the same activities, but we won't use your data for research.

Your privacy is important. We'll keep your data safe. Remember, you don't have to join if you don't want to, and it won't affect your grades.

What if I have questions?

Your participation is voluntary. You have the right to stop participating in this study at any time and for any reason. You can also ask questions at any time. If you have any questions about this study, you can contact Badruliman Bin Batni, researcher, on 0137145363.

If you agree to participate in this study, please sign your name below. Thank you.

Student Name : _____

Signature : _____

Date : _____

Appendix F: Lesson Plan Review Form

Borang Semakan Rancangan Pengajaran
Aktiviti Pembelajaran: Cabaran Susun Cawan / Bina Menara / Kad Bersyarat

Aspect	Questions/Considerations	Agree	Agree (With improvement)	Disagree
Content Clarity	a. Are the learning objectives clear and measurable?			
	b. Are the instructions for each step clear and concise?			
Engagement and Interaction	a. Are the group activities well-designed for collaboration?			
	b. Does the game provide an engaging way for students to grasp the concept?			
Timing and Pace	a. Is the estimated time for each step realistic?			
	b. Does the lesson flow smoothly from one step to the next?			
Differentiation	a. Does the grouping consider varied skill levels?			
	b. Are the rules progressively challenging?			
Assessment	a. Are their clear methods for assessing student understanding?			
	b. Is their provision for feedback during or after activities?			
Alignment with Standards	a. Does the lesson align with the curriculum or standards?			
Additional Comments				
Recommendations				

Reviewer's Name	
Reviewer's Signature	

Appendix G: Lesson plan for unplugged activity

(Building Tower, Conditional card game and Stacking cup)

AKTIVITI PEMBELAJARAN: MEMBINA MENARA			
MINGGU:		TARIKH:	HARI:
TINGKATAN:		MASA:	MATA PELAJARAN: ASK
BIDANG/TEMA/TAJUK	1.0 Konsep Asas Pemikiran Komputasional		
STANDARD KANDUNGAN	1.1 Asas Pemikiran Komputasional		
STANDARD PEMBELAJARAN	1.1.1 Menerangkan teknik Leraian (Decomposition), Pengecaman Corak (Pattern Recognition), Peniskalaan (Abstraction) dan Pengitlakan (Generalisation) dalam pemikiran komputasional untuk menyelesaikan masalah. 1.1.3 Mengesan unsur persamaan dan perbezaan untuk pengecaman corak dalam sesuatu situasi. 1.1.4 Membuat keputusan bagi menentukan aspek penting dalam sesuatu permasalahan. 1.1.5 Menentukan ciri-ciri kesamaan dalam sesuatu permasalahan.		
OBJEKTIF PEMBELAJARAN/ KRITERIA KEJAYAAN	Murid dapat mengaplikasikan Kemahiran pemikiran komputasional melalui cabaran membina menara secara praktikal dengan: i. Menggunakan 4 teknik dalam asas pemikiran komputasional dengan betul ii. Mengesan sekurang-kurangnya 3 unsur persamaan dan perbezaan dalam proses pengecaman corak pembinaan menara. iii. Memberikan sekurang-kurangnya 1 justifikasi aspek penting dalam membuat keputusan. iv. Menentukan sekurang-kurangnya 2 ciri kesamaan dalam proses pembinaan Menara.		
LANGKAH/AKTIVITI			
SET INDUKSI	1. Murid diperkenalkan dengan konsep asas pemikiran komputasional sebagai pendekatan penyelesaian masalah dalam sains komputer. 2. Murid ditegaskan untuk menggunakan 4 teknik pemikiran komputasional dalam menyelesaikan tugas bagi sesuatu masalah.		
PERKEMBANGAN	1. Murid dibahagikan kepada 3-5 orang bagi setiap kumpulan. 2. Murid mendengar penerangan guru tentang cabaran untuk membina menara tertinggi. 3. Murid dikehendaki untuk membina sebuah Menara yang paling tinggi dalam masa 30 minit dengan menggunakan 20 batang straw dan stapler. 4. Murid ditugaskan untuk menghubungkan kemahiran Pemikiran Komputasional secara khusus dengan cabaran ini dengan: a. Menangani cabaran dalam penyelesaian masalah. b. Mengaplikasikan teknik leraian bagi keseluruhan tugas. c. Mengenalpasti corak pembinaan dalam memastikan menara untuk kekal stabil. d. Menilai keberkesanan dan kesesuaian Teknik yang digunakan dalam pembinaan menara. 5. Tugas kumpulan secara pertandingan : Murid membina Menara yang tinggi dan stabil dengan mengaplikasikan konsep pemikiran komputasional. 6. Murid digalakkan untuk saling berkomunikasi semasa proses pembinaan. 7. Kumpulan murid dibantu dan dibimbing guru sekiranya menghadapi masalah dalam proses pembinaan. 8. Murid membentangkan hasil kerja kumpulan dengan menyatakan strategi penyelesaian masalah, teknik dan corak yang diaplikasikan dalam proses pembinaan menara.		

	9. Guru menyemak dan membuat penilaian keatas tugas yang dihasilkan oleh setiap kumpulan. 10. Guru mengumumkan pemenang dalam pertandingan membina menara.
PENUTUP	1. Murid membuat rumusan pembelajaran hari ini pembinaan menara tertinggi. 2. Guru membuat peneguhan konsep yang dipraktikkan boleh diaplikasikan dalam pelbagai situasi dalam proses penyelesaian masalah yang lain.
ABM/BBM/ SUMBER	Buku teks, buku rujukan, stapler, LCD
REFLEKSI	

AKTIVITI PEMBELAJARAN: KAD BERSYARAT			
MINGGU:		TARIKH:	HARI:
TINGKATAN:		MASA:	MATA PELAJARAN:
Standard Kandungan	3.1 Pembangunan Algoritma		
Standard Pembelajaran	3.1.1 Menulis pseudokod dan melukis carta alir pelbagai pilihan dalam penyelesaian masalah. 3.1.2 Menulis pseudokod dan melukis carta alir yang melibatkan ulangan.		
Objektif Pembelajaran/ Kriteria Kejayaan	Pada akhir pengajaran dan pembelajaran, murid dapat : i. mengenal pasti sekurang-kurangnya 2 pilihan (if/else) dalam menulis pseudokod dan carta alir. ii. mengaplikasikan sekurang-kurangnya 2 konsep asas pernyataan pelbagai pilihan (if/else) dalam penyelesaian masalah.		
LANGKAH-LANGKAH/ AKTIVITI			
SET INDUKSI	1. Murid diperkenalkan dengan konsep permainan kad bersyarat. 2. Murid dipilih secara rawak untuk bersoal jawab tentang konsep dasar pernyataan pilihan (if/else) dalam penyelesaian masalah.		
PERKEMBANGAN	1. Murid dibahagikan pelajar kepada 3-5 orang bagi setiap kumpulan. 2. Guru mengedarkan 2 set kad UNO yang berlainan warna bagi setiap kumpulan. (Merah — Biru atau Hijau — Biru). 3. Murid mendengar penerangan guru tentang cabaran permainan dan memahami peraturan Pusingan 1 iaitu, pernyataan if / else dalam permainan. 4. Murid dilibatkan semasa guru menerangkan contoh permainan. 5. Pusingan 1: Murid mencabut 5 kad secara ;awak setelah kad dikocak. 6. Murid berbincang sesama ahli kumpulan dan mengira mata yang diperolehi berdasarkan kad yang dicabut. 7. Guru membimbing murid memahami syarat untuk menentukan markah dengan betul. 8. Pusingan 2: Murid diperkenalkan dengan syarat yang lebih mencabar dan kompleks iaitu, pernyataan pelbagai pilihan (if / else) 9. Guru merangsang murid untuk mengaitkan permainan kad dengan Pseudokod dan carta alir yang melibatkan ulangan dan pelbagai piihan. 10. Murid dalam kumpulan menulis Pseudokod dan melukis carta alir berdasarkan permainan yang telah dijalankan pada kertas mahjung. 11. Murid menampal jawapan kumpulan masing-masing di papan putih.		

	12. Setiap kumpulan murid bergerak dari satu jawapan ke jawapan yang lain dan melakukan banding beza. (Galery walk) 13. Guru menyemak, menilai dan berbincang tentang hasil kerja setiap kumpulan bersama murid.
PENUTUP	1. Murid berkongsi pengalaman dan pandangan mereka terhadap proses pembelajaran hari ini dan seterusnya membuat rumusan serta kesimpulan. 2. Guru membuat pengukuhan konsep pernyataan pilihan (if ~ else) yang diaplikasikan dalam permainan kad tersebut boleh dipraktikkan dalam proses membuat keputusan dalam kehidupan harian sebenar bagi penyelesaian masalah.
ABM/BBM/ SUMBER	Buku teks, buku rujukan, kad UNO, LCD
REFLEKSI	

AKTIVITI PEMBELAJARAN : CABARAN SUSUN CAWAN			
MINGGU:		TARIKH:	HARI:
TINGKATAN:		MASA:	MATA PELAJARAN ASK
STANDARD KANDUNGAN	1.1 Asas Pemikiran Komputasional		
STANDARD PEMBELAJARAN	1.1.1 Menerangkan teknik Leraian (Decomposition), Pengecaman Corak (Pattern Recognition), Peniskalaan (Abstraction) dan Pengitlakan (Generalisation) dalam pemikiran komputasional untuk menyelesaikan masalah. 1.1.2. Menyelesaikan sesuatu tugas menggunakan teknik leraian dan menentukan langkah secara tertib.		
OBJEKTIF PEMBELAJARAN/ KRITERIA KEJAYAAN	Pada akhir pengajaran dan pembelajaran, Murid dapat : 1. Merangka tugas kompleks (susunan cawan) kepada sekurang-kurangnya 5 langkah yang lebih ringkas dan mudah dilaksanakan. 2. Membuat dan menyempurnakan arahan langkah demi langkah (algoritma) untuk menyelesaikan cabaran susun cawan dengan efisien.		
LANGKAH/ AKTIVITI			
SET INDUKSI	1. Guru merangsang dan mengemukakan persoalan kepada murid tentang isu robotik, Robot dalam kehidupan manusia. 2. Murid berbincang secara berpasangan dan mengemukakan respon kepada isu yang dibangkitkan guru. 3. Murid dipilih secara rawak untuk mengemukakan pendapat masing-masing tentang penggunaan robot dalam kehidupan seharian.		
PERKEMBANGAN	1. Murid dibahagikan kepada 3-5 orang bagi setiap kumpulan. 2. Setiap kumpulan diberikan sebanyak 8 cawan plastik. 3. Murid ditugaskan untuk mencipta satu set arahan, langkah demi langkah untuk menyusun cawan dengan cepat dan tepat. 4. Murid dikehendaki mengembangkan idea dalam pelaksanaan algoritma secara perbincangan kumpulan. 5. Murid seterusnya menghasilkan pengaturcaraan yang tepat dengan arahan lebih interaktif dengan memperkenalkan butang ikon mewakili arahan seperti Angkat Cawan, Letakkan Cawan, Langkah Ke Kanan, Langkah Ke Kiri, dan Putar. 5. Kumpulan murid akan dibantu dan dibimbing oleh guru sekiranya hadapi masalah. 6. Kumpulan murid yang berjaya menghasilkan tugas dengan betul digalakkan membimbing rakan kumpulan yang lain.		

	7. Kumpulan mempamerkan hasil kerja masing-masing, dan ahli kumpulan lain boleh mencuba dan menguji algoritma sesebuah kumpulan. 8. Murid dikehendaki menilai keberkesanan algoritma setiap kumpulan yang lain dengan menekankan kepada ketepatan dalam memahami arahan. 9. Guru membuat penilaian bagi setiap kumpulan berdasarkan strategi yang digunakan.
PENUTUP	10. Guru menyoal murid untuk menguji tahap pemahaman dan penguasaan konsep yang dipraktikkan. 11. Murid membuat kesimpulan dan rumusan terhadap aktiviti yang telah dijalankan.
ABM/BBM/ SUMBER	Buku teks, buku rujukan, LCD
REFLEKSI	

Appendix H: Contents of The Learning Activities in The Classroom for Each Week

Week/ Duration	UA+BBP group	BBP group
Week 1 (1 Hour)	Building Tower Teams constructed the tallest tower using twenty straws and plasticine. Students strategized, identified patterns, evaluated progress, and decomposed tasks.	Introduction to BBP and Scratch user interface; problem decomposition and creating algorithms as a set of instructions; using abstractions.
Week 1 (1 Hour)	Stacking Cup Introduced CT through algorithm development for precise cup stacking. Students worked in small groups to develop, test, and refine stacking algorithms. Post-activity debriefings encouraged reflections on algorithmic insights and the importance of clear instructions.	Conditionals and decision-making in programming (if and if-else statements); using mathematical operators in programming.
Week 2 (1 Hour)	Conditional Card Game Interactive pedagogical tool resembling a modified Uno card game with two suits, introducing an element of chance. Employs if/else conditional rules to teach programming conditions. Links to broader concepts like pseudocode and flowcharts, incorporating	Loops and different repetition structures in programming; using nesting in programming; using logical operators in programming.

	elements of choice and decision-making. CT skills: pattern recognition, abstraction, and algorithmic design.	
Week 2 (1 Hour)	Introduction to BBP and Scratch user interface; problem decomposition and creating algorithms as a set of instructions; using abstractions.	Creating, using, and manipulating different types of variables in programming; event-driven programming with broadcasting in Scratch
Week 3 (2 Hour)	Project 1: Drawing Snowflakes Shapes Learn coding concepts like sequencing and looping. Utilization of the Scratch pen tool for programming and drawing. Review knowledge of polygons and related geometric concepts. Focus on the correlation between coding and geometric principles. Applies coding skills to visually represent and comprehend geometric shapes and patterns.	
Week 4 (2 Hour)	Project 2: Math Race Game Integrate math and coding to develop a game in Scratch. Engagement in a gamified learning experience cantered on math. Demands problem-solving skills to design game mechanics grounded in mathematical principles.	

Appendix I: Students Feedback Form

Name : _____

Class : _____

Activity : _____

1. Did you understand the objective of the study? ☐ Yes ☐ NO

2. What was your favourite part of the activity?

3. What new skills did you learn through this activity?

4. How much did you enjoy this activity? ☐ A lot ☐ A little ☐ Not at all

5. Would you like to a similar activity again? ☐ Yes ☐ NO

6. Any suggestions to improve this activity?

Appendix J: Approval from Dr. Marcos Roman Gonzalez, the original author of the CT test (Faculty of Education at Universidad Nacional de Educación a Distancia (UNED), Spain)

RE: Permission to Use Your Computational Thinking Test (Item) in My Research

MARCOS ROMAN GONZALEZ <mroman@edu.uned.es>

Thu 11/9/2023 5:04 PM

To: BADRULIMAN BIN BATNI@BATENI <22020154@siswa.unimas.my>

Cc: Syahrul Nizam bin Junaini <syahruln@unimas.my>; badrul8903@gmail.com <badrul8903@gmail.com>

Dear Badruliman Bin Batni,

I am very pleased to share the CTtest for research and/or educational innovation purposes...

...the easiest way to share the test is through a Google Form in Drive...

...do you have a Gmail account (or equivalent) in order to do the sharing?

Best wishes,



Marcos Román González

Coordinador de Comunicación, Virtualización y Tecnologías

Facultad de Educación, Departamento MIDE I

C/ Juan del Rosal, 14, Despacho 2.18. 28040 Madrid

T. +34 91 398 90 37

mroman@edu.uned.es



MARCOS ROMAN GONZALEZ <mroman@edu.uned.es>

To: BADRULIMAN BIN BATNI@BATENI

Especificacion_items_TPC (v ...
14 KB

TPC_(english_version) - final....
3 MB

TPC_(english_version) - final....
4 MB

3 attachments (6 MB) Save all to OneDrive - UNIMAS Download all

Hi again,

I've just shared the CTt with you as a Google Form through Drive.

As you'll see, the form has an initial section named "Personal Information", which you can edit to adapt to your research purposes.

Besides, you can find attached:

- A .docx / .pdf file with the test (including correct answers).
- A .xlsx file with the specifications of the 28 items.

P.S. Here you have some references about validity studies of the test:

- Content Validity: https://www.researchgate.net/publication/290391277_COMPUTATIONAL_THINKING_TEST_DESIGN_GUIDELINES_AND_CON
- Descriptives, Percentiles, Reliability and Criterion Validity (1): <https://www.sciencedirect.com/science/article/pii/S0747563216306185>
- Criterion Validity (2): <https://www.sciencedirect.com/science/article/pii/S0747563217305563>

Appendix K: Computer Thinking Test

COMPUTATIONAL THINKING TEST (CTt)

Welcome to the Computational Thinking Test (CTt)

* Indicates required question

PERSONAL INFORMATION

Please fill out the following personal information:

1. First Name *

2. Surname *

3. Gender *

Mark only one oval.

Boy

Girl

4. School *

Write the name of the school you are currently attending.

INSTRUCTIONS

The test consists of 28 questions, divided into 7 pages with 4 questions each.

All questions have 4 answer options (A, B, C or D) from which only one is correct.

From the start of the test you have 45 minutes to do the best you can. It is not essential that you answer all questions.

To move forward from one page of the test to another, at the bottom of the page you must click on the 'Continue' button. VERY IMPORTANT: when you finish or the time ends you must move forward to the last page and click on the 'Submit' button to save your answers.

If you need to zoom in on any question to enlarge it, press 'Ctrl+' on your keyboard (or 'Ctrl-' to make it smaller).

Before starting the test, we are going to see three examples so you can familiarise yourself with the kind of questions that you will find, and the characters that will appear.

GOOD LUCK!



'Pac-Man'



Ghost



Artist

EXAMPLE I

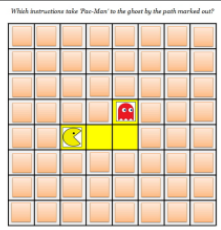
In the first example you are asked which instructions take 'Pac-Man' to the ghost by the path marked out.

That is, to take 'Pac-Man' EXACTLY to the square where the ghost is (without going over or stopping short), and strictly following the path marked in yellow (without touching the walls, represented by the orange squares).

The correct answer in this example is B. Click the "B" button below the question.

Example I

Which instructions take 'Pac-Man' to the ghost by the path marked out?



Options A

→→↓

Options B

→→↑

Options C

→↑↑

Options D

→↓↓

5. Example I *

Select the correct answer (in this example, the correct answer is B)

Mark only one oval.

A

B

C

D

EXAMPLE II

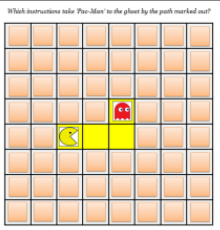
In the second example you are asked again which instructions take 'Pac-Man' to the ghost by the path marked out. However, in this question the options are presented as groups of instruction blocks rather than arrows.

Remember that the question asks you to take 'Pac-Man' EXACTLY to the square where the ghost is (without going over or stopping short), and strictly following the path marked in yellow (without touching the walls, represented by the orange squares).

The correct answer in this example is C. Click the "C" button below the question.

Example II

Which instructions take 'Pac-Man' to the ghost by the path marked out?



Options A

move forward
turn left 90°
move forward
move forward

Options B

move forward
turn right 90°
move forward
move forward

Options C

move forward
move forward
turn left 90°
move forward

Options D

move forward
move forward
turn right 90°
move forward

6. Example II *

Select the correct answer (in this example, the correct answer is C)

Mark only one oval.

A

B

C

D

EXAMPLE III

In the third example you are asked which instructions the artist should follow to draw the shape on screen. That is, how to MOVE the pencil to draw the shape.

The MOVE instruction pushes the pencil whilst drawing, while the JUMP instruction makes the artist jump to another location without drawing.

The grey arrow indicates the direction of the first movement of the pencil.

The correct answer in this example is A. Click the "A" button below the question.

176

Example III

Which instructions should the enter follow to draw the shape? The short side measures 50 pixels and the long side measures 100 pixels.

Option A move forward by 50 pixels turn left by 90 degrees move forward by 100 pixels	Option B move forward by 50 pixels turn right by 90 degrees move forward by 100 pixels
Option C move forward by 100 pixels turn left by 90 degrees move forward by 50 pixels	Option D move forward by 100 pixels turn right by 90 degrees move forward by 50 pixels

7. **Example III ***

Select the correct answer (in this example, the correct answer is A)

Mark only one oval.

☐ A

☐ B

☐ C

☐ D

QUESTIONS 1 - 4

On this page you will find questions 1 to 4.

Question 1

Which instructions take Pac-Man to the ghost by the path marked out?

Option A → → → ↓
Option B → → → ↓
Option C → → → ↓ ↓
Option D ↓ ↓ ↓ ↓

Question 2

Which step is missing in the instructions below to take Pac-Man to the ghost by the path marked out?

Option A →
Option B ←
Option C ↑
Option D ↓

9. **Question 2**

Select the correct answer

Mark only one oval.

☐ A

☐ B

☐ C

☐ D

Question 3

The instructions should take Pac-Man to the ghost by the path marked out. In which step of the instructions is there a mistake?

move forward	→ Step A
turn left	→ Step B
move forward	→ Step C
turn left	→ Step D
move forward	

10. **Question 3**

Select the step in which there is a mistake

Mark only one oval.

☐ A

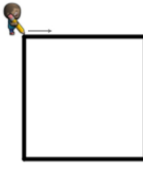
☐ B

☐ C

☐ D

Question 4

Which instructions should the writer follow to draw the square? Each of the sides of the square measures 100 pixels.



Option A move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels	Option B move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels
Option C move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels	Option D move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels turn left 90 by 90 degrees move forward 100 by 100 pixels

11. Question 4
Select the correct answer
Mark only one oval.

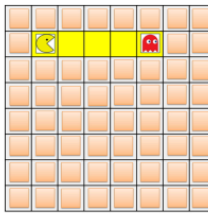
☐ A
☐ B
☐ C
☐ D

QUESTIONS 5 - 8

On this page you will find questions 5 to 8.

Question 5

Which instructions take 'Pac-Man' to the ghost by the path marked out?



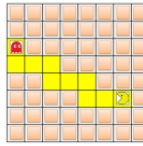
Option A + 5 	Option B + 3 
Option C + 4 	Option D + 2 

12. Question 5
Select the correct answer
Mark only one oval.

☐ A
☐ B
☐ C
☐ D

Question 6

How many times must the sequence be repeated to take 'Pac-Man' to the ghost by the path marked out?



Option A	× 2
Option B	× 1
Option C	× 4
Option D	× 3

13. Question 6
Select the correct answer
Mark only one oval.

☐ A
☐ B
☐ C
☐ D

Question 7

The instructions should make the writer draw the following rectangle once (100 pixels wide and 100 pixels high). In which step of the instructions is there a mistake?



repeat 1 times
 do
 move forward 100 by 100 pixels
 turn left 90 by 90 degrees → Step B
 move forward 100 by 100 pixels → Step C
 turn left 90 by 90 degrees → Step D

14. Question 7
Select the step in which there is a mistake
Mark only one oval.

☐ A
☐ B
☐ C
☐ D

Question 8

Which instructions take 'Pico-Max' to the ghost by the path marked out?

Option A Repeat 3 times do: repeat 3 times do: move forward turn left 90° repeat forward	Option B Repeat 3 times do: repeat 3 times do: move forward turn left 90° repeat forward
Option C Repeat 3 times do: repeat 3 times do: move forward turn right 90° repeat forward	Option D Repeat 3 times do: repeat 3 times do: move forward turn right 90° repeat forward

15. Question 8
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

QUESTIONS 9 - 12

On this page you will find questions 9 to 12.

Question 9

Which instructions take 'Pico-Max' to the ghost by the path marked out?

Option A Repeat until the... do: move forward turn right 90°	Option B Repeat until the... do: move forward turn left 90°
Option C Repeat until the... do: move forward turn left 90°	Option D Repeat until the... do: move forward turn right 90°

16. Question 9
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 10

Which step is missing in the instructions below to take 'Pico-Max' to the ghost by the path marked out?

Repeat until the...
 do:
 repeat 3 times
 do:
 move forward
 turn left 90°
 repeat forward
 turn right 90°
 repeat forward

Option A turn left 90°	Option B turn right 90°
Option C move forward	Option D Not missing any step

17. Question 10
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 11

The instructions should take 'Pico-Max' to the ghost by the path marked out. In which step of the instructions is there a mistake?

Repeat until the...
 do:
 move forward
 turn right 90°
 repeat forward
 turn left 90°
 repeat forward

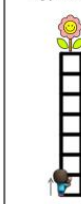
Step A: Repeat until the...
 Step B: move forward
 Step C: turn right 90°
 Step D: repeat forward

18. Question 11
Select the step in which there is a mistake
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 12

Which instructions should the artist follow to draw the ladder that reaches the flower? There are 30 pixels between each rung.



Option A <pre>repeat until the flower do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>	Option B <pre>repeat until the flower do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>
Option C <pre>repeat until the flower do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>	Option D <pre>repeat until the flower do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>

19. Question 12
Select the correct answer
Mark only one oval.

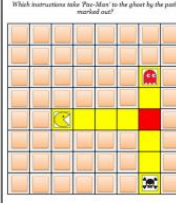
- ☐ A
☐ B
☐ C
☐ D

QUESTIONS 13 - 16

On this page you will find questions 13 to 16.

Question 13

Which instructions take Pac-Man to the ghost by the path marked red?



Option A <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>	Option B <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>
Option C <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>	Option D <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>

20. Question 13
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 14

Which instructions take Pac-Man to the ghost by the path marked red?



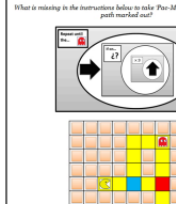
Option A <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>	Option B <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>
Option C <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>	Option D <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>

21. Question 14
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 15

What is missing in the instructions below to take Pac-Man to the ghost by the path marked red?



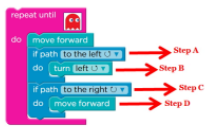
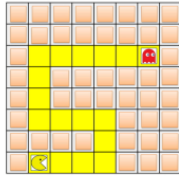
Option A <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>
Option B <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>
Option C <pre>repeat until the ghost do repeat 3 times do move forward 30 by 300 pixels turn right 90 by 90 degrees turn forward 90 by 300 pixels</pre>
Option D Both option A and option C are correct

22. Question 15
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 16

The instructions should take Pac-Man to the ghost by the path marked out. In which step of the instructions is there a mistake?



23. Question 16

Select the step in which there is a mistake

Mark only one oval.

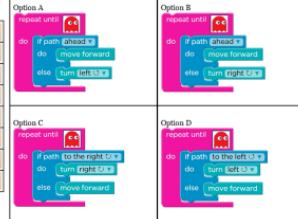
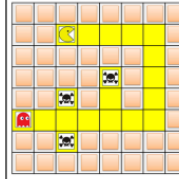
- ☐ A
☐ B
☐ C
☐ D

QUESTIONS 17 - 20

On this page you will find questions 17 to 20.

Question 17

Which instructions take Pac-Man to the ghost by the path marked out?



24. Question 17

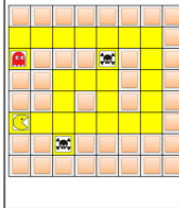
Select the correct answer

Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 18

Which instructions take Pac-Man to the ghost by the path marked out?



25. Question 18

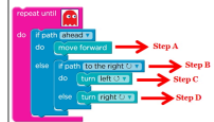
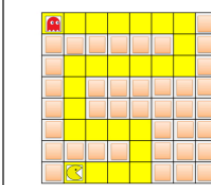
Select the correct answer

Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 19

The instructions should take Pac-Man to the ghost by the path marked out. In which step of the instructions is there a mistake?



26. Question 19

Select the step in which there is a mistake

Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 20

Which step is missing in the instructions below to take 'Pao-Man' to the phone by the path marked out?

Option A move forward	Option B turn right 90 degrees
Option C turn left 90 degrees	Option D Not missing any step

27. Question 20
Select the correct answer
Mark only one oval.

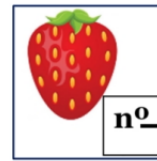
- ☐ A
☐ B
☐ C
☐ D

QUESTIONS 21 - 24

On this page you will find questions 21 to 24.

IMPORTANT: READ CAREFULLY

In this group of questions the image of a strawberry appears in some squares. The number on the bottom right of the image shows how many strawberries are in the square.



Number of strawberries that are in the square

Question 21

Which instructions take 'Pao-Man' to the strawberry by the path marked out and tell 'Pao-Man' to eat all the strawberries ahead?

Option A while path ahead do: move forward repeat 3 times do: eat 1 strawberry	Option B while path ahead do: move forward repeat 3 times do: eat 3 strawberries
Option C while path ahead do: move forward repeat 3 times do: eat 1 strawberry	Option D while path ahead do: move forward repeat 3 times do: eat 3 strawberries

28. Question 21
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 22

Which instructions take 'Pao-Man' to the strawberries by the path marked out and tell 'Pao-Man' to eat all the strawberries ahead?

Option A while path ahead do: repeat 3 times do: move forward do: eat 1 strawberry	Option B while path ahead do: move forward repeat 3 times do: eat 1 strawberry
Option C while path ahead do: repeat 3 times do: move forward do: eat 1 strawberry	Option D while path ahead do: move forward repeat 3 times do: eat 3 strawberries

29. Question 22
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 23

What is missing in the instructions below to take 'Pao-Man' to the strawberry by the path marked out and tell 'Pao-Man' to eat all the strawberries ahead?

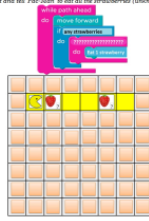
Option A 1 time
Option B 2 times
Option C 3 times
Option D 5 times

30. Question 23
Select the correct answer
Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 24

Which step is missing in the instructions below to take 'Pico-Man' to the strawberries by the path marked out and tell 'Pico-Man' to eat all the strawberries (unknown number)?



Option A

While path ahead

Option B

While no path ahead

Option C

While any strawberries

Option D

While no strawberries

31. Question 24

Select the correct answer

Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

QUESTIONS 25 - 28

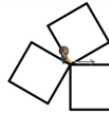
On this last page you will find questions 25 to 28. Don't forget to click on the 'SUBMIT' button when you are finished.

Question 25

The following set of instructions is called 'my function', and draws one square of 100 pixels each side.



Which instructions should the artist follow to draw the following design? Each side of each square measures 100 pixels.



Option A

ask 3 times
do: my function
end

Option B

ask 3 times
do: my function
end

Option C

ask 3 times
do: my function
end

Option D

ask 3 times
do: my function
end

32. Question 25

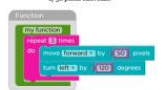
Select the correct answer

Mark only one oval.

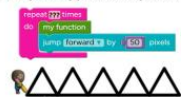
- ☐ A
☐ B
☐ C
☐ D

Question 26

The following set of instructions is called 'my function', and draws one triangle of 100 pixels each side.



The instructions below should make the artist draw the following design. Each side of each triangle measures 100 pixels. What's missing in the instructions?



Option A

15

Option B

5

Option C

4

Option D

3

33. Question 26

Select the correct answer

Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

Question 27

The following set of instructions is called 'get 5'.



Which instructions take 'Pico-Man' to the strawberries by the path marked out and tell 'Pico-Man' to eat all the strawberries (known)?



Option A

ask 5 times
do: my function
end

Option B

ask 5 times
do: my function
end

Option C

ask 5 times
do: my function
end

Option D

ask 5 times
do: my function
end

34. Question 27

Select the correct answer

Mark only one oval.

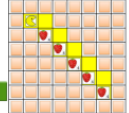
- ☐ A
☐ B
☐ C
☐ D

Question 28

The following set of instructions is called 'move and get 4'.



What is missing in the instructions below to take 'Pac-Man' to the strawberries by the path marked out and tell 'Pac-Man' to eat all the strawberries along the way?



repeat (222) times
do move and get 4

Option A 3	Option B 4
Option C 5	Option D 6

35. Question 28

Select the correct answer

Mark only one oval.

- ☐ A
☐ B
☐ C
☐ D

SELF-ASSESSMENT

Please answer these two short questions honestly. Then click on the 'Submit' button to save your answers and finish the test.

36. From 0 to 10, how do you think you did on the test? *

Mark only one oval.

- 0 1 2 3 4 5 6 7 8 9 10
Awful ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ Excellent

37. From 0 to 10, how do you think you get on with computers? *

Mark only one oval.

- 0 1 2 3 4 5 6 7 8 9 10
Awful ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ Excellent

This content is neither created nor endorsed by Google.

Google Forms

Appendix L: Cronbach's Alpha Analysis for Attitudinal Survey

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17
Q1	1.000	0.334	0.400	0.391	0.569	0.598	0.420	0.276	0.501	0.493	0.098	0.160	0.442	0.358	0.232	0.468	0.377
Q2	0.334	1.000	0.535	0.268	0.397	0.361	0.276	0.376	0.319	0.286	0.290	0.282	0.349	0.656	0.213	0.474	0.258
Q3	0.400	0.535	1.000	0.439	0.417	0.407	0.196	0.322	0.380	0.359	0.351	0.405	0.410	0.600	0.199	0.400	0.154
Q4	0.391	0.268	0.439	1.000	0.347	0.326	0.353	0.433	0.538	0.457	0.119	0.235	0.452	0.321	0.305	0.271	0.411
Q5	0.569	0.397	0.417	0.347	1.000	0.446	0.467	0.411	0.380	0.516	0.157	0.164	0.417	0.413	0.202	0.358	0.446
Q6	0.598	0.361	0.407	0.326	0.446	1.000	0.416	0.461	0.486	0.439	0.136	0.252	0.330	0.413	0.234	0.369	0.233
Q7	0.420	0.276	0.196	0.353	0.467	0.416	1.000	0.657	0.514	0.563	0.219	0.268	0.523	0.370	0.137	0.261	0.320
Q8	0.276	0.376	0.322	0.433	0.411	0.461	0.657	1.000	0.578	0.498	0.119	0.337	0.593	0.440	0.153	0.192	0.307
Q9	0.501	0.319	0.380	0.538	0.380	0.486	0.514	0.578	1.000	0.616	0.256	0.417	0.624	0.378	0.262	0.272	0.324
Q10	0.493	0.286	0.359	0.538	0.516	0.439	0.563	0.486	0.616	1.000	0.061	0.161	0.695	0.453	0.181	0.295	0.373
Q11	0.098	0.290	0.351	0.119	0.157	0.136	0.219	0.156	0.261	0.061	1.000	0.601	0.137	0.212	0.249	0.156	0.093
Q12	0.160	0.282	0.405	0.235	0.164	0.252	0.268	0.337	0.417	0.161	0.601	1.000	0.148	0.187	0.162	0.229	0.085
Q13	0.442	0.349	0.410	0.452	0.417	0.330	0.523	0.593	0.624	0.637	0.137	0.148	1.000	0.461	0.104	0.269	0.297
Q14	0.358	0.656	0.600	0.321	0.413	0.413	0.370	0.440	0.538	0.412	0.287	0.161	0.461	1.000	0.227	0.332	0.292
Q15	0.232	0.213	0.199	0.305	0.202	0.34	0.37	0.53	0.62	0.181	0.249	0.162	0.104	0.227	1.000	0.087	0.279
Q16	0.468	0.474	0.400	0.271	0.358	0.369	0.261	0.192	0.272	0.295	0.156	0.029	0.269	0.432	0.087	1.000	0.185
Q17	0.377	0.258	0.154	0.411	0.446	0.233	0.320	0.307	0.324	0.373	0.093	0.085	0.297	0.292	0.279	0.185	1.000

Appendix M: Computer Programming Attitude Scale (Five-point Likert scale)

Survey Question	Dimension	Attitude Scale				
1. I enjoy learning programming.	Interest and enjoyment	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
2. I am open to trying new programming languages and tools to enhance my skills.	Interest and enjoyment	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
3. I expect to use programming after completing my school-level education.	Interest and enjoyment	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
4. I would consider a career in programming.	Interest and enjoyment	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
5. I am sure of myself when I do programming.	Challenge and confident	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
6. I am confident that I can find and correct errors in my programming project.	Challenge and confident	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
7. I am confident that when I have a programming task, I can work independently to find a solution.	Challenge and confident	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
8. I know I can do well in programming.	Challenge and confident	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
9. I am sure I could handle advanced programming tasks.	Challenge and confident	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
10. I will need programming for my future course or study.	Challenge and confident	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree
11. How much do you agree that practicing programming regularly makes it easier over time?	Interest and enjoyment	Strongly Disagree	Disagree	Neither Agree nor Disagree	Agree	Strongly Agree

12. Compared to my other classes, I find programming:	Interest and enjoyment	Much harder	Harder	Not sure	Easier	Much easier
13. How would you rate your overall experience with programming so far?	Interest and enjoyment	Very Negative	Negative	Average	Positive	Very Positive
14. When a program I made doesn't work, I feel:	Challenge and confident	Very frustrated	Frustrated	Somewhat frustrated	Slightly frustrated	Not at all frustrated
15. How frustrated do you feel when it takes a lot of time to understand a programming concept?	Challenge and confident	Very frustrated	Frustrated	Somewhat frustrated	Slightly frustrated	Not at all frustrated
16. How often do you seek help from teachers or peers when you encounter programming difficulties?	Challenge and confident	Never	Rarely	Sometimes	Often	Very Often
17. How often do you think about pursuing a job that involves programming?	Challenge and confident	Never	Rarely	Sometimes	Often	Very Often

Appendix N: Script to Recruit Students to Volunteer for the Interview

Script to Recruit Students to Volunteer for the Interview

Dear students,

Thank you for participating in this study. We really appreciate your cooperation. As I mentioned earlier, participation in this study will not affect your class grade in any way. I hope you enjoyed the learning activities and have a better understanding of programming. As the final part of this study, I would like to conduct one-on-one interviews with twelve student volunteers. Participation in the interview is voluntary and lasts about 15 minutes. In the interview I will ask you questions about the activity and project you have involved over the past few weeks.

The purpose of these interviews is to better understand how the activities helped you learn about computational thinking and programming. I will record the interviews and later transcribe them for data analysis. No information about your identity is recorded during the conversations or assigned later. Interviews will take place either during lunchtime (after you have finished lunch) or after school, depending on availability in the school computer lab. If you would like to volunteer, please raise your hand and I will write your name. Thank you very much.

Appendix O: Interview Question

1. Describe a specific activity or lesson that you found particularly exciting. What do you like about it?
2. How do you think this activity or lesson impacted your understanding of computational thinking (CT)?
3. How, if any, did the lesson improve your understanding of complex programming concepts?
4. Can you recall a specific incident during the lesson that was particularly memorable or significant? Why was it effective?
5. Do you think participating in activities without the use of computers has improved your understanding of CT and programming? Please explain your reasons.
6. How did your attitude toward programming change before and after taking this lesson? Describe any differences in your views or feelings.

Appendix P: Published Article, “Redefining Computational Thinking: Synergizing Unplugged Activities with Block-Based Programming”

Education and Information Technologies
<https://doi.org/10.1007/s10639-024-12869-8>



Redefining computational thinking: Synergizing unplugged activities with block-based programming

Badruliman Batni¹  · **Syahrul Nizam Junaini¹** 

Received: 24 January 2024 / Accepted: 13 June 2024
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

In the dynamic educational context of Malaysia, this study examines the impact of integrating Unplugged Activities (UA) with Block-Based Programming (BBP) on improving the computational thinking (CT) skills of secondary students in full boarding schools in Northern Peninsular Malaysia. Using a quasi-experimental design and mixed-methods analysis, the research evaluates the impact of these teaching methods on students' CT skills and attitudes toward programming. This research compares the results between a group that uses only BBP and another that combines both UA and BBP. The results indicate that CT skills improved in both groups, while students in the UA+BBP group showed more significant gains in confidence and a more positive attitude toward programming. These results provide valuable insights into pedagogical strategies within digital education and highlight the benefits of an integrated approach that combines tactile learning experiences with digital technologies. By combining hands-on activities with technology-based instruction, this approach not only deepens students' understanding of CT concepts but also positively changes their perception and engagement with programming.

Keywords Computational thinking · Block-based programming · Unplugged activities · Pedagogical strategies · Computer science education

Appendix Q: Published Article, “Current Research Trends of Scratch Block Based Programming for K12: A Systematic Review”

<i>Journal of Advanced Research in Applied Sciences and Engineering Technology</i> 51, Issue 2 (2025) 138-152	
 SEMARAK ILMU PUBLISHING Journal of Advanced Research in Applied Sciences and Engineering Technology	Journal of Advanced Research in Applied Sciences and Engineering Technology Journal homepage: https://semarakilmu.com.my/journals/index.php/applied_sciences_eng_tech/index ISSN: 2462-1943
	
Current Research Trends of Scratch Block based Programming for K-12: A Systematic Review	
Badruliman Batni ¹ , Syahrul Nizam Junaini ^{1*} , Jonathan Sidi ¹ , Wan Azani Mustafa ² , Zamhar Iswandono Awang Ismail ³	
¹ Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, 94300 Kota Samarahan, Sarawak, Malaysia ² School of Engineering Technology, Kampus Sg. Chuchuh, Universiti Malaysia Perlis, 02100 Padang Besar, Perlis, Malaysia ³ Faculty of Computer Information Science, Higher Colleges of Technology, Abu Dhabi, United Arab Emirates	
ARTICLE INFO	ABSTRACT
Article history: Received 5 December 2023 Received in revised form 5 July 2024 Accepted 21 August 2024 Available online 19 September 2024	In the evolving landscape of K-12 education, the introduction of programming skills through block-based environments such as Scratch has become increasingly common. This approach meets the growing need for computer literacy among students, but its effectiveness in improving learning outcomes remains controversial. The aim of this paper is to systematically review and summarize current research on the use of Scratch as a teaching and learning tool in K-12 education. Using the PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses) methodology, this paper analyses 27 relevant articles to assess the impact and use of Scratch in educational settings. The review aims to identify the current trends, methods and foci in Scratch-related educational research. Our results show that 17 studies focus on the use of Scratch in teaching CS and ICT subjects, while 10 examine its application in other academic disciplines. The review shows generally positive results of Scratch programming in an educational context. However, it also highlights the need for more comprehensive empirical research. This includes conducting studies with larger and more diverse student samples over longer periods of time to gain a deeper understanding of how Scratch programming can effectively improve learning outcomes in K-12 education.
Keywords: Teaching; education; learning; scratch; K-12	