Three Essays on the Impact of Coding Training on Students' Learning Behavior and Outcomes

vorgelegt von
M. Sc.
Hai Anh Vu
ORCID: 0009-0007-9856-5825

an der Fakultät VII – Wirtschaft und Management
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Wirtschaftswissenschaften
- Dr. rer. oec. -

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Astrid Cullmann (TU Berlin)
Gutachter:	Prof. Dr. Axel Werwatz (TU Berlin)
Gutachter:	Dr. Arne Thomas (University of Amsterdam)

Tag der wissenschaftlichen Aussprache: 27. Februar 2024

Berlin 2024

Abstract

Motivated by the substantial changes instigated by the Fourth Industrial Revolution and its significant implications for education, especially in developing countries like Vietnam, this dissertation meticulously explores the influence of coding education within Vietnam's educational reforms. This comprehensive study consists of three distinct essays, each addressing crucial aspects: (1) evaluating the impact of coding courses on secondary school students' engagement in self-directed coding activities, (2) assessing the effect of teaching coding through visual programming languages on students' computational thinking abilities, and (3) establishing and validating the Coding Motivation Scale for Middle School Students (CMS-M), a reliable instrument designed to measure students' motivation, attitudes, and interests in coding within Vietnam's educational context. The study conducted multiple rounds of surveys in Vietnam to collect data for its analysis. These data sets were then analyzed using both quantitative and qualitative approaches. In detail, the study employed panel data estimation and multivariate statistical techniques, as well as conducted focus group discussions and in-depth interviews. The core findings reveal that coding education exerts a substantial impact. Mastery of coding not only fosters self-directed learning behaviors but also enhances students' computational thinking skills. These outcomes are primarily attributed to the solidification of foundational knowledge and the students' increased interest cultivated through hands-on coding projects.

Zusammenfassung

Motiviert durch die bedeutenden Veränderungen, die durch die vierte industrielle Revolution ausgelöst wurden, und deren erhebliche Implikationen für die Bildung, insbesondere in Entwicklungsländern wie Vietnam, untersucht diese Dissertation sorgfältig den Einfluss der Programmierausbildung im Rahmen der Bildungsreformen Vietnams. Diese umfassende Studie besteht aus drei unterschiedlichen Aufsätzen, die jeweils wichtige Aspekte behandeln: (1) die Bewertung der Auswirkung von Programmierkursen auf das Engagement von Sekundarschülern bei selbstgesteuerten Programmieraktivitäten, (2) die Beurteilung der Effekte des Lehrens von Programmierung durch visuelle Programmiersprachen auf die Fähigkeiten der Schüler im berechnenden Denken, und (3) die Entwicklung und Validierung der Skala zur Motivationsmessung beim Programmieren für Mittelschüler (CMS-M), ein zuverlässiges Instrument, das darauf abzielt, die Motivation, Einstellungen und Interessen der Schüler am Programmieren im Bildungskontext Vietnams zu messen. Die Studie führte mehrere Umfragerunden in Vietnam durch, um Daten für ihre Analyse zu sammeln. Diese Datensätze wurden dann sowohl mit quantitativen als auch qualitativen Ansätzen analysiert. Im Detail verwendete die Studie Schätzungen mit Paneldaten und multivariate statistische Techniken sowie die Durchführung von Fokusgruppendiskussionen und ausführlichen Interviews. Die Kernergebnisse zeigen, dass die Programmierausbildung einen erheblichen Einfluss ausübt. Die Beherrschung der Programmierung fördert nicht nur selbstgesteuerte Lernverhalten, sondern verbessert auch die Fähigkeiten der Schüler im berechnenden Denken. Diese Ergebnisse sind hauptsächlich auf die Festigung des Grundlagenwissens und das durch praktische Programmierprojekte gesteigerte Interesse der Schüler zurückzuführen.

Acknowledgements

This research journey has marked my first deep dive into serious academic work. It has provided me with invaluable insights into the mechanics of research and the satisfaction of resolving complex research challenges. Moreover, this thesis has played a significant role in shaping my career interests, offering a clearer path forward. Reflecting on this accomplishment fills me with immense gratitude for those who supported me, enabling the successful completion of this thesis.

Firstly, I extend my deepest appreciation to Professor Axel Werwatz, my primary advisor at Fakultät VII - Wirtschaft und Management, Technische Universität Berlin. His supervision has been pivotal in providing me with a profound understanding of econometrics and invaluable guidance throughout this journey. His investment of time, expertise, and assistance to extend my study has been invaluable. Many techniques employed in this thesis stemmed from his remarkable lectures and expert advice.

I also wish to express sincere thanks to my second advisor, Arne Thomas from the Faculty of Economics and Business, University of Amsterdam. His consistent encouragement and unwavering support were invaluable in helping me overcome the challenges I encountered during my time in Germany. Completing my study within the given timeframe would have been unattainable without his assistance.

Special acknowledgment is owed to my colleagues at the School of Economics, UEH University in Vietnam, for their indispensable assistance in data collection and administrative management throughout this research project.

My heartfelt gratitude extends to my love, whose constant support, encouragement, and constant companionship have been a continuous source of strength. Her role as a passionate and tireless mentor significantly contributed to this thesis's success. I am deeply indebted to my parents for their enduring support and financial assistance, without which extending my studies and acquiring knowledge would not have been possible.

Lastly, I am immensely thankful to The Dariu Foundation for their invaluable contribution through research funding. Their support has been instrumental in enabling the crucial data collection necessary for this thesis.

Contents

Preface

The Fourth Industrial Revolution, often referred to as Industry 4.0, is a global transformation with profound effects across sectors. While it offers opportunities like increased productivity and resource efficiency, it also poses challenges such as labor market disruptions and changing investment patterns, especially impactful in developing countries. To tackle these challenges, developing nations must prioritize education strategies that reskill and upskill their workforce to meet evolving job demands.

As a typical developing country, Vietnam has recognized its vulnerability to the challenges posed by Industry 4.0 and has taken proactive steps to address them. Vietnamese authorities have heeded the earlier recommendations and initiated educational reforms with a focus on enhancing digital and analytical skills among the younger generation. This dedication is evident in the government's strong emphasis on STEM and informatics education, an approach widely supported by parents and educators who recognize the importance of coding skills in preparing students for Industry 4.0. Despite the widespread enthusiasm for coding education in Vietnam, there exists a notable research gap in assessing its impact on the abilities, interests, and attitudes of Vietnamese students toward STEM and coding. To address this gap, this thesis, consisting of three independent essays, seeks to analyze the effectiveness of coding education within Vietnam's new teaching and learning reforms.

In the first essay, we examine how secondary school students engage in self-study coding activities after completing basic coding courses without external incentives. The objective is to comprehend whether program participation influences students' commitment to self-study and the underlying mechanisms. We employ a mixed-method approach, combining fixed effect estimation with qualitative methods, and focus on two types of self-learning: self-online learning and peer-to-peer self-learning. Our research reveals that coding training predominantly enhances students' dedication to self-online learning. Importantly, this effect is not primarily due to changes in students' motivation or increased access to online resources but is closely tied to the development of a more substantial and robust knowledge foundation.

In the same manner, we analyze the impact of teaching coding with visual programming languages on students' computational thinking skills in the second essay. Our research demonstrates that coding instruction has a positive influence on students' computational thinking scores, especially among Grade 7 students learning MakeCode, whereas the effect is less pronounced among Grade 6 students using Scratch. These findings gain strong support from qualitative insights obtained through interviews with IT educators and educational experts. The disparity in outcomes can be attributed to MakeCode for Micro:bit's emphasis on robotics, sensors, and wearables, which fosters a more robust connection between the digital and physical realms compared to Scratch. This emphasis not only heightens students' interest but also substantially enhances their computational thinking skills.

The third essay turns to another research area, involving the development and validation of a reliable scale for measuring students' motivation, attitudes, and interests towards coding in Vietnam. Guided by (Situated) Expectancy Value Theory and informed by existing scales in the field, we crafted the initial version of the Coding Motivation Scale for Middle School Students (CMS-M). Subsequently, the first version of CMS-M underwent iterative adjustments and validation steps, leading to the final version of the scale. The ultimate CMS-M comprises 36 items, which are grouped into 7 latent constructs. These factors encompass coding confidence, coding interest, gender perception, general usefulness, academic usefulness, academic self-efficacy and attitudes, and social perception. The final CMS-M exhibits commendable psychometric properties, including satisfactory reliability and a robust internal structure. Especially, the measurement invariance property of the CMS-M across different gender and age groups holds and has practical implications for evaluation study in Vietnam.

While the primary focus of the third essay does not lie in the direct analysis of the impacts of coding training, it does, however, introduce a valuable instrument for gauging students' motivation, attitudes, and interests in coding within the context of Vietnam. This instrument serves as a cornerstone for assessing the efficacy of coding education in light of the new teaching and learning reforms taking place in Vietnam. In contrast to the third essay, the first and second essays thoroughly explore the analysis of changes resulting from the acquisition of coding skills. These changes are assessed from different perspectives, with the first essay specifically investigating alterations in self-directed learning time and the second essay placing a strong emphasis on the development of computational thinking skills. Despite the distinct approaches to measuring outcomes, both essays contribute to the establishment of a causal link between learning how to code and these transformative outcomes. Ultimately, this collaborative research aims to advance our understanding of coding education within the K12 educational framework, particularly in developing countries such as Vietnam.

Lists of Tables

Chapter 1

Chapter 2

Chapter 3

Lists of Figures and Boxes

Chapter 1

Chapter 2

Chapter 3

Lists of Abbreviations

Abbreviation	Meaning	Page
ADB	Asian Development Bank	2
BTS	Bartlett's Test of Sphericity	85
CFA	Confirmatory Factor Analysis	68
CFGP	Coding for Future with Google Project	5
CFI	Comparative Fit Index	83
CMS-M	The Coding Motivation Scale for Middle School Students	63
CPAS-M	Computer Programming Attitude Scale for Middle Schoolers	75
CPSES	Computer Programming Self-Efficacy Scale	72
CS	Computer Science	44
CSTA	Computer Science Teachers Association	36
CT	Computational Thinking	34
Df	Degree of freedom	103
E-CSA	Elementary Computer Science Attitudes	75
EFA	Exploratory Factor Analysis	68
EVT	Expectancy Value Theory	70
Google APAC	Google Asia Pacific Pte. Ltd.	5

Abbreviation	Meaning	Page
ILO	International Labor Organization	2
Industry 4.0	The Fourth Industrial Revolution	7
IT	Information Technology	6
K-12	Kindergarten to 12th grade education	26
KMO	Kaiser-Meyer-Olkin	82
MCAR	Missing Completely at Random	107
MG-CS	The Computer Science Attitudes Scale for Middle-Grade Students	71
MIT	Massachusetts Institute of Technology	35
MLR	Robust Maximum Likelihood	83
NGO	Non-Governmental Organization	34
OECD	Organisation for Economic Co-operation and Development	4
PES	Programming Empowerment Survey	72
RMSEA	Root Mean Square Error of Approximation	85
SCAPA	Self-Concept and Attitude toward Programming Assessment	71
SEM	Structural Equation Modeling	69
SEVT	Situated Expectancy Value Theory	70
SRMR	Standardized Root Mean Square Residual	83
STEM	Science, Technology, Engineering, and Mathematics	34
STEM-CIS	STEM Career Interest Survey	75

Abbreviation	Meaning	Page
TDF	The Dariu Foundation	5
TLI	Tucker–Lewis Index	83
UNDP	United Nations Development Programme	64
UNESCO	United Nations Educational, Scientific, and Cultural Organization	34
UNIDO	United Nations Industrial Development Organization	64
WLSMV	Weighted Least Squares Mean and Variance Adjusted	83

Chapter 1

The Impact of a Brief Coding Program on the Secondary School Students' Engagement in Coding Self-Learning Activities

Abstract: Using a mixed method design combining a fixed effect design and qualitative methods, we analyze the effect of coding courses provided within a digital literacy program on students' self-learning behavior in Lam Dong province of Vietnam after the program finished. Following the recommendations provided by Hong and Park (2012), we measure self-study efforts through the amount of time that students spend on two coding learning activities: self-online learning and peer-to-peer self-learning. Our analysis reveals that among two forms of self-learning activities, coding training only has a positive effect on students' self-online learning effort. Our explorations show further that this effect is less likely to be driven by changes in students' motivation or access to additional online resources. Instead, it appears to be linked to a more solid knowledge foundation.

1. Introduction

The remarkable development of digital technology in recent years has led to widespread digitization and automation across all areas of the world economy. These technological innovations present new opportunities for economic growth, providing people with greater access to flexible services and employment opportunities at lower costs, thereby contributing to poverty reduction and income growth (ADB, 2021; ILO, 2019). However, digitalization also imposes significant challenges, particularly for labor-intensive economies that rely on unskilled or low-skilled workers. One of the most pressing issues among these challenges is how to address skills mismatches in an ever-changing labor market (ADB, 2021; Charles, Xia & Coutts, 2022).

In a series of reports on the Future of Work, the World Economic Forum estimated that more than 65% of children at primary school today would eventually work in jobs that have not yet existed (Schwab & Zahidi, 2020). Moreover, by 2025, approximately 50% of current employees worldwide will require reskilling to keep up with new technologies by 2025 (Schwab & Samans, 2016). To meet the demands of the future workforce, the World Bank (2019) emphasizes the need for new knowledge and skills training, particularly in digital literacy and coding, to enable employees to adapt to new technologies and automation.

Recognizing the importance of digital and coding skills in the digitized economy, many governments and educational institutions worldwide explore various approaches providing coding education programs. Some countries have already integrated coding into their formal curricula for young children, while others have launched similar but informal initiatives complementary to formal school activities (Balanskat & Engelhardt, 2014; Sáez-López et al., 2016). However, in developing countries, these programs often face resource constraint, including a lack of trained teachers, inadequate infrastructure, and limited time for delivering the training, which makes them challenging to sustain over the long term.

One possible solution for this problem is to provide training for students on fundamental coding concepts and help them develop a coding mindset. By building a strong foundation through such training, students may be motivated to pursue self-study and deepen their knowledge in the field. Similar low-cost programs have been implemented for other subjects like literacy or mathematics with disadvantaged children in remote areas or out of school in developing countries, demonstrating the feasibility of this approach (Asadullah, 2016; Sawada et al., 2022).

Advancements in technology have also made this solution more feasible. The progress in internet accessibility, facilitated by 3G/4G/5G systems, along with the affordability of tablets and smartphones, has significantly democratized self-learning opportunities for students. Additionally, the abundance of online educational resources has further enhanced the accessibility, efficiency and convenience of self-learning. Therefore, it is reasonable to expect that this solution is also feasible for coding training. Yet a critical remaining issue is whether students will engage in self-study after completing the basic coding courses, especially when external incentives are absent.

In this study, we try to examine this issue and investigate the effects of a short-term coding program on secondary school students' engagement in coding self-learning activities after the coding training finished. Particularly, we seek answers to two research questions: does participation in the program affect students' coding self-study efforts afterward? And if there is an effect, what are the mechanisms driving it? By answering these questions, our study can provide insights into the effectiveness and feasibility of low-cost, short-term coding education programs in enhancing students' coding skills and knowledge.

Using a mixed method design combining a fixed effect design and qualitative methods, we analyze the effect of coding courses provided within a digital literacy program on students' self-learning behavior in Lam Dong province of Vietnam after the program finished. We measure self-study efforts through the amount of time that students spend on two coding learning activities: self-online learning and peer-to-peer self-learning (Hong & Park, 2012). Our analysis reveals that among two forms of self-learning activities, coding training only has a positive effect on students' self-online learning effort. Our explorations show further that this effect is less likely to be driven by changes in students' motivation or access to additional online resources. Instead, it appears to be linked to a more solid knowledge foundation.

The remainder of this paper is structured as follows. Section 2 presents the theoretical background of our research. Section 3 outlines our research setting, including a description of the coding program, study design, data collection procedures, and variable measurement. Section 4 details our econometric approach, while section 5 presents the results of our analysis. In section 6, we discuss our results and conclude our paper.

2. Theoretical Background

2.1. Self-Learning

In education literature, the concept of self-learning is often associated with several other concepts such as independent learning (Harden, 2009), self-regulated learning (Zimmerman, 2013) and self-directed learning (Dewey, 2022; Knowles, 1975; Rogers & Freiberg, 1994; Tough, 1989). Despite potentially carrying distinct connotations, these concepts share many similar aspects and are frequently used interchangeably. At their core, these concepts emphasize a learning process in which students take charge of designing and controlling their learning independently and study on their own. In this sense, self-learning can be defined as both the ability and the learning process by which students learn and advance new knowledge on their own, without the need for direct teaching or instruction (Knowles, 1975).

Self-learning can occur in various settings and formats, each tailored to accommodate diverse learning preferences and objectives. Traditionally, students engage in self-study by dedicating time outside the classroom to work independently, utilizing textbooks and other reference materials (OECD, 2011). Alternatively, self-study can manifest collaboratively in group settings, where students join forces to exchange knowledge and support each other's learning endeavors. In recent years, the increasing flexibility and accessibility of online learning resources have empowered students to engage in self-online learning more conveniently and efficiently, making a transformative shift in the learning landscape (Future Learn, 2022).

Developing self-learning is crucial for students to succeed in both academics and their future careers. Research demonstrates that engaging in self-study enhances students' comprehension of regular school lessons, potentially resulting in improved academic performance (Kistner et al., 2010; Paris & Paris, 2001; Zimmerman, 2002). In today's rapidly changing world, where continuous skill development and lifelong learning is essential for professional growth, self-learning ability is more critical, as it allows individuals to adapt to new challenges, acquire new skills, and stay competitive (Barbosa et al., 2016; Fontana et al., 2015). Moreover, with the growing accessibility and flexibility of digital technology and online learning platforms, self-learning has emerged as a cost-effective approach to enrich children's knowledge and skills, especially in developing nations where access to high-quality formal education might be constrained (Jha & Ghatak, 2023; Sawada et al., 2022).

2.2. Engaging in Self-Learning

Self-learning, although highly desirable (OECD, 2011), does come with significant challenges. Engaging in self-learning activities demands considerable effort from students and is influenced by a multitude of both internal and external factors. Externally, the creation of an “enabling environment” (Meyer, 2008, pp.21) is essential to support students’ self-learning processes. This environment includes but is not limited to physical spaces conducive to study, access to appropriate learning resources, and necessary assistance from teachers and peers during the learning journey (MacBeath, 1993).

Internally, students must possess the necessary knowledge foundation and cognitive skills to effectively undertake self-study (Meyer, 2008). Equally vital as intellectual capabilities are intrinsic motivation, comprised of essential components such as interest, confidence, and outcome expectations (Zimmerman, 2002). Interest refers to students’ curiosity, attention and liking that students have towards the subject. Confidence pertains to students’ beliefs in their ability to learn and solve problems independently. Outcome expectations refer to students’ perceptions of the usefulness of their learning (Bandura, 1977, 1997; Zimmerman, 2002). Research shows that students with high self-efficacy, strong interest in the subject, and high expected values of knowledge and skills acquired from learning are more likely to actively engage in self-learning activities (Harden, 2009; Meyer, 2008; Zimmerman, 2002).

3. Research Setting

3.1. The “Coding for Future with Google” Project

From 2019 to 2021, The Dariu Foundation (TDF) and Google APAC launched a digital literacy project named “Coding for Future with Google” (CFGP). The project aimed to help children develop the digital skills needed in the digital age by providing free coding courses. CFGP had a focus on poor, remote communities where digital access and training for children was limited. By doing so, it hoped to empower rural youth to successfully participate in the digital economy and thus improve their overall well-being.

CFGP was implemented in two stages. In the first stage, CFGP worked with local government officials in rural areas to select and invite targeted schools to participate in the project. If these schools agreed to join, CFGP then lent free laptops to them and trained their IT teachers to teach coding using Scratch and Micro:bit.

In the second phase, the trained teachers organized coding courses for their students at the targeted schools, using equipment and materials provided by CFGP. The teaching curriculum and materials were prepared and validated by IT educational experts of the University of Technology Ho Chi Minh City. Teachers were able to modify CFGP's recommended curriculum to suit local circumstances, but all courses were designed to cover all compulsory learning content and concepts. Each standard course comprised eight 45-minute sessions over one semester. At the end of each course, students were evaluated by an online assessment or a written exam.

3.2. Study Design

The data used in this study was collected through an impact evaluation of the "Coding for Future with Google" Project in 2020. The evaluation was designed and implemented by a group of researchers from the University of Economics, Ho Chi Minh City, of which the author is a member. The study received the Institutional Review Board (IRB) approval from the University of Economics Ho Chi Minh City (IRB Study No. 1660 /QD-DHKT-QLKHHTQT).

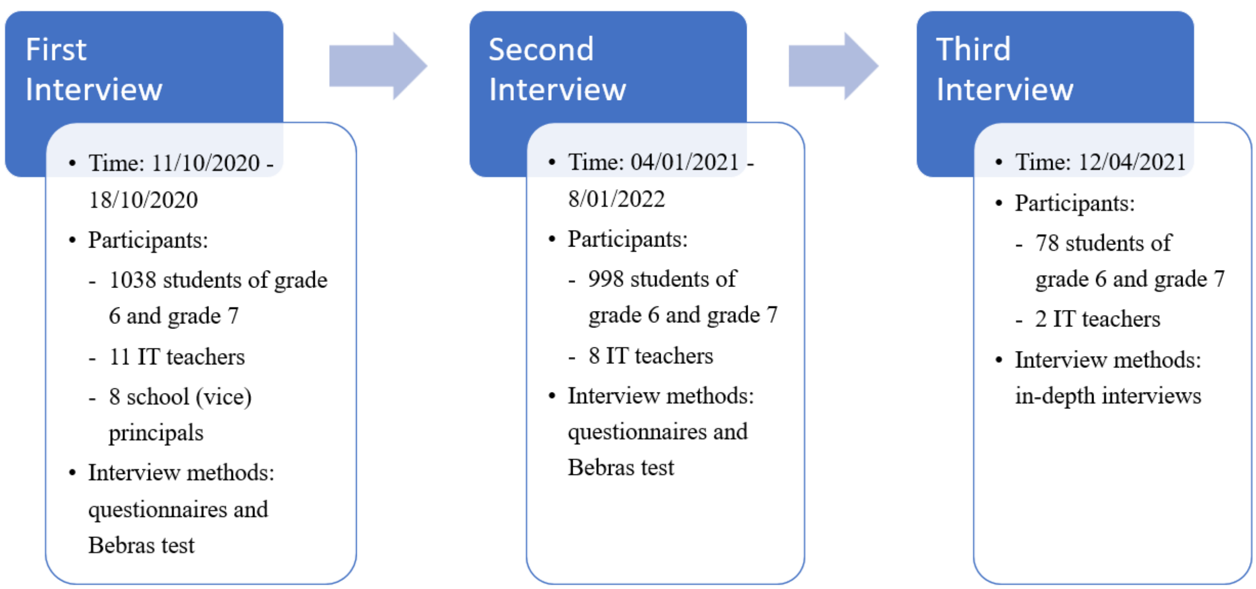
The objective of the evaluation is to assess the short-term impact of the coding training provided by CFGP on students' learning behavior and computational thinking skills. Eight schools in Lam Dong province were chosen for this study. They were schools participating in the project but had not carried out any activities at the time the evaluation study began, creating a controlled setting for the evaluation.

Ideally, randomization would be used to assess the impact of an intervention on beneficiaries. However, due to the local situation and the agreement between TDF and the targeted schools, it was difficult for the evaluation team to implement any form of randomization. To overcome this problem, the evaluation team adopted a quasi-experimental approach and designed a control-treatment pre-posttest study.

Particularly, at the beginning of the academic year 2020 - 2021 (October 2020), we conducted the baseline interview with 1038 students from 8 schools in Lam Dong. At that time, none of these students had participated in any coding training by the project. Half of the students in this sample were then assigned to the treatment group and received coding training by CFGP at their schools during the first semester of the year 2020 - 2021 (October 2021 - January 2021). The other half were assigned to the delayed-entry control group and only received the training in the second semester of the year 2020 - 2021. The schools were responsible for the treatment-control assignment on a non-random basis, convenient basis.

We conducted the follow-up interview at the end of the first semester in January 2021 - two weeks after all coding courses ended. The number of students re-interviewed was 998, indicating an impressively low dropout rate of around 3.85%. Finally, in April 2021, an in-depth interview was organized with the participation of 78 students from the treated group. Figure 1 summarizes the timeline of our evaluation study.

Figure 1: Timeline of the Evaluation



Note: The interviews were conducted in one-on-one and in-person format.

The unit of analysis is individual students. Our final data set includes information on 998 secondary school students between the ages of 12 and 13. The longitudinal nature of the data allows us to estimate the effect of coding training on students' learning behavior although we could not conduct a randomized control evaluation.

3.3. Variables

Dependent Variable: Self-learning variable

Self-learning requires students to make a variety of choices, from selecting learning content to determining the pace and location of their learning (Harden, 2009). Among these choices, the decision regarding the amount of time students devote to self-study is perhaps the most fundamental. It reflects not only the level of self-discipline of students but also their commitment to self-learning. Moreover, since coding courses in our setting were extracurricular and not included in the final grade of students, spending more time self-studying coding strongly demonstrates students' willingness to study (OECD, 2011). Therefore, in this study, we utilize the amount of time students devote to self-study as a measure of their self-learning effort. Following OECD (2011), we measure students' self-study time on coding through a self-reported question. We ask students to estimate how much time they spend on self-study after they attend the coding course. The precise phrasing of two questions is displayed in Box 1.

Box 1: Questions to Measure Students' Self-study Time on Coding

In the last two weeks, how much time do you typically dedicate to self-online learning for coding?				
1 Never (No time)	2 Little (Less than 2 hours per week)	3 Somewhat (from 2 to 4 hours per week)	4 Much (from 4 to 6 hours per week)	5 A Great Deal (more than 6 hours per week)
In the last two weeks, how much time do you typically dedicate to self-learning with your friends for coding?				
1 Never (No time)	2 Little (Less than 2 hours per week)	3 Somewhat (from 2 to 4 hours per week)	4 Much (from 4 to 6 hours per week)	5 A Great Deal (more than 6 hours per week)

We emphasize that the self-study time recorded here is the study time after students finished the coding course to avoid artificial effects as students may study more when they were offering the course in the class (although the class was an extra course). We focus on two formats of self-study: self-online learning and self-learning with peers. Particularly, for each format, students were asked to select one of five categories: "Never (No time) "Little (less than 2 hours per week)", "Somewhat (from 2 to 4 hours per week)", "Much (from 4 – 6 hours per week)", "A Great Deal (more than 6 hours per week)". Since students who reported spending less than 2 hours appeared to have difficulty describing their activities during this self-learning time, suggesting that they may not have engaged in meaningful self-learning activities, I converted this variable to 0 (spent less than 2 hours) and 1 otherwise.

Treatment variable: Coding training

Our treatment variable, *Coding training*, is a binary variable assigned a value of 1 if a student has received coding training, and 0 if they have not. Consequently, at the initial time point (t = 0), *Coding training* is set to 0 for all students. Then, at the subsequent time point (t = 1), it is adjusted to 1 for students belonging to the treated group who received coding training during the semester, while it remains at 0 for students who did not participate in coding training during the same period.

Control variable

Two important time-varying[1] factors are used as the control variables in our regressions. First, students' access to computers and laptops may influence their coding learning and consequently their cognitive skills. Therefore, we control computer access in our analysis by using a dummy variable, *PC*. Students were asked to state how frequently they used computers and laptops outside of school (at home or at public shops) on a five-point scale. The dummy variable *PC* is equal to 1 if students rated their usage frequency from 3 to 5, and 0 otherwise.

Second, the acquisition of cognitive skills through coding learning may also be subject to the time students spend on household responsibilities to assist their families. Students who must allocate their time and cognitive resources to household tasks may find it challenging to dedicate the focused attention necessary for cognitive development through coding training. To account for this potential effect, we include a control variable that captures the amount of time students spend on daily household chores. Students were particularly asked to select one of six categories representing their

[1] Since we use fixed-effect models to solve the problem of unobserved confounders, we do not include in our estimations all time-invariant control variables such as gender, age, family's income or parents' educational levels.

daily household chore time: “No household chore time”, “Up to 1 hour”, “Up to 2 hours”, “Up to 3 hours”, “Up to 4 hours”, and “More than 4 hours”. These categories were then coded into an ordinal variable ranging from 0 to 5, where 0 represented “No household chore time” and 5 denoted “More than 4 hours”.

4. Econometric Consideration

4.1. Identification Strategy

Our interest is in how participating in coding courses influences students’ self-study time. Ideally, if the coding courses were delivered randomly, a simple comparison between self-study time of the treated group and the control group would be enough. However, since the treatment assignment process was not fully random, the treatment-control comparison is subject to endogeneity problems.

For instance, numerous studies have identified a connection between socioeconomic factors, parental engagement, and students' academic dedication. Maltais et al. (2021) emphasize that parents' concern for academic performance can catalyze student motivation, thereby influencing their self-directed learning habits. This insight aligns with the findings of Davis-Kean (2005), who highlighted that socioeconomic status, specifically parents' education and income, indirectly shapes children's academic performance through parental beliefs and behaviors.

Additionally, Muilenburg and Berge (2005) identified multiple factors influencing students' engagement with online learning, prominently featuring aspects like technology accessibility and the cost of internet connectivity. Notably, students' online learning time is markedly affected by the availability of computers and reliable internet at home. Thus, it becomes imperative to account for these socioeconomic variables in regression analyses.

Due to the inability to fully control these variables, the estimates are susceptible to bias arising from endogeneity. To address this issue, we employ the individual fixed effect approach, utilizing repeated observations on students within our dataset. The core assumption of this approach is that unobserved confounding variables remain constant over time. In our study, this assumption holds true due to the short interval between the baseline survey conducted in October 2021 and the post-treatment survey, which took place only four months later, after all coding courses had concluded. Given the short time span, significant changes are unlikely to occur. Therefore, we are confident that the identification assumption of the fixed-effect approach is applicable to our case.

4.2. Econometric Specification

Since our identification strategy is based on canceling time-invariant unobserved confounders, we need to use a linear, additive function form, or a nonlinear model with a statistic that can drop out these time-invariant terms during transformation. Therefore, we first use a linear probability fixed-effect model with robust standard errors clustered at the school level to estimate our treatment effect. We then estimate the logit model as a robustness check of our results.

The model for estimation is:

$$Y_{it} = \beta D_{it} + X_{it}\gamma + \mu_i + \lambda_t + \varepsilon_{it}$$

in which Y_{it} represents the measurement of self-study time; D_{it} denotes the treatment variable and X_{it} are the control variables. With this model, we conclude that coding training (D) has an effect on students' computational thinking skills (Y) when $\hat{\beta}$, the estimate of β, is statistically different from zero.

5. Results

5.1. Sample Composition

Table 1 provides a comprehensive overview of the sample composition. The data encompasses 1038 students who participated in the first round of interviews. Approximately 52.12% of the sample comprises students in grade 6, with the remaining students (47.88%) belonging to grade 7. Gender distribution in the sample is well-balanced, with approximately half (50.58%) being male students and the other half (49.42%) being female students.

At the time of the first interview, none of the students had received any coding training from the project. Subsequently, a coding training session was delivered to a portion of the students, following which all students were re-interviewed during the second survey. However, out of the total 1038 participants, only 998 students provided valid and complete responses. Therefore, the final sample used for the analysis comprises a balanced dataset of 998 observations, with a dropout rate of 3.85%.

Table 1: Composition of the Sample

		First Round (October 2020)		Second Round (January 2021)	
		Freq. (N)	**Perc. (%)**	**Freq. (N)**	**Perc. (%)**
Total student:		**1038**	**100%**	**998**	**100%**
Grade					
	- *Grade 6*	541	52.12%	523	52.40%
	- *Grade 7*	497	47.88%	475	47.60%
Gender					
	- *Male*	525	50.58%	503	44.72%
	- *Female*	513	49.42%	495	55.28%

Note: First round and second round of surveys were conducted in Lam Dong Province, in Vietnam

In the final sample, 6th and 7th-grade students account for 52.40% and 47.60%, respectively, while approximately 55.28% of the students are female, and the remaining are male. Table 2 presents a tabulation of the final sample based on gender and grade, illustrating an equal distribution of students across sub-groups.

5.2. Descriptive Statistics of Variables

Table 3 provides a concise overview of all variables included in the regression analysis. Among the students in the final sample, approximately 46.9% had participated in the coding training. Interestingly, only 46% of students had access to a computer during the initial round of interviews, but this figure slightly increased to 47.6% in the second survey.

Regarding household chores, the time students spent on these activities remained quite consistent between the two rounds. Only 4.8% and 6.3% of students did not engage in any household chores during the first and second surveys, respectively. Conversely, a considerable proportion of students, 63% in the first survey and 62% in the second survey, allocated less than 1 hour to attend to household responsibilities. Moreover, the percentage of students involved in household chores decreased significantly as the working time increased

Table 2: Tabulation of the Final Sample by Gender and Grade

	Gender	Grade	No. of Observations (Collected)	No. of Observations (Analysis)	Percentage (Analysis column)
First Round			**1038**	**998**	**100 %**
	Boy				
		6	276	265	26.55%
		7	249	238	23.85%
	Girl				
		6	265	258	25.85%
		7	248	237	23.75%
Second Round			**998**	**998**	**100 %**
	Boy				
		6	265	265	26.55%
		7	238	238	23.85%
	Girl				
		6	258	258	25.85%
		7	237	237	23.75%

Note: First round and second round of surveys were conducted in Lam Dong Province, Vietnam

Table 3: Summary of Variables in the Regression

	Survey Round					
	Round 1		Round 2		Total	
	No.	%	No.	%	No.	%
Coding training						
No	998	100.0	530	53.1	1,528	76.6
Yes	0	0.0	468	46.9	468	23.4
Total	998	100.0	998	100.0	1,996	100.0
Grade						
Grade 6	523	52.4	523	52.4	1,046	52.4
Grade 7	475	47.6	475	47.6	950	47.6
Total	998	100.0	998	100.0	1,996	100.0
Online self-learning time (per week)						
Less than 2 hours	829	83.1	732	73.3	1,561	78.2
Higher than or equal to 2 hours	169	16.9	266	26.7	435	21.8
Total	998	100.0	998	100.0	1,996	100.0
Computer access						
No	539	54.0	523	52.4	1,062	53.2
Yes	459	46.0	475	47.6	934	46.8
Total	998	100.0	998	100.0	1,996	100.0
Time for working household chores per day						
No household chore time	48	4.8	63	6.3	111	5.6
Up to 1 hour	629	63.0	619	62.0	1,248	62.5
Up to 2 hours	213	21.3	229	22.9	442	22.1
Up to 3 hours	66	6.6	58	5.8	124	6.2
Up to 4 hours	26	2.6	17	1.7	43	2.2
More than 4 hours	16	1.6	12	1.2	28	1.4
Total	998	100.0	998	100.0	1,996	100.0

Note: Coding interest (single item): Assessed via one question about overall interest; Coding confidence (single item): Assessed via one question about overall confidence; Coding interest (multiple items): Calculated as the average of multiple items from the ESCAS scale; Coding confidence (multiple items): Calculated as the average of multiple items from the ESCAS scale.

Especially, prior to undergoing coding training, about 16.9% of students allocated a minimum of 2 hours per week to engage in online coding learning. After the training, this metric observed a significant rise to 26.7%. Nevertheless, this upsurge in participation cannot be singularly regarded as compelling proof of a direct cause-and-effect relationship between the training and the extension of students' online self-learning duration. A range of additional factors, both evident and concealed, holds the potential to skew these results. For instance, variables such as parental educational background and household income might exert an influence.

Upon examining the correlation coefficients outlined in Table 4, several key findings emerge. To begin with, a robust and positive correlation emerges between students' online self-learning time and their participation in the training program. In simpler terms, those who took part in the training tended to spend more time engaged in self-directed online learning compared to their untrained counterparts. However, no such correlation was found between students' self-learning time with peers and their participation in the training.

Table 4: Correlations of Variables in the Regression

	Polychoric Correlation					
	1	2	3	4	5	6
1. Online self-learning time	1					
2. Self-learning time with peers	0.69	1				
3. Coding training	0.30***	0.01	1			
4. Grade	-0.15***	-0.17***	0.04	1		
5. Computer access	0.37***	0.50***	-0.06	0.06	1	
6. Time for working household chores	0.05	0.06*	-0.09	-0.01	0.01	1

*Note: * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$*

Furthermore, both self-learning time with peers and online demonstrate statistically significant and positive correlations with computer access. This observation is in line with expectations, as students who have access to computers typically invest more time in online learning activities.

Additionally, the correlations reveal a noteworthy negative relationship between grade level and self-learning time, both with peers and online. This suggests that older students generally allocate less time to self-directed learning. Lastly, the remaining variables scrutinized in the analysis do not exhibit significant correlations with each other. This implies that they are relatively independent of one another within the context of this study.

5.3. Effects of Coding Training on Self-Learning Activities

Table 5 presents the results of our fixed effects regression models, which aim to assess the impact of coding training on students' online self-study time. In particular, columns 1 and 2 present the estimates from the linear model and logit model, respectively, using a panel data fixed effect approach. Both models utilize a dummy variable indicating whether students spend more than 2 hours on online self-study as the dependent variable.

Our findings reveal a significantly positive effect of coding training on the dependent variable in both model estimations, accompanied by a very small p-value (<0.001). Based on the linear model estimates, there was an 18.3 percentage point increase in the likelihood that students would spend over 2 hours on online self-study when engaged in the coding course. This aligns with the positive result from the logit model, where the estimate for the coding participation variable is 1.463, equivalent to an odds ratio of 4.317. In practical terms, this means that when a student transitions from not learning how to code to acquiring coding skills, their odds of spending more than 2 hours on online self-study are multiplied by 4.317.

However, it is important to note that studying how to code did not lead to a significant increase in self-study time when students were working with their peers. The estimate results in both columns 1 and 2 of Table 6 indicate that although the coding variable has a positive estimate in both models, the values are very small and not significantly different from 0. In summary, our analysis suggests that coding training significantly enhances students' self-study time in an online mode but does not have a notable impact on their self-study time with peers.

Table 5: Effect of Treatment on the Students' Online Self-Study Time

	Dependent Variable: Online self-study time	
	LINEAR MODEL	LOGIT MODEL
	(1)	(2)
Round	0.00995	0.0589
	(0.032)	(0.287)
Coding training	0.183**	1.463***
	(0.040)	(0.329)
Computer access	0.0358	0.479
	(0.040)	(0.297)
Time for working household chores		
Up to 1 hour	-0.0487	-0.0445
	(0.023)	(0.330)
Up to 2 hours	-0.0107	0.151
	(0.027)	(0.440)
Up to 3 hours	-0.0714	-0.135
	(0.038)	(0.555)
Up to 4 hours	-0.000829	-0.0131
	(0.074)	(0.650)
More than 4 hours	0.0284	0.843
	(0.144)	(1.099)
Constant	0.190***	
	(0.028)	
N	1996	526
adj. R^2	0.069	

Standard errors in parentheses * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

Table 6: Effect of Treatment on the Students' Self-Study Time with Peers

	Dependent Variable: Self-study time with peers	
	LINEAR MODEL	LOGIT MODEL
	(1)	(2)
Round	0.0334	0.493***
	(0.016)	(0.132)
Coding training	0.00596	0.0136
	(0.018)	(0.191)
Computer access	0.0660	0.939
	(0.043)	(0.728)
Time for working household chores		
Up to 1 hour	-0.0626	-0.710
	(0.052)	(0.764)
Up to 2 hours	-0.0500	-0.484
	(0.049)	(0.776)
Up to 3 hours	-0.0428	-0.658
	(0.057)	(0.687)
Up to 4 hours	-0.127	-2.112
	(0.098)	(1.401)
More than 4 hours	-0.134	-1.820
	(0.131)	(1.659)
Constant	0.125*	
	(0.052)	
N	1996	278
adj. R^2	0.020	

Standard errors in parentheses

* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

5.4. Examining Mechanisms

In this section, we explore further the mechanisms underlying the impact of coding training on self-online learning. As we discussed in our theoretical section, various internal and external factors play a crucial role in students' engagement in self-learning. Consequently, coding training can act as a catalyst for change on several fronts: first, by providing students with the knowledge and cognitive foundation essential for autonomous coding self-study; secondly, by opening doors to online resources that may have previously eluded them; and finally, by instigating shifts in their motivation towards coding. Through a series of analyses, our aim is to understand which of these three mechanisms primarily drive the effect of coding training on students' self-online learning for coding.

Box 2: Questions to Measure Students' Interest, Confidence, and Perception of Benefits

<table>
<tr><td colspan="5">On a scale of 1 – 5, with 1 being "Not at all interested" and 5 being "Extremely interested", how interested are you in learning coding?</td></tr>
<tr><td>1
Not at all interested</td><td>2
Slightly interested</td><td>3
Moderately interested</td><td>4
Very interested</td><td>5
Extremely interested</td></tr>
<tr><td colspan="5">On a scale of 1 – 5, with 1 being "Not at all confident" and 5 being "Extremely confident", how confident are you in your ability to learn coding well?</td></tr>
<tr><td>1
Not at all confident</td><td>2
Slightly confident</td><td>3
Moderately confident</td><td>4
Very confident</td><td>5
Extremely confident</td></tr>
<tr><td colspan="5">On a scale of 1 – 5, with 1 being "No benefits at all" and 5 being "Tremendous benefits", how much benefit do you expect to gain from learning coding for your study, life and future careers?</td></tr>
<tr><td>1
Not benefits at all</td><td>2
Slight benefits</td><td>3
Moderate benefits</td><td>4
Significant benefits</td><td>5
Tremendous benefits</td></tr>
</table>

Note: All questions were asked in Vietnamese.

We start our mechanism exploration by examining whether coding training leads to changes in students' learning motivation towards coding, and thus changes in students' self-online learning activities. To do so, we use three single-item questions to measure three motivational variables: students' interest, confidence, and perception of the learning benefits towards coding on a five-point scale, ranging from "Not at all" to "Extremely". The precise phrasing of two questions is displayed in Box 2.

While the single-item questions may not be able to capture the breadth of students' interest, confidence and outcome expectation, their simplicity enabled our students to understand and answer them easily. Therefore, we use the variables constructed from these three questions as the primary mechanism variables in our analysis. Similar to our outcome variable, we then converted this variable to a binary indicator, assigning a value of 1 if the rating is between 3 and 5, and 0 otherwise. As a robustness check, we also employ multi-item scales based on the Elementary Student Coding Attitudes Survey (Mason & Rich, 2020) to measure students' interest, confidence and outcome expectation. All results remain consistent when employing the multi-item scales.

The following three tables (Tables 7, 8, and 9) provide a comprehensive summary of the findings derived from our linear and logit models. These models were employed to evaluate the influence of acquiring coding skills on students' levels of interest, self-efficacy, and perceived advantages related to coding. In each of these tables, we will find estimations derived from linear models in columns 1 and 2, as well as estimations from logit models in columns 3 and 4. These estimations encompass outcome variables, with single-item indicators featured in columns 1 and 3, and variables assessed through multiple-item evaluations in columns 2 and 4. Our analysis consistently reveals that the coefficients of the treatment variable tend to be either negative or statistically insignificant across all three tables. Consequently, we can confidently conclude that learning how to code did not lead to an increase in students' interest, confidence, or perception of the benefits associated with coding. These results suggest that the observed increase in students' online self-learning behavior is unlikely to be caused by improved motivation or attitudes toward coding.

Table 7: Impacts of Coding Training on Coding Interests (Single and Multiple Items)

	LINEAR MODEL		LOGIT MODEL	
	DV: Coding interest (single item) (1)	*DV: Coding interest (multiple items)* (2)	*DV: Coding interest (single item)* (3)	*DV: Coding interest (multiple items)* (4)
Round	-0.01	-0.09**	-0.09	-0.89***
	(0.011)	(0.023)	(0.269)	(0.264)
Coding training	-0.04*	-0.01	-0.93***	-0.14
	(0.015)	(0.034)	(0.276)	(0.409)
Computer access	0.04	0.00	0.79	0.06
	(0.025)	(0.029)	(0.563)	(0.330)
Time for working household chores				
Up to 1 hour	-0.03	0.01	-0.36	0.41
	(0.041)	(0.040)	(0.547)	(0.375)
Up to 2 hours	-0.01	0.04	0.28	0.75
	(0.026)	(0.053)	(0.668)	(0.546)
Up to 3 hours	-0.04	0.07	-0.40	1.24
	(0.056)	(0.080)	(0.951)	(1.221)
Up to 4 hours	-0.03	0.09	-0.04	1.74*
	(0.048)	(0.079)	(1.506)	(0.782)
More than 4 hours	-0.03	0.07	0.00	14.04***
	(0.037)	(0.074)	(.)	(2.005)
Constant	0.96***	0.83***		
	(0.028)	(0.047)		
N	1996	1996	184	424
adj. R^2	0.014	0.039		

Standard errors in parentheses

*+ p < 0.10, * p < 0.05, ** p < 0.01, *** p < 0.001*

Table 8: Impacts of Coding Training on Coding Self-Efficacy (Single and Multiple Items)

	LINEAR MODEL		LOGIT MODEL	
	DV: Self - efficacy (single item)	*DV: Self - efficacy (multiple items)*	*DV: Self - efficacy (single item)*	*DV: Self - efficacy (multiple items)*
	(1)	(2)	(3)	*(4)*
Round	-0.04 (0.024)	-0.02 (0.024)	-0.36^{+} (0.199)	-0.19 (0.195)
Coding training	-0.05^{+} (0.028)	0.04 (0.030)	-0.38^{+} (0.209)	0.41 (0.281)
Computer access	0.05 (0.051)	0.03 (0.030)	0.35 (0.343)	0.25 (0.281)
Time for working household chores				
Up to 1 hour	0.03 (0.054)	0.03 (0.042)	0.41 (0.330)	0.43 (0.373)
Up to 2 hours	0.02 (0.062)	0.04 (0.040)	0.30 (0.319)	0.49 (0.383)
Up to 3 hours	0.03 (0.068)	0.08 (0.063)	0.27 (0.549)	0.80 (0.510)
Up to 4 hours	-0.06 (0.069)	0.04 (0.112)	-0.70 (0.765)	0.37 (0.876)
More than 4 hours	0.05 (0.157)	0.01 (0.149)	0.56 (0.893)	0.32 (1.138)
Constant	0.77^{***} (0.047)	0.77^{***} (0.028)		
N	1996	1996	536	406
adj. R^2	0.019	0.001		

Standard errors in parentheses

*+ $p < 0.10$, * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$*

Table 9: Impacts of Coding Training on Perceived Benefit (Single and Multiple Items)

	LINEAR MODEL		LOGIT MODEL	
	DV: perceived benefit (single item) (1)	*DV: perceived benefit (multiple items)* (2)	*DV: perceived benefit (single item)* (3)	*DV: perceived benefit (multiple items)* *(4)*
Round	-0.02 (0.018)	-0.02 (0.063)	-0.53 (0.348)	-0.27 (0.572)
Coding training	-0.00 (0.019)	-0.06 (0.063)	0.13 (0.370)	-0.41 (0.579)
Computer access	-0.04 (0.025)	0.02 (0.023)	-0.82 (0.633)	0.24 (0.254)
Time for working household chores				
Up to 1 hour	0.02 (0.038)	0.01 (0.049)	0.27 (0.836)	0.18 (0.428)
Up to 2 hours	0.02 (0.040)	0.06 (0.044)	0.41 (0.906)	0.54 (0.405)
Up to 3 hours	0.02 (0.036)	0.02 (0.051)	0.40 (0.860)	0.13 (0.564)
Up to 4 hours	0.01 (0.086)	0.09 (0.125)	0.57 (1.753)	1.37 (0.994)
More than 4 hours	-0.03 (0.047)	0.02 (0.068)	-13.83*** (1.350)	0.43+ (0.245)
Constant	0.95*** (0.044)	0.81*** (0.062)		
N	1996	1996	180	444
adj. R^2	0.007	0.016		

Standard errors in parentheses

*+ $p < 0.10$, * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$*

We now shift our focus to the information mechanism, in which coding training enhances self-online learning by providing students more access to online resources. While it is challenging to directly measure the information and resources available to students, we examine this mechanism through an indirect approach. Specifically, we conducted estimations for a sub-group of students who, prior to undergoing coding training, were already skilled at finding and using online learning materials. Our expectation is that if the primary driver behind the effect of coding training on self-online learning is the information mechanism, we should observe weaker effects within this particular sub-group.

We measure students' ability to locate and use online learning resources prior coding training with the following question: *"Do you often use the Internet to search for online learning materials for your self-study?"* Students could respond with either *Yes* or *No*. Those who answered *Yes* are categorized as proficient in using online learning resources, while those who responded *No* are placed in the group that are less familiar with searching for online resources. We then ran two separate models for these two groups to examine the information mechanism. The estimation results of these models are presented in Table 10. In columns 1 and 2, the results of the linear and logit models for students with limited experience in using online study materials are displayed. Similarly, columns 3 and 4 show the outcomes of the linear and logit models for individuals who possess prior experience in searching for online resources.

According to our estimations, coding training has a consistently positive impact on online self-directed learning behaviors for both groups of students. Especially, the p-values associated with the estimates of the treatment variable for the group having experience in online learning are exceptionally low. These p-values are below 1%, and even as low as 0.1%, indicating a high level of statistical significance. In contrast, for the group lacking prior experience, statistical significance is only attained at the 5% level.

If we consider a significance level of 1%, the impact on the group with no prior experience in using online resources becomes statistically insignificant, while it retains its significance for students with experience. In this case, we can deduce that the provision of online resources in the coding course had a limited or negligible impact on the self-learning behavior of students unfamiliar with using online materials. In simpler terms, the information mechanism did not have any observable influence on altering students' self-directed learning behaviors.

Table 10: Effect of Treatment on the Students' Online Self-Study Time - Split by Prior Experience in Using Online Resources

Dependent Variable: Online Self-Study Time	**GROUP 1: NO EXPERIENCE**		**GROUP 2: HAVING EXPERIENCE**	
	LINEAR MODEL	LOGIT MODEL	LINEAR MODEL	LOGIT MODEL
Round	0.01 (0.026)	0.10 (0.325)	0.01 (0.036)	0.07 (0.268)
Coding training	0.12* (0.036)	1.03* (0.432)	0.23** (0.062)	1.88*** (0.217)
Computer access	0.05 (0.026)	0.39 (0.410)	0.04 (0.048)	0.76* (0.314)
Time for working household chores				
Up to 1 hour	-0.14+ (0.062)	-1.74+ (0.973)	0.02 (0.058)	0.91** (0.319)
Up to 2 hours	-0.10 (0.084)	-1.77 (1.272)	0.05 (0.059)	1.24** (0.401)
Up to 3 hours	-0.22* (0.073)	-3.27* (1.493)	0.03 (0.074)	1.12 (0.757)
Up to 4 hours	-0.09 (0.191)	-1.73 (1.517)	0.06 (0.103)	0.94 (0.818)
More than 4 hours	-0.11 (0.134)	-1.37 (1.124)	0.13 (0.179)	1.90+ (1.151)
Constant	0.22* (0.064)		0.16+ (0.075)	
N	776	146	1220	380
adj. R^2	0.060		0.079	

Standard errors in parentheses

*+ $p < 0.10$, * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$*

Conversely, for a case of using a significance level of 5%, the impact maintains statistical significance for all groups. It means the information mechanism positively changed students' self-directed learning behaviors. However, because the estimations for both groups employ identical variables, we can directly compare the estimates of the treatment variable, it becomes evident that the estimates derived from the group experienced in using online learning resources are larger than those from the group lacking experience in finding online materials. This discovery deviates from what one might expect if the primary influencing mechanism were solely the information channel, where weaker or no effects would be anticipated for the group with prior experience. Therefore, in this case, we can infer that learning how to code must exert a positive effect on students' online self-learning time through a different mechanism, one that is distinct from or complementary to the information mechanism.

In summary, when considering three potential influence channels - providing fundamental coding knowledge, offering online resources for self-learning, and fostering positive attitudes toward coding - our analysis indirectly demonstrates that the increase in time devoted to online self-learning programming after coding training is closely linked to a growing knowledge and foundation in coding. It appears that these other mechanisms may not be the primary drivers of enhancements in students' online self-directed learning within the field of coding.

6. Discussion

In this study, we investigate the effect of participating in a coding training program on students' engagement in self-learning for coding in Vietnam. We specifically focus on two forms of self-learning: self-online learning and peer-to-peer self-learning. Our quasi-experimental analysis reveals that coding training positively affects students' self-study time dedicated to online learning for coding but does not significantly influence their self-study time with peers. Further analyses suggest that this effect is more likely to be attributed to coding training's role in helping students establish a knowledge foundation for their later self-study, rather than changing their access to online resources or motivation beliefs.

In term of literature contribution, the study expands upon the current body of research on the impact of coding education for K-12 students by delving into a relatively unexplored area: the influence of coding training on students' initiation into self-directed learning activities post-program. While existing literature has extensively examined the effects of such training on a range of outcomes, including cognitive and computational thinking skills (Relkin et al., 2021; Lye & Koh, 2014; Lockwood & Mooney, 2018), social skills and collaboration (Fessakis et al., 2013; Falloon, 2016), and various academic achievements (Hayes and Stewart 2016; McClure et al., 2017; Sáez-López et al 2016), the shift from structured learning environments to autonomous coding studies after the completion of training programs has been less studied. Our research diverges from prior work by specifically investigating this transition and suggesting potential mechanisms for the observed effects, thus initiating a dialogue on the possibility of sustained impacts of coding training in contexts with limited resources.

To the best of our knowledge, only three studies have considered the impact of coding training on students' active learning behaviors (Fallon, 2016; Kalelioglu, 2015; Sáez-López et al., 2016). These studies, however, largely concentrate on the cultivation of such behaviors within the training period itself and do not explore the extent to which students independently continue their coding studies thereafter. Furthermore, these investigations have predominantly utilized teacher observations or qualitative methodologies, which, although insightful, fall short of establishing causality in a broad context. Our study addresses this gap by employing a quasi-experimental design, offering strong causal evidence of the impact of coding training on the self-learning behaviors of students.

Our findings highlight the crucial role of initial self-study engagement following coding training, suggesting that even short educational interventions can encourage the first steps towards a lifelong journey in coding education. This insight is crucial for educators and policymakers aiming to foster both technical skills and an active learning mindset in an increasingly digital world. Although our results do not directly confirm the long-term persistence of these behaviors, they underscore the significance of the immediate post-training phase as a pivotal moment for promoting continued coding engagement. This insight opens new research pathways on how brief educational interventions could serve as a foundation for enduring learning habits, providing a detailed perspective that could guide the development and execution of future coding education programs.

Our study is subject to certain limitations that suggest directions for future research. First, due to the research project's time constraints, we focused on short-term effects of coding training on students' self-study practices, rather than delving into the potential long-term impact. Given the necessity of transforming self-study activities into enduring habits to sustain the benefits of a coding training initiative, further studies should aim to conduct follow-up assessments over an extended period to determine whether these positive effects endure over time.

Second, in this study, we measure self-study efforts in terms of time spent on self-study activities, rather than evaluating the quality or effectiveness of the learning that took place. While increased time spent on self-learning is a positive indicator, it may not necessarily translate to a high level of depth and effectiveness in the learning process itself. Future research should incorporate assessments of the quality and effectiveness of the self-study activities, to gain a more comprehensive understanding of the impact of coding training.

Finally, the study's design did not allow for a direct test of the underlying mechanisms driving the observed effects. While we speculate that the positive influence of coding training on self-study time is related to the establishment of a robust knowledge foundation, we were not able to conduct specific analyses to confirm this hypothesis. Future research could further investigate these mechanisms to provide a more nuanced understanding of how coding training contributes to enhanced self-learning in different contexts.

References

Alhloul, A., & Kiss, E. (2022). Industry 4.0 as a Challenge for the Skills and Competencies of the Labor Force: A Bibliometric Review and a Survey. *Sci*, *4*(3), 34. MDPI AG. Retrieved from http://dx.doi.org/10.3390/sci4030034

Asadullah, M. N. (2016). Do pro-poor schools reach out to the poor? Location choice of BRAC and ROSC schools in Bangladesh. *Australian Economic Review*, *49*(4), 432-452.

Asian Development Bank (2021). *Reaping the benefits of industry 4.0 through skills development in high-growth industries in Southeast Asia: insights from Cambodia, Indonesia, the Philippines, and Viet Nam*. Asian Development Bank, Manila, Philippines.

Balanskat, A. and Engelhardt, K. (2014): Computing our future: computer programming and coding - priorities, school curricula, and initiatives across Europe. European Schoolnet.

Bandura, A. (1977). Self-efficacy: toward a unifying theory of behavioral change. *Psychological review*, *84*(2), 191.

Bandura, A. (1997). Self-Efficacy: The Exercise of Control. *Journal of Cognitive Psychotherapy, 13*, 158 - 166.

Barbosa, J., Silva, Á., Ferreira, M. A., & Severo, M. (2016). Transition from secondary school to medical school: the role of self-study and self-regulated learning skills in freshman burnout. *Acta medica portuguesa*, *29*(12), 803-808.

Charles, L., Xia, S. & Coutts, A. P. (2022). *Digitalization and employment: a review*. ILO, Geneva. Retrieved from https://www.ilo.org/employment/Whatwedo/Publications/WCMS_854353/lang--en/index.htm on 27 Feb 2023.

Davis-Kean, P. E. (2005). The influence of parent education and family income on child achievement: the indirect role of parental expectations and the home environment. *Journal of family psychology*, *19*(2), 294.

Dewey, J. (1986). Experience and education. In *The educational forum* (Vol. 50, No. 3, pp. 241-252). Taylor & Francis Group.

Dewey, J. (2022). *How We Think & Other Works on Logic: Leibniz's New Essays; Essays in Experimental Logic; Creative Intelligence; Human Nature & Conduct*. DigiCat.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning*, *32*(6), 576-593.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education, 63*, 87-97.

Fontana, R. P., Milligan, C., Littlejohn, A., & Margaryan, A. (2015). Measuring self-regulated learning in the workplace. *International Journal of Training and Development*, *19*(1), 32-52.

Future Learn (2022). The future of learning report. Retrieved from https://www.futurelearn.com/info/thefutureoflearning

Harden, R. M. (2009). Independent learning. *A practical guide for medical teacher. 3rd ed. London, UK: Elsevier Churchill Livingstone*, 168-74.

Hayes, J., & Stewart, I. (2016). Comparing the effects of derived relational training and computer coding on intellectual potential in school-age children. *British Journal of Educational Psychology, 86*(3), 397-411.

Hong, S. C., & Park, Y. S. (2012). An analysis of the relationship between self-study, private tutoring, and self-efficacy on self-regulated learning. *KEDI journal of educational policy*, *9*(1).

International Labor Organization (ILO) (2019). *Preparing for the future of work: National policy responses in ASEAN +6*. Regional Economic and Social Analysis Unit International Labour Organization , Bangkok. Retrieved from https://www.ilo.org/asia/publications/WCMS_717736/lang--en/index.htm

Jha, J., & Ghatak, N. (2023). Open Schools in Developing Countries: Virtual and Open or Distant and Closed? In *Handbook of open, distance and digital education* (pp. 493-508). Singapore: Springer Nature Singapore.

Kalelioglu, F. (2015). A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior, 52*, 200-210.

Kistner, S., Rakoczy, K., Otto, B., Dignath-van Ewijk, C., Büttner, G., & Klieme, E. (2010). Promotion of self-regulated learning in classrooms: Investigating frequency, quality, and consequences for student performance. *Metacognition and learning*, *5*, 157-171.

Knowles, M. S. (1975). *Self-directed learning: A guide for learners and teachers.*

Li, L. Reskilling and Upskilling the Future-ready Workforce for Industry 4.0 and Beyond. *Inf Syst Front* (2022). https://doi.org/10.1007/s10796-022-10308-y

Lockwood, J., & Mooney, A. (2018). Developing a computational thinking test using Bebras problems.

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. https://doi.org/10.1016/j.chb.2014.09.012

MacBeath, J. (1993). *Learning for Yourself: Supported Study in Strathclyde Schools*.

Maltais, C., Bouffard, T., Vezeau, C. & Dussault, F. (2021). Does parental concern about their child performance matter? Transactional links with the student's motivation and development of self-directed learning behaviors. *Soc Psychol Educ* **24**, 1003–1024 (2021). https://doi.org/10.1007/s11218-021-09642-x

Mason, S. L., & Rich, P. J. (2020). Development and analysis of the Elementary Student Coding Attitudes Survey. *Comput. Educ., 153*, 103898.

McClure, E. R., Guernsey, L., Clements, D. H., Bales, S. N., Nichols, J., Kendall-Taylor, N., & Levine, M. H. (2017). STEM starts early: Grounding science, technology, engineering, and math education in early childhood. In *Joan Ganz Cooney center at sesame workshop*. Joan Ganz Cooney Center at Sesame Workshop. 1900 Broadway, New York, NY 10023.

Meyer, B., Haywood, N., Sachdev, D., & Faraday, S. (2008). Independent learning: Literature review. *Learning and Skills Network*.

Muilenburg, L. Y., Berge, Z. L. (2005). Student barriers to online learning: A factor analytic study. *Distance Education*, 26(1), 29-48. doi:10.1080/01587910500081269

Organisation for Economic Co-operation and Development (OECD) (2011). *Quality time for students: Learning in and out of school*. PISA report. Paris: OECD Publishing.

Paris, S. G., & Paris, A. H. (2001). Classroom applications of research on self-regulated learning. *Educational Psychologist, 36*(2), 89–101. https://doi.org/10.1207/S15326985EP3602_4

Rogers, C. R., & Freiberg, H. J. (1994). *Freedom to learn*. Merrill/Macmillan College Publishing Co.

Sáez-López, J. M., Román-González, M., & Vázquez-Cano, E. (2016). Visual programming languages integrated across the curriculum in elementary school: A two-year case study using "Scratch" in five schools. *Computers & Education*, 97, 129-141.

Sawada, Y., Aida, T., Griffen, A. S., Kozuka, E., Noguchi, H., & Todo, Y. (2022). Democratic institutions and social capital: Experimental evidence on school-based management from a developing country. *Journal of Economic Behavior & Organization*, *198*, 267-279.

Schwab, K., & Samans, R. (2016). World Economic Forum (2016). *The future of jobs: Employment, skills and workforce strategy for the fourth industrial revolution*. Global Challenge Insight Report.

Schwab, K., & Zahidi, S. (2020). The future of jobs report 2020. World Economic Forum, October 2020. Retrieved from http://www3.weforum.org/docs/WEF_Future_of_Jobs_2020.pdf

Tough, A. M. (1989). Self-directed learning: Concepts and practice. In *Lifelong education for adults* (pp. 256-260). Pergamon.

World Bank (2019). *World Development Report 2019: The Changing Nature of Work.* Washington, DC: World Bank. doi:10.1596/978-1-4648-1328-3

Zimmerman, B. J. (2002). Becoming a self-regulated learner: An overview. *Theory into practice*, *41*(2), 64-70.

Zimmerman, B. J. (2013). Theories of self-regulated learning and academic achievement: An overview and analysis. *Self-regulated learning and academic achievement*, 1-36.

Chapter 2

The Influence of a Short-Term Coding Program on Computational Thinking Skills among Secondary School Students in Vietnam

Abstract: Employing a mixed-method design incorporating both quasi-experimental and qualitative approaches, the research explores the influence of two coding programs, Scratch and MakeCode for Micro:bit, on the development of computational thinking skills in secondary school students in Vietnam. While Scratch emphasizes interactive story creation, animation, and game design, MakeCode for Micro:bit focuses on merging the digital and physical realms. The study evaluates students' computational thinking skills by using the well-established Bebras test (Lockwood & Mooney, 2018) and discovers that both programs enhance these skills. However, this effect is exclusive to Grade 7 students engaged in coding with MakeCode for Micro:bit, while Grade 6 students undergoing training with Scratch did not show a similar effect. Insights from qualitative interviews conducted with IT teachers and educational experts serve to reinforce these conclusions. These discussions highlight the crucial role of students comprehending programming concepts through problem-solving as a fundamental element in nurturing computational thinking skills during coding training.

1. Introduction

In today's digitized world, technology has become an indispensable part of our daily lives, influencing both personal and professional spaces. With machines and algorithms becoming increasingly ubiquitous, it has become essential for individuals to understand how they function to leverage them effectively and avoid negative consequences. This has led to a growing recognition of the importance of cultivating "computational thinking" (CT), a skill set that allows individuals to think "like a computer scientist when confronted with a problem" (Grover & Pea, 2013; UNESCO, 2022). As a result, many countries worldwide have implemented curricular reforms to train CT in K-12 education, thus equipping students with the necessary skills for the digital age.

Although CT training can be integrated into various subjects such as mathematics or science, coding is widely considered the most effective way to develop this skill set owing to its code-centric nature (Kite et al., 2021). However, the effectiveness of teaching CT through coding in K-12 education hinges on the ability to make coding accessible and engaging for students. This implies that students should be able to focus on the problem-solving aspects of coding rather than technicalities (Repenning et al., 2010).

In Vietnam, recent years have also seen a movement to shift from teaching coding with a focus on syntax to teaching it with a focus on problem-solving. The goal is not only to equip students with coding literacy but also to develop CT skills. This movement involves formal reforms towards STEM education by the Vietnamese Ministry of Education and Training, as well as many initiatives by educational institutions and NGOs. At the same time, visual programming languages have also been introduced in Vietnam as part of this effort. With block-based, graphical elements and drag-and-drop features, visual programming languages can reduce syntactic problems and make it easier for students to practice and self-study compared with traditional programming languages (Koh et al., 2010). Consequently, visual programming is expected to be more effective than traditional programming languages in fostering CT skills in students.

While many empirical studies have confirmed the positive effect of coding training with visual programming languages on developing CT skills in students (Lye & Koh, 2014), research also suggests that the effectiveness of visual programming languages depends on several factors, such as learning culture and teaching methods (Su et al., 2020; Weintrop, 2016). Therefore, further research is needed to understand the impact of visual programming languages on the development of CT skills for Vietnamese students. This could provide valuable insights to inform policymakers, educators and other stakeholders in Vietnam about the effective approaches for CT education.

This study is among the first attempts to examine this issue. Using a mixed method design combining a quasi-experimental approach and qualitative method, we investigate the effects of a short-term coding program using visual programming languages on the development of CT skills of secondary school students in Lam Dong province of Vietnam. We measure students' computational thinking through a test using pre-existing Bebras problems (Lockwood & Mooney, 2018). We focus on two visual programming languages: Scratch and MakeCode for Micro:bit. Scratch, developed by the MIT Media Lab, is a block-based visual programming language that enables students to program by arranging and stacking blocks of code. On the other hand, MakeCode for Micro:bit, created by Microsoft, combines block-based visual programming with a microcontroller board called Micro:bit. Both tools facilitate learning programming concepts through block arrangement and stacking. However, Scratch places more emphasis on the creation of interactive stories, animations and games, while MakeCode for Micro:bit highlights the development of projects involving robotics, sensors and wearable devices, allowing students to bridge the gap between the digital and physical worlds. Through our study, we seek to shed light on the features and benefits of each tool in fostering CT skills in K-12 education in Vietnam.

The results of our study show that coding training has a positive effect on students' CT scores. However, this effect is only significant for Grade 7 students who received coding training using MakeCode for Micro:bit, in contrast to Grade 6 students who received coding training with Scratch . Qualitative insights derived from interviews with IT teachers and educational experts further support our findings and highlight the importance of students acquiring programming concepts through problem-solving as a fundamental factor in effectively nurturing CT skills during coding training.

The paper is organized as follows: Section 2 provides the theoretical background of our research. Section 3 describes our research setting, including the coding program, study design data collection procedures, and our CT assessment approach. Section 4 explains our econometric approach, and Section 5 presents the results of our analysis. We discuss our results and conclude our paper in Section 6.

2. Theoretical Background

2.1. Computational Thinking Definition

Computational thinking is a relatively new concept that has gained a significant amount of attention in recent years. The terms can be traced back to Seymour Papert's work on teaching computational processes to children using LOGO programming in the early 1980s (Papert, 1980, 1981). However, it was Jeannette Wing's influential article in 2006 that brought widespread recognition and usage of the term (Wing, 2006). In her paper, Wing defined "computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to computer science" (Wing, 2006), and emphasized the relevance of these skills in all domains.

Although Wing's work generated enthusiasm among many scholars, educators and policymakers, it also presented several challenges. One of the primary issues stems from the ongoing lack of consensus on the nature of the concept, despite numerous studies in this field. Zhang and Nouri (2019) conducted a review of different definitions and categorized them into three groups. The first category comprises "generic definitions" by Wing (2006, 2011) and Aho (2012), which define CT as a general problem-solving skill set without specifying concise components. The second category includes "operational definitions" provided by the Computer Science Teachers Association (CSTA, 2011) and Selby and Woollard (2013), which outline CT skills encompassing abstraction, decomposition, data analysis and representation, algorithmic thinking, evaluation and generalization. Finally, the "educational definitions" proposed by Barr and Stephenson (2011) and Brennan and Resnick (2012) introduce a framework with three key dimensions: computational concepts (e.g., loops or conditions), computational practices (e.g., testing and debugging), and computational perspectives (e.g., questioning and connecting).

While both operational definitions and educational definitions offer tractable insights into computational thinking, they diverge in terms of their focus (Kong, 2016; Zhang & Nouri, 2019). Operational definitions primarily emphasize the essential dimensions of universal thinking skills in CT. In contrast, educational definitions place excessive emphasis on the specific knowledge used and applied when students create code (Falloon, 2016). This distinction highlights the different perspectives and priorities in approaching computational thinking.

2.2. Computational Thinking and Coding Training

Despite the recent integration of computational thinking in various subjects (Ye et al., 2023), coding remains the primary method for fostering computational thinking in K-12 education. Numerous studies have shown the positive effects of coding on the development of CT skills among students (Xu et al., 2022). However, due to the ongoing lack of consensus on the specific components of CT, the impact of coding on computational thinking development exhibits significant heterogeneity. Engaging in coding tasks has been shown to lead to multiple outcomes, from the comprehension of computational concepts, the application of thinking skills in real-world contexts, to the cultivation of positive social emotions (Grover & Pea, 2018; Hooshyar et al., 2021; Perez-Marin et al., 2020; Weintrop et al., 2016).

When it comes to training students in computational thinking, teaching programs often distinguish between unplugged and plugged activities. Unplugged activities refer to hands-on learning experiences that do not involve the use of computers yet are designed to impart foundational coding knowledge. These activities can take diverse forms, such as games, puzzles, physical manipulatives, and interactive exercises. As unplugged activities require minimal resources, this approach is accessible to students across various educational settings. Additionally, the utilization of unplugged activities fosters collaborative and communicative environments, encouraging students to enhance their interpersonal and teamwork skills alongside their computational thinking abilities (Polat & Yilmaz, 2022).

On the other hand, plugged activities involve the utilization of computers and/or digital devices for coding training. Traditionally, coding has been taught using text-based programming languages like Java, C++, or Python. Nevertheless, in recent years, various new generation programming tools and languages have developed and been used. These include web-based simulation tools (Repenning et al., 2014), educational robots (Kerimbayev et al., 2023; Lee et al., 2017), and visual programming tools (Brennan & Resnick, 2012; Juskeviciene et al., 2021; Stewart & Baek, 2023).

Literature suggests that new generation programming tools and languages are particularly well-suited for teaching coding to students at an early age (Lye & Koh, 2014). They offer an intuitive interface that alleviates the complexity of syntax and allows young students (elementary and middle-school aged learners) to focus more on problem-solving and algorithmic reasoning (Garcia-Penalvo & Mendes, 2018, Grover & Pea, 2018). However, research also noted that the effectiveness of these tools relies on appropriate teaching approaches, with the teacher's role shifting to from explaining programming concepts to providing support for students' learning processes (Weintrop & Wilensky, 2016; Xu et al., 2022). As a result, teachers accustomed to traditional methods may face challenges in effectively utilizing these new tools to foster the development of CT skills among students.

2.3. Computational Thinking Assessment

The measurement of CT skills is of utmost importance for evaluating the effectiveness of educational interventions. However, due to the ongoing debate on the nature of computational thinking, there is currently a lack of standardized methods for measuring this concept. This poses a challenge for researchers and educators who aim to assess CT abilities in a reliable manner across different contexts and age groups.

In literature, two common approaches to measuring CT skills have emerged. The first approach involves developing tools specifically tailored to a particular programming language or tool. These tools, such as Dr. Scratch (Moreno-Leon & Robles, 2015) for Scratch or the Computational Thinking Patterns CTP-Graph (Koh et al., 2010) for AgentSheets, utilize extensive data collected during the learning process to evaluate learners' progress and provide feedback. By focusing on specific programming tools, these assessments offer valuable insights into the development of CT skills. However, they may not effectively capture computational thinking with broader aspects and across different contexts. Therefore, it is difficult to use these measures to evaluate pre- and post-intervention outcomes (Lockwood & Mooney, 2018).

The second approach focuses on the development of general assessment tools that are independent of specific programming languages, tools, or subject areas. Measures in this group have been designed to capture different aspects of computational thinking, ranging from aptitude assessments (Roman-Gonzalez, 2019) to attitude scales (Korkmaz et al., 2017). Notably, these assessment tools can be administered under "pure pre-test conditions" (Roman-Gonzalez et al., 2017), allowing for direct measurement of students' computational thinking without relying on specific interventions or programming environments.

Within the second approach, the Bebras International Challenge (Dagiene & Futschek, 2008; Dagiene et al., 2015) emerges as an assessment tool that aims to evaluate students' application of CT skills in solving real-life problems (Lookwood & Mooney, 2018). The Bebras test presents a series of challenging problems that encompass a wide range of topics, including decomposition, abstraction, algorithmic thinking, pattern recognition, and generalization. It is specifically designed to be language-agnostic, allowing it to be administered to students across different age groups and educational contexts, regardless of their prior programming experience. Therefore, the Bebras test can effectively measure the development progress of students' CT skills over time and is thus suitable for pre- and post-intervention evaluation.

3. Research Setting

3.1. The "Coding for Future with Google" Project

In the three years from 2019 to 2021, The Dariu Foundation (TDF) and Google APAC collaborated on a digital literacy initiative called "Coding for Future with Google" (CFGP). The project aimed to equip children with the digital skills necessary for the future by providing free coding courses. CFGP was targeted at underprivileged communities where it was difficult for children to gain access to digital education. Through this project, TDF and Google APAC expected that rural youth were prepared and empowered better to participate successfully in the digital economy.

There were two stages in the implementation of CFGP. First, the project collaborated with provincial governments to identify targeted schools. Subsequently, CFGP provided free laptops to these schools on a lending basis and trained their IT teachers on how to teach coding with Scratch and Micro:bit. In the second phase, the trained teachers conducted coding courses for their students at the schools, using materials and equipment provided by CFGP. The teaching curriculum and materials were developed and validated by IT educational experts from the University of Technology Ho Chi Minh City. While teachers had the flexibility to adapt CFGP's recommended curriculum to local needs, the project ensured that all courses covered all essential learning content and concepts. An overview of the learning contents of both coding languages, Scratch and Micro:bit, is provided in Table 1.

A typical coding course consisted of eight 45-minute sessions. The sessions were held over one semester and concluded with an online assessment or a written exam to evaluate student performance. Additionally, students were also provided access to free online coding materials on the project's website and YouTube channel.

Table 1: Summary of Learning Contents

Lesson	LEARNING CONTENTS	
	Scratch	**MakeCode (Micro:bit)**
1	Drag-and-Drop Programming Language	Introduction to Micro:bit Programming Board
2	Basic Operations on Scratch Commands	Interacting with the Display Screen on Micro:bit
3	Drawing Images on Scratch	Synthesizing Display Commands
4	Loops in Scratch	Controlling Buttons on Micro:bit
5	Animating in Scratch Programming	Interaction between Micro:bit and User Behavior
6	Input-Output Programming in Scratch	Sensors on Micro:bit
7	Conditional Statements in Scratch	Wireless Data Transmission between Micro:bit Boards
8		Loop and Conditional Structures on Micro:bit

Note: Summarized from the Dariu documents

3.2. Study Design

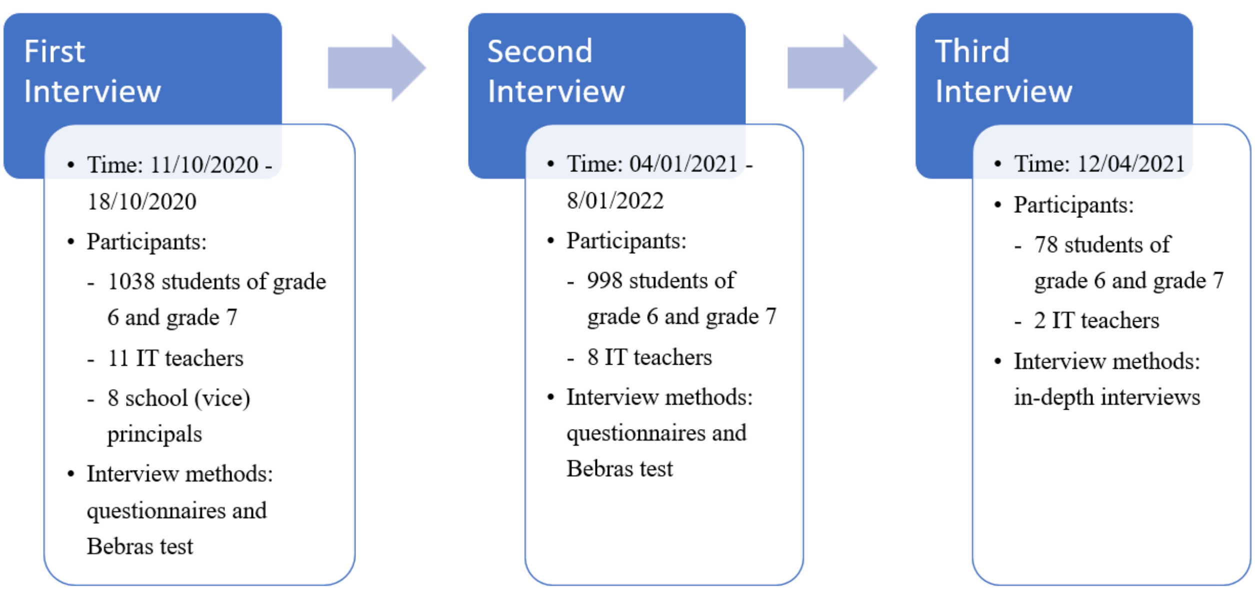
In this research, we utilized the data collected in the impact evaluation of CFGP conducted by a team of researchers from the University of Economics, Ho Chi Minh City in 2020. The author of this study is one of the team members. The evaluation obtained approval from the University of Economics Ho Chi Minh City's Institutional Review Board (IRB Study No. 1660 /QD-DHKT-QLKHHTQT).

The evaluation aimed to estimate the effects of coding courses offered by CFGP on students' learning behavior and CT skills. The data was collected from 8 schools in Lam Dong province. These schools joined the project at the time the evaluation began and therefore, they had not yet conducted any activities before that. This allowed us to collect the pre-test data for our study.

To estimate the effect of an intervention on its beneficiaries, randomization would be the most preferable approach. However, due to the local requirements (or local authority's requirements), the evaluation team faced significant challenges in implementing a randomized control study. To tackle this problem, we employed a quasi-experimental approach instead and designed a pre-test post-test study.

Specifically, in October 2020, we conducted the baseline interview with 1038 students from 8 schools in Lam Dong, who had not received any coding training from the project at that time. The students were then assigned to two groups: a treatment group and a delayed-entry control group. The treatment group, comprising half of the students, participated in coding courses by CFGP at their schools during the first semester of the academic year 2020 – 2021, from October 2020 to January 2021. The remaining students in the delayed-entry control group only joined coding training activities by the project in the second semester of the year 2020 – 2021. The treatment-control assignment was carried out by the schools through a non-random, convenient process. In addition, the training curriculum was designed to teach Micro:bit to 7th-grade students and to introduce 6th-grade students to the world of Scratch. After completing the training phase, we conducted the follow-up survey to collect the post-treatment data in January 2021. The timeline of the whole evaluation study is summarized in Figure 1.

Figure 1: Timeline of the Evaluation



Note: The interviews were conducted in 1-on-1 and in-person format.

Notably, as we were able to obtain valid and comprehensive responses from 998 students out of the initial 1038 participants in the first phase. It means our attrition rate was low, at just 3.85%. Therefore, our final sample includes longitudinal data of 998 secondary school students from the ages of 12 to 13. We will exploit the panel data structure of our data to estimate the impact of coding training despite the non-random treatment assignment.

3.3. Variables

We measure all variables at two time points: initially, at the beginning of the semester during the first interview in October 2020 (t = 0), and subsequently, at the end of the semester during the second interview in January 2020 (t = 1).

Dependent Variable: Computational thinking measure

To assess the impact of the coding program on students' computational thinking, it was crucial to measure their cognitive skills before and after the training. In our study, since we aim to understand how coding contributes to the development of universal CT skills that enable students to comprehend and collaborate with machines, we choose to adopt the operational definition of computation thinking (CSTA, 2011; Selby & Woollard, 2013). This definition characterizes CT skills as encompassing abstraction, decomposition, data analysis and representation, algorithmic thinking, evaluation and generalization. Consequently, we designed two computational thinking tests based on the Bebras UK Challenges to measure these dimensions.

The standard UK challenge for secondary students consists of 18 questions in 40 minutes. The questions in the Bebras UK challenges are categorized into six age groups, three difficulty levels, and cover multiple dimensions of CT skills. To ensure that both tests were comparable and aligned with the operational definition, we reviewed all questions from the Bebras UK challenges from 2014 to 2018, and then selected the most appropriate questions based on the age group, difficulty rating and topics covered.

We ultimately developed two separate tests, each comprising five questions. While using a short test with only five questions may have limitations in assessing students' CT skills, the constraints of our larger survey necessitated a concise test format. These tests were administered in a paper and pencil format to ease administration and minimize any technical problems. Table 2 provides a comparison of the two tests

Table 2: The Comparison of Two Tests

No.	Test for the First Round (August 2020)				Test for the Second Round (January 2021)			
	Name	Level	Computational Thinking	Source	Name	Level	Computational Thinking	Source
1	Bird Colours	Kits: C Castors: B Juniors: A	CT Skills: Generalisation CS Domain: Data, data structures and representations	UK Bebras Test Booklet 2018 Page: 24	Beaver code	Kits: Castors: B Juniors: A	CT Skills: Algorithmic Thinking, Decomposition, Generalisation CS Domain: Data, data structures and representations	UK Bebras Test Booklet 2016 Page: 20
2	Bottles	Kits: C Castors: B Juniors: A	CT Skills: Abstraction, Evaluation CS Domain: Data, data structures and representations	UK Bebras Test Booklet 2016 Page: 14	Beaver Tournament	Kits: Castors: B Juniors: A	CT Skills: Abstraction, Algorithmic Thinking, Evaluation CS Domain: Data, data structures and representations	UK Bebras Test Booklet 2017 Page: 23
3	Party Guest	Kits: Castors: B Juniors: A	CT Skills: Algorithmic Thinking, Decomposition CS Domain: Algorithms and programming	UK Bebras Test Booklet 2016 Page: 17	Rubbish Robots	Kits: Castors: B Juniors: A	CT Skills: Algorithmic Thinking CS Domain: Algorithms and programming	UK Bebras Test Booklet 2018 Page: 20

No.	Test for the First Round (August 2020)				Test for the Second Round (January 2021)			
	Name	Level	Computational Thinking	Source	Name	Level	Computational Thinking	Source
4	Parking Lot	Kits: Castors: Juniors: B	CT Skills: Algorithmic Thinking, Decomposition, Evaluation. CS Domain: Data, data structures and representations	UK Bebras Test Booklet 2017 Page: 20	Five Sticks	Kits: Castors: Juniors: B	CT Skills: Abstraction, Algorithmic Thinking, Decomposition CS Domain: Algorithms and programming	UK Bebras Test Booklet 2017 Page: 27
5	News Editing	Kits: Castors: C Juniors: B	CT Skills: Abstraction, Algorithmic Thinking, Evaluation CS Domain: Data, data structures and representations	UK Bebras Test Booklet 2017 Page: 33	Roundabout City	Kits: Castors: C Juniors: B	CT Skills: Abstraction, Algorithmic Thinking, Evaluation. CS Domain: Interactions, systems and society	UK Bebras Test Booklet 2017 Page: 35

Note: Juniors = Grade 6 & 7 (age 10 - 12); Castors = Grade 4 & 5 (age 8 - 10); Kits = Grade 2 & 3 (age 6 - 8)
CS = Computer Science; CT = Computational Thinking

The problems come in three levels of difficulty: A, B and C. The "A" problems are the easiest. "C" problems the hardest.
CT Skills include 5 skills: Abstraction; Algorithmic Thinking; Decomposition; Evaluation; Generalization.
CT Domains include 5 domains: Algorithms and programming; Data, data structures and representations; Computer processes and hardware; Communication and networking; Interactions, systems and society

A time limit of 25 minutes was allocated for students to complete the tests. This time allocation took into consideration the standard 3-minute design of Bebras questions and accounted for potential challenges in question comprehension arising from translation. Prior to the main study, a pilot test was conducted to assess the appropriateness of the time frame. The pilot results demonstrated that students were able to complete the questions comfortably within the allocated time.

To mitigate any potential ambiguities or misunderstandings that could arise from language barriers, a validation process was employed. Each test underwent a two-way translation process and received a final check from a secondary teacher. This ensured that the test items were accurately translated and that the content remained consistent and comprehensible for the target students.

The final computational thinking scores of students ranged from 0 to 5. To simplify our subsequent analysis, we converted these scores to two binary indicators, assigning a value of 1 if a student's computational thinking score is higher than the average of 3, and 0 otherwise. This conversion allowed us to perform appropriate estimations.

Treatment variable: Coding training

Our treatment variable, *Coding training*, is a dummy variable set to 1 if a student had already received coding training and 0 otherwise. Therefore, at time t = 0, *Coding training* is set to 0 for all students. Subsequently, at time t = 1, *Coding training* is set to 1 for students who were in the treated group and received coding training during the semester, while it remains 0 for students who did not undergo coding training during the same period.

Control variable

Two important time-varying[2] factors are used as the control variables in our regressions. First, students' access to computers and laptops may influence their coding learning and consequently their cognitive skills. Therefore, we control computer access in our analysis by using a dummy variable, *PC*. Students were asked to state how frequently they used computers and laptops outside of school (at home or at public shops) on a five-point scale. The dummy variable *PC* is equal to 1 if students rated their usage frequency from 3 to 5, and 0 otherwise.

[2] Since we use fixed-effect models to solve the problem of unobserved confounders, we do not include in our estimations all time-invariant control variables such as gender, age, family's income or parents' educational levels.

Second, the acquisition of cognitive skills through coding learning may also be subject to the time students spend on household responsibilities to assist their families. Students who must allocate their time and cognitive resources to household tasks may find it challenging to dedicate the focused attention necessary for cognitive development through coding training. To account for this potential effect, we include a control variable that captures the amount of time students spend on daily household chores. Students were particularly asked to select one of six categories representing their daily household chore time: “No household chore time”, “Up to 1 hour”, “Up to 2 hours”, “Up to 3 hours”, “Up to 4 hours”, and “More than 4 hours”. These categories were then coded into an ordinal variable ranging from 0 to 5, where 0 represented “No household chore time” and 5 denoted “More than 4 hours”.

4. Econometric Consideration

4.1. Identification Strategy

Our focus is on examining how engagement in coding courses affects the development of students’ computational thinking skill. In an ideal scenario with random assignment of coding courses, a straightforward comparison between the computational thinking score of the treated group and the control group would suffice. However, due to the non-random intervention assignment process, the comparison between the treatment and control groups is vulnerable to endogeneity issues.

To tackle this problem, we employ a methodological approach that leverages our panel data structure and utilizes the individual fixed effect technique. The underlying assumption of this approach is that the unobserved variables that could potentially bias the results remain constant over time. In our study, we can reasonably assume the validity of this assumption due to the short time period between our baseline (October 2020) and post-treatment (January 2021) surveys. Given the limited timeframe, it is improbable that significant changes in the unobserved factors took place during this period. As a result, we have confidence in the validity of the identification assumption of the fixed-effect approach.

4.2. Econometric Specification

To account for the cancellation of time-invariant unobserved confounders, we must employ a linear, additive function form, or a nonlinear model that allows for the exclusion of these time-invariant terms through transformation. Therefore, in this study, we use a linear probability fixed effect model with robust standard errors clustered at the school level to estimate our treatment effect.

As a robustness check, we also estimate the logit model to validate our findings. The model for estimation is:

$$Y_{it} = \beta D_{it} + X_{it}\gamma + \mu_i + \lambda_t + \varepsilon_{it}$$

in which Y_{it} represents the measurement of computation thinking skills; D_{it} denotes the treatment variable and X_{it} are the control variables. With this model, we conclude that coding training (D) has an effect on students' computational thinking skills (Y) when $\hat{\beta}$, the estimate of β, is statistically different from 0.

5. Results

5.1. Sample Composition

Table 3 provides a comprehensive overview of the sample composition. The data encompasses 1038 students who participated in the first round of interviews. Approximately 52.12% of the sample comprises students in grade 6, with the remaining students (47.88%) belonging to grade 7. Gender distribution in the sample is well-balanced, with approximately half (50.58%) being male students and the other half (49.42%) being female students.

Table 3: Composition of the Sample

		First Round (October 2020)		**Second Round (January 2021)**	
		Freq. (N)	**Perc. (%)**	**Freq. (N)**	**Perc. (%)**
Total student:		**1038**	**100%**	**998**	**100%**
Grade					
	- *Grade 6*	541	52.12%	523	52.40%
	- *Grade 7*	497	47.88%	475	47.60%
Gender					
	- *Boy*	525	50.58%	503	44.72%
	- *Girl*	513	49.42%	495	55.28%

Note: First round and second round of surveys were conducted in Lam Dong Province, in Vietnam

Four months after the initial interview, a follow-up round of interviews was conducted during the second survey, encompassing all students. However, from the total pool of 1038 participants, only 998 students contributed with valid and complete responses. This outcome shaped the final sample for analysis, creating a balanced dataset of 998 observations and reflecting a dropout rate of 3.85%.

Within this participant cohort, 6th and 7th-grade students constitute 52.40% and 47.60%, respectively. Furthermore, approximately 55.28% of the students identify as female, leaving the remaining participants as male. The distribution of students across gender and grade is presented in Table 4, showcasing an equitable spread among various sub-groups.

Table 4: Tabulation of the Final Sample by Gender and Grade

	Gender	Grade	No. of Observations (Collected)	No. of Observations (Analysis)	Percentage (Analysis column)
First Round			**1038**	**998**	**100 %**
	Boy				
		6	276	265	26.55%
		7	249	238	23.85%
	Girl				
		6	265	258	25.85%
		7	248	237	23.75%
Second Round			**998**	**998**	**100 %**
	Boy				
		6	265	265	26.55%
		7	238	238	23.85%
	Girl				
		6	258	258	25.85%
		7	237	237	23.75%

Note: First round and second round of surveys were conducted in Lam Dong Province, Vietnam

5.2. Descriptive Statistics of Variables

Table 5 provides a concise overview of all variables included in the regression analysis. Among the students in the final sample, approximately 46.9% had participated in the coding training. Interestingly, only 46% of students had access to a computer during the initial round of interviews, but this figure slightly increased to 47.6% in the second survey.

Table 5: Summary of Variables in the Regression

	Survey Round			
	Round 1		**Round 2**	
	No.	%	No.	%
Coding training				
No	998	100.0	530	53.1
Yes	0	0.0	468	46.9
Total	998	100.0	998	100.0
Computational Thinking score				
Less than or equal to 2	547	54.8	684	68.5
Higher than 2	451	45.2	314	31.5
Total	998	100.0	998	100.0
Grade				
Grade 6	523	52.4	523	52.4
Grade 7	475	47.6	475	47.6
Total	998	100.0	998	100.0
Computer access				
No	539	54.0	523	52.4
Yes	459	46.0	475	47.6
Total	998	100.0	998	100.0
Time for working household chores per day				
No household chore time	48	4.8	63	6.3
Up to 1 hour	629	63.0	619	62.0
Up to 2 hours	213	21.3	229	22.9
Up to 3 hours	66	6.6	58	5.8
Up to 4 hours	26	2.6	17	1.7
More than 4 hours	16	1.6	12	1.2
Total	998	100.0	998	100.0

Note: calculated from the collected data.

Regarding household chores, the time students spent on these activities remained quite consistent between the two rounds. Only 4.8% and 6.3% of students did not engage in any household chores during the first and second surveys, respectively. Conversely, a considerable proportion of students, 63% in the first survey and 62% in the second survey, allocated less than 1 hour to attend to household responsibilities. Moreover, the percentage of students involved in household chores decreased significantly as the working time increased.

Especially, prior to the coding training, 45.2% of students exhibited a computational thinking score higher than 2. However, after undergoing the training, this proportion decreased to 31.5%. It is essential to recognize that this decrease alone does not provide conclusive evidence of a causal relationship between the training and the decline in students' scores because other observed and unobserved factors may confound these results.

Table 6: Correlations of Variables in the Regression

	Polychoric Correlation				
	1	2	3	4	5
1. Coding training	1	-0.233***	0.017	-0.062	-0.090
2. Computational thinking score	-0.233***	1	0.059	0.145***	-0.030*
3. Grade	0.017	0.059	1	0.055	-0.015
4. Computer access	-0.062	0.145***	0.055	1	0.014
5. Time for working household chores per day	-0.090	-0.030*	-0.015	0.014	1

Note: calculated from the collected data. Polychoric correlation is employed due to the ordinal scale of variables.

Upon examining the correlation coefficients presented in Table 6, four key findings come into focus. First, there is a significant negative correlation between the computational thinking score and the training. This suggests that students who participated in the training tend to have lower scores compared to those who did not undergo the training. Second, the score demonstrates a significant and positive correlation with computer access. It means students who have access to computers generally achieve higher coding scores. Third, the correlations also reveal a significant negative relationship between the score and the number of hours spent on working household chores per day. This indicates that a lower score is observed when students spend more time on household chores. Finally, while the computational thinking score is significantly correlated with other variables, the remaining variables in the analysis do not exhibit significant correlations with each other.

5.3. Impact of Learning How to Code on Computational Thinking

Table 7 presents the outcomes of the fixed effects regression models, analyzing the impact of coding training on students' Computational Thinking Scores. Specifically, columns 1 and 2 display the estimates of the linear model and logit model, respectively, utilizing the panel data fixed effect approach. Both models use the dummy variable for having a Computational Thinking Score higher than 2 as the dependent variable.

Our findings reveal a significantly positive effect of coding training on the dependent variable in both estimations, with a very small p-value (<0.001). According to the estimates of the linear model, the probability of having a Computational Thinking Score higher than 2 increased by 8.47% when students participated in the coding course. This aligns with the positive result from the logit model, where the estimate of the coding participation variable is 0.389, equivalent to an odds ratio of 1.475. This means that if a student transitions from not learning how to code to learning how to code, their odds of having a higher CT Score are multiplied by 1.475. In other words, coding training significantly helps students increase their CT Scores in both estimations.

5.4. Impact of Different Training Programs on Computational Thinking

After establishing the overall positive impact of coding training, our study delves further into comparing the effectiveness of two different programming tools - Scratch and MakeCode for Micro:bit - on students' Computational Thinking Scores. While both tools utilize block-based visual programming, MakeCode for Micro:bit additionally incorporates physical computing, which potentially yields distinct effects on student learning outcomes. It is worth noting that, due to administrative constraints, we were unable to randomize the assignment of these tools among our samples. Consequently, Scratch was exclusively offered to students in Class 6, while MakeCode for Micro:bit was only introduced to students in Class 7. Despite this limitation, we believe that our comparative analysis still offers valuable insights into the development of computational thinking skills through coding training and the applications of these programming tools in the context of K-12 education in Vietnam.

Table 7: Linear Probability and Logit Model of Treatment on the Main Outcome

	Dependent Variable: Computational Thinking Score	
	LINEAR MODEL	LOGIT MODEL
	(1)	(2)
Round	-0.175***	-0.909***
	(0.017)	(0.075)
Coding training	0.0847**	0.389***
	(0.022)	(0.106)
Computer access	-0.0261	-0.195
	(0.026)	(0.151)
Time for working household chores		
Up to 1 hour	0.0775	0.335
	(0.045)	(0.269)
Up to 2 hours	0.0514	0.228
	(0.042)	(0.256)
Up to 3 hours	0.0855	0.390
	(0.078)	(0.482)
Up to 4 hours	0.193+	0.951+
	(0.087)	(0.560)
More than 4 hours	-0.188	-0.803
	(0.153)	(0.692)
Constant	0.396***	
	(0.028)	
N	2036	778
adj. R^2	0.051	

Notes: This table reports estimation results from the linear probability model and logit model. The dependent variable in all models is a dummy variable for having Computational Thinking Score higher than 2. Column (1) reports the results of the linear probability regression (Equation 2). Column (2) reports the results of the logit regression. Robust standard errors clustered at the team level are in parentheses. Significance levels: $^{+}$ *p<0.1; * p<0.05; ** p<0.01; *** p<0.001.*

Table 8: Estimations of Treatment on the Main Outcome by Teaching Curriculums

	Grade 6 (Scratch) *Dependent Variable: CT Score*		Grade 7 (Micro:bit) *Dependent Variable: CT Score*	
	LINEAR MODEL (1)	LOGIT MODEL (2)	LINEAR MODEL (3)	LOGIT MODEL (4)
Round	-0.220***	-1.189***	-0.128***	-0.711***
	(0.036)	(0.180)	(0.023)	(0.143)
Coding training	0.0575	0.138	0.118*	0.721***
	(0.046)	(0.272)	(0.035)	(0.211)
Computer access	0.00884	0.0808	-0.0523	-0.411+
	(0.049)	(0.314)	(0.031)	(0.221)
Time for working household chores				
Up to 1 hour	0.0817	0.263	0.0712	0.457
	(0.064)	(0.336)	(0.061)	(0.399)
Up to 2 hours	0.0426	-0.0249	0.0629	0.540
	(0.067)	(0.299)	(0.079)	(0.422)
Up to 3 hours	0.143	0.565	-0.0275	-0.0665
	(0.087)	(0.620)	(0.105)	(0.518)
Up to 4 hours	0.289	1.833*	0.134	0.737
	(0.163)	(0.814)	(0.135)	(0.644)
More than 4 hours	0.0125	0.0422	-0.430	-1.804
	(0.233)	(1.058)	(0.274)	(1.483)
Constant	0.383***		0.409***	
	(0.051)		(0.059)	
N	1046	410	950	368
adj. R^2	0.103		0.033	

*Note: This table reports estimation results for different teaching curriculums. The dependent variable in all models is a dummy variable for having Computational Thinking Score higher than 2. Column (1) and (2) reports the results of estimations for students learning Scratch. Column (3) and (4) report the results of the and logit regression. Significance levels: * p<0.05; ** p<0.01; *** p<0.001*

Table 8 presents the results of our comparative analysis. In columns 1 and 2, the estimates for Scratch training show positive but statistically insignificant impacts on students' CT scores. This suggests that learning coding using Scratch has a positive effect on CT scores, but the difference is not statistically significant. On the other hand, the estimates in columns 3 and 4 demonstrate that learning how to code with Micro:bit has a significantly positive impact on students' CT scores.

In detail, the coefficient for Micro:bit is 0.118 in column 3, indicating that Micro:bit training led to an 11.8% increase in the probability of students achieving a high CT score, with a significance level of 5%. Likewise, in column 4, the coefficient for Micro:bit is 0.721, corresponding to an odds ratio of 2.056. This means that students, who transition from not learning Micro:bit to learning it, have their odds of achieving a higher CT Score multiplied by 2.056. In other words, learning a programming language with integrated physical computing, like Micro:bit, significantly boosts students' CT Scores, while learning coding without physical computing (Scratch) does not demonstrate a significant impact. However, caution should be exercised when interpreting these findings due to the potential grade-level confounding. In the next section, additional supporting evidence is provided to support this assertion.

5.5. Further Evidence from In-depth Interview

To draw conclusions regarding the relative impact of Scratch and MakeCode for Micro:bit, it is important to establish that students in Class 7 and Class 6 do not significantly differ in their ability to learn coding concepts and develop their computational thinking skills through coding. This ensures that any variations in the impact of the two tools can be attributed to their unique characteristics rather than disparities in students' ability stemming from their respective grade levels. While we cannot completely rule out the possibility of disparities, our in-depth interviews with IT teachers and educational experts strongly support the likelihood that such differences are minimal or negligible between the two grade levels. An IT teacher participating in our study explained:

> *"I don't think there are any differences between Grade 6 and Grade 7 students in their capacity to learn coding and develop computational thinking skills. First, in this school year (school year 2020 – 2021,) we continue using the old curriculums for both Grade 6 and Grade 7. In Grade 6, students acquaint themselves with computer fundamentals, such as computer usage basics and introductory MS Word skills. Meanwhile, Grade 7 students learn how to use MS PowerPoint, MS Excel (only*

basic functions such as sum or average) and other simple applications. These contents do not encompass any programming or computational knowledge. Second, based on my teaching experience, students in both grades demonstrate an equal aptitude for learning coding with tools like Scratch or MakeCode, encountering no notable difficulties. The curriculum structures in Grade 6 and Grade 7 are fundamentally similar, providing foundation knowledge of the secondary level. Therefore, I believe that students in Grade 6 and Grade 7 share a similar ability to learn coding and comprehend computational concepts."

Our qualitative interviews with IT teachers and students also shed light on the divergent impact of coding training with Scratch and MakeCode for Micro:bit. An IT teacher discussed the differences between learning coding with Scratch and with MakeCode for Micro:bit as follows:

> *"When students learn coding through Scratch, their activities revolve around creating basic animations and small-scale computer games, primarily within the digital sphere. In contrast, students who engage with Micro:bit find themselves working with real-world hardware to design physically-operating applications. While these applications may be very simple, students still need to think about how code interacts with hardware components and how programming concepts can be used to solve real-work problems. This practical approach may make coding with Micro:bit more effective for skill development."*

An educational expert from the project commented further on the differences between Scratch and Micro:bit:

> *"Effectively teaching coding through visual block-based tools requires an active teaching approach. Teachers must create a learning environment in which students actively engage with programming concepts by solving problems, avoiding the mere mechanical manipulation of graphical blocks without a profound understanding of the underlying logic and algorithms. Achieving this can be challenging in our context (Vietnam,) where large class sizes and traditional teaching methods prevail. Therefore, teaching coding with Micro:bit may prove more effective, as it fosters increased exploration from students and compels teachers to transition towards more active teaching approaches."*

Hence, the underlying mechanisms driving the divergent impacts of Scratch versus MakeCode for Micro:bit center on the capacity to connect programming concepts with problem-solving scenarios during the coding training. The integration of Micro:bit and physical devices into coding training seems to render the learning experience more tangible and concrete, thereby enhancing its effectiveness in nurturing computational thinking skills.

6. Discussion

In this study, we examine how participating in a coding training program affects the development of students' computational thinking skills in Vietnam. We focus on the impact of two visual block-based programming tools: Scratch and MakeCode for Micro:bit. Our quasi-experimental analysis reveals that coding training has a positive impact on students' computational thinking scores. However, this impact occurs exclusively among Grade 7 students who engaged in coding with MakeCode for Micro:bit, while no such effect was observed for Grade 6 students who underwent coding training with Scratch. Qualitative data obtained through interviews with IT teachers and educational experts provides further evidence supporting these findings. Insights from these interviews highlight the importance of students acquiring programming concepts through problem-solving as a key factor in fostering computational thinking skills during coding training.

The findings of our study contribute new evidences to the growing body of literature on the development of computational thinking through coding education among K-12 students. Despite the noted prevalence of computational thinking educational programs and coding initiatives aimed at younger students, Lye and Koh (2014) highlighted that a majority of studies have been concentrated in higher education settings. Lockwood and Mooney (2018) documented the increasing spread of such programs in secondary schools, thereby emphasizing the need for more extensive research to understand these initiatives' effectiveness thoroughly. Our research responds to this call by clarifying the effectiveness of coding programs.

Additionally, our research enriches the literature on computational thinking skill development among K-12 students by conducting a comparative analysis of two visual block-based programming tools, Scratch and MakeCode for Micro:bit. Our examination of the impacts of these tools on fostering CT skills, alongside the exploration of the mechanisms underlying their different effects, provides critical insights. Particularly, we spotlight the capacity of MakeCode for Micro:bit to connect coding concepts with real-world applications and its role in driving pedagogical innovation. This contribution is especially significant, aligning with the call from Fraillon et al. (2018) and Tang et al. (2020) for more in-depth understanding of effective practice in promoting CT skill development.

Our findings hold significant implications for educational policymakers. First, our results indicate that coding training significantly contributes to the development of students' computational thinking skills - a competence increasingly vital in our technology-driven world. Thus, the early introduction of coding courses into the curriculum is preferable to equip students with essential skills for the future. Second, the differences we observed between Scratch and MakeCode for Micro:bit suggest that to develop computational thinking skills, it is necessary for students to actively engage with programming concepts to solve (real-world) problems. Policymakers should take these insights into account when designing coding curricula, and teacher training programs should also incorporate them to equip educators with the knowledge and pedagogical strategies necessary to impart computational thinking effectively.

Our study does exhibit several limitations, which in turn, present opportunities for future research. First, our research was confined to a single province in Vietnam and concentrated solely on two specific grades, Grade 6 and Grade 7, within the educational system. This localized context could constrain the generalizability of our findings to other regions and educational levels. Future research could broaden their scope to incorporate a more diverse range of educational settings. Second, while the identification assumption of our comparative analysis is supported by qualitative data, ideally, we would have implemented randomization in the assignment of coding programs among students at the same grade to compare the effects of different tools. Incorporating true randomization into such studies would enhance the validity of our findings. Lastly, although our qualitative data hints at the underlying mechanism of the impact of coding training on computational thinking scores, emphasizing the importance of students actively utilizing programming concepts to solve problems, we do not directly test these mechanisms or specific teaching methods. Future research should delve deeper into the teaching methodologies employed during coding training to uncover the causal mechanisms underpinning the observed effects and further refine pedagogical strategies in this context.

References

Aho, A. V. (2012). Computation and computational thinking. *The Computer Journal, 55*(7), 832-835.

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.

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada* (Vol. 1, p. 25).

Computer Science Teachers Association (CSTA) Standards Task Force (2011). *CSTA K-12 computer science standards* (pp. 9). Retrieved from: https://members.csteachers.org/documents/en-us/46916364-83ab-4f51-85fb-06b3b25b417c/1/

Dagienė, V., & Futschek, G. (2008). Bebras international contest on informatics and computer literacy: Criteria for good tasks. In *Informatics Education-Supporting Computational Thinking: Third International Conference on Informatics in Secondary Schools-Evolution and Perspectives, ISSEP 2008 Torun Poland, July 1-4, 2008 Proceedings 3* (pp. 19-30). Springer Berlin Heidelberg.

Dagienė, V., Pelikis, E., & Stupurienė, G. (2015). Introducing Computational Thinking through a Contest on Informatics: Problem-solving and Gender Issues. *Information Sciences/Informacijos Mokslai*, *73*.

Falloon, G. (2016). An analysis of young students' thinking when completing basic coding tasks using Scratch Jnr. On the iPad. *Journal of Computer Assisted Learning*, *32*(6), 576-593.

Fraillon, J., Ainley, J., Schulz, W., Friedman, T., & Duckworth, D. (2020). *Preparing for life in a digital world: IEA international computer and information literacy study 2018 international report* (p. 297). Springer Nature.

Garcia-Penalvo, F. J., & Mendes, A. J. (2018). Exploring the computational thinking effects in pre-university education. *Computers in Human Behavior*, *80*, 407-411.

González, M. R. (2015). Computational thinking test: Design guidelines and content validation. In *EDULEARN15 Proceedings* (pp. 2436-2444). IATED.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher, 42*(1), 38-43. https://doi.org/10.3102/0013189X12463051

Grover, S., & Pea, R. (2018). Computational thinking: A competency whose time has come. *Computer science education: Perspectives on teaching and learning in school*, *19*(1), 19-38.

Hooshyar, D., Malva, L., Yang, Y., Pedaste, M., Wang, M., & Lim, H. (2021). An adaptive educational computer game: Effects on students' knowledge and learning attitude in computational thinking. *Comput. Hum. Behav., 114*, 106575.

Juskeviciene, A., Stupuriene, G., & Jevsikova, T. (2021). Computational thinking development through physical computing activities in STEAM education. *Computer Applications in Engineering Education*, *29*(1), 175-190.

Kerimbayev, N., Nurym, N., Akramova, A., & Abdykarimova, S. (2023). Educational Robotics: Development of computational thinking in collaborative online learning. *Educ Inf Technol*. https://doi.org/10.1007/s10639-023-11806-5

Kite, V., Park, S., & Wiebe, E. (2021). The Code-Centric Nature of Computational Thinking Education: A Review of Trends and Issues in Computational Thinking Education Research. *SAGE Open, 11*(2). https://doi.org/10.1177/21582440211016418

Koh, K. H., Basawapatna, A., Bennett, V., & Repenning, A. (2010). Towards the Automatic Recognition of Computational Thinking for Adaptive Visual Language Learning. In *Proceedings of the 2010 IEEE Symposium on Visual Languages and Human-Centric Computing (VLHCC '10)*. *IEEE Computer Society*, USA, 59 - 66. https://doi.org/10.1109/VLHCC.2010.17

Kong, S. C. (2016). A framework of curriculum design for computational thinking development in K-12 education. *Journal of Computers in Education*, *3*, 377-394.

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in human behavior*, *72*, 558-569.

Lee, S., Kim, J. & Lee, W. (2017). Analysis of Factors Affecting Achievement in Maker Programming Education in the Age of Wireless Communication. *Wireless Pers Commun*, 93, 187–209. https://doi.org/10.1007/s11277-016-3450-2

Lockwood, J., & Mooney, A. (2018). *Developing a computational thinking test using Bebras problems*. In CC-TEL 2018 and TACKLE 2018 Workshops, Leeds. Retrieved from https://mural.maynoothuniversity.ie/10316/

Lockwood, J., & Mooney, A. (2018). Developing a computational thinking test using Bebras problems.

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. https://doi.org/10.1016/j.chb.2014.09.012

Moreno-León, J., & Robles, G. (2015). Computer programming as an educational tool in the English classroom a preliminary study. In *2015 IEEE global engineering education conference (EDUCON)* (pp. 961-966). IEEE.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic books.

Papert, S. (1981). Computers and computer cultures. *Creative Computing*, *7*(3), 82-92.

Perez-Marin, D., Hijon-Neira, R., Bacelo, A., & Pizarro, C. (2020). Can computational thinking be improved by using a methodology based on metaphors and scratch to teach computer programming to children? *Comput. Hum. Behav., 105*, 105849.

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.

Repenning, A., Webb, D. C., Brand, C., Gluck, F., Grover, R., Miller, S., ... & Song, M. (2014). Beyond Minecraft: Facilitating computational thinking through modeling and programming in 3d. *IEEE Computer Graphics and Applications*, *34*(3), 68-71.

Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM technical symposium on Computer science education (SIGCSE '10)*. Association for Computing Machinery, New York, NY, USA, 265 - 269. https://doi.org/10.1145/1734263.1734357

Roman-Gonzalez, M., Moreno-Leon, J., & Robles, G. (2017). Complementary tools for computational thinking assessment. *Proceedings of International Conference on Computational Thinking Education (CTE 2017).*

Roman-Gonzalez, M., Moreno-León, J., & Robles, G. (2019). Combining assessment tools for a comprehensive evaluation of computational thinking interventions. *Computational thinking education*, 79-98.

Scherer, R., Siddiq, F., & Viveros, B. S. (2020). A meta-analysis of teaching and learning computer programming: Effective instructional approaches and conditions. *Computers in Human Behavior*, *109*, 106349.

Selby, C., & Woollard, J. (2013). Computational thinking: The developing definition. Retrieved from http://eprints.soton.ac.uk/356481

Stewart, W., & Baek, K. (2023). Analyzing computational thinking studies in Scratch programming: A review of elementary education literature. *International Journal of Computer Science Education in Schools*, *6*(1), 35-58.

Su, Y. S., Shao, M., & Zhao, L. (2022). Effect of mind mapping on creative thinking of children in scratch visual programming education. *Journal of Educational Computing Research*, *60*(4), 906-929.

Tang, X., Yin, Y., Lin, Q., Hadad, R., & Zhai, X. (2020). Assessing computational thinking: A systematic review of empirical studies. *Computers & Education, 148*, 103798.

United Nations Educational, Scientific, and Cultural Organization (UNESCO) (2022). Computational thinking, artificial intelligence and education in Latin America (programme and meeting document). Retrieved from https://unesdoc.unesco.org/ark:/48223/pf0000381761

Weintrop, D. (2016). *Modality matters: Understanding the effects of programming language representation in high school computer science classrooms* (Doctoral dissertation, Northwestern University).

Weintrop, D., & Wilensky, U. (2016). Bringing blocks-based programming into high school computer science classrooms. In *Annual Meeting of the American Educational Research Association (AERA). Washington DC, USA*.

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, 127-147.

Wing, J. (2011). Research notebook: Computational thinking - What and why. *The link magazine*, *6*, 20-23.

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

Xu, W., Geng, F., & Wang, L. (2022). Relations of computational thinking to reasoning ability and creative thinking in young children: Mediating role of arithmetic fluency. *Thinking Skills and Creativity*, *44*, 101041.

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.

Zhang, L., & Nouri, J. (2019). A systematic review of learning computational thinking through Scratch in K-9. *Computers & Education*, *141*, 103607.

Chapter 3

The Development and Validation of a Reliable Scale for Measuring Students' Motivation, Attitudes, and Interests towards Coding in Vietnam

Abstract: In response to the challenges posed by digital transformation and the emergence of Industry 4.0, there is a growing emphasis on teaching coding to young learners in Vietnam. Many anticipate that this training will greatly benefit students. However, a notable gap exists: there is no specialized scale in Vietnamese to assess the impact of coding education on students. Addressing this gap was the primary objective of our study, which aimed to create and validate the Coding Motivation Scale for Middle School Students (CMS-M). Using Confirmatory Factor Analysis and Reliable Analysis, we validated this scale by gathering responses from 1038 students aged 11 to 12 in 6th and 7th grades. This process unveiled a 36-item scale consisting of 7 constructs that effectively capture young learners' attitudes toward coding. These constructs encompass coding confidence, coding interest, gender perception, general usefulness, academic usefulness, academic self-efficacy and attitudes, and social perception. In addition, the measurement invariance analysis yielded an important result: the CMS-M exhibited consistency across various gender and grade levels. This underscores its reliability in evaluating coding attitudes among diverse groups of young learners.

1. Introduction

The Fourth Industrial Revolution (Industry 4.0) driven by the fusion of physical, digital, and biological domains has far-reaching impacts on all disciplines, economies and industries (Schwab, 2017; UNDP, 2018). On one hand, Industry 4.0's core technologies have the potential to increase productivity, expand consumer choices, enhance resource usage efficiency and help to achieve circular economy models (ADB, 2021; UNIDO, 2019). On the other hand, these technologies, especially automation and Artificial Intelligence, can lead to substantial labor market disruption, reverse flows of foreign direct investment as well as reshoring and near-shoring trends (Chang, Rynhart & Huynh, 2016; UNIDO, 2019). These problems are particularly exacerbated in developing countries where the traditional economic growth models based on a comparative advantage of low-skilled labor force, foreign investment and export-oriented industrialization are now no longer effective (ADB, 2021; UNDP, 2018).

To maximize the benefits and minimize the risks posed by Industry 4.0 in developing countries, education strategies that focus on reskilling and upskilling the workforce to align with emerging job requirements are crucial (ADB, 2021; UNIDO, 2019). Education policies should target two skill groups, namely analytical skills and self-management skills. The first group, which is consistent year-over-year demand, includes analytical thinking, complex problem solving and critical thinking. The second group, which is newly emerging in recent years, consists of stress tolerance, resilience, flexibility and active learning (Schwab & Zahidi, 2020). Therefore, education reform aimed at equipping young generations with digital and analytical skills are now strongly advised to developing countries' governments. This will enable them to leapfrog Industry 4.0's technology waves and avoid a repeat of falling behind as seen in the previous industrial revolutions (UNIDO, 2018).

Vietnam, a typical developing nation whose economic growth has been relying on a young population, cheap labor and high foreign direct investment, is obviously facing significant threats from Industry 4.0. As a result, Vietnam's government is urged to implement reskilling and upskilling programs to maintain its position as a country having the second highest growth rate per capita in the world since 1990 (ADB, 2021; UNDP, 2018). Various education policies with the aim of enhancing digital and analytical skills for the new generation were issued in the K-12 Education System of Vietnam (Cammaert, Nguyen & Tanaka, 2020; UNDP, 2018). For example, the Prime Minister's Directive No. 16/CT-TTg, dated May 4, 2017, emphasizes the importance of promoting STEM (Science, Technology, Engineering, and Mathematics) and informatics in the general education program. To implement this directive, the Ministry of Education and Training has taken steps to provide teacher training in STEM education and integrate STEM education into the new general education curriculum from 2020 (Official dispatch no.3089/BGDĐT-GDTrH). Moreover, local authorities in various provinces across Vietnam have provided favorable support to non-government organizations that offer free programs aimed at enhancing K-12 students' digital literacy, coding experience, and computer thinking skills (Vu, 2023).

These efforts highlight the government's dedication to education reform towards STEM, with digital literacy and coding literacy training being an important component. This is echoed by the views of many parents and teachers in Vietnam who believe that learning how to code is essential in preparing students for Industry 4.0. Despite the widespread support for coding education, there is limited research on evaluating its impact on students' abilities, interests, and attitudes towards STEM and coding. To the best of my knowledge, there are currently no specific questionnaires or scales designed to measure the changes in feelings and opinions of K-12 students in Vietnam after participating in computer courses. To fill this gap, this study aims to develop and validate a reliable scale to measure students' motivation, attitudes and interests towards coding in Vietnam. This scale especially needs to have the properties of measurement invariance across groups for meaningful and valid comparisons across different groups of students. In this way, this instrument is providing a fundamental stone for assessing the effectiveness of coding education in the new teaching and learning reforms of Vietnam as recommended by Cammaert, Nguyen and Tanaka (2020).

The subsequent sections of this paper are organized as follows: Section 2 introduces the general process of developing a new scale. Section 3 outlines the theoretical background for creating new measures and items, covering the primary psychological framework, relevant prior scales, and our item selection. Section 4 provides a detailed account of our data collection procedures, which involved three surveys, along with descriptions of the participants. Moving forward, Section 5 elaborates on our analysis strategy, and Section 6 presents the results of our analysis. In Section 7, we engage in a comprehensive discussion of our results and the final new scale, followed by an exploration of the limitations of this research in Section 8. Finally, Section 9 serves as the conclusion, summarizing the key findings and contributions of our study.

2. Theoretical Framework: The Scale Development Process

To develop the scale, we followed the scale development processes proposed by various studies, including DeVellis (2017), Boateng et al. (2018), Furr (2011), Hinkin (2005), Messick (1995), Trochim (2006). Although the processes outlined in these studies vary in the number of steps and their definitions, they share common principles in developing a scale. Typically, scale development may be divided into two stages, namely "developing new measures" and "testing the new measures". Each stage consists of several steps that are summarized in Figure 1.

Figure 1: Scale Development Process

STAGE 1: DEVELOPING THE SCALE

- Step 1: Construct Identification
- Step 2: Item Generation
- Step 3: Item's Content Validation
- Step 4: Pilot Test

STAGE 2: TESTING THE NEW SCALE

- Step 5: Survey Administration
- Step 6: Exploratory Factor Analysis
- Step 7: Reliability and Validity Assessment

The first step of the scales development process is to determine what to measure by using a suitable theoretical framework or empirical evidence. This step is often referred to identification of domain or constructs, which involves defining the concept, attribute, or unobserved behavior being studied (Devellis, 2017; Hinkin, 2005). At this step, it is important to achieve proper conceptualization and identification of the constructs because failing to do so can lead to a deficient scale that only partially captures the targeted constructs. A well-conceptualized domain can provide researchers with more knowledge about the phenomenon being studied, such as the description of the domain, the existence of similar scales, the dimensions of the domain and the methods of assessing these dimensions (Boateng et al, 2018). These understandings serve as a comprehensive background for the next step in the process.

Once the domain is delineated, the next step is to generate a pool of items or questions that measure the constructs of interest. This process is labeled as question development or item generation, in which the number of the questions, the wording of the item, the form of the questions (open, closed items or the combination of them) and the format of responses (dichotomous or Likert scales) should be seriously considered. In general, studies can use either deductive or inductive methods, or a combination of both to generate questions, given that construct underrepresentation and construct-irrelevant variance are avoided. Failure to avoid these pitfalls can lead to the invalidation of the scale due to inaccurate measurement of the intended constructs (Boateng et al, 2018; Hinkin, 2005).

The third step is to conduct a "theoretical analysis" to evaluate the content validity of the scale. This step is meant to assess the extent to which the item pool is suitable for evaluating the domain or constructs of interest. In addition, checking content validity aims to guarantee a high level of content relevance and content representations, which allows the items to effectively capture the relevant experience and characteristics of the population being examined. Content validity can be checked through comments of experts or feedback from the target population. While experts with in-depth knowledge can provide theoretical and empirical evaluations of the domain and items, the target population can provide feedback on their experience as end-users in understanding and answering included items. It is ideal to combine both the opinions of expert and end-user, but the first one is preferred in case resources are limited. It is also encouraged to increase the number of experts and end-users participating in this step to reduce bias. At the end of the step, a draft survey with potential questions is generated.

The fourth step, which is the last step of the first stage, is a pilot test that involves administering the draft survey to a small sample of the target population. This step can be thought of as pre-testing the scale before it is officially distributed (Boateng et al, 2018; Carpenter, 2018). Unlike the third step, which involved administering draft items to the target population through office meetings, the pilot test involves a larger number of participants under realistic conditions in which the survey will be conducted. The sample size for the pilot test can vary from 5 to 100 people according to the diversity of the population (Carpenter, 2018). The goal of the pilot test is to refine the draft items by assessing the wording and meaning of the items, and then rephrasing them to be more easily understood, thereby reducing the cognitive burden on research participants.

During the pilot test, participants are not only asked to answer the draft items but also to describe their mental process of answering the questions, i.e., verbalize how they understand and respond to the questions. This approach is known as cognitive interviewing and is used to gather qualitative data that helps to clarify whether the items are understood as intended by developers and to ensure that respondents are able to provide answers that reflect their genuine thoughts, thus ensuring that items can generate the targeted information accurately (Boateng et al, 2018; Carpenter, 2018). Furthermore, quantitative data collected in the pilot test can be analyzed with the Exploratory Factor Analysis (EFA) method if the sample size ranges from 50 to 100 respondents. Results from the EFA help to identify whether items belong to intended factors, eliminate irrelevant items and add more items if necessary. Once the draft scale is adjusted to achieve the final instrument, it is ready for full-scale administration and further testing in stage 2 (Carpenter, 2018; Hinkin, 2005).

The fifth step of a scale development, which is the initial phase of stage 2, "testing the new measures", involves administering the adjusted instrument on a large scale to the target population. This step is crucial and serves as the foundation for the rest of the stage since data gathered during this stage is used in subsequent steps. Therefore, to ensure the quality of the collected data, it is important to carefully select a sample that is representative of the target population and has an appropriate size. It is also important to carefully choose the data collection method, as there are various options available such as paper and pen interviewing, computer-assisted personal interviewing, and computer-assisted web interviewing. These methods can be divided into two categories based on their interviewer involvement, i.e., with or without interviewers. Each approach has its own advantages and disadvantages, and the chosen method should be based on several factors such as population attributes, scale characteristics, sample size, and research conditions.

The sixth step after data collection is to perform preliminary EFA to assess and further refine the new scale if needed. This analysis involves extracting latent factors that represent the common variance in items' responses, with the assumption that relevant items belonging to a common latent should have systematic interdependence. Through estimating the best-fit latent constructs based on the interrelationships among items, the hypothetical factor structure of the scale can be verified. Factor analysis also allows for the creation of a concise survey by removing inadequate items that have low shared variance (<10%) or low loading (<0.3) or high cross-loading on multiple factors. At the end of this step, a concise scale matching the predetermined domains is generated. However, it is important to note that the factor structure found in this step is still hypothetical and requires various tests through Confirmatory Factor Analysis (CFA) in the last step.

The final step involves conducting several tests to ensure the reliability and validity of the new scale. These tests include evaluations of dimensionality, internal consistency, and measurement invariance. They help to verify that the items, theoretical domains, and extracted factor structure function appropriately and consistently across different samples. Typically, data for these tests is collected from a new sample of the target population or by re-interviewing the previous sample from the preliminary factor analysis at a different time. The newly acquired data is then subjected to CFA using Structural Equation Modeling (SEM) and Cronbach's Alpha Analysis.

SEM is a popular method for testing dimensionality and measurement invariance in CFA. In this approach, the hypothetical model derived from the factor structure identified in the preliminary EFA is evaluated by assessing the goodness of fit between the hypothetical model and the sample data. A good fit indicates that the factor structure is appropriate for the data. SEM also helps to assess the measurement invariance property of the scale, which refers to the extent to which the scale can be used to measure the same psychometric domains across groups and over time. To examine measurement invariance, five sequential tests are conducted: configural, metric, scalar, strict (residual), and structural tests. Since each subsequent test was built on the previous one, further tests are not necessary if the configural invariance is not established.

Once the dimensionality of the scale has been confirmed, it is necessary to assess the internal consistency of a number of items that supposedly form a construct. The commonly used method for this assessment is Cronbach's Alpha Analysis, which involves calculating Alpha coefficients for each factor. A high value of Alpha coefficient provides strong evidence for the existence of a latent factor and indicates that the set of items is closely related. Conversely, if an item does not form the latent construct, its exclusion leads to an increase in the Alpha coefficient, so it should be dropped from the factor. This examination of internal consistency helps to ensure that the set of items together captures the underlying factor adequately as predetermined.

3. Developing New measures and Items

3.1. Theoretical Framework

In the field of education, numerous psychological theories have been adopted to explain and measure the driving force, i.e., motivation, behind students' learning interests and attitudes. However, due to differences in analysis approach and modeling, motivation is defined in multiple ways in these theories (Di Serio, Ibáñez & Kloos, 2013). Consequently, researchers in educational contexts have used various definitions, which describe motivation as "the psychological driving force that enables action in the pursuit of that goal" (Lewin, 1935), "that which activates and directs behavior toward certain goals" (Berhenke et al., 2011), and "the process whereby goal-directed activities are instigated and sustained" (Schunk, Meece & Pintrich, 2014). While many definitions exist, they all highlight four key characteristics: it is a process; it is goal oriented; it involves both physical and mental activities; and it entails both the initiation and persistence of actions directed towards attaining the goal (Cook & Artino, 2016; Schunk, Meece & Pintrich, 2014).

There are five major theories of motivation frequently referenced in the educational literature, which are Expectancy Value Theory (EVT), Social-cognitive Theory, Goals and Goal Orientations, Attribution Theory, and Self-determination Theory (Cook & Artino, 2016; Graham & Weiner, 2012; Schunk, Meece & Pintrich, 2014). Among these theories, Expectancy Value Theory of Eccles et al. (1983) is well-established and has been supported by many empirical studies in educational settings (Schunk, Meece & Pintrich, 2014). Brophy (2010) even claimed that EVT is more comprehensible to understand students' performance and choices. Due to its comprehensiveness and clarity, EVT is selected as the foundational theory which sheds light to motivation and other relevant constructs used to develop the scale in this study.

The EVT was primarily developed by Eccles, Wigfield and their colleagues by integrating two perspectives: Lewin's level of aspiration and Atkinson's achievement motivation (Atkinson, 1957; Lewin, 1942; Schunk, Meece & Pintrich, 2014). The theory asserts that people's expectations for success and subjective task values are critical factors that determine their activity choice, performance, and persistence, i.e., motivation (Eccles et al., 1983; Eccles & Wigfield, 2020; Wigfield, 1994; Wigfield & Eccles, 2000). Additionally, task-specific beliefs, such as ability beliefs, perceived task difficulty, personal goals, self-schema, and affective memories, shape expectations and values. These belief variables are in turn influenced by individuals' perceptions of their previous experiences and various socialization factors, such as academic self-concepts, gender roles, social perceptions (Eccles & Wigfield, 2020; Wigfield & Eccles, 2000;). Until recently, the theory has been upgraded to the Situated Expectancy Value Theory (SEVT) to emphasize the importance and primacy of situational factors in shaping individuals' behaviors, acknowledging the situated nature of all aspects in the model (Eccles & Wigfield, 2020).

The SEVT also claimed that expectancy and value are distinct constructs consisting of multiple components (Eccles & Wigfield, 2020). Specifically, Eccles et al. (1983) proposed two components for expectancy: (1) ability beliefs, which reflect how students perceive their current capabilities; and (2) expectancy beliefs, which reflect how students view their future potential. For value, there are four components, of which three positively contribute to students' behaviors: (1) intrinsic value, which refers to the enjoyment derived from doing activities; (2) utility value, which is the achievement of present or future goals by engaging in activities; and (3) attainment value, which involves the achievement of personal identity by engaging in activities. The fourth value component is perceived cost, which negatively impacts student motivation because it reflects the sacrifices students must make to choose and perform activities. Perceived cost can be complex and multidimensional, including effort, opportunity, and emotional/psychological costs. Due to its complexity and multi-dimensionality, the cost component has been studied less in the literature than intrinsic and utility value (Eccles & Wigfield, 2020).

3.2. Prior Studies

Various scales have been developed to assess the attitude and interest of K-12 students towards coding, such as the Elementary Student Coding Attitudes Survey (ESCAS) by Mason and Rich (2020) and the Computer Science Attitudes Scale for Middle-Grade Students (MG-CS Attitudes) by Rachmatullah et al. (2020). Both scales are grounded in EVT and were carefully designed and validated using large sample sizes. The ESCAS interviewed 6040 students in grades 4-6 and the MG-CS Attitudes surveyed 663 students in grades 6-8. However, the two scales differ in the number of items and constructs they rely on to assess students' motivation for programming. The MG-CS Attitudes measures 2 constructs, namely self-efficacy and outcome expectancy, with 9 items, while the ESCAS has 23 items and captures 5 constructs, including coding confidence, coding interest, utility, social influence, and perceptions of coders. Despite their differences, these scales were both developed and validated in the USA and are only available in English. This poses potential issues of measurement error and content invalidity when applied to non-Western countries, such as Vietnamese students. Therefore, careful consideration is needed when adopting these instruments in Vietnam.

There are several other instruments developed and validated outside of the USA, with prominent examples in Germany, Hong Kong, and Türkiye. In Germany, Leifheit et al. (2020) developed the Self-Concept and Attitude toward Programming Assessment (SCAPA), which includes 50 items and assess 7 constructs: programming understanding, ability self-concept, intrinsic value belief, attainment value belief, utility value belief, cost belief, programming compliance and persistence. Although their study does not explicitly state any philosophical framework, the included constructs appear to align with the EVT by Wigfield and Eccles (2000). It is evident that SCAPA shares a similar approach and constructs with the two scales developed in the USA, indicating a high level of similarity between them.

In Türkiye, the Computational Thinking Scales (CTS) by Korkmaz et al. (2017) and the Computer Programming Self-Efficacy Scale (CPSES) by Kukul et al. (2017) are two widely referenced instruments. Unlike the ESCAS and MG-CS Attitudes, the CTS (29 items) and CPSES (31 items) are not fully grounded in a psychological theory; rather, they are primarily based on the concept of self-efficacy from Bandura's Social Cognitive Theory (Schunk, Meece & Pintrich, 2014). These instruments were designed to evaluate students' self-efficacy towards various computational thinking concepts and skills, such as problem-solving, creativity, and algorithmic thinking. Since the scales' items were developed to assess in-depth understanding of coding, students may find the questions challenging to answer if they have only heard about or have a basic understanding of programming.

A similar effort by Tsai et al. (2018) was undertaken with the CPSES, which also has a lack of psychological framework for its development and focus on measuring self-efficacy of students in 5 programming practices like Turkish scales. These coding skills, including algorithm, logical thinking, debugging, control and cooperation, are assessed by a set of 16 items. Although CPSES were developed and validated with a sample of 106 college students, the authors claim that the instrument aims at all students above middle school. Another scale developed in Hong Kong is the Programming Empowerment Survey (PES) by Kong et al. (2018) with a sample of 287 primary school students. The PES relies on the framework of programming empowerment to develop a main list of 15 items to assess 4 components of students' programming empowerment, namely programming self-efficacy, creative self-efficacy, meaningfulness and impact. Additionally, the scale measures students' interest and attitude towards programming collaboration (4 items for each construct). Compared to ESCAS by Mason and Rich (2020), the constructs of programming self-efficacy, meaningfulness, and interest of PES overlap with coding confidence, utility, and interest in the ESCAS.

Since the CPSES and PES instruments may not have been originally written in English, the phrasing of their items shown in publications may differ from their authentic text. As word choice can affect how respondents comprehend and respond to questions, there is a concern that the validation of these scales may not be applicable to their English versions (Messick, 1995). However, despite this potential issue, these instruments could still be a useful reference for developing a new scale in

Vietnam since they were formulated and validated with samples of Hong Kong students who share many cultural and perceptual similarities with Vietnamese students.

In general, the new scale that is suitable for the Vietnam context could be developed by incorporating the strengths of all the existing instruments mentioned above and addressing their limitations. For instance, it could include a more comprehensive psychological framework for its development and assess a broader range of constructs, rather than just self-efficacy and coding skills. Especially, the new scale should be designed in a way that it can be easily comprehended and answered by students with little or no programming experience, such as beginners and those who have not received coding instruction.

3.3. Construct Selection and Item Inclusion

Given the complexity of the factors that influence students' motivation, this study will rely on several key constructs to develop a scale for assessing students' motivation in learning how to code. Drawing on the guidance of (Situated) Expectancy Value Theory and prior scales in the field, we have identified seven key constructs to include in the scale. These constructs are coding confidence, academic self-efficacy and attitudes, coding interest, usefulness value, gender perception, and social perception. In the upcoming section, we explain the rationale for choosing these constructs, and present potential items for measuring them in Table 1.

The first construct included in the new scale is the students' perceived confidence or self-efficacy in learning how to code. According to SEVT, one of the two main determinants of motivation is expectancy of success, which is made up of two elements: ability beliefs and expectancy beliefs. While ability beliefs reflect the students' current confidence in their coding skills, expectancy beliefs represent their confidence in their future coding potential. Therefore, the construct of confidence should include items that capture both the current and future confidence in learning to code. Since our scale is intended for students who have little or no experience in coding, the items are not designed to target specific coding skills or knowledge. Moreover, these items must be tailored to the programming domain rather than computer science or other school subjects, as efficacy varies by context and subject (Eccles & Wigfield, 2020).

Self-concept and attitudes toward academics are assessed on the second construct. Academic self-concept and attitudes have been defined as a person's beliefs and attitudes toward their ability to achieve high academic performance in school (Eccles & Wigfield, 2020). While academic self-concept is close to the concept of self-efficacy, Eccles and Wigfield (2020) distinguished them conceptually in their framework. They argued that these constructs should be treated distinctly instead of combining them into one. They also suggest that an individual's previous and other academic self-concepts and attitudes can influence their motivation and performance in learning other subjects. Their views are supported by outcomes from empirical studies that have shown having a positive academic self-concept and attitudes can significantly improve a student's motivation to learn how to code (Leifheit et al., 2019; Mason & Rich, 2020). This is why we have developed a new scale including a list of items that specifically assess academic self-concept and attitudes.

The third construct used to assess the intrinsic value of programming is called coding interest. It refers to the extent to which a student is intrinsically interested in the subject (Eccles & Wigfield, 2020; Leifheit et al., 2019, Mason & Rich, 2020). This construct is one of four components that make up the subjective task value, which is a main determinant of a student's motivation, as suggested by SEVT. Most existing scales include this construct because both empirical and theoretical studies have shown that students with a high intrinsic value for a task are typically more engaged in it and capable of maintaining that engagement for extended periods (Eccles & Wigfield, 2020).

The next construct is usefulness value, which is used to assess both attainment value and utility value of participating coding courses. In the SEVT, Eccles and Wigfield (2020) identify utility value and attainment value as components of subjective task value. They suggest that utility value can take on various forms specific to the domain and activity, and it may overlap with attainment value in certain situations, such as pursuing a particular occupation. As a result, our scale includes the construct usefulness value to measure not only the positive effects of coding on academic performance but also its broader impact on areas such as job preparation.

The fifth construct, gender perception, is used to evaluate students' perceived gender stereotypes regarding coding. According to Eccles and Wigfield's (2020) theory, gender role perceptions can impact students' motivation and attitudes towards a subject. For instance, if coding is perceived as a masculine subject or career, or if the stereotype exists that boys are better than girls in coding, it may decrease girls' self-efficacy and expectations of success (Mason & Rich, 2020). Many studies also showed that male students have significantly higher interest in computing than female students (Mason & Rich, 2020; Sullivan & Bers, 2019; Witherspoon et al., 2016). Hence, we have included items in our scale that evaluate the gender perception of students towards coding ability and interest.

The final construct of this study, social influence, is included to evaluate the social perception of coding. As SEVT emphasizes, an individual's choice of activity and performance is not only determined by their expectancy and value but also by their social, cultural, and environmental context. In other words, the motivation of children towards programming can be influenced by various factors around them, such as their family, teachers, and friends (Eccles and Wigfield, 2020). If students perceive that their parents, teachers, and peers value coding highly, this perception may increase their attitudes towards and participation in coding (Mason & Rich, 2020).

After identifying the constructs constituting our scale, we move to the stage of item development to measure these constructs. The items in our scale were developed in two ways: incorporating items from prior scales and generating new items. Items incorporated into our scale were not solely derived from the scales reviewed above, but also from other relevant scales. Although these scales are not directly related to programming, they are still pertinent to coding and other school subjects, such as computer science and STEM. The scales we drew items from include the CPAS-M by Gul, Cetin and Ozden (2022), the E-CSA by Vandenberg et al. (2021), the CS survey by Hoegh and Moskal (2009), the STEM instrument by Mahoney (2010), the CS survey by Dorn and Ellitott Tew (2015), the CS scale by Forssen, Moskal and Harriger (2011), and the STEM-CIS by Kier et al. (2014).

We decided to use a 5-point Likert rating scale for the response format, ranging from 1 to 5 (where 5 = agree completely; 4 = agree; 3 = rather agree; 2 = disagree; 1 = do not agree at all). None of the constructs included any items with reversed wording. Since word choice can significantly affect how respondents understand and answer questions, we translated the selected items from existing scales into Vietnamese with careful consideration (Messick, 1995).

We then had four experts evaluate the translated items, including a language expert and three domain experts (a secondary school principal, an IT teacher from the same school, and a primary school literature teacher). The experts assessed the items for clarity, coherence, grammar, and appropriateness. Based on their suggestion, we rephrased three items before administering the scale to 10 students in grades 4 to 7 on a voluntary basis. We monitored their response time to ensure that the scale was not too time-consuming. Additionally, we asked them to evaluate the comprehensibility of the items and provide descriptions of the items' meanings. We found that they spent less than 12 minutes answering the scale and understood all the items fully. Hence, we developed the initial version of the Coding Motivation Scale for Middle School Students (CMS-M), which comprises 33 items (Table 1).

Table 1: Initial Subscales and Items

Subscales and Items	Item Code	Sources / Prior Scales
Coding Confidence	**CC**	
I think I have enough ability to learn how to code	CC1	MR, KO, KI
I think I am good at learning how to code	CC2	MR, KO, LH, RM
If programs written by me don't work, I can find my mistakes and fix them.	CC3	MR, KK, TS
I feel I can code to control computers or robots to do the tasks I want	CC4	MR
Academic Self-Concept and Attitudes	**AT**	
I am good at learning Math	AT1	MR
I am good at learning Physics	AT2	Self-developed
I am good at learning Technology	AT3	Self-developed
I am good at learning Literature	AT4	MR
I like to learn Math	AT5	MR
I like to learn Physics	AT6	Self-developed
I like to learn Technology	AT7	Self-developed
I like to learn Literature	AT8	MR
Coding Interest	**CI**	

Subscales and Items	Item Code	Sources / Prior Scales
I think coding is interesting	CI1	MR, KO, MA, FO, KI
I do not feel boring when I learn how to code	CI2	FO
I would like to learn more about coding	CI3	MR, MA, DO, FO
I like working with computers	CI4	Self-developed
I would like to choose a job that is related to coding or computer in the future	CI5	MA, FO, KI
Usefulness Value	**UV**	
Learning how to code is beneficial for my future jobs	UV1	MR, LH, RM, VA, FO, KI
Knowing how to code will help me to create useful computer applications for people	UV2	MR, VA
Learning how to code will make me better in math.	UV3	MR, VA
Learning how to code will make me better in science	UV4	MR, VA
Learning how to code will make me better in technology	UV5	ME, VA
Learning how to code will make me better in language arts	UV6	MR
Gender Perception	**GP**	
I think boys are better at learning to code than girls	GP1	MR, FO
I think girls are better at learning to code than boys	GP2	MR, FO
I think boys like to learn coding more than girls do	GP3	MR
I think girls like to learn coding more than girls do	GP4	MR
I think both boys and girls can do well in coding classes	GP5	MR

Subscales and Items	Item Code	Sources / Prior Scales
I think both boys and girls like to learn coding equally	GP6	FO
Social Perception	**SP**	
I believe people who know how to code are intelligent	SP1	MR
My friends believe kids who can code are smart	SP2	Self-developed
My parents think learning how to code is important.	SP3	MR
My teachers think learning how to code is important.	SP4	MR

Total number of items: 33; Total number of constructs: 6

Note: DO = Dorn & Elliott Tew (2015); FO = Forssen, Moskal & Harriger (2011); HO = Hoegh & Moskal (2009); KK = Kukul (2018); KI = Kier et al. (2014); KO = Kong (2018); LH = Leifheit (2020); MA = Mahoney (2010); MR = Mason & Rich (2020); RM = Rachmatullah et al. (2020); TS = Tsai et al. (2019); VA = Vandenberg et al. (2021).

After constructing this version, we conducted a pilot survey, administering it to 321 middle school students in Vietnam's Mekong Delta region. This sample size significantly surpasses Carpenter's (2018) suggested threshold of 100 participants for a pilot survey. The data collected from this survey was utilized to evaluate the scale's construct structure through EFA. Analysis results helped to create the second version of CMS-M by indicating which items should be modified, removed or added. Subsequently, we moved to the testing phase as outlined in section 2 to validate the second version of CMS-M. We subjected this new version to two separate samples to test the reliability, construct validity, and measurement invariance of the scale. In the next sections, we will provide further details about data collection and analysis results.

4. Data Collection

The data used in this study were collected through three surveys conducted in 2020 and 2021. The first survey was a pilot survey conducted in the Mekong Delta region of Vietnam and included students in grades 4, 6, and 7. The second survey was administered in Lam Dong province and consisted of students in grades 6 and 7. The third survey was undertaken in Ho Chi Minh City and involved students of the same age range as the students in the second survey. All three samples consist of completely different participants. The distribution of characteristics of respondents participating in three surveys are shown in Table 2 and Table 3.

Sample 1 comprised 334 middle-aged students from 6 schools, and the data collected from this sample was used for the pilot study. However, after removing incomplete and inconsistent responses, the analysis was based on the sample of 321 observations. This sample consisted of 198 females (61.68%) and 123 males (38.32%). Fourth graders constituted approximately 15% of the sample, while sixth and seventh graders made up about 42% and 43%, respectively.

Table 2: Summary of the Student Sample

		Pilot Survey		Second Survey		Third Survey	
		Freq. (N)	**Perc. (%)**	**Freq. (N)**	**Perc. (%)**	**Freq. (N)**	**Perc. (%)**
Total student:		**321**	**100%**	**989**	**100%**	**1165**	**100%**
Grade							
	- *Grade 4*	48	14.95%	0	0%	0	0%
	- *Grade 6*	134	41.74%	497	50.25%	511	43.86%
	- *Grade 7*	139	43.30%	492	49.75%	654	56.14%
Gender							
	- *Male*	123	38.32%	501	50.66%	521	44.72%
	- *Female*	198	61.68%	488	49.34%	644	55.28%

Note: Pilot survey in Mekong Delta region; Second survey in Lam Dong Province; Third survey in Ho Chi Minh City

Sample 2 was composed of 989 students from 8 middle schools, and the EFA was conducted on the data collected from this sample. The sample was nearly evenly split by gender, with 488 females (49.34%) and 501 males (50.66%). About half of the students (51%) were in the sixth grade, while the remaining 49% were in the eighth grade. It's worth noting that in sample 2, although 1038 students were interviewed, data from only 989 students was used for analysis due to incomplete answers, invalid responses, or interview rejection. The dropping rate is very low because only 49 students (4.7%) dropped.

Sample 3 included 1165 middle school students from 3 schools, comprising 644 females (55.28%) and 521 males (44.72%). The proportion of the sixth grade and seventh grade students in the sample was roughly 56% and 44%, respectively. Unlike the first and second surveys, the third survey was conducted using both online and offline modes. This means that the data collected for sample 3 includes responses gathered at schools and at home via electronic devices. The responses collected offline accounted for 35.7% of the total, while online responses constituted 64.3%. The collected data was then subjected to CFA, internal reliability analysis and measurement invariance analysis for further analysis and interpretation.

Table 3: Tabulation of Sample by Gender and Grade

By	Gender	Grade	No. of Observations (Collected)	No. of Observations (Analysis)	Percentage (Analysis column)
Survey 1			**334**	**321**	**100 %**
	Boy		**130**	**123**	**38.32%**
		4	25	21	6.54%
		6	50	49	15.26%
		7	55	53	16.51%
	Girl		**204**	**198**	**61.68%**
		4	28	27	8.41%
		6	88	85	26.48%
		7	88	86	26.79%

By	Gender	Grade	No. of Observations (Collected)	No. of Observations (Analysis)	Percentage (Analysis column)
Survey 2			**1038**	**989**	**100 %**
	Boy			**501**	**50.66%**
		6	276	255	25.78%
		7	249	246	24.87%
	Girl			**488**	**49.34%**
		6	265	242	24.47%
		7	248	246	24.87%
Survey 3			**1165**	**1165**	**100 %**
	Boy		**521**	**521**	**44.72%**
		6	230	230	19.74%
		7	291	291	24.98%
	Girl		**644**	**644**	**55.28**
		6	281	281	24.12%
		7	363	363	31.16%

Note: Survey 1 and Survey 2 were collected offline; Survey 3 was collected using both online and offline

Because data sets from sample 1 and sample 2 were used for EFA, it is important to consider the sample size requirements for factor analysis. Differing viewpoints exist regarding the minimum sample size needed for unbiased results. Some researchers suggest a minimum of 500 subjects, while others recommend having at least 10 subjects for each item (Comrey & Lee, 2013; Watkins, 2018). In the current study, samples 2 meet the recommended sample size requirements for conducting factor analysis. However, sample 1 falls slightly short of the desired sample size, with only 321 complete observations instead of the desired 330 observations for the 33 items or a minimum 500 subjects. Although this smaller sample size may limit the accuracy in revealing the expected constructs of the

CMS-M scale, it is important to note that the primary objective of the pilot study is to re-evaluate the items and factors rather than determining the scale's structure. Hence, the sample size of 321 observations may suffice for conducting an effective EFA in the pilot survey.

5. Analysis Strategy

5.1. Exploratory Factor Analysis:

To investigate the factor structure of the CMS-M scale, EFA was performed on the data collected from sample 1 and sample 2, using the command "factor" in Stata 14. Prior to conducting the analysis, Kaiser-Meyer-Olkin (KMO) and Bartlett's Test of Sphericity (BTS) were employed to assess the suitability of the data for factor analysis. Since the item responses were ordinal and not normally distributed, which violated the linearity and normality assumptions of EFA, Polychoric correlations were utilized instead of Pearson correlations. Polychoric correlations are preferable in such cases as they assume the existence of an unobservable, normally distributed continuous variable underlying each observed ordinal variable, and then estimate the Pearson correlation between these underlying hypothetical variables (Comrey & Lee, 2013; Howard, 2016; Watkins, 2018).

Regarding the estimation method, Watkins (2018) recommended the principal axis factoring method for sample sizes that are relatively small (≤300). Conversely, for larger sample sizes, the maximum likelihood approach was suggested. Following this recommendation, the principal axis method is employed for the EFA conducted on the pilot data, while the maximum likelihood approach was utilized for sample 2.

Once the factor estimation method was determined, the next crucial step involved deciding the number of retained factors. To achieve this, the parallel test was conducted as it is considered the most accurate empirical estimate of the number of factors to retain. However, since no method has been found to be correct in all situations, multiple criteria were used to determine the number of factors, including eigenvalues greater than 1, the Scree test, the total amount of common variance explained, and the conceptual interpretability of the factor structure (Comrey & Lee, 2013; Gul, Cetin & Ozden, 2022; Watkins, 2018).

For factor interpretation, Promax rotation was employed instead of Varimax since the psychological constructs of the scale cannot be assumed to be uncorrelated. When removing items from the scale, it is recommended to apply different cut-off points for item loadings based on the specific research field. As a rule of thumb, a minimum loading of 0.333 is considered satisfactory, indicating approximately 10% shared variance with the other items in the same factor. However, items with loadings of 0.5 and above are considered strong and more desirable (Costello & Osborne, 2005; Watkins, 2018).

5.2. Reliability Analysis:

To assess the internal consistency and coherence of the construct measurements, a reliability analysis was conducted using the "psych" package in R Programming. This analysis involved calculating Cronbach's alpha coefficients using the data from sample 33. Cronbach's alpha is a measure that ranges from 0 to 1, with higher scores indicating stronger internal consistency among the items within a scale (Boateng et al., 2018; Hinkin, 2005; Devellis, 2017). For this analysis, the "alpha" command was utilized in both Stata 14 and R Programming. It was applied to all scales and subscales within the CMS-M, allowing for a comprehensive assessment of their reliability.

5.3. Confirmatory Factor Analysis:

To evaluate the validity of the scales, a CFA was performed on the data from sample 3 using the "psych" and "lavaan" package in R Programming. The CFA was carried out using the SEM. In assessing the goodness of fit of the theoretical factor model, several model fit indices were utilized, including the Chi-square ($\chi 2$) test, Root Mean Square Error of Approximation (RMSEA), Standardized Root Mean Square Residual (SRMR), Comparative Fit Index (CFI) and Tucker Lewis Index (TLI). A better fit for the model is indicated by lower values of RMSEA and SRMR, higher values of CFI and TLI, and non-significant Chi-square tests. Therefore, the model is considered to have a good fit if the corresponding indices fall below or exceed their thresholds suggested by Hu and Bentler (1999), Brown (2006), and Harrington (2009).

Various estimation methods have been developed for CFA, with Weighted Least Squares - Mean and Variance Adjusted (WLSMV) being preferred for categorical or ordinal variables. Li (2016) demonstrated that both Robust Maximum Likelihood (MLR) and WLSMV can be utilized in CFA when variables are non-continuous and not normally distributed. However, WLSMV was found to have less bias compared to MLR. Thus, for this study, we employed the WLSMV estimator in the CFA analysis.

5.4. Measurement Invariance:

To evaluate measurement invariance across groups, stepwise multi-group CFAs were applied on the data of survey 3, using the "psych" package in R Programming. This analysis involved estimating and comparing increasingly constrained CFA models, including configural invariance, metric invariance, scalar invariance, and strict invariance. The plausibility of each model was assessed using tests such as the Chi-square difference test and fit indices. Typically, full measurement invariance is considered established when the strict measurement invariance model is supported. However, because subsequent levels of invariance build upon the previous one, all previous invariance models need to be established before examining the strict measurement invariance (Brown, 2006; Harrington, 2009; Van De Schoot et al., 2015).

Therefore, establishing configural invariance is the initial step before progressing to higher levels of invariance. To test configural invariance, the configural model was fitted to each group, allowing all factor loadings and item intercepts to vary freely (Brown, 2006; Harrington, 2009; Van De Schoot et al., 2015). If the model-fit indices indicated a good fit for the configural model, further analysis was conducted by comparing four nested invariance models. These comparisons involved progressively constraining parameters and examining the Chi-square difference test as well as changes in CFIs and TLIs (ΔCFI and ΔTLI).

The first comparison involved constraining the factor loadings to be equivalent across groups, testing metric invariance (configural vs metric). If the increase in $\chi 2$ was not statistically significant, and ΔCFI and ΔTLI were smaller than 0.01, it was concluded that the factor loadings were invariant across groups. The second comparison constrained the item intercepts to be equivalent across groups, testing scalar invariance (metric vs. scalar). Scalar invariance was supported if the conditions were satisfied. Finally, scalar invariance was compared to strict invariance to examine whether item residuals differed across groups. If all three conditions were met, it indicated that residual invariance was established and confirmed the measurement invariance of the scale (Brown, 2006; Harrington, 2009; Van De Schoot et al., 2015).

6. Results

6.1. The Pilot Survey

The pilot survey involved administering the scales in a one-to-one setting with 334 students. By utilizing one-to-one cognitive interviews, we ensured that students' feedback and comprehension were comprehensively captured, enabling a meticulous analysis of the phrasing of items. Furthermore, the responses to the items were subjected to EFA to explore the factor construct of the items. The combined analysis of students' comprehension and the outcomes of the EFA led to several adjustments being made to the CMS-M scales.

Despite the modest sample size of 321 observations used in the EFA, the results of the KMO measure indicated a value of 0.73, suggesting that the sample was adequate for conducting the analysis. Additionally, the Bartlett's sphere test produced a significant result (model Chi-square $\chi2(528) = 2876.78$, $p < 0.001$), further supporting the suitability of the sample for EFA (Howard, 2016; Watkins, 2018). It was concluded that EFA can be conducted. The principal axis factoring and Promax method with the cut-off point of 0.33 was used for the factor extraction and factor rotation. Because the parallel test suggests that the number of factors is at least 11, EFA was first conducted with 11 factors.

However, the model consisting of 11 factors does not adequately fit the data due to several issues. Firstly, many extracted factors contain only 1 or 2 items, falling short of the recommended criterion of at least 3 items per factor (Howard, 2016; Watkins, 2018). Moreover, numerous items exhibit cross-loadings, loading at 0.33 or higher on two different factors. Notably, several items load onto unexpected factors that lack support from established psychological frameworks. Therefore, alternative rules are employed to determine the appropriate number of factors.

The rule of retaining factors with eigenvalues greater than 1 suggests the presence of either 5 factors (the fifth eigenvalue = 1.171) or 6 factors (the sixth eigenvalue = 0.962, near to 1). The elbow rule based on the Scree plot indicates the potential for 6 or 7 factors. Subsequently, EFA was conducted using 3 models of 5, 6, and 7 factors, with the 6-factor model emerging as the most suitable choice. It exhibits no cross-loading items and aligns with existing psychological theory. The 6-factor model explains 79.88% of the total variance, and the absolute values of factor loadings range from 0.33 to 0.79 after rotation (refer to Table 4). The results of the EFA presented in Table 4 reveal several noteworthy findings.

Table 4: Factor Loadings for the Initial Items

Item Code	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6	Uniqueness
CC1	0.71						0.45
CC2	0.54						0.55
CC3	0.44						0.78
CC4	**0.12 ***	0.38					0.81
CI1	0.74						0.46
CI2	**0.38**			0.33			0.63
CI3	0.83						0.35
CI4	0.73						0.44
CI5	0.46						0.61
GP1						**0.68**	0.54
GP2					**-0.52**		0.55
GP3						**0.78**	0.39
GP4					**-0.55**		0.50
GP5					**0.66**		0.51
GP6					**0.71**		0.49
UV1		0.42					0.59
UV2		0.54					0.58
UV3		0.41					0.71

Item Code	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6	Uniqueness
UV4		0.52					0.73
UV5		0.46					0.76
UV6		0.65					0.66
SP1				**0.71**			0.46
SP2				**0.69**			0.55
SP3		**0.47**					0.76
SP4		**0.39**					0.73
AT1			**0.79**				0.40
AT2			**0.73**				0.46
AT3			**0.54**				0.65
AT4			**0.53**				0.61
AT5			**0.45**				0.70
AT6		**0.44**					0.68
AT7		**0.29 ***					0.74
AT8		**0.49**					0.72

** Loadings were provided for clarification, even though they were below 0.33.*
Values highlighted in bold indicate areas of concern
Note: Principal Axis Factoring method was employed; Promax rotation was applied; The cut-off point was 0.33.

Firstly, it is observed that both the “CC” (coding confidence) and “CI” (coding interest) items load onto a single factor, specifically factor 1 as shown in Table 4. This indicates that instead of two distinct latent factors representing “coding confidence” and “coding interest” as expected by the SEVT framework, there appears to be only one underlying factor. There are two possible explanations for this result. The first possibility is that Vietnamese students may exhibit higher interest in subjects where they have a high level of self-efficacy, so these constructs are grouped together. The second possibility is that the sample size of the pilot study might not be large enough to differentiate between these constructs. To address this, we have decided to treat “CC” and “CI” as distinguishable factors and plan to investigate their distinctiveness further in the second survey, which will involve a larger sample size.

In addition, we have made adjustments to the items associated with these constructs. Due to its loading below 0.32 and its similarity in meaning to item “CI1”, we have excluded “CI2” from the analysis. However, despite the “CC4” having relatively low loading of 0.12 on factor 1, we have opted to retain it. This decision is based on its loading of 0.32 on factor 2, indicating that it may be influenced by the sample size and could potentially be more appropriately assigned to the correct construct when a larger sample is considered. To compensate for the removed items, we have introduced 2 new items for the “coding confidence” construct and 2 new items for the “coding interest” construct (refer to Table 5).

Secondly, the EFA results reveal that the six items (“GC”) associated with the “gender perception” construct load on two different factors instead of the expected single factor. This discrepancy raises concerns regarding the inclusion of the “gender perception” construct or the adequacy of item wording. The first possibility can be dismissed as it deviates from the expectations of the SEVT theory and contradicts numerous empirical studies. Extensive research has consistently shown that gender stereotypes in STEM subjects emerge as early as middle school (Alonso et al., 2021; Charlesworth & Banaji, 2019; Gnambs, 2021; Montuori et al., 2022), or even earlier during preschool and elementary school years (Aesaert & van Braak, 2015; Kersey et al., 2018; Master et al., 2021).

The second possibility is quite plausible considering the results of cognitive interviews conducted with students. These interviews revealed that students exhibited hesitancy and reluctance when responding to the gender-related items, especially two items “GC5” and “GC6”. Although they comprehended the meaning of the items, they encountered challenges in maintaining consistency and rationality across them. However, it's necessary to note that the issues associated with this construct may be attributed to the limited sample size. Therefore, to further examine the existence of this construct, we propose replacing “GC5” and “GC6” with two new items (refer to Table 5) and assessing it with a larger sample size. These new items were derived from the study of Forssen, Moskal and Harriger (2011).

The construct "usefulness value" initially appeared promising as all the items were correctly assigned to a single factor. However, during cognitive interviews with students, it was revealed that some other aspects of usefulness were not addressed in the scale. For instance, students mentioned that coding not only helps in learning mathematics but also in other subjects. To address this, the item “AU5” was added to the scale. Furthermore, students emphasized that learning how to code has implications for their future careers and daily lives. Therefore, three additional items “GU2”, “GU4”, and “GU5” were included (Table 5), focusing on the impact of coding on students' future occupations, as suggested by Leifheit et al. (2020).

Given the SEVT theory's suggestion of potential overlap or distinction between utility value and attainment value, the construct “usefulness value”, which initially encompassed both values, was divided into two separate constructs. This decision was made after including the four new items. As a result, the original items and the newly added items were reassigned to two distinct constructs: “utility value” and “attainment value”. To gather evidence regarding the existence of these constructs, an EFA with a large sample size will be conducted. The results of this analysis will provide support for the presence of either one or two distinct constructs.

Finally, both the constructs “social perception” and “academic self-efficacy and attitudes” encountered issues as their items were assigned to two factors instead of one. However, feedback from cognitive interviews with students indicated that they had no difficulty in understanding and responding to the items. It means there is no reason to attribute these issues to the phrasing of the items. As a result, we have decided to retain these constructs as they are. We anticipate that a larger sample size will facilitate the proper assignment of items to their respective factors. Additionally, we have included one new item derived from Mason and Rich (2020) in the “social perception” construct. This addition increases the total number of items assessing “social perception” from 4 to 5, reducing the likelihood of the construct having fewer than 3 items after removing low-loading items in the subsequent EFA.

The remaining items from the initial scale, as well as the newly added items, were combined to form a new scale, as presented in Table 5. This revised scale had 41 items and was administered to two groups of students in separate surveys. The first group comprised 1038 students in grades 6 and 7, while the second group consisted of 1165 students of the same age. The responses provided by the students were then subjected to subsequent EFA, CFA, and other relevant tests to further evaluate the scale's psychometric properties.

Table 5: Subscales and Items after the Pilot Survey

Subscales and Items	Item Code in the Pilot	New Item Code	Note (Prior Scales)
Coding Confidence	**CC**	**CC**	
I think I have enough ability to learn how to code	CC1	CC1	
I think coding is an easy school subject		*CC2*	New item: derived from LH
I think I am good at learning how to code	CC2	CC3	
Everybody say that I am good at coding		*CC4*	New item: derived from MR
If programs written by me don't work, I can find my mistakes and fix them.	CC3	*CC5*	
I feel I can code to control computers or robots to do the tasks I want	CC4	CC6	
Academic Self-Concept and Attitudes	**AT**	**AA**	
I like to learn Math	AT5	AA1	
I like to learn Literature	AT8	AA2	
I like to learn Physics	AT6	AA3	
I like to learn Technology	AT7	AA4	
I am good at learning Math	AT1	AA5	

Subscales and Items	Item Code in the Pilot	New Item Code	Note (Prior Scales)
I am good at learning Literature	AT4	AA6	
I am good at learning Physics	AT2	AA7	
I am good at learning Technology	AT3	AA8	
Coding Interest	**CI**	**CI**	
I think coding is interesting	CI1	CI1	
I like learning how to code		*CI2*	New item: derived from MR, LH, KI
I like solving coding problems		*CI3*	New item: derived from MR, DO
I would like to learn more about coding	CI3	CI4	
I like working with computers	CI4	CI5	
I would like to choose a job that is related to coding or computer in the future	CI5	CI6	
General Usefulness	**UV**	**GU**	
Learning how to code is beneficial for my future jobs^^	UV1	GU1	
I think coding is important and necessary in my daily life		*GU2*	New item: derived from KO, LH
Knowing how to code will help me to create useful computer applications for people	UV2	GU3	
Learning programming helps me get a good preparation for careers in the field of computer and engineering.		*GU4*	New item: derived from FO, KI

Subscales and Items	Item Code in the Pilot	New Item Code	Note (Prior Scales)
Learning how to code provides many benefits in my life.		*GU5*	New item: derived from LH, DO
Academic Usefulness	**UV**	**AU**	
Learning how to code will make me better in math.	UV3	AU1	
Learning how to code will make me better in language arts	UV6	AU2	
Learning how to code will make me better in science	UV4	AU3	
Learning how to code will make me better in technology	UV5	AU4	
Learning how to code will make me better in other courses		AU5	New item: derived from MR
Gender Perception	**GP**	**GP**	
I think boys are better at learning to code than girls	GP1	GP1	
I think girls are better at learning to code than boys	GP2	GP2	
I think boys like to learn coding more than girls do	GP3	GP3	
I think girls like to learn coding more than girls do	GP4	GP4	
I think coding-related jobs are more suitable for boys than girls		*GP5*	New item: derived from FO

Subscales and Items	Item Code in the Pilot	New Item Code	Note (Prior Scales)
I think coding-related jobs are more suitable for girls than boys		*GP6*	New item: derived from FO
Social Perception	**SP**	**SP**	
I believe people who know how to code are intelligent	SP1	SP1	
My friends believe kids who can code are smart	SP2	SP2	
My parents think learning how to code is important.	SP3	SP3	
My teachers think learning how to code is important.	SP4	SP4	
People around me told me that coders are successful and talented		*SP5*	New item: self-developed

Total number of items: 41; Total number of constructs: 7

Italic items are newly added items.

Note: DO = Dorn & Elliott Tew (2015); FO = Forssen, Moskal & Harriger (2011); KI = Kier et al. (2014); KO = Kong (2018); LH = Leifheit (2020); MR = Mason & Rich (2020).

6.2. Exploratory Factor Analysis for The Second Survey

A total of 1038 middle school students participated in the second survey, which was conducted in a one-to-one offline setting. The students' responses were found to be suitable for performing EFA because it produced a very high KMO value of 0.89 and the significant result of BTS (model Chi-square $\chi2(820) = 23028.71$, $p < 0.000$). Considering the ordinal nature of the data, factor extraction was performed using Polychoric correlations and the maximum likelihood method. For factor rotation, the Promax rotation method with a cutoff point of 0.5 was chosen. Initially, EFA was conducted with 7 factors based on the results obtained from the preferred parallel test (Howard, 2016; Watkins, 2018).

The EFA result revealed a robust 7-factor model, as depicted in Table 6. Specifically, each factor was well represented by a minimum of three items with strong loadings (item loading ≥ 0.5). In addition, there were no concerns of cross-loading, and all retained items aligned with their respective factors as anticipated by the theoretical framework. However, it is noted that 4 items, namely “GP2”, “GP4”, “GP6” and “SP1”, were removed from the scale due to their low loadings. This elimination was reasonable considering that the three omitted "GP" items shared similarities with the retained ones, and the “SP1” (“I believe people who know how to code are intelligent”) did not appropriately align with the construct of “social perception”. Consequently, it was determined that only three items were sufficient to measure the constructs of “gender perception” and “social perception”.

Table 6: Factor Loadings for the Final Items

Item Code	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6	Factor 7	Uniqueness
CC1			0.79					0.21
CC2			0.51					0.77
CC3			0.62					0.40
CC4			0.86					0.28
CC5			0.72					0.58
CC6			0.79					0.35
AA1	0.81							0.36
AA2	0.59							0.62
AA3	0.83							0.34
AA4	0.85							0.34

Item Code	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6	Factor 7	Uniqueness
AA5	0.67							0.49
AA6	0.54							0.70
AA7	0.83							0.31
AA8	0.58							0.63
CI1		0.90						0.16
CI2		0.91						0.17
CI3		0.70						0.38
CI4		0.75						0.41
CI5		0.89						0.30
CI6		0.78						0.34
GU1				0.89				0.21
GU2				0.76				0.41
GU3				0.65				0.51
GU4				0.55				0.65
GU5				0.78				0.42
AU1					0.60			0.55
AU2					0.80			0.41
AU3					0.71			0.45

Item Code	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5	Factor 6	Factor 7	Uniqueness
AU4					0.72			0.44
AU5					0.63			0.60
GP1							0.74	0.48
GP2								**0.91**
GP3							0.76	0.44
GP4								**0.88**
GP5							0.63	0.61
GP6								**0.94**
SP1								**0.74**
SP2						0.66		0.55
SP3						0.80		0.28
SP4						0.70		0.44
SP5						0.62		0.56

Note: Values highlighted in bold indicate low loading items, which will be dropped.
Maximum Likelihood estimation was employed; Promax rotation was applied; The cut-off point was 0.50.

Alternative models with 6 and 8 factors were also considered during the analysis. However, upon examination of the extracted loadings, it became evident that these models did not yield satisfactory EFA solutions. In the 6-factor model, only the item "GU5" from the "general usefulness" construct had a sufficiently high loading to be retained. Conversely, in the 8-factor model, all six items assessing the "gender perception" construct were kept, but they loaded on two separate factors instead of a single, coherent one. In comparison, the model with 7 factors emerged as the most meaningful and acceptable solution. The 7-factor model explains 52.15% of the total variance, and the values of factor loadings range from 0.51 to 0.91 (refer to Table 6).

EFA results from sample 2 yielded a 7-dimensional attitude scale with 37 items. In the next survey, reliability analysis and CFA were used to examine internal consistency reliability and factor structure of the 37-item scale. Results of these analyses are presented in the next sections.

6.3. Reliability Analysis

The dataset for sample 3 consists of 1165 students from grades 6 and 7. To assess the internal consistency of each construct and the overall scale, the data underwent Cronbach alpha analysis. The results, presented in Table 7, indicate that the Cronbach alpha coefficients for the individual constructs ranged from 0.832 to 0.925, reflecting a level of reliability ranging from reliable to excellent (Taber, 2018).

Moreover, excluding any item from the constructs did not result in an increase in Cronbach Alpha, except for the item "CC2" within the "coding confidence" construct. By removing this item, the Cronbach alpha of the construct would increase from 0.832 to 0.869. This finding suggests that removing this item from the scale would enhance the overall reliability of the construct.

Table 7: Cronbach's Alpha for Subscales

Items	Cronbach's Alpha if the item is dropped	Cronbach's Alpha for the subscale
Coding Confidence		**0.832**
CC1	0.787	
CC2 ****	0.869	
CC3	0.785	
CC4	0.790	
CC5	0.793	
CC6	0.794	

Items	Cronbach's Alpha if the item is dropped	Cronbach's Alpha for the subscale
Coding Interest		**0.917**
CI1	0.903	
CI2	0.891	
CI3	0.900	
CI4	0.899	
CI5	0.908	
CI6	0.911	
Gender Perception		**0.867**
GP1	0.799	
GP3	0.812	
GP5	0.828	
Academic Self-Concept and Attitudes		**0.911**
AA1	0.905	
AA2	0.905	
AA3	0.898	
AA4	0.899	
AA5	0.901	
AA6	0.900	
AA7	0.896	

Items	Cronbach's Alpha if the item is dropped	Cronbach's Alpha for the subscale
AA8	0.897	
Social Perception		**0.834**
SP2	0.828	
SP3	0.783	
SP4	0.757	
SP5	0.788	
General Usefulness		**0.902**
GU1	0.880	
GU2	0.876	
GU3	0.875	
GU4	0.904	
GU5	0.868	
Academic Usefulness		**0.925**
AU1	0.910	
AU2	0.903	
AU3	0.898	
AU4	0.907	
AU5	0.921	

*Note: **** Item drop leads to an increase in Cronbach's Alpha of the subscale*

Table 8 presents a summary of the Cronbach alpha values for the entire scale when all items were included, as well as when each item was individually excluded. The overall scale demonstrated excellent reliability, with a Cronbach alpha coefficient of 0.948. When examining the exclusion of each item, the alpha values did not show significant changes, ranging between 0.946 and 0.948.

Table 8: Cronbach's Alpha for the Entire Scale

Items (1)	**Cronbach's Alpha if the item is dropped (2)**	**Difference between column (2) and 0.948**
Cronbach's Alpha for the entire scale: 0.948		
CC1	0.946	
CC2 *	0.949	-0.001212 [a]
CC3	0.946	
CC4	0.946	
CC5	0.946	
CC6	0.946	
CI1	0.946	
CI2	0.945	
CI3	0.945	
CI4	0.945	
CI5	0.946	
CI6	0.946	
GP1	0.948	-0.000413 [b]

Items (1)	Cronbach's Alpha if the item is dropped (2)	Difference between column (2) and 0.948
GP3	0.948	-0.000423 [b]
GP5	0.948	-0.000527 [b]
AA1	0.947	
AA2	0.946	
AA3	0.946	
AA4	0.946	
AA5	0.946	
AA6	0.946	
AA7	0.946	
AA8	0.946	
SP2	0.947	
SP3	0.946	
SP4	0.946	
SP5	0.946	
GU1	0.946	
GU2	0.945	
GU3	0.946	
GU4	0.947	

Items (1)	Cronbach's Alpha if the item is dropped (2)	Difference between column (2) and 0.948
GU5	0.945	
AU1	0.946	
AU2	0.946	
AU3	0.946	
AU4	0.946	
AU5	0.946	

Note: [a] *The difference is not 0.001 due to rounding of the alpha values*
[b] *The difference is not 0.000 due to rounding of the alpha values*

It is good to observe that dropping most items resulted in a decrease in the overall alpha value, except for the four items: "CC2", "GP1", "GP3", and "GP5". Therefore, it is recommended to exclude these four items from the scale. However, since the increase in overall alpha for dropping these items was minimal, they can still be retained, except for "CC2". Only the item "CC2" was eliminated because its removal contributed to an increase in the Cronbach alpha of the "coding confidence" construct. Furthermore, the meaning conveyed by the item "CC2" (I think coding is an easy school subject) is already encompassed by the item "CC1" (I think I have enough ability to learn how to code). Therefore, retaining only "CC1" is sufficient for capturing students' perceptions regarding this aspect of coding. The elimination of "CC2" decreased the number of items to 36 and increased the scale's overall alpha to 0.949.

6.4. Confirmatory Factor Analysis

CFA was performed to evaluate the construct validity of the 36-item scale, utilizing data gathered from 1165 middle school students in the third survey. In Table 9, the KMO score was found to be 0.81 and the BTS yielded $\chi 2$ (820) = 43332.62, p =.000, suggesting the data is suitable for factor analysis. After running CFA on the 7-factor model, the model Chi-square test showed that the model did not fit the data ($\chi2 = 1361.132$, Df = 573, $p < 0.000$). However, considering that the Chi-square test is sensitive to sample size, it is important to consider additional goodness-of-fit indices (Brown, 2006; Hu & Bentler, 1999; Kline, 2005).

Table 9: Confirmatory Factor Analysis

Measures	Actual Calculation	Benchmark	Conclusion
KMO	0.81	(0.80 – 0.90) good	Suitable for factor analysis
BTS	$\chi2 = 43332.62$ (Df = 820, $p < 0.000$)	p-value should be small	Suitable for factor analysis
Chi-square test	$\chi2 = 1361.132$ (Df = 573, $p < 0.000$)	p-value should be large	Not fit
$\chi2$ / Df	2.37	should be maller than 3.0	Good Fit
CFI	0.99	(0.95 – 1.00) desirable	Good Fit
TLI	0.99	(0.95 – 1.00) desirable	Good Fit
RMSEA	0.058	0.06	Good Fit
SRMR	0.034	0.08	Good Fit

Benchmark suggested by Brown (2006), Hu & Bentler (1999), Sun (2005), and Vandenberg et al. (2021)

Overall, the 7-factor model demonstrated a good fit based on widely used fit indices. Firstly, the Chi-square to degree of freedom ratio ($\chi2$ / Df) was 2.37 (=1361.132 / 573), falling below the threshold of 3.0 and indicating a good fit (Sun, 2005; Vandenberg et al., 2021). Secondly, both CFI and TLI were 0.99, significantly surpassing the desirable threshold of 0.95 and approaching a perfect fit (CFI and TLI values close to 1). Additionally, RMSEA yielded a value of 0.058, and the SRMR was 0.034. These values indicated a good fit because the RMSEA and SRMR of a good model should be below 0.06 and 0.08 respectively (Brown, 2006; Hu & Bentler, 1999; Sun, 2005). Furthermore, all estimates of item loadings were positive and significant, and the variances exhibited positive signs. These findings are desirable in CFA, as negative estimates of variances or loadings are considered unacceptable (Kline, 2005).

Therefore, the findings provide strong evidence supporting the construct validity of the CMS-M scale. In other words, the scale effectively assesses students' interest and attitude toward coding across its seven constructs: coding confidence, coding interest, gender perception, general usefulness, academic usefulness, academic self-efficacy and attitudes, and social perception.

6.5. Invariance Measurement

Stepwise multi-group CFAs were applied on the third data set to evaluate whether the factor structure of the scale performed consistently across groups, such as gender, grade levels, and survey administration settings. The initial step involved testing the existence of configural models for these groups. The fit indices presented in Table 10 demonstrated a good model fit for each group. This indicates that the overall factor structure of the scale was similar across all groups, establishing configural invariance. Subsequently, the validity of higher levels of invariance was assessed by examining the significance of Chi-square tests, as well as the magnitude of ΔCFI and ΔTLI.

The results in Table 10 indicate that full measurement invariance was achieved across gender and grade levels. Notably, the changes in model $\chi2$ were not statistically significant, and the ΔCFI and ΔTLI values were below the threshold of 0.01 across the invariance models. Additionally, all models demonstrated good model-fit indices. These findings highlight the consistency of underlying factor structure when comparing males and females, as well as sixth and seventh grade students. Therefore, it can be concluded that the scale successfully captured and represented the interests and attitudes towards coding among these diverse groups.

However, full measurement invariance across survey administration modes was not achieved in the analysis. When transitioning from the metric invariance to the scalar invariance model, the significant increase in $\chi2$ ($\Delta\chi2 = 76.30$, Df = 36, $p < .001$) indicated that imposing the constraint of equal intercepts led to a poorer model fit. This suggests that the factor structure of the scale differed between the modes of survey due to difference in scalars of the factor structure. However, negligible changes in the fit indices of TLI, CFI and RMSEA between the invariance models may be indicative of $\chi 2$ over-sensitivity to sample size, and thus the measurement invariance across survey modes could be established (Brown, 2006).

Table 10: Measurement Invariance across Groups

Group	Model	$\chi 2$	Df	RMSEA	SRMR	CFI	TLI	$\Delta\chi 2$
Gender (Males vs Females)	M1-Configural	1258.41	1146	0.022	0.035	0.96	0.96	
	M2-Metric	1290.94	1175	0.022	0.039	0.96	0.96	36.46
	M3-Scalar	1323.92	1204	0.022	0.039	0.96	0.96	38.51
	M4-Strict	1361.67	1240	0.022	0.040	0.96	0.96	42.59
Grade Level (Grade 6 vs Grade 7)	M1-Configural	1220.23	1146	0.018	0.034	0.97	0.97	
	M2-Metric	1262.16	1175	0.019	0.037	0.97	0.97	34.40
	M3-Scalar	1297.23	1204	0.020	0.037	0.97	0.97	39.80 *
	M4-Strict	1336.51	1240	0.020	0.038	0.97	0.97	42.76
Survey Distribution (Online vs Offline)	M1-Configural	1251.87	1146	0.021	0.036	0.96	0.96	
	M2-Metric	1303.23	1175	0.023	0.040	0.96	0.95	41.02
	M3-Scalar	1351.78	1204	0.025	0.041	0.95	0.95	76.3 ***
	M4-Strict	1388.21	1240	0.024	0.041	0.95	0.95	39.26

X2 = Model Chi-square; Df = degree of freedom; RMSEA = Root Mean Square Error of Approximation; SRMR = Standardized Root Mean Square Residual; CFI = Comparative Fit Index; TLI=Tucker–Lewis Index; $\Delta\chi 2$ = the chi square difference test

** = significant, $p<0.01$; *** = significant, $p<0.001$; All statistics $\chi 2$ are significant, $p<0.001$*

$\Delta\chi 2$ (when using WLSMV) is not simply the difference between 2 observed Chi-square statistics, but it is calculable

7. Discussion

The objective of this study was to create a coding motivation scale that is both valid and reliable for middle school students. The construction and validation process consisted primarily of two phases. In the first phase, the initial version of the Coding Motivation CMS-M was developed under the SEVT framework, comprising 33 items and six constructs. This phase involved several key steps, including concept determination, review of existing scales, selection and refinement of appropriate items, expert evaluation, and field testing, as well as the administration of a pilot survey and EFA. Upon completion of the first phase, the results indicated the need to eliminate certain items and introduce new ones. As a result, a revised version of the CMS-M was created, consisting of 41 items.

In the second phase, the new CMS-M was administered to two separate samples of students from middle schools. Once data was collected, a series of analyses were conducted to refine and validate the scale. Firstly, EFA was performed on the dataset of one sample, resulting in the identification of 37 items representing seven distinct dimensions: coding confidence, coding interest, gender perception, general usefulness, academic usefulness, academic self-efficacy and attitudes, and social perception.

To ensure the reliability and validity of these constructs, the second dataset was next utilized for reliability analysis, CFA, and measurement invariance analysis. Cronbach's alpha coefficient was initially calculated for each construct to assess their internal consistency. The results indicated that all constructs exhibited strong internal consistency, as all alpha values surpassed the threshold of 0.8. Next, the reliability analysis was conducted for the entire scale. During this analysis, it was determined that one item should be removed from the scale. As a result, the final version of the CMS-M consisted of 36 items. Importantly, the scale demonstrated excellent internal consistency, with a remarkably high alpha value of 0.948.

Furthermore, the results of CFA strongly support the construct validity of the 36-item CMS-M. All model-fit indices exceeded the thresholds recommended by experts, indicating a good fit between the hypothesized model and the collected data. Notably, the factor structure of the CMS-M demonstrated invariance across different gender and grade levels. This is a crucial characteristic of the CMS-M as it ensures that the same underlying construct is being measured consistently among both male and female students, as well as across 6th-grade and 7th-grade students (Brown, 2006;

Harrington, 2009). Therefore, researchers can rely on the CMS-M to yield reliable and comparable results, enabling a comprehensive understanding of coding motivation across different gender and grade groups.

However, the analysis of measurement invariance revealed that the CMS-M did not demonstrate complete invariance across different survey modes. It only achieved metric invariance, which implies that the relationships between the items and the underlying construct were equivalent across survey modes. This conclusion was derived from the significant Chi-square difference test observed when transitioning from metric invariance to scalar invariance.

As Chi-square and its associated indices are sensitive to sample size, it is recommended to consider other model-fit indices such as changes in TLI, CFI and RMSEA (Brown, 2006). Following this recommendation, the differences between the two invariance models across survey modes, as indicated by the statistically significant change in the Chi-square value, were trivial and not of substantive importance. Based on this approach, the scale's ability to establish measurement invariance across survey modes could be reasonably supported because changes in TLI, CFI and RMSEA are nearly negligible (refere to Table 10). However, this conclusion should be used with caution, as the validity of this approach is still under examination (Brown, 2006).

8. Limitations and Future Research

While the findings regarding the reliability and construct validity of the CMS-M are promising, it is important to acknowledge certain limitations that should be addressed in future research. The primary limitation pertains to the fact that the reliability and validity tests of the scale were conducted based on the Vietnamese version. Consequently, it is crucial to re-evaluate the English version of the CMS-M through a validation study involving native English-speaking middle school students. This step will ensure the cross-cultural applicability and generalizability of the scale.

Furthermore, all analyses in the current study handled missing values using the listwise deletion method. However, this approach can lead to information loss or biased estimates if the missing data is not missing completely at random (MCAR) (Nassiri et al., 2018). Although the missing rate in the dataset was approximately 5%, it is recommended to employ more advanced techniques, such as imputation or full information maximum likelihood, to handle incomplete data. This will allow for a more robust examination of the estimates and enhance the accuracy of the results.

Another limitation to consider is the length of the CMS-M. The scale, designed in line with the psychological framework, comprises multiple constructs and a relatively large number of items. While this design choice ensures comprehensive coverage and a solid foundation, it also leads to a fairly lengthy instrument. Consequently, participants may face logistical time pressures and respondent fatigue during its completion (Maloney et al., 2011). To address this concern, conducting a subsequent study aimed at validating a shorter version of the CMS-M would be advantageous. This streamlined version would retain the scale's psychometric properties while alleviating potential burdens on respondents, striking a better balance between comprehensiveness and participant convenience.

Lastly, it is worth noting that the CMS-M does not currently include the cost component theorized by SEVT. This omission of perceived costs in the CMS-M is a result of its complex and multidimensional nature. However, the potential inclusion of this construct could provide a more comprehensive understanding of students' motivation toward coding. Therefore, a careful and extensive review regarding the cost component will be conducted in the next study, given the current lack of consensus and complete research regarding the cost items.

9. Conclusion

Based on the model-fit indices and results of validation tests, the present study concluded that the 36-item CMS-M has appropriate and promising psychometric properties. This implies adequate reliability and a solid internal structure of seven latent factors, namely coding confidence, coding interest, gender perception, general usefulness, academic usefulness, academic self-efficacy and attitudes, and social perception. With the emergence of 4.0 technologies and involvement of programming in future works, the current study contributes a much needed psychometrically reliable and valid measure for evaluating programming courses incorporated in middle school curriculum. Most importantly, the measurement invariance property of the CMS-M in different gender and age groups has practical implications for evaluation study in Vietnam.

References

Aesaert, K., & van Braak, J. (2015). Gender and socioeconomic related differences in performance-based ICT competences. *Comput. Educ., 84*, 8-25. doi: 10.1016/j.compedu.2014.12.017

Alonso, M. T., Barba-Sánchez, V., López Bonal, M. T., & Macià, H. (2021). Two perspectives on the gender gap in computer engineering: from secondary school to higher education. *Sustainability, 13*, 10445. doi: 10.3390/su131810445

Asian Development Bank (2021). *Reaping the benefits of industry 4.0 through skills development in high-growth industries in Southeast Asia: insights from Cambodia, Indonesia, the Philippines, and Viet Nam*. Asian Development Bank, Manila, Philippines.

Atkinson, J. W. (1957). Motivational determinants of risk-taking behavior. *Psychol. Rev., 64*, 359-372. doi: 10.1037/h0043445

Berhenke, A., Miller, A. L., Brown, E., Seifer, R., & Dickstein, S. (2011). Observed emotional and behavioral indicators of motivation predict school readiness in Head Start graduates. *Early Childhood Research Quarterly, 26*(4), 430–441. doi:10.1016/j.ecresq.2011.04.001

Boateng, G. O., Neilands, T. B., Frongillo, E. A., Melgar-Quiñonez, H. R., & Young S. L. (2018). Best Practices for Developing and Validating Scales for Health, Social, and Behavioral Research: A Primer. *Front Public Health, 11*, 6-149. doi: 10.3389/fpubh.2018.00149. PMID: 29942800; PMCID: PMC6004510.

Brophy, J. E. (2010). Motivating students to learn. *New York, NY: Routledge*. doi:10.1111/j.1467-8535.2010.01135_2_1.x

Brown, T. A. (2006). Confirmatory factor analysis for applied research. *New York: The Guilford Press.*

Cammaert, R., Nguyen, T. P. L., & Tanaka, S. (2020). Kindergarten to Grade 12 Reforms in Viet Nam. In B. Panth, R. Maclean (Ed.) *Anticipating and Preparing for Emerging Skills and Jobs. Education in the Asia-Pacific Region: Issues, Concerns and Prospects,* vol 55. Springer, Singapore. https://doi.org/10.1007/978-981-15-7018-6_12

Carpenter, S. (2018). Ten steps in scale development and reporting: A guide for researchers. *Communication methods and measures*, *12*(1), 25-44.

Chang, J. H., Rynhart, G., & Huynh, P. (2016). ASEAN in transformation how technology is changing jobs and enterprises. *ILO Working Papers* 994909343402676. International Labour Organization. https://www.ilo.org/actemp/publications/WCMS_579553/lang--en/index.htm

Charlesworth, T. E. S., & Banaji, M. R. (2019). Gender in science, technology, engineering, and mathematics: issues, causes, solutions. *J. Neurosci, 39*, 7228-7243. doi: 10.1523/JNEUROSCI.0475-18.2019

Comrey, A. L., & Lee, H. B. (2013). *A first course in factor analysis*. Lawrence Erlbaum, Hillsdale, NJ.

Cook, D. A., & Artino, A. R., Jr (2016). Motivation to learn: an overview of contemporary theories. *Med Educ, 50*, 997-1014. https://doi.org/10.1111/medu.13074

Costello, A. B., & Osborne, J. (2005). Best practices in exploratory factor analysis: Four recommendations for getting the most from your analysis. *Practical assessment, research, and evaluation*, *10*(1), 7.

DeVellis, R. F., & Thorpe, C. T. (2021). *Scale development: Theory and applications*. Sage publications.

Di Serio, Á., Ibáñez, M. B., & Kloos, C. D. (2013). Impact of an augmented reality system on students' motivation for a visual art course. *Computers & Education, 68*, 586-596. doi:10.1016/j.compedu.2012.03.002

Dorn, B., & Elliott Tew, A. (2015). Empirical validation and application of the computing attitudes survey. *Computer Science Education, 25*(1), 1-36. https://doi.org/10.1080/08993408.2015.1014142

Durak, H. Y., & Saritepeci, M. (2018). Analysis of the relation between computational thinking skills and various variables with the structural equation model. *Computers & Education, 116*, 191-202. https://doi.org/10.1016/j.compedu.2017.09.004.

Eccles, J. S., & Wigfield, A. (2020). From expectancy-value theory to situated expectancy-value theory: A developmental, social cognitive, and sociocultural perspective on motivation. *Contemporary Educational Psychology, 61*, 101859.

Eccles, J. S., Adler, T. F., Futterman, R., Goff, S. B., Kaczala, C. M., Meece, J. L., & Midgley, C. (1983). Expectancies, Values, and Academic Behaviors. In J. T. Spence (Ed.), *Achievement and Achievement Motivation* (pp. 75-146). San Francisco, CA: W. H. Freeman.

Forssen, A., Moskal, B. M., & Harriger, A. (2011). Measuring the Impact of a High School Intervention on Students' Attitudes in Information Technology: Validation and Use of an Attitude Survey.

Furr, M. (2011). Scale construction and psychometrics for social and personality psychology. *Scale Construction and Psychometrics for Social and Personality Psychology*, 1-160.

Gnambs, T. (2021). The development of gender differences in information and communication technology (ICT) literacy in middle adolescence. *Comput. Human Behav, 114*, 106533. doi: 10.1016/j.chb.2020.106533

Graham, S., Weiner, B. (2012). Motivation: past, present, and future. In K. R. Harris, S. Graham, T. Urdan (Ed.), *APA Educational Psychology Handbook,* Vol 1: Theories, Constructs, and Critical Issues. Washington, DC: American Psychological Association 2012;367-97.

Gul, D., Cetin, I., & Ozden, M. Y. (2022). A scale for measuring middle school students' attitudes toward programming. *Comput Appl Eng Educ., 30*, 251-258. https://doi.org/10.1002/cae.22454

Gul, D., Cetin, I., & Ozden, M. Y. (2022). A scale for measuring middle school students' attitudes toward programming, *Comput Appl Eng Educ., 30*, 251-258. https://doi.org/10.1002/cae.22454

Harrington, D. (2009). Confirmatory Factor Analysis, *Oxford University Press: Oxford*. UK.

Hinkin, T. R. (2005). Scale Development Principles and Practices. In R. A. Swanson & E. F. Holton III (Ed.), *Research in Organizations: Foundations and Methods of Inquiry* (pp. 161-179). Berrett-Koehler Publishers.

Hoegh, A., & Moskal, B. M. (2009, October). Examining science and engineering students' attitudes toward computer science. In *2009 39th IEEE Frontiers in Education Conference* (pp. 1-6). IEEE.

Howard, M. C. (2016). A review of exploratory factor analysis decisions and overview of current practices: What we are doing and how can we improve? *International journal of human-computer interaction*, *32*(1), 51-62.

Hu, L. T., & Bentler, P. M. (1999). Cutoff criteria for fit indexes in covariance structure analysis: Conventional criteria versus new alternatives. *Structural equation modeling: a multidisciplinary journal*, *6*(1), 1-55.

Kersey, A. J., Braham, E. J., Csumitta, K. D., Libertus, M. E., & Cantlon, J. F. (2018). No intrinsic gender differences in children's earliest numerical abilities. *Npj Sci. Learn., 3*, 1-10. doi: 10.1038/s41539-018-0028-7

Kier, M. W., Blanchard, M. R., Osborne, J. W., & Albert, J. L. (2014). The development of the STEM career interest survey (STEM-CIS). *Research in Science Education*, *44*, 461-481.

Kline, R. B. (2005). *Principles and practice of structural equation modeling* (2nd ed.). Guilford Press.

Kong, S. C., Chiu, M. M., & Lai, M. (2018) A study of primary school students' interest, collaboration attitude, and programming empowerment in computational thinking education. *Computers & Education.* doi: 10.1016/j.compedu.2018.08.026.

Korkmaz, Ö., Çakir, R., & Özden, M. Y. (2017). A validity and reliability study of the computational thinking scales (CTS). *Computers in Human Behavior*. https://doi.org/10.1016/j.chb.2017.01.005.

Kosovich, J. J., Hulleman, C. S., Barron, K. E., & Getty, S. (2015). A Practical Measure of Student Motivation: Establishing Validity Evidence for the Expectancy-Value-Cost Scale in Middle School. *The Journal of Early Adolescence, 35*(5-6), 790-816. https://doi.org/10.1177/0272431614556890

Kukul, V., Gökçearslan, S., & Günbatar, M. S. (2017). Computer programming self-efficacy scale (CPSES) for secondary school students: Development, validation and reliability. *Eğitim Teknolojisi Kuram ve Uygulama, 7*(1). Retrieved from http://dergipark.ulakbim.gov.tr/etku/article/view/5000195912.

Kyriazos, T. A., & Stalikas, A. (2018). Applied Psychometrics: The Steps of Scale Development and Standardization Process. *Psychology, 9*, 2531-2560. https://doi.org/10.4236/psych.2018.911145

Leifheit, L., Tsarava, K., Moeller, K., Ostermann, K., Golle, J., Trautwein, U., & Ninaus, M. (2019, October). Development of a questionnaire on self-concept, motivational beliefs, and attitude towards programming. In *Proceedings of the 14th Workshop in Primary and Secondary Computing Education* (pp. 1-9).

Leifheit, L., Tsarava, K., Ninaus, M., Ostermann, K., Golle, J., Trautwein, U., & Moeller, K. (2020, June). SCAPA: Development of a Questionnaire Assessing Self-Concept and Attitudes Toward

Programming. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education* (pp. 138-144).

Lewin, K. (1935). *A dynamic theory of personality: Selected papers* (D. E. Adams & K. E. Zener, Trans). New York, NY: McGraw Hill.

Lewin, K. (1942). Field theory and learning. In N. B. Henry (Ed.), *The forty-first yearbook of the National Society for the Study of Education: Part 2, The psychology of learning* (pp. 215–242). The University of Chicago Press.

Li, C. H. (2016). Confirmatory factor analysis with ordinal data: Comparing robust maximum likelihood and diagonally weighted least squares. *Behavior research methods, 48*(3), 936-949. https://doi.org/10.3758/s13428-015-0619-7

Mahoney, M. P. (2010). Students' Attitudes toward STEM: Development of an Instrument for High School STEM-Based Programs. *Journal of Technology Studies*, *36*(1), 24-34.

Maloney, P., Grawitch, M. J., & Barber, L. K. (2011). Strategic item selection to reduce survey length: reduction in validity? *Consult*. *Psychol. J. Pract. Res., 63*(3), 162-175. https://doi.org/10.1037/a0025604

Mason, S. L., & Rich, P. J. (2020). Development and analysis of the Elementary Student Coding Attitudes Survey. *Comput. Educ., 153*, 103898.

Master, A., Meltzoff, A. N., & Cheryan, S. (2021). Gender stereotypes about interests start early and cause gender disparities in computer science and engineering. *Proc. Nat. Acad. Sci., 118*, e2100030118. doi: 10.1073/pnas.2100030118

Messick, S. (1995). Validity of psychological assessment: validation of inferences from persons' responses and performances as scientific inquiry into score meaning. *Am. Psychol., 50*(9), 741-749.

Montuori, C., Ronconi, L., Vardanega, T., & Arfé, B. (2022). Exploring gender differences in coding at the beginning of primary school. *Frontiers in Psychology*, *13*, 887280.

Nassiri, V., Lovik, A., Molenberghs, G., & Verbeke, G. (2018). On using multiple imputation for exploratory factor analysis of incomplete data. *Behavior research methods*, *50*(2), 501-517.

Putnick, D. L., & Bornstein, M. H. (2016). Measurement Invariance Conventions and Reporting: The State of the Art and Future Directions for Psychological Research. *Developmental review : DR, 41*, 71-90. https://doi.org/10.1016/j.dr.2016.06.004

Rachmatullah, A., Wiebe, E., Boulden, D., Mott, B., Boyer, K., & Lester, J. (2020). Development and validation of the Computer Science Attitudes Scale for middle school students (MG-CS attitudes). *Computers in Human Behavior Reports, 2*, 100018. doi:10.1016/j.chbr.2020.100018

Schunk, D. H., Meece, J, L., & Pintrich, P. R. (2014). *Motivation in Education: Theory, Research, and Applications* (4th ed.). Upper Saddle River, NJ: Pearson.

Schwab, K. (2017). *The fourth industrial revolution*. Currency.

Schwab, K., & Zahidi, S. (2020). The future of jobs report 2020. *World Economic Forum*. Retrieved from https://www3.weforum.org/docs/WEF_Future_of_Jobs_2020.pdf

Sullivan, A., & Bers, M. U. (2019). VEX robotics competitions: Gender differences in student attitudes and experiences. *Journal of Information Technology Education, 18*, 97-112. https://doi.org/10.28945/4193

Sun, J. (2005). Assessing goodness of fit in confirmatory factor analysis. *Measurement and Evaluation in Counseling and Development, 37*(4), 240-256. doi:10.1080/07481756.2005.11909764

Taber, K. S. (2018). The Use of Cronbach's Alpha When Developing and Reporting Research Instruments in Science Education. *Research in Science Education 48, 6*, 1273-1296. https://doi.org/10.1007/s11165-016-9602-2

Trochim, W. M. (2006). The research methods knowledge base.(2nd eds.). *Cincinnati: Atomic Dog Publishing*.

Tsai, M. J., Wang, C. Y., & Hsu, P. F. (2018). Developing the Computer Programming Self-Efficacy Scale for Computer Literacy Education. *Journal of Educational Computing Research,* 073563311774674. doi:10.1177/0735633117746747

United Nations Development Programme – UNDP (2018). *Development 4.0: Opportunities and Challenges for Accelerating Progress towards the Sustainable Development Goals in Asia and the Pacific*. Sustainable Development Series, UN Development Programme, New York, USA.

United Nations Industrial Development Organization - UNIDO (2018). *Panel discussion at 17th UNIDO General Conference: Industry 4.0 - The opportunities behind the challenge*

United Nations Industrial Development Organization - UNIDO (2019*). Report from the 1st Regional Conference on Industrial Development: Unlocking the Potential of Industry 4.0 for Developing Countries – Asia Pacific. Vienna.*

Van De Schoot, R., Schmidt, P., De Beuckelaer, A., Lek, K., & Zondervan-Zwijnenburg, M. (2015). Editorial: Measurement Invariance. *Frontiers in psychology, 6*, 1064. https://doi.org/10.3389/fpsyg.2015.01064

Vandenberg, J., Rachmatullah, A., Lynch, C., Boyer, K. E., & Wiebe, E. (2021). Interaction effects of race and gender in elementary CS attitudes: A validation and cross-sectional study. *International Journal of Child-Computer Interaction*, *29*, 100293.

Vu, H. A., Nguyen, T., Nguyen, A., & Truong, N. (2023). *Final Report: Mid-Term Evaluation for the Digital Literacy Programme of the DARIU Foundation in Vietnam*. Research Report.

Watkins, M. W. (2018). Exploratory Factor Analysis: A Guide to Best Practice. *Journal of Black Psychology, 44*(3), 219-246. https://doi.org/10.1177/0095798418771807

Wigfield, A. (1994). Expectancy-value theory of achievement motivation: A developmental perspective. *Educational Psychology Review, 6*, 49-78

Wigfield, A., & Eccles, J. S. (2000). Expectancy–value theory of achievement motivation. *Contemporary educational psychology*, *25*(1), 68-81.

Witherspoon, E. B., Schunn, C. D., Higashi, R. M., & Baehr, E. C. (2016). Gender, interest, and prior experience shape opportunities to learn programming in robotics competitions. *International Journal of STEM Education, 3*(1), 18.

Concluding Remarks

This thesis comprises three essays that represent pioneering efforts to explore the impact of introducing programming education to students in Vietnam. In the first essay, we observed a positive effect on students' engagement in online self-learning as a result of learning how to code. This effect, notably, cannot be solely attributed to changes in students' motivation or increased access to online resources; rather, it is intricately connected to the cultivation of a more profound and resilient knowledge foundation. In the second essay, our research revealed that coding training led to an increase in students' computational thinking scores. This effect was more pronounced among Grade 7 students learning MakeCode, in comparison to Grade 6 students learning Scratch. This discrepancy can be attributed to MakeCode for Micro:bit's emphasis on robotics, sensors, and wearables, which facilitates a more robust connection between the digital and physical realms, as opposed to Scratch. In the final essay, we developed and validated the Coding Motivation Scale for Middle School Students (CMS-M). This scale comprises 36 items organized into 7 latent constructs and can effectively gauge students' motivation, attitudes, and interests towards coding in the Vietnamese context. This scale demonstrates satisfactory reliability, a robust internal structure, and measurement invariance.

Building upon the positive impact of programming education on students' engagement, it is crucial to consider the policy implications of our findings. As Vietnam continues its efforts to modernize and adapt to the digital age, the integration of coding into the education system should be more emphasized. We recommend that educational policymakers allocate resources and support for the development of coding curriculum and teacher training. Within the coding curriculum, the emphasis should extend beyond technical skills to foster a comprehensive knowledge foundation that transcends mere coding. Prioritizing teacher training is essential, as many educators may lack exposure to coding education during their previous bachelor of education studies, leaving them inadequately prepared in this domain. In addition, because teachers can adapt to changing needs and guarantee long-term support for students, training the teachers can also be cost-effective, reaching more students over a longer period. Therefore, the "train-the-teachers" approach is considered more effective, efficient, scalable, and sustainable than "train-the-students" approach.

Furthermore, to sustain this level of engagement, schools should actively seek partnerships with technology companies and coding education organizations, particularly NGOs in Vietnam dedicated to empowering young and disadvantaged individuals in the digital age. These partnerships can provide students with real-world exposure, mentorship, and opportunities to apply their coding skills. The government should incentivize and facilitate such collaborations, ensuring that students have access to practical experiences. These practical experiences not only make students more curious and interested in coding, but also align with the demands of the evolving job market. Policymakers should also consider offering a variety of coding languages and online materials to both students and teachers, allowing them to choose the one that aligns best with their interests, goals, learning styles and time allocation. Moreover, our research indicates the importance of introducing coding education early in the curriculum. We propose that coding and computational thinking skills be integrated into primary education. This will not only improve computational thinking skills but also lay the groundwork for more advanced learning in computer science and related fields.

Finally, as our development of the Scale CMS-M provides a powerful tool for assessing and understanding students' motivation, attitudes, and interests in coding, we suggest that the government and educational institutions regularly implement CMS-M assessments to evaluate the effectiveness of coding education programs. This regular evaluation will enable policymakers and educators to assess the impact of their initiatives, identify areas of improvement, and make data-driven decisions that foster the growth of coding education. Moreover, the data obtained from these assessments can inform the development of tailored interventions to enhance the quality of coding education, ensuring that it matches to the ever-evolving needs and aspirations of students.

In summary, our research has illuminated a promising pathway towards a more robust and impactful coding education system in Vietnam. It is crucial for educational policymakers and institutions to heed these recommendations and proactively implement measures to unleash the full potential of coding education. This has the potential to shape a brighter, technology-driven future for both our students and the nation as a whole. However, it's important to note that the findings presented in this thesis are derived from a small and imbalanced sample, employing a relatively simple impact estimation technique. Hence, to further strengthen our understanding and draw more comprehensive insights, future studies on coding education should be prioritized. Expanding the sample size and utilizing advanced causal evaluation designs, such as randomization, can significantly enhance our understanding of the causal impact of coding teaching on students' outcomes. These improvements can lead to more robust insights and recommendations for the advancement of coding education in Vietnam.

Appendix A: Ethical Approval for Research

MINISTRY OF EDUCATION AND TRAINING
UNIVERSITY OF ECONOMICS HCMC

SOCIALIST REPUBLIC OF VIETNAM
Independence - Freedom - Happiness

No: 1660 /QĐ-ĐHKTQLKH

Ho Chi Minh City, July 02nd, 2020

DECISION
On Ethical Approval for Research

THE PRESIDENT OF UNIVERSITY OF ECONOMICS HCMC

Pursuant to the Education Law dated on June 18th, 2012 and the Amended Law, adding some articles to the Education Law dated on November 19th, 2018.

Pursuant to the Decree No 99/2019/NĐ-CP dated on December 30th, 2019 of the Government on the detailed regulation and implementation of the Amended Law, adding some articles to the Education Law.

Pursuant to the Regulation No 40/QC-DHKT-HDT dated on December 30th, 2019 of the University Council on the organization and operation of the University of Economics Ho Chi Minh City.

Considering the Conclusion of the Ethics Committee pursuant to the Decision No 709/QĐ-ĐHKT-QLKHHTQT on March 09th, 2020.

On the recommendation of the Head of Department of Research Administration and International Relations,

DECIDES:

Article 1. The Ethics Committee has examined the proposal titled **"Effects of a Coding Program on Attitudes and Learning Performance of Pupils – Evidence from a Programming Project in Vietnam"** proposed by Dr. Nguyen Hoang Bao and found the information contained therein follows the current ethical requirements, and the required forms have been filled out properly.

Article 2. The Committee recommends that the project may be continued for the duration mentioned in the proposal.

Article 3. Head of Department of Research Administration and International Relations and all members of the project shall be obliged to comply with this decision in legal proceedings and accordance with the law from the day of signature../-

Received:
- As the Article 3;
- File: VT, QLKH.

PP. PRESIDENT
VICE PRESIDENT

(signed and stamped)

Prof. Nguyen Trong Hoai