Title	Conceptual educational frameworks for improving programming learning efficiency
Sub Title	
Author	葛, 佳慧(Ge, Jiahui) 岸, 博幸(Kishi, Hiroyuki)
Publisher	慶應義塾大学大学院メディアデザイン研究科
Publication year	2020
Jtitle	
JaLC DOI	
Abstract	
Notes	修士学位論文. 2020年度メディアデザイン学 第802号
Genre	Thesis or Dissertation
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO40001001-00002020-0802

Master's Thesis
Academic Year 2020

Conceptual Educational Frameworks for Improving Programming Learning Efficiency

Keio University
Graduate School of Media Design

Jiahui Ge

A Master's Thesis
submitted to Keio University Graduate School of Media Design
in partial fulfillment of the requirements for the degree of
Master of Media Design

Jiahui Ge

Master's Thesis Advisory Committee:

Hiroyuki Kishi	(Main Research Supervisor)
Akira Kato	(Sub Research Supervisor)

Master's Thesis Review Committee:

Hiroyuki Kishi	(Chair)
Akira Kato	(Co-Reviewer)
Kai Kunze	(Co-Reviewer)

Abstract of Master's Thesis of Academic Year 2020

Conceptual Educational Frameworks for Improving Programming Learning Efficiency

Category: Social Science / Humanities

Summary

This thesis investigates the current trend of global programming learning. After an extensive literature review discussing the current standard in the related field and the introduction of essential learning frameworks by different theorists that are used in the field of programming learning and IT education, this thesis proposes two conceptual frameworks to improve programming learning efficiency. Conceptual framework 1 (for children) is tested in a real life learning environment to see if it can aid students to improve overall studying motivation and learning performance. Conceptual framework 2 (learner centric) is proposed as an alternative for older teenage learners to emphasize real time collaboration and code accessibility. Both conceptual frameworks offer new approaches to effective programming learning and can can be used to improve learning environments in elementary/primary, secondary and higher level educational institutions.

Keywords:

Education, Programming Leaning System, Conceptual Learning Framework

Keio University Graduate School of Media Design

Jiahui Ge

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank everyone who was involved in the process of writing this thesis including professors, family, friends and the test participants. My thesis would not have been possible without the encouragement and guidance from these people. I am in debt to Professor Hiroyuki Kishi, Professor Akira Kato and Professor Kai Kunze for guiding not only my research but many aspects of my life.

Chapter 1
Introduction

1.1. Background

The trend of learning computer programming skills is steadily increasing. Kids as young as three years of age are being encouraged to acquire the skills to write computer program codes (AFP-JIJI 2019). While some people are learning to code as part of their course curriculum, there are others who are learning it as a tool to land lucrative jobs or simply to create useful applications and games. For instance, there are countries (e.g. United Kingdom) where computer programming has been integrated into the primary school curriculum as a mandatory course. And then there are still countries (i.e. Denmark) where digital literacy is not mandatory but is actively promoted by learning and applying the knowledge while building technologies. Programming is used to code software tools, games and utility applications for computers and the related computing devices (i.e. smart phones, tablets, wearables devices, etc.). It is an area of computer science that in Japan was used to be taught to only those students who took computer science as their major discipline in higher level education.

In 2017, however, the Ministry of Education, Culture, Sports, Science and Technology (MEXT), decided to make programming learning a mandatory part of Japan's public primary school education curriculum by April of 2020 (Sano 2019). This means, irrespective of whether students want to pursue a computer science career in future, it would still be mandatory for them to learn programming just as they would learn a foreign language like English.

The factors that drive individuals to learn coding voluntarily are different from the factors that drive governments to force students to learn coding. This thesis aims to investigate the difference between these two sets of factors. It shall explore how the aspect of adding the programming learning courses into Japan's formal

educational curriculum would relate to the country's economy and the general acumen of students.

The choice of programming language to teach and the education level to teach it in is a decision that the educational institutions should not merely base on the educational strategies and politics. It should be a decision based on best practices, opinions of educational experts, researchers, scientists and practitioners i.e. teachers and assistants (Jancheski 2017). This work aims to explore the situation. Based on the study of the effects different code teaching techniques have on the learning outcome in students, an analysis of the education settings that are most lucrative to the introduction of programming, and the neurological perspectives regarding the increase or decrease of learning efficiency in humans with age, an effective code learning educational framework shall be presented.

The motivation behind the research is that the design of an effective code learning framework would help in nurturing future generations that are sufficiently skilled to effectively fill in the difference between the supply and demand of IT workforce in the Japanese job market. This in turn would lift up the country's overall society as well as economy.

Existing teaching approaches utilized in the country to teach coding shall be observed in person by participating in a boot camp activity. The educational institutes do not share a common system of learning. So, in this work, three learning frameworks shall be proposed that can be implemented by three types of educational institutes or environments. Depending on the type of educational environment, each framework will have its own set of objectives. Also the market shall be explored for tools that provide collaboration and learning.

1.2. Research Objectives

The thesis explores the following research question: How to improve Japanese teenagers' computer programming learning efficiency?

The thesis shall explore through literature why the need of learning programming has become so urgent. It will highlight the reasons behind the rising global trend of learning the seemingly high-tech programming skills which seem an appropriate skill for only some selective career paths. The thesis will also review some

of the existing code learning frameworks. Based on the research findings, it will suggest two educational conceptual frameworks. Conceptual framework 1 will be evaluated through the hallway test, while the author will suggest a test-setup for Conceptual framework 2 to be conducted at a later point in time, due to current restrictions because of Covid-19.

The hypothesis is: Effective learning of coding skills is possible by teaching through a visualized practical informatics and programming learning system.

1.3. Research Contribution

As the trend of acquiring digitally skilled societies is global, the thesis shall serve as a basic manual to understand the reasons underlying the rising urgency in gaining coding skills. The research shall outline the consequences societies will face by gaining the coding skills or not gaining them. The thesis has proposed two conceptual frameworks for programming learning and successfully tested one, which showed that it can be used to improve overall learning efficiency. Its work can be used as a reference for educational institutes to ensure effective programming learning and benefit the upcoming generations of professionals in Japan.

1.4. Research Methodology

The thesis shall use literature review as a major research methodology. The data gathered would be from both the primary (on field research) and secondary sources (internet, books and surveys). Based on the study of literature, it shall propose two conceptual frameworks to facilitate the learning of programming languages. The author will conduct a field research to analyse current problems and difficulties for programming learning. Data will be collected, including techniques of observation and a questionnaire. The author shall also conduct a small scale hallway test to measure the difference in learning outcomes by using the conceptual framework 1 and observing the learning responses of two student groups who are taught programming lessons. For documenting the research and findings, the thesis uses the Chicago-Style citation format.

1.5. Thesis Structure

The thesis will start by providing the motivation behind the research. It shall then confirm or negate the research hypothesis through a literature review of the social, economic, market and psychological factors in relation to teaching programming language earlier than the higher education level. Based on the findings after reviewing the literature, the thesis shall present two conceptual frameworks to improve the students' learning efficiency, each framework achieving a specific set of learning goals. A conducted hallway test and a proposed test design will be discussed for one of the conceptual frameworks respectively. Lastly, the thesis will conclude the research in relation to the research hypothesis.

Chapter 2
Literature Review

This section reviews some of the available literature on the topic of programming learning. It aims to highlight its significance, the underlying motivation, the social support and resistance it has as part of curriculum, the procedure to implement it in educational systems, its social and economic effects, learning tools that can help in acquiring the skill and the market research of computing technology.

2.1. ICT vs Computing

ICT is short for Informations and Communications Technology. STEM refers to a curriculum that includes Science, Technology, Engineering and Mathmatics. The shortage of digital skills has turned the focus of the world toward teaching STEM education. 'T' in stem has been assumed to mean programming learning and so many governments have made it part of formal primary education (Sterling 2016). Some argue in favour of this measure, while some show resistance.

One of the biggest issues that has surfaced with reference to the teaching of computing in schools is that the concept gets mistaken to be ICT. So, the students are taught the basic skillset pertaining to office applications such as word-processing and spreadsheets. Another issue is that the teachers have no personal formal education of their own in computing. So, they lack to convey the interest or the basic comprehension regarding the things that happen inside a computer in order to make it work. In schools where there is a course aiming to provide an introduction to programming, the teaching style is such that the student gets intimidated rather than being enlightened. Consequently, students get frustrated and bored, and eventually leave the class with the opinion that programming is not easy and confusing. Due to these reasons it may not come as a surprise when these students

do not pursue a course related to computing in their higher education studies or even in their professional workplaces (Wilson and Mofat 2010).

2.2. Factors Involved in Voluntary Code Learning

Computer literacy (computer, internet, communication, ICT) along with digital skills (cognition, critical thinking and problem solving) have become merits for employees to land a lucrative job, secure it and excel in it (Pirzada and Khan 2013).

The high demand for qualified IT specialists means that a career in Computer Science is well-paid. In the developed economic countries such as United States, in 2018, the average annual salary of IT related occupations is exceptionally high. Although Japanese IT specialists are not as highly paid as the Americans, their salaries are still more lucrative than many other jobs in Japan.

The recruitment chances of individuals with digital skills is higher (Ossiannilsson and Ioannides 2017). Roles in the research and career areas where having coding skills is considered essential include knowledge constructors, to create teaching artifacts from personal experiences, innovation designers, to create new and imaginative solutions and computational thinkers, to develop and test solutions (Schmidt-Crawford et al. 2019).

Learning coding because the society needs professional coders should not be the biggest motivation. Coding should be considered as a class similar to drawing, music, dramatics or sports that is learnt as an extra-curricular activity in schools. Just as a music class does not make every student a musician, one coding class does not make a programmer. However, these classes help identify hidden talents and interests (Sterling 2016).

2.3. Factors Involved in Government Initiatives

Many countries have integration programming learning into their education curriculum as a mandatory course. The countries include Spain, France, Austria,

United Kingdom, Slovakia, Bulgaria, Israel, Czech Republic, Poland, Denmark, Estonia, Hungary, Ireland, Malta, Portugal, Lithuania, South Korea, Turkey and some states of America (Uzunboylu, Kinik and Kanbul 2017). The rationale behind governments taking this measure is that formal code learning education would promote fostering individuals with skills such as logical thinking, problem solving, coding skills, ICT employability and other competencies.

We live in a digital era where computer and internet accessibility is on a constant rise worldwide. According to Statista findings (Alsop 2020), a computer is present in nearly half of the private households worldwide. The penetration rate of computers is almost one-third in the developing countries, while it is over 80% in the developed countries. Where the computer is absent, the house still has connection to internet through other computing devices such as smart phones, tablets, laptops, wearable devices, etc. (ITU 2019, 7). According to the report by Internet World Stats (2020), on Internet usage data till May 31, 2020, 59.6 % of the global population is in one way or another online i.e. connected to the Internet. In Japan, 93.3% of Japan's population uses the Internet (2018).

Realizing the deep-rooted status of technology in the fabric of societies worldwide, and considering the fast pace of technological advancements, the demand for computer and IT experts has skyrocketed in all types of industries. While the demand for IT specialists may have increased, the supply of competent and skilled computing professionals is not enough to meet the required skill demands. To compensate for the gap between the computer skills demand and the skilled labor, different developed and developing countries are heading towards making the study of computer science a compulsory subject in their formal education frameworks. By making the learning of programming a mandatory part of school curriculum, the government aims to equip the future professionals with the required IT skills.

Digital skills have become a necessity for all employment sectors (Leahy and Wilson 2014). Digital skills include technical skills, communication, work-related IT skills, personal skills (cognitive, motor, sociological, and emotional) and personal attitude (the ethical, secure, critical, creative and innovative use of technology).

To observe the level of digital skills the workforce generally possesses, Ester van

Laar, et al. (2020) conducted a survey of 87 professionals belonging to creative industries such as project managers, engineers, designers, journalists, architects, photographers, etc. An assessment was designed to see the level of presence for four digital skills: critical thinking, information evaluation, problem solving and creativity. Most of the participants showed low digital skills pertaining to information evaluation and problem solving.

Talent mismatch is a global phenomenon that is constantly on the rise. In Japan the situation is constantly deteriorating as for every job seeker, there are 1.5 job positions available. There is no talent to fill in the vacant positions. As most of the country's labor force comprises of elder population, who have already worked to their full potential, the job vacancies are expected to rise further. As per a report by Ministry of Economy, Trade and Industry of Japan, by the year 2015, the country was already in shortage of 170,000 technical workers. This number is expected to rise to 590,000 by 2030 (Dressler 2017).

The government's decision in 2017, to make computer learning courses mandatory in primary education was based on the abovementioned numbers. The country also wants to follow the global trend of developing generic skills and aptitudes in the country. By instating a system to add technological thinking, knowledge and skills in the instincts of future generations, the ability to apply learning aptly will be boosted. According to MEXT, the main reason for the educational reforms in Japan is to instate technological perspectives and thinking styles which allow viewing society and daily life phenomena with respect to their connection with technology and rationalizing how technology can be optimized to facilitate a society's needs, safety demands, environmental issues, economic competence, etc. (Yata, Ohtani and Isobe 2020).

In the countries that have not yet made programming learning a mandatory part of their education curriculum, the government is still playing an active part in raising the awareness and assigning funds to institutions and educational organizations to ease the access to digital skills. Courses and certificates in educational institutes serve as proof of an individual possessing a standard set of digital skills that coincide with current technological skill requirements (Chenoy, Shobha and Shiv Kumar 2019).

With regard to teaching students the fundamentals of programming, the people

who show resistance to the idea present the argument that most of the students who are learning programming may not grow up to be computer experts or scientists. So the time of the student that is invested in doing a futile activity (i.e. coding) can be shifted to doing something more meaningful and valuable. People who support the idea of teaching programming do agree with the fact that not everyone will end up becoming a computer professional. However their side of the argument is that the exercise of learning to code itself is a beneficial task as it allows a student to gain an understanding of the basic working of a computer. This in turn allows a student to learn skills such as problem solving, computational thinking, and earn him/her the knowledge to control computers. Students can think of themselves as creators of computers rather than consumers (Schmidt-Crawford, Lindstrom and Thompson 2019).

2.4. Programming Language to Teach

Depending on the complexity, ease of learning and the feature sets, the programming languages can be organized into the following main levels:

1. Novice – These are simple text-based or visual block-based languages that are focused at teaching algorithm building rather than code syntax. For instance, Logo, Scratch, Code.org, Tynker.

2. Intermediate – These include some of the visual programming languages where the interfaces of software applications can be made through a visual interface and the coding is done behind the scene. For instance, Visual Basic, Visual C++. Some other simple languages also fall in this category, that are easy to learn. For instance, Python, JavaScript, HTML, etc.

3. Expert – These are object oriented languages that have rich features set, syntax rules and extensive libraries. For instance, C++, Java, C Sharp, etc.

The underlying intention behind programming learning is to gain cognitive skills that teach an individual critical thinking and problem solving skills. All the programming languages stated above can help acquire these skills.

2.5. Appropriate Age for Code Learning

It is an open question as to when a child can learn to code. The learning of traditional programming languages such as Java, C Sharp, Python, C++, Delphi, etc. requires syntax learning. This is a rather daunting task which causes the younger children of primary level to get discouraged and uninterested in learning programming altogether. Even the easier visual programming languages (Visual Basic and Visual C) are unsuitable for younger programmers. The older primary student may be a suitable level to introduce these traditional languages. For the younger students, the most suitable programming languages are the simple text based language such as Logo or the visual programming languages such as Scratch (Jancheski 2017).

As to the question of the appropriate age or level (elementary, primary or secondary) at which these languages should be introduced, it is on ongoing debate (Jancheski 2017). Learning of text based programming language requires children to be considerablly patient and mostly follow the teacher. There have been cases of as low as eight year olds reported to have learnt text based coding (Jancheski 2017). At ten years of age a child is more mindful and is able to make model longer code snippets that lead to certain outputs. A few children at the age of ten are able to tackle the basic elements of traditional languages, Python and C (Ghose 2016).

Based on the research explored, no age is too early to learn programming skills. Similar to the learning of foreign language, the sooner the programming is learnt, the better. Even before a child can learn to read (Jancheski 2017). There are visual programming tools such as ScratchJr that 5 year olds can use to make simple stories and games. Other visual tools to help learn coding concepts can be used by children such as The Foos, Lightbot, Tynker, etc. (Ghose 2016).

Studies of neurology with regards to learning efficiency reveal that children learn faster than adults. This is because due to the immature cognitive abilities, children younger than seven years use implicit learning mechanisms rather than explicit ones, as is the case with adults. Unlike the explicit learning which is a conscious effort, during the implicit learning phase the learning is unintentional. The learnt lessons become part of the child's instincts and are triggered automatically when the need arises (Lichtman 2016).

With view to adults, there is no upper limit to learn programming. Even adults can start from the visual tools designed for children to learn coding. As programming is the most needed skill of the 21^{st} century, for the adult programming illiterates, it is better to learn it late than never. In view of this notion, some of the notable universities in the US have taught the visual programming language (Scratch) to high level students in the introductory subjects of computer science (Jancheski 2017).

2.6. Effect of Coding in Skill Development

Programming is now being taught all over the world at K-9 level in a voluntary educational setup or as part of formal curriculum. In a Swedish study by Nouri and Zhang (2020) the skills that programming teaches to the learners were explored. Based on the experience of 19 teachers, programming not only taught the computing technology (CT) specific skills i.e. computational concepts, practices and perspectives, but it also taught the digital skills of 21st century i.e. cognitive, creativity, language, collaborative and problem solving skills.

Programming is the visual representation of an action plan in a language the computer understands (Myers and Ko 2009). And the action plan is based on an algorithm which is a sequence of conditional steps realized while solving a real world problem (Uzunboylu, Kinik and Kanbul 2017). Research shows that children who studied computer science have better performance in logical thinking and vertical thinking. These skills could be beneficial for a life time.

Skills associated with learning to code includes increased digital literacy, creativity, research tendencies. Coding boost problem-solving, outcome and process based thinking, critical and spatial thinking, collaboration and social skills. Since coding gives students a sense of confidence, it may inspire students to attend school with more enthusiasm. Coding becomes part of instincts, so the knowledge acquired utilizes a student's long term memory (Uzunboylu, Kinik and Kanbul 2017).

Computational thinking is a skill that is independent of programming. But programming helps to polish this skill. As a programmer understands how computers work following instructions, he/she refines the proposed solutions until they be-

come computer-perfect (Sterling 2016). Computational thinking is built from the skills such as human behaviour comprehension with regarding to computer science fundamentals, problem solving and the designing of system solutions (Wilson and Moffat 2010).

To study the effect computer programming has on an individual's spatial thinking capability and algorithmic problem solving skills, a case study was conducted on 5-6-year-old kindergarten children in a whole-class social mode (Fessakis, Gouli and Mavroudi 2012). After some preliminary playing to get the children familiar with the symbols of game and the interactive software environment, the children were then given seven tasks, six of which were analogous path problems while one was a maze solving task. The children showed active participation and excitement, and learnt lessons on problem solving, mathematical concepts and social skills.

Student engagement is an important factor that plays a crucial role in determining the student's performance/success in school. The amount of technology enhanced learning (TEL) has an effect on the high and low levels of student engagement. And the level of student engagement in turn has an effect on the level of digital skills a student acquires. This eventually affects the student's overall outcome/performance. In their study, Bergdahl, Nouri and Fors (2020) found that the high levels of digital skills were related to engagement in TEL. But the disengagement was not found to relate to digital skills. The variation is digital skills and way of engagement while use of TEL can strengthen the educational and socioeconomic gap.

2.7. Code Teaching Techniques

Effective computer science programming learning techniques in children as well as adults were studied.

2.7.1 Teaching Coding as Part of STEM

Sterling (2016) is of the view that coding should be taught as part of STEM courses in the curriculum as it relates to mathematics and science subjects. Examples of computer systems that apply the mathematical and scientific knowledge can

help the student relate the three technical knowledge areas, instigating in them a curiosity and interest of the field just like science and mathematics.

2.7.2 Teaching Coding as a Language

According to Bers (2019), there is a need for teachers to move out of their comfort zones and go beyond the simple adoption of the conventional teachings methods that have been using to teach the tradition STEM courses for years. Computer programming should be considered as "coding as another language" (CAL). There are six stages of coding that a child goes through while learning CAL. The underlying principle behind CAL is basically understanding how programming learning is using a new language to express and communicate functions. As language and literacy play a crucial functional role in child's early years, the same models of instructions can be used to supplement computer science.

With the view of giving programming a place as an optional foreign language in the school's courses curriculum, Sterling (2016) does not support the view. According to Sterling, it is more appropriate to categorize it as part of the compulsory STEM courses than being considered as part of cultural courses of communication. As per Sterling, the communication between humans is not the same as the communication between and humans and computers. In the curriculum of digital technology standards, coding should be given a significant place.

2.7.3 Teaching Coding in Earlier Years

As computers are being used by children even before the age of 3. By providing the right type of computer exposure, mathematics can be taught to children from birth to 3 years of age (Clements 2002).

Introducing programming at early ages lays the foundation of children's personal capacities in future. They learn the skills of designing and solving problems which earn them computational thinking, a skill they would use in future. Depending on their experience, there is a higher chance of them pursuing a coding career, benefitting society and economy. Many young children use tablets these days. By exposing them to programming environment at home can extend their classroom learning to home (Sterling 2016).

Regarding the use of ICT in classrooms, the results from a survey of 100 early education teachers suggest that teachers find it suitable to use computers to teach students in their earlier years. They have been effectively using computers in their curriculum 1-2 times a week to teach young students of primary and nursery levels. Through the computer based activities, the teachers aim to target the cognitive skills (such as perception, problem solving and decision making) and language development. And in their view, the activities have shown positive results in these skill development areas (Yurt and Cevher-Kalburan, 2011).

Based on the literature, it is found that visualization has a significant role in a student's programming learning process. In contrast to the traditional theoretical learning environments, students who have access to visualizations have a change of gaining a higher efficiency level in different skills (Ghose 2016).

The recommended duration of computer usage in school level education is also 1-2 times a week. According to a report by BBC (Coughlan 2015), in cases where the teaching relied heavily on computer technology, for instance, more than twice a week, the students' performance results deteriorated.

Children sitting in front of computer screens for long durations to learn coding is not an effective way of learning to program. Rather, the use of shorter interactive visual applications seems to be more effective for both teachers in their teaching as well as students in their learning (Ghose 2016). This approach has been used both by educational games and visual coding programs (Jancheski 2017). The enthusiasm and effort of the teacher dictate the coding experience of the students (Sterling 2016).

2.7.4 Game Oriented Teaching

According to a study by Schindler et al. (2017) from the available computer-based technology, digital games offer the highest form of student engagement (emotional, behavioral and cognitive).

According to a study by Kiss (2013), students perform better in their programming learning endeavors when the learning lessons and exercises are game oriented. In the study, the output of a programming learning course where the examples were game oriented, produced better learning results. While learning the basic programming constructs, in contrast to writing traditional applications,

the students showed higher motivation towards writing game programs.

2.7.5 Constructivist Based Teaching

When courses are packed with a lot of content, the teachers end up simply reciting their personal knowledge and students tend to "cram" information. There is no deeper understanding of the subject nor any development of essential life skills such as problem solving, critical thinking and communication. Memorizing facts does not constitute learning. Learning rather it is the ability to use one's memory to find, assess and apply knowledge. Active processing of knowledge leads to learning, not the passive reception of information. To learn, students need to develop their own sense of concepts, relationships and procedures. The teachers must foster them during their learning phase by reducing the amount of information that is required to memorize concepts. Passive formats of lecture delivery should be revised, and teachers should invest their efforts towards encouraging students to become active self-learners, problem designers and solvers. The teachers should organize and conduct collaborative and interactive activities, arrange educational games and create and provide a learning setting where the students are encouraged to make inquiries about their confusions (Lujan and DiCarlo 2006).

Papavlasopoulou, Giannakos and Jaccheri (2019) carried out a design-based research workshop for the children aged 8-12 to study the effectiveness of constructionism based coding activities in developing children's technical learning and social, cognition, creativity and emotional development. In constructionism based activities, the learners are active knowledge constructors rather than passive knowledge recipients. Constructionism based activities gave the children coding as an object to think and build with. The children learned how to program while building games with a Scratch tool. As children of the current digital age are accustomed to playing video games, the task of developing games is more involving than project-based activities. Game making activity enhanced the student's skills in problem solving, collaboration and critical thinking. In the first activity, the children learned how to apply programming concepts of conditions and loops on real robots that they could touch and design. This activity was followed by a paper-based tutorial containing inspiring examples and visualizations to de-

velop games. Thereafter, children worked in groups of twos and threes, imagined their games, created and played with them using the Scratch tool and shared and reflected upon their works while having fun.

2.7.6 Teaching Using Tools

According to Keen (2011), tool usage in children is an important step to build a child's problem solving skills. Through tools, the progress of children can easily be measured as their level of thinking, planning and goal development skills can be evaluated against a directed behavior comprising of certain sequential steps. The curiosity and explorative nature of children are their driving force to learning. When the actions are expected to be goal oriented, students can change strategies in pursuit of the defined goal. These two factors together make tool usage an ideal choice for teaching and instating a desired skill in children.

2.7.7 Four Approaches to Teaching Coding

Sellby (2011) describes four methods that have been used to teach beginners level programming learning. These methods teach the aspects of procedural, object oriented programming, language paradigm and visual programming.

1. Code Analysis Approach – helps recognize parts of code and their meaning. Similar to how learners begin to read prior to writing, coders learn to understand code first to produce their own.

2. Building Blocks Approach – It is similar to how vocabulary, nouns and verbs are learnt in a language before creating sentences.

3. Simple Units Approach – Learners learn solving small problems first. Then they apply the logics learnt to build solutions to complex problems.

4. Full Systems Approach – similar to how a foreign language is learnt through immersion, the learners start by designing the solution for a significant portion of problem. The language constructs and programming concepts are added when the solution requires to apply them.

Only by mastering these four approaches can a learner become a competent programmer. By using them together, a learner gets the opportunity to gain complete understanding, skills and knowledge of programming. Teachers should encourage the students to actively create knowledge based on their experience, i.e. support constructivism and artefact creation. Code walkthrough, reading, debugging, etc. are a few constructivist activities that they can direct the student to.

2.8. Shortage of Qualified Teachers

The problem with most educational systems is that up to K-12 level, outside the traditional core STEM areas i.e. mathematics, chemistry, biology and physics, the standards of education in engineering and computer science are not well established. And so majority of the traditional STEM teachers are not confident that they can train students programming or computer science efficiently (Kay and Moss 2012).

A training of teachers is required. In this regard, Kay and Moss (2012) designed and carried out a study wherein twenty K-12 mathematics, science as well as computer science teachers were given a three-day robot programming classes after school. The teachers taught at elementary and high school level. 89% of the teachers had little to no programming experience and doubted their learning ability. The workshop taught them programming skills and gave them the confidence to teach programming to hundreds of students (revealed through a follow-up survey 9 months later).

2.9. Market Analysis

In order to design an effective conceptual framework, it is important to realize the technology usage trends of the target population. Analyzing the existing IT education industry will reveal the demands of the market.

The general global trend of computing devices connecting to the internet shall keep rising (Help Net Security 2019). The market for smart wearable devices has also been on the rise (Statista 2020). From the devices that access Internet,

smartphones are the most widely used devices (Chaffey 2019). A survey of 9 global markets reveals that the majority of the digital audience are multi-platformers i.e. they use both desktop/laptop and smartphones to access online content (Chaffey 2019). According to the data collected by Global Stats (2020) between May 2019 to May 2020, Google Chrome has the biggest share in the global browser market (63.91%) in contrast to the Safari (18.2%) and Firefox (4.39%) browsers.

A large percentage of the global population uses technology for gaming purposes. Based on the survey by Limelight Networks (2018), responses from young gamers from the countries of Japan, France, Germany, UK, USA and South Korea, who play video games at least once a week, were recorded. On an average, the gamers spent over 5.96 hours each week gaming. Instead of computers (laptops/desktops), gaming consoles and tablets, most of them used mobile phones to play games. Free games are generally downloaded and played more than paid content. In all the countries worldwide, casual single-player games are played the most (see Figure 2.1).

Country	Casual Single-Player games (like Candy Crush or Angry Birds)	Casual Multi-Player games (like Words With Friends)	First-Person Shooter games (like Call of Duty)	Single-Player Role-play games (like The Elder Scrolls)	Massive Multiplayer Online games (like World of Warcraft or League of Legends)
France	2.43	0.90	1.18	1.10	0.90
Germany	1.78	0.78	1.07	1.11	0.98
Japan	1.25	0.81	1.07	1.46	0.97
South Korea	1.74	1.33	1.51	1.47	1.66
U.K.	2.06	1.10	1.37	1.23	0.94
U.S.	2.39	1.28	1.20	1.05	0.90
Global	**1.94**	**1.03**	**1.23**	**1.24**	**1.06**

(Source: Limelight Networks 2018)

Figure 2.1 Game Types Played Worldwide on Computing Devices.

Based on the market survey, it can be deducted that the adoption of smart devices is human societies is on a constant rise. Coding tools, software applications, websites and learning methodologies, with designs that are flexible enough to compensate for the innovative technology have a sound place in the future.

The above market trend also shows that the future is secure for technology specialists and application/game developers as the technology market is expected to boom even more. A career pursuit in this direction could be a lucrative choice as the technology is holding a huge share of the global economy.

Education of society through games could be a way to incite curiosity to learn more. Educational games with the design that follows the casual-single player template may contribute towards this venture.

2.10. Existing Technology Education Frameworks

Many theoretical and conceptual frameworks exist to suggest the optimal settings to ensure effective learning. This section presents four of these approaches that can apply to programming learning in the different educational environments.

2.10.1 Mobile Learning Framework

Ossiannilsson and Ioannides (2017) propose a theoretical framework and a learning methodology that supports the innovative mobile technology of the 21^{st} century. The framework combines all the different learning approaches possible in mobile learning (heutagogy, authentic, challenge-based, self-determined, personal, ubiquitous and collaborative learning) into one holistic, contextualized approach that has capability to facilitate learning across all disciplines (see Figure 2.2). The framework allows learner-centered learning in which the learning potential is set by an individual himself. The implementation of the framework in education sector would require reforms in not just the curriculum, course and delivery design and development sector. The framework offers guidelines to revise the assessment mannerisms as well. In contrast to the traditional educator learner hierarchical association, the framework works by emphasizing the collaboration of the two parties where each party can take up the roles of both the producer and consumer of knowledge.

(Source: Ossiannilsson and Ioannides 2017)

Figure 2.2 Mobile Learning Framework.

2.11. Learning Assessment Framework

McGill and Volet (1997) propose a theoretical framework to analyze the programming knowledge in students. The framework has incorporated the three knowledge areas of programming skills that students learn through computing courses i.e. syntactic (facts and rules of programming language), conceptual (constructs and principles of programming language), and strategic (problem solving using the programming language). The framework relates these three skills with three distinct knowledge areas the literature of cognitive psychology i.e. declarative knowledge, procedural and conditional. The highest level of knowledge is the

strategic skillset in programming and conditional skills in cognitive psychology. The next in hierarchy is the conceptual skillsets (procedural cognition) that enables students visualize the working of the program e.g. the arithmetic involved, the data structures used, general sequence of program execution (algorithm), etc. The lowest form of skillset is the syntactic (declarative cognition knowledge) and are programming language dependent not problem dependent. These skills are used for expressing a solution designed for a problem. Five categories of programming knowledge were defined (see Figure 2.3).

Based on the supporting data, knowledge testing questions and findings from a previous study, the usefulness and potential of the framework model was found. The framework allows diagnoses of problem areas by assessing the programming knowledge of beginners during a teaching course. Based on the knowledge level, the course instructions can be revised to ensure effective learning.

A Conceptual Framework of the Various Components of Programming Knowledge

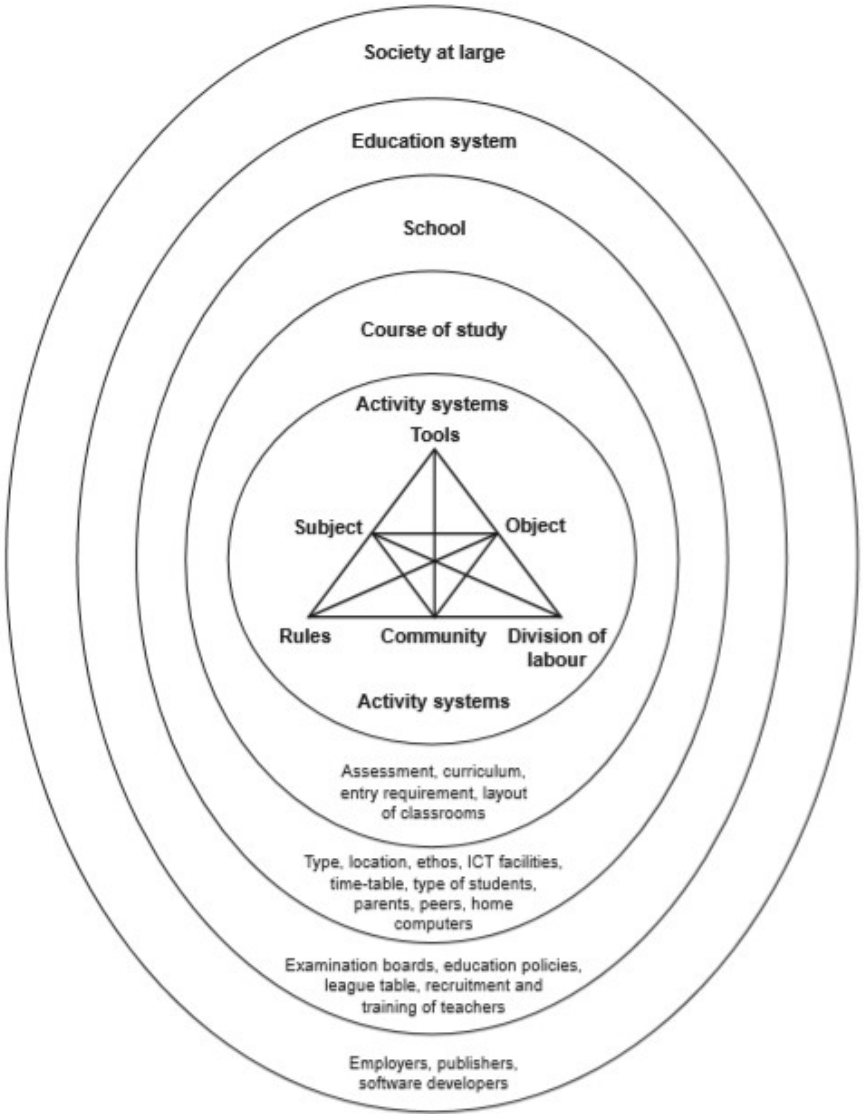
	Declarative Knowledge	Procedural Knowledge
Syntactic Knowledge	Knowledge of syntactic facts related to a particular language, such as: • Knowing that a semicolon is needed to end each statement in Pascal. • The ability to explain the syntactic differences between a procedure and a function in BASIC	Ability to apply rules of syntax when programming, such as: • Ability to write a syntactically correct REPEAT statement in Pascal. • The ability to open a text file and read from it using BASIC.
Conceptual Knowledge	Understanding of and ability to explain the semantics of the actions that take place as a program executes, such as: • The ability to explain what a fragment of pseudocode does. • Knowing the way in which the result of a function activation is returned in a particular language.	Ability to design solutions to programming problems, such as: • The ability to design a procedure to compute the mean of some data. • The ability to modify a program that prints a 1-D array to print a 2-D array.
Strategic/Conditional Knowledge The ability to design, code, and test a program to solve a novel problem.		

(Source: McGill and Volet 1997)

Figure 2.3 Programming Knowledge Testing Framework.

2.12. Sociocultural Framework

Lim (2002) proposes a theoretical framework that studies ICT in schools through a sociocultural approach which rejects the notion of learning ICT in isolation but should be learnt in a broader context. The framework employs the sociocultural perspective and presents a holistic design based on an adapted version of the activity theory. Activity theory dictates that an activity system represents only a single operational unit that is bounded by ecological circles of various levels. Lim proposes a concentric model in which the garden-as-culture metaphor is applied to the activity systems (see Figure 2.4). The underlying idea shared by both garden and culture is the same that the young organisms grow in an artificial environment that is maintained through optimal surroundings facilitated by useful tools and other organisms.



(Source: Lim 2002)

Figure 2.4 Sociocultural Framework for ICT Learning in Schools.

The ICT based lesson represents the activity system. The outermost circle represents the society at large. Each circle has its own activity systems. Each circle impacts the circle that it is outside of it and vice versa. All activity systems are interdependent. A change that is initiated by an activity system has an influence on all the related activity systems. Based on this framework, the argument is

presented that in order to nurture ICT learned societies, restructuring will be needed in all the levels of the related entities (i.e. the lessons, courses, schools, educational system and society) so that the alteration triggered in the innermost levels are able to reach the outermost levels.

2.13. Community of Inquiry Framework

Garrison, Anderson and Archer (1999) suggested a conceptual framework that caters to the computer mediated communication (CMC) in the higher education levels. In CMC, the medium of communication may be textual, audio or video, where the impact factor of each depends on the disciplinary context of communication. The model represents the education system as a community of inquiry where the key participants are the teachers and students.

Cognitive presence is a crucial element of an effective learning system. It is represented as the critical thinking that leads through a process to an outcome. Social presence represents participants of the learning system i.e. the teachers and the students. Teaching presence represents two functions that either of the participants can perform: designing and facilitation. Designing involves structuring the course outline, objectives, activities and assessment criteria. This function is generally carried out only by the teacher. Facilitation involves supporting the social and cognitive presence. This functionality is shared by both the teachers as well as the students.

Cognitive presence is best achieved through collaborative constructive perspective which is dependent on the social presence. And the collaborative activities are determined by the teaching presence. Some examples for the three elements (cognitive presence, social presence and teaching presence) in an educational community of inquiry are given in Table 2.1.

Table 2.1 Community of Inquiry Indicators

Elements	*Categories*	*Indicators (examples)*
Cognitive Presence	triggering event	Recognitions of the problem, feeling confused
	exploration	exchange of information, discussion of uncertainties
	integration	relating ideas, design and production of solutions
	resolution	Application of new ideas, critically evaluate solutions
Social Presence	emotional expression	Emoticons, real-life descriptions
	open communication	Expressing freely, recognizing others
	group cohesion	Positive collaboration, help and support
Teaching Presence	instructional management	Structuring of content, Setting up of discussion topics, making student groups
	Building understanding	Sharing of personal values, showing agreement, taking student opinions
	Direct instruction	Directing discussions, replying to questions, making amendments, summarization of results/problems

(Source: Garrison, Anderson and Archer 1999)

2.14. Existing Code Teaching Tools

Many tools exist to aid the code learning process. But the criterion to filter through this list is the feature set offered by the tool versus the features required for effective learning.

Table 2.2 Collaborative Program Learning Tool

Tool	Group Formation and Member Addition	Sharing of Code	Instant feedback	Communication
CodePen	Yes	Yes	Yes	Text
JS Fiddle	Yes	Yes	Yes	Audio and Text
CodeSandbox	Yes	Yes	Yes	No

(Source: Adán-Coello, J. and others 2011)

For the novice learners or beginners, a tool that focuses on teaching the skill of problem solving and algorithm development is the best choice. Table 2.3. shows a a few tools that are appropriate tools for novice learners (Ghose 2016).

Table 2.3 Programming Learning Tools for Novices

Tool	*Description*	*Age*
BeeBot	A real-world robot toy, BeeBot, that has only left- and right-buttons on its body. Kids learn sequencing the left and right commands to move the robot from one end of a room to the other while avoiding the hindrances that come in the way.	1+
Robot Turtles	A physical board game that teaches the use of directions, while guiding the turtle through a maze towards a jewel the turtle can eat.	4+
Light Bot	A smart phone app (supports iPhone and Android) that teaches navigating a robot through a maze while turning on lights on the way.	4–8
Dash & Dot	A robot pack that is programmable.	5+
ScratchJr	A free app (supports Android and iPhone) that allows use of simple icons to code stories and games that are interactive.	5–7
The Foos	A free app (supports iPhone) with symbolled icons (e.g. monsters, speech bubbles, arrows) used for learning to solve adventures (e.g. chase a donkey thief or rescue lost puppies in space.	5–10
Tynker	An online visual interactive course that teaches coding (fundamentals and designing) through stories and games.	7+
Scratch	A free to use coding language that teaches basic coding concepts (repeating loops, if-then statements) using blocks of textual commands.	8 – 16
Lego Mindstorms	Combination of LEGOs with sensors, motors and remote controls to construct robots that can walk, talk and follow commands.	10+

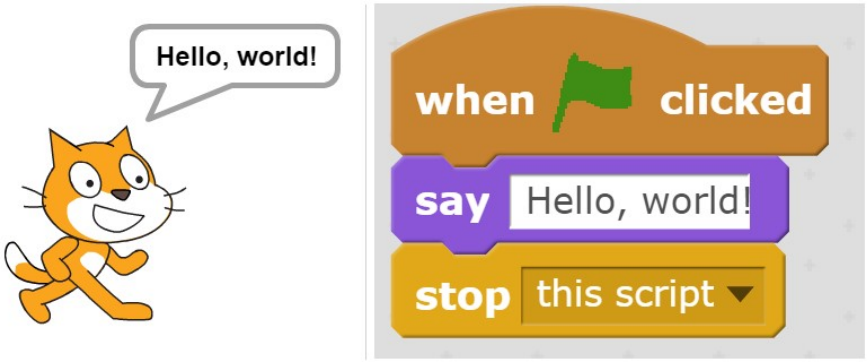
(Source: Ghose 2016)

From the above list in table 2.3, ScratchJr. and Scratch are the two most widely used programming learning tools. Currently many educational institutions worldwide are using both its versions for teaching children coding as part of mandatory course or voluntarily.

Regarding the effectiveness of Scratch, Wilson and Moffat (2010) give details of the boot camp they conducted and the results they achieved while teaching Scratch to children (age 8) of a class in Glasgow, Scotland. In the first visit, a concept of algorithm (steps) was taught. Thereafter, every week there was one hour-long lesson. And there were in total eight lessons. In each class a programming concept was taught through examples (demos on whiteboard or through toy robot) and then the students (in groups of twos) would reproduce what was taught in their Scratch programs. The order of the concepts taught was introduction to Scratch, introduce sequence, first test, iteration, selection, coordination and synchronization, Scratch card, second test and lastly a final test in week nine. The assessment measurements revealed that in contrast to the other ICT lessons that were ongoing at school, the weekly Scratch lessons were found to be extra cognitive (i.e. they augmented the skillset to solve problems and think critically) and affective (i.e. they augmented the enjoyment levels and inner motivation). By the end of eight lessons, students were able to code programs of elementary level. Contrary to the frustration and anxiety which is the general sentiment after the first programming learning experience, although the cognitive progress was reasonable, the factor of enjoyment made the programming learning experience a positive one.

Scratch is a visual programming object oriented tool that can be effectively used by novice program learners of all age groups. The interactive interface of Scratch is such that a learner can never misspell a command. Having being freed from the burden of learning the command, the learner can focus on the problem solving and algorithm design part. The environment of the program is such that it provides instant feedback. During the programming, instant display of results plays a significant part in keeping the learner engaged and motivated (Sterling 2016). It is also important in situations where the logic starts to go in the wrong way. The mistake can be corrected easily. Making computer science seem easy, Scratch can motivate learners inciting in them the curiosity and urgency to learn

more. The tool encourages the students towards self-directed learning. Figures 2.5 and 2.6 show a simple program created in Scratch (Speranza 2018).



(Source: Speranza 2018)

Figure 2.5 The Simple Interface of the Scratch Application

(Source: Speranza 2018)

Figure 2.6 Scratch Application Interface

For the learning of somewhat advanced programming languages, an important feature that promotes effective cognitive learning is the collaborated code development. This feature is important during programming learning tasks where the students are in groups. In such scenarios, the features expected to be present in the tool include formation of private groups, addition of participants to a group, sharing of code, instant feedback, communication through voice or message.

For simple web development program learning, two languages HTML and JavaScript can be learnt. So, from the code aiding tools available in the market, finding a tool that support the required type of effective learning collaboration is very important. The features provided by a few HTML and JavaScript learning tools versus the features required for effective learning are summarized in the Table (Ekene 2019).

2.15. Difficulties in Learning Programming

Having skills in computer programming is required in most of the fields including Science, Mathematics and Engineering at tertiary levels. However, students are facing different kinds of challenges during the learning process and the subject itself is considered difficult to learn. A research conducted by Rosminah, Siti md derus, siti rosminah Mohamad Ali, Ahmad Zamzuri Ali, Mohamad (2012) showed that students' biggest fear of learning programming is that they have difficulty in understanding the basic concepts of programming structure, furthermore, they are having a hard time designing a program to solve certain tasks.

Nature of Difficulties	Mean	SD
Understanding basic concepts of programming structure	3.40	0.884
Designing a program to solve certain tasks	3.36	0.952
Learning the programming language syntax	3.35	0.980
Gaining access to a computer	3.33	0.916
Using program development environment	3.22	0.930
Finding bugs from my own program	3.19	1.066

(Source: Rosminah and others 2012)

Figure 2.7 Difficulties in Learning Programming .

2.16. Factors that Lead to Poor Performance in Programming

A research conducted by Tan, Phit Huan Yee, Ting and Ling, Siew-Woei (2009) identified the reasons why students have difficulties in learning programming. The following figure 2.8 shows the factors that lead to poor performance in programming from the research.

Factors	Mean	SD
Less examples of practical use are shown	3.38	1.032
Computers provided in labs are not functioning well	3.32	1.157
Teaching methodology is less effective	3.30	1.099
Presentation of instructors and their attention on students	3.25	1.099
Students' lack of interest to learn	3.22	0.961
Syllabus focuses too much on theory	3.13	1.048
Syllabus coverage per semester is too wide	3.11	0.934
Learning environment that is not conducive	3.04	0.990

(Source: Tan, Phit Huan Yee, Ting and Ling, Siew-Woei (2009)

Figure 2.8 Factors that Lead to Poor Performance in Programming

As we can see from the figure, the biggest factor that leads to poor performance in programming learning is that students do not have enough examples for practical use, students also think the teaching methodology is not effective (Schwartz and Efklides 2012).

2.17. Overall Picture of Current Japanese Programming Education Scene

The programming education market for children is expanding rapidly due to MEXT's "Programming as mandatory course in elementary schools" policy. Recently, along with the trend, various services related to programming education for children have appeared, equivalent to other types of teaching materials for programming learning and the advent of programming certification for children (COETECO 2020).

(Source: COETECO 2020)

Figure 2.9 Services Dedicated to Teaching Programming for Children.

According to the report of COETECO (2020), the current Japanese programming education service can be divided into three types. "Classroom type", "com-

munication/online type" and "short term/event type. Before 2010, programming schools were mainly "short-term/event-type", which some companies established for a certain period of time, and there were not many "classroom-type" services. Over time, in response to the compulsory programming education at elementary schools, the number of classrooms increased rapidly as a result of the advent of franchise development in 2017 for a series of "classroom-type" schools. Another feature of 2018 is the entry of non-educational industries such as railways and electronics retail stores. The market for programming schools has rapidly increased since around 2017 and is expected to grow significantly as part of the education industry. On the other hand, due to the rapid increase of number of schools, we can see that the data for 2018 shows a decrease in the average number of students per school, and there are many issues such as the oversupply of schools where the quality is not catching up. However, most recently, with the advance of compulsory education, consumers' awareness of programming education is increasing, and there is an increasing impression that efforts are being made to improve the curriculum of each school and provide high-quality content. While there are high expectations for businesses in the programming education market to grow exponentially during the 21st century, as different industries enter, the number of choices will increase and the market will keep responding to diverse consumer needs.

2.18. Difficulties of Current Japanese Programming Education

In the classroom, there are specialized teachers and the environment is quite well maintained, so students can take relatively high quality lessons. However, research showed that there are some common problems in programming classes in Japan (ARSCHOOL 2020):

1. Too much emphasis is put on programming skills instead of letting students understand the programming concept. No one can ignore the importance of learning programming, however, instead of learning the programming concepts, learning programming techniques becomes the trend. There are

very few parents with programming experience and most of them don't know what programming is, why it is needed, or what it will be like to learn. It seems to be similar to Japanese English education. As a result of the importance of memorizing grammar and rules and taking high scores in tests, even after studying English for 6 years in junior high school and high school, many students can not speak English.

2. Lack of suitable learning environment. Some schools are already working on programming education even before it becomes compulsory. On the other hand, there are some who question whether the learning environment will be improved in the same way as other areas in schools where there are few teachers and students in depopulated areas. The Central Education Council Curriculum Section Primary School Section reported that the pressing issue is to improve the environment nationwide. In a depopulated school, the number of students is very small, the number of teachers is limited, and there are no specialized teachers. It will be necessary in the future considering that students in such an environment will be able to take the same classes as in other areas by using ICT.

3. Many of the teachers in programming classes are engineers. However, in programming learning, especially when teaching teenage students, it is important to have high "teaching skills" rather than being an engineer, which means to have high programming skills. Those who are good at developing the potential of children are better suited than engineers. For example, those who have studied Montessori education, and elementary school teachers. In most cases, programming that young students learn is not so difficult and requires no expertise. Rather, it is more important that the child is motivated and able to communicate with people.

Chapter 3
Field Research

Beginning the code learning process can be a daunting task for the children and teenagers. By personally interning as a tutor in an IT education company's boot camp and observing the existing education activities, allowed to learn the factors that make the code learning process seem hard for the students.

3.1. Study Environment

This was an observatory experiment that was conducted in May 2019, at Life is Tech, Inc., Tokyo Office / Bootcamp School. The program of participation was Online Camp.

3.1.1 Objective

Observe the general working of online teaching methods and analyze the code learning margin of students.

3.1.2 Materials Being Used

The company has self-produced a web-based educational platform called "Tekunorojia mahō gakkō". The platform is made based on Scratch but altered with original contents. The platform has collaborated with one of the biggest mass media and entertainment companies in the world and used famous characters to build the storyline in order to guide the students. The field research was conducted by testing product users and trial users.
The web-based program contains 76 lessons for entry level and beginner level, the program completion takes about 100 study hours (about 40 hours of entry level,

60 hours of beginner level). The program contains 3 main areas: web-design, game and media art.

Web-Design: While designing web pages featuring animated characters, students will learn HTML5 and CSS3 from the basics of web production languages. There is also a higher-grade web design lesson that uses JavaScript. Students will also be able to understand how the website they normally see is made, so that they can refer to the knowledge they gained and apply it in other contexts.

Game: Using the computer programming languages Processing and JavaScript, students will learn game production in various genres, from action games, racing games to puzzle games. Students will be able to feel the usefulness of mathematics and physics, as well as programming basics and algorithm basics, through experience. It is supposed to develop students creative skills rather than just playing the game.

Media art: CG is generated by programming and algorithm to create a moving art work. Processing is used very often in the media art field, even beginners can produce surprisingly rich expressions. In the latter half of the course, students will be asked to try to create more advanced visual expression using Shader, a programming language that is strong in graphics processing.

3.1.3 Research Questions

2251 questionnaires were sent to the users' emails. The questions are shown in the following figures 3.1, 3.2 and 3.3:

1. Who is filling the questionnaire? [单选题] *

○Mom

○Dad

○Myself

○Others

2. You or your kid's gender? [单选题] *

○Male

○Female

3. You or your kid's age? [单选题] *

○4-10 years old

○11-15 years old

○15-18 years old

○18-25 years old

○Above 25

4. Are you satisfied with your or your kid's needs by using those online education platforms? [单选题] *

○Yes

○No

○I don't know

Figure 3.1 Questionnaire Part 1

5. If you are not satisfied with your needs, what is missing? [单选题] *

○The contents are not interesting enough to make me motivated

○The contents are too easy

○The contents are too difficult

○The contents don't have the learning materials that I am looking for

○There are too many contents

6. What's the completion rate of your learning process? [单选题] *

○Below 10%

○10% ~ 20%

○20% ~ 30%

○30% ~ 40%

○40% ~ 50%

○50% ~ 60%

○60% ~ 70%

○70% ~ 80%

○80% ~ 90%

○90% ~ 100%

7. Were you motivated to learn with program? [单选题] *

○Yes

○No

○I don't know

Figure 3.2 Questionnaire Part 2

8. What was your biggest challenge in the learning process? [填空题] *

_____

9. If you haven't finished the program, what section of the product made your feel the most difficult to keep motivated? [填空题] *

_____

10. If you or your kid are not motivated to finish the program, what's the reason of being unmotivated? [单选题] *

○No 'real' teacher feedback

○No visual feedback

○No students' interation

○Not used to online learning environment, prefer real classroom

○Don't know the practical use for the program

Figure 3.3 Questionnaire Part 3

3.1.4 Participants and Observations

135 answers were collected one week after sending the questionnaire. 20 answers were chosen randomly from the library. The age distribution is shown below. 45% of the users are not satis ed with the platform and 20% choose the answer 'I don't know'. For Q5, 7 out of 13 said the contents are not interesting to make them motivated. 70% of students agreed that using visualization tools while explaining programs would help them learn the basic programming courses. The most challenged section for students to be motivated is the web design section that uses javascript.

The company was aware of the pressing demand of creating IT skilled professionals and had instated a course of online code learning. But the company was having a hard time in keeping the students motivated enough to learn coding. Analysis of the existing online teaching setup showed that over 50% IT education programs were being taught online. In all the programs combined, only 6% of the students actually finished the program. The underlying problem was that there was no motivation and no feedback for the students, therefore they were having difficulties understanding the programming concepts.

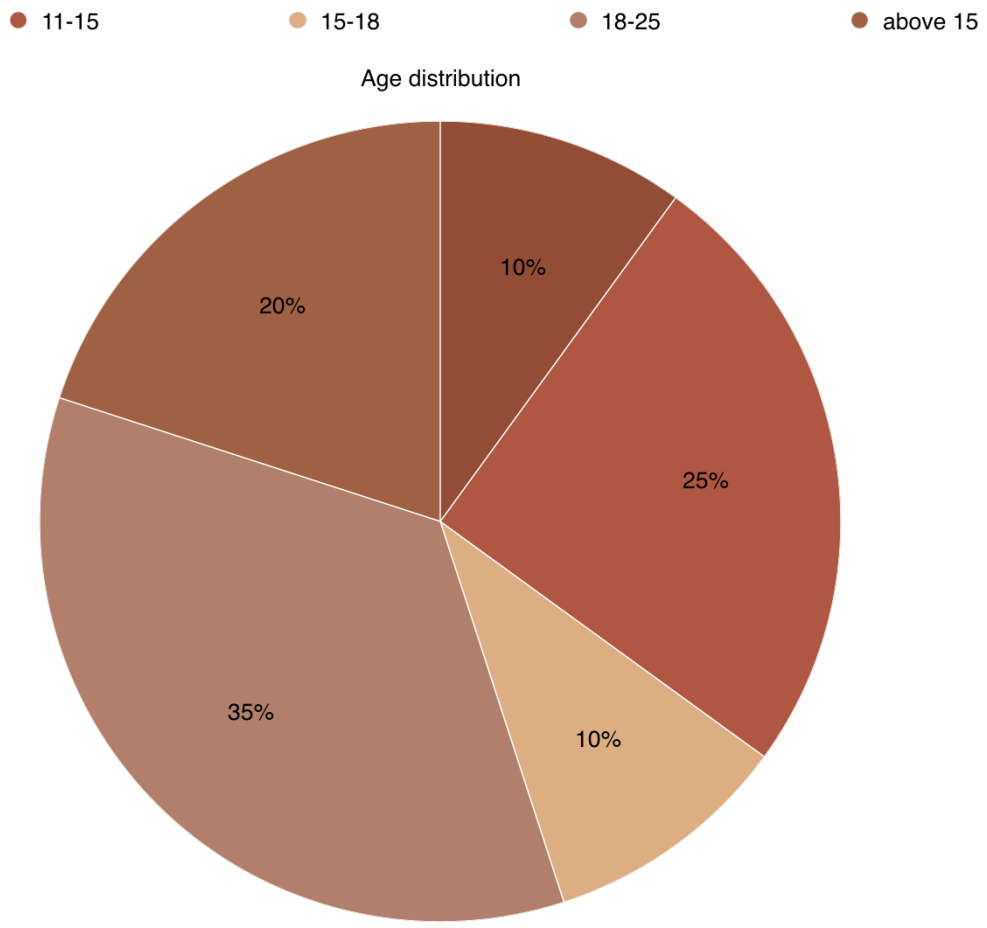


Figure 3.4 Age Distribution

3.1.5 Findings

A lack of visualization during programming learning poses a lot of difficulty for students that are learning how to program. This is because without the visualizations, students are unable to relate and visualize the order of process and execution in the code. So ultimately, when the coded programs do not execute properly, the students cannot identify where and what problem occurred in the code.

Therefore, by developing and using effective learner support mechanisms, students will get a chance to obtain an easier and effective learning environment that optimally identifies and helps handle their shortcomings.

Student can also use visualization tools to seek help in learning programming and understanding the most influencing programming factors that are associated to programming behaviors and play an important part during the execution of a computer programs. Additionally, students can also use the visualization tools as learning models to help them link the new information with their old programming and world knowledge.

Chapter 4
Programming Learning Concept Design

This section presents a theoretical conceptual framework that can be used for implementing an effective programming learning environment in elementary/primary, secondary and higher level educational institutions. It highlights the factors that affected the preliminary design. Choosing a prototype system from existing web applications elaborated in Chapter 2, the conceptual framework 1 shall be implemented and evualted. The conceptual framework 2 is still in the test design stage, because the author was unable to conduct testing due to Covid-19.

4.1. Design Considerations

Based on the review of literature, analysis of the market and findings from the field study, it can be deducted that program learning is a big trend nowadays, especially for teenagers. Learning computer programming can develop technology skills, cognitive skills, logical thinking skills and problem solving skills.

While more and more people start learning computer programming, the difficulties of learning it become more are more obvious as well. Research showed that the current learning methods put too much emphasis on programming skills instead of letting students understand the programming concept. Therefore, students always have a hard time keeping themselves motivated. Furthermore, most of the programming teachers are engineers who are good at programming but lack teaching skills, the learning efficiency is fairly low due to the unmatched needs between students and teachers. Through the field research, the author could observe that although students appreciate online learning methods with various entertainment contents and stories, the low completion rate shows that students

still lack motivation to learn programming and an objective visualized description is needed to help students understand the programming concept and be motivated to learn programming more efficiently. The following factors were considered while designing the conceptual framework for effective programming learning:

1. As most of the devices that connect to the Internet are smart phones, it would be effective to include the technology in the conceptual framework design.

2. The aim would be to design a program that allows the students to understand computer science concepts efficiently and facilitates in memorizing the skills that are taught to them.

3. An interactive system where a student learns to code and instantly sees its value worth in a real life situation must be ensured.

4. The conceptual framework should support constructionism based activities that support collaboration to ensure cognitive development.

5. The conceptual framework must ensure a system of feedback and acknowledgement is in place.

6. The conceptual framework must encompass interactions with learning tools.

7. The conceptual framework must allow learning coding by solving real life questions as it helps a student relate to the importance and practicality of programming.

4.2. Conceptual Framework for Children

To teach programming learning to the younger children, the vital participants in the education environment are the teachers and the students. While both the teacher and students have social presence, the teacher can also design a course, set up activities and assessments. The teachers teach the students some lesson from the course. Then the students are asked to solve a real world problem. The objective of the programming learning is to acquire the cognitive skills. By

working in groups, as the collaborations are constructivist, the students can learn the skill of cognition which is promoted by discussions, sharing of ideas, helping one another, etc.

All the above learning environment requirements can be addressed by the Community of Inquiry model rather successfully (Garrison, Anderson and Archer 1999), presented in section 2.8.4. No essential change is required to the existing model (see Figure 4.1). However, the proposed conceptual framework encompasses the McGill and Volet's (1997) learning knowledge assessment model within the element of teacher presence. The teacher is responsible to carry out the assessments that are designed on the basis of the five categories of learning levels. Once the students undertake the assessments (through class assignments and observations), then, based on the estimated knowledge level that is achieved by the students, the teacher can scaffold instructions, modify the assessment criteria to match the knowledge status of students in a manner that ensures effective learning. The communication medium the conceptual framework for the children includes video, audio (supported by visuals), text-based and the face-to-face communication. For the elementary children, the medium is mostly face-to-face.

Based on the roles of the two participants, some indicators for the three elements in the proposed framework for children are given in Table 4.1.

4.2.1 Implementation

The proposed conceptual framework can be implemented in the same manner as the case study by Papavlasopoulou, Giannakos and Jaccheri (2019) was conducted, described in Section 2.6. For the elementary children, the course content is visual and can be shared in face-to-face meetings through whiteboard or on paper. For the primary and secondary level students, as most of the students have Internet connection, the course sources can be shared through online classes, or through digital media technology i.e. online video lectures, portable files, content management systems (CMS), etc.

In a coding camp, children can be given some visual representations of a few game ideas on paper and an introduction of a programming tool. Once the children have processed the ideas and gotten accustomed to the tool, they can then be divided into groups of twos and requested to create a game using the same tool.

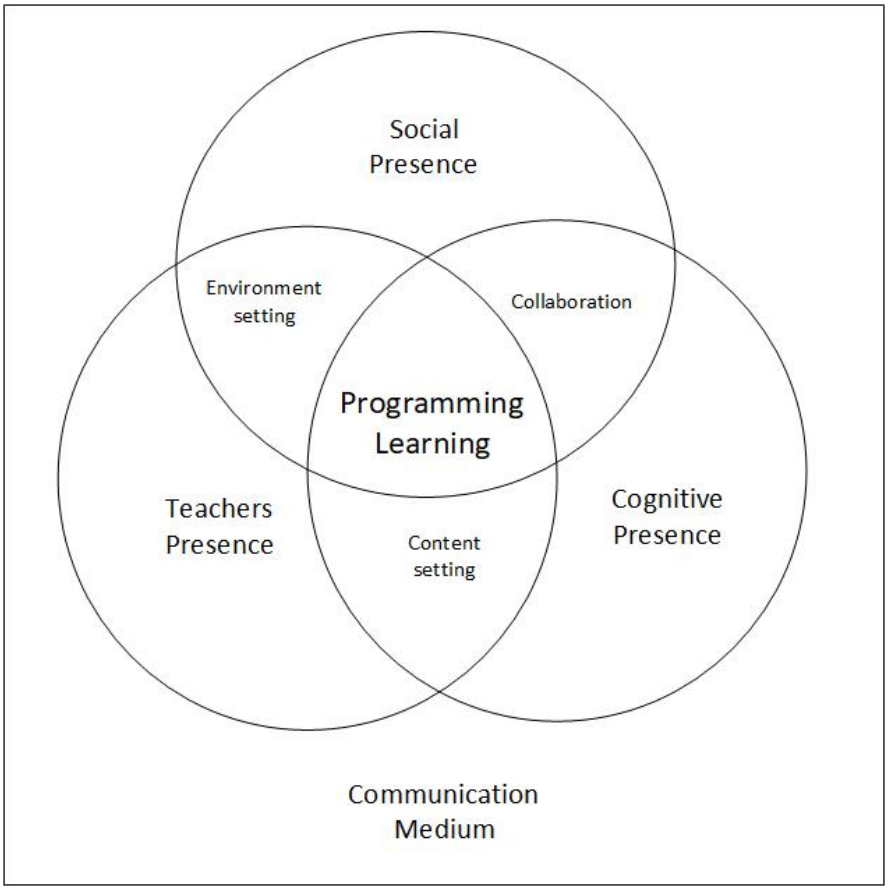


Figure 4.1 Proposed Conceptual Framework for Children

The students get to work in groups and the author predicts that the collaboration could also successfully contribute to develop students' cognitive skills. In case, the collaboration is online (i.e. between the older children of primary and secondary levels), in theory, similar results can still be achieved. The students can hold online or offline meetings between themselves, discuss over the phone, exchange textual messages and after discussion and support from one another, submit the intended task. In case of any queries, the students can send email to the teachers, or post their confusions on the assigned discussion board.

(Source: Garrison, Anderson and Archer 1999)

Table 4.1 Community of Inquiry Indicators

Elements	*Categories*	*Indicators (examples)*
Cognitive Presence	triggering event	recognizing the problem, feeling confused
	exploration	exchange information, discuss doubts
	integration	connect ideas, produce solutions
	resolution	apply new ideas, critically evaluate solutions
Social Presence	emotional expression	Emoticons, real-life descriptions
	open communication	Express freely, recognizing others
	group cohesion	Positive collaboration, help and support
Teaching Presence	instructional management	Structure content, devise assessments based on five knowledge levels, set up discussion topics, make groups
	Building understanding	Share personal values, show agreement, take opinions
	Direct instruction	Directing discussion, answering questions, making corrections, summarizing results/problems

4.2.2 Prototype System

Scratch can serve as the programming learning tool to implement the proposed conceptual framework. It is one of the simplest, yet most effective and most widely used programming learning tools designed for children or novice programming learners. Scratch was used in the case study by Papavlasopoulou, Giannakos and Jaccheri (2019). Although, the students in the case study were between the ages 8 to 12, the same settings can be extended to include children of lesser and higher age range.

Similar to the case study, the younger learners can be provided with paper-based visual game ideas to simulate their minds. The in person presence of the teachers can help in guiding on the spot.

4.2.3 Hallway Test

The author conceptualised a hallway test to see if the conceptual framework for children could be applied to aid the learning of programming skills. The experiment was conducted in May 2019, at Life is Tech, Inc., Tokyo Office / Bootcamp School, which was a programming learning bootcamp for children. The program was called English IT camp. During the time at the camp, students were encouraged to write their own programs on their laptops, while the tutors walked around the desks. In case a student had a question, the tutors would instantly help them face to face. At that time, the author was working at Life is Tech, Tokyo Office as a part-time tutor and market researcher. Life is Tech kindly allowed the author to use a time frame in the Bootcamp schedule to conduct testing for this thesis. The testing time was set at 2 hours and 30 minutes. 10 students and 2 tutors participated. The students got to sit in a separate room to shield them from additional distractions. The author randomly picked students as experiment participants aged between eight and thirteen year old. The participants were randomly assigned to a group.

Objective

The objective was to see if the Proposed Conceptual Framework for Children could be beneficial to enhance learning performance in children, by combining an extensive teacher presence with a visual representation (objects) of a complex programming concept.

Test Design

The participants were confronted with the task of the "Knapsack Problem" to understand the concept of "proof of exhaustion". They had to use programming to solve the problem. Instead of learning the abstract concept first, students in the experimental condition were taught "hands-on" with a backpack and objects of different values. Group 1 (Control) was subject to a a traditional teaching method (textbook), while Group 2 (Experimental) was subject to a teaching method that reflected the Proposed Conceptual Framework for Children (textbook + teacher presence + real world objects).

The Knapsack Problem

The Knapsack Problem starts out with a backpack that holds at maximum 15kg of weight and 10 of size. The student is asked to put in as many objects as possible, whilst not exceeding the limitations of weight and size of the backpack. Initially, the objects have just two variables (size and weight). The concept that is taught is called "proof of exhaustion" - this means that students will try every possible combination in order to eventually achieve the right combination for both weight and size to solve the problem. The students of Group 2 get to experience this complex concept by trial and error in the real world. Later on, when they are confronted with the actual code in Python, it is easier for them to understand and solve the problem because of their hands-on experience.

The objects represented in the Knapsack Problem had the following values for weight and size (See figure 4.2):

Object	A	B	C	D	E	F
Weight	3.5kg	3.0kg	0.6kg	5.0kg	4.0kg	1.0kg
Size(Value)	1	4	3	5	3.5	4

Figure 4.2 Object Values Categorized by Weight and Size

In Group 2, students would be presented with a scenario that they have to go camping, for which they need to pack their belongings. Students would first make solutions for the problem in their minds by touching the objects and calculating. The teacher had a key role in explaining (Extensive Teacher Presence) and in that aided with a social interactive component to transfer knowledge about a complex concept by integrating real world objects. By being confronted with a real-life problem and real objects, the students were able to abstract the problem and their handling of the solution in order to understand the concept. In this way, students were able to benefit from "learning by doing" with an added teacher presence. The students then had to proof that they understood the concept by executing the program.

Experimental Set-Up

Each group was comprised of 5 students. The target programming language to learn was Python.

Group 1 followed only the textbook to write the solution to the Knapsack problem in Python. The students were then asked to write the programs again after lunchtime on the same day. The students were then asked to write the program again after 2 weeks, from their memory.

Group 2 followed the textbook, the teacher instructions and interacted with real life objects. They then were asked to write the solution to the Knapsack problem in Python.

Methodology

The students were then asked to write the program again after lunchtime on the same day. The students were then asked to write the programs again after 2 weeks, from their memory.

At the end of the course, all the students from the two groups were handed questionnaires that asked them three questions:

1. How well were you able to understand the programming?
2. Did you find the course interesting?
3. How satisfied are you with your learning?

4.2.4 Results

In Group 1, only one student could rewrite the complete code contents. Four of the students forgot the contents almost completely, they had to use the text book again to remind themselves. In Group 2, two students could rewrite the contents. The remaining students were able to produce nearly 40% correct output. The response of the survey showed that the students from Group 1 did not enjoy the learning process too much as their satisfaction rate was below 60%. The response of survey for Group 2 students revealed that the students considered the course interesting. The programming was easier to learn and the satisfaction rate was nearly 80%.

4.2.5 Findings

Based on the hallway test, the following observations were made:

1. When the text book based traditional teaching method was used to teach programming, the majority of the students showed difficulty in understanding the basic concept of programming structure.
2. One of the reasons that students often lose interest in learning programming is because there are presented with little to no examples showing practical use of the coding purpose.
3. With traditional teaching methods, the students will not have a visual feedback in reality and find it difficult to understand the practical use of programming.
4. Adding teacher presence made it easier for students to get real-time feedback to better understand the abstract concept.

4.2.6 Improvements

The conceptual framework did indeed show promising results during this hallway test. At this stage in the research phase, no improvements are necessary.

4.3. Learner Centric Conceptual Framework

The design of the Learner Centric Conceptual Framework is shown in Figure 4.3. The author envisioned the Learner Centric Conceptual Framework to have a suitable alternative for educational institutes where the students are independent in their learning and do not rely on the coursework or the teacher to learn the programming knowledge. In this conceptual framework, the teacher does not provide the course contents. Only the assessments are designed by the teacher. The learners learn everything themselves and eventually take the assessments. The students have easy access to mobile computing technology i.e. smart phones, tablets, wearable devices, etc.

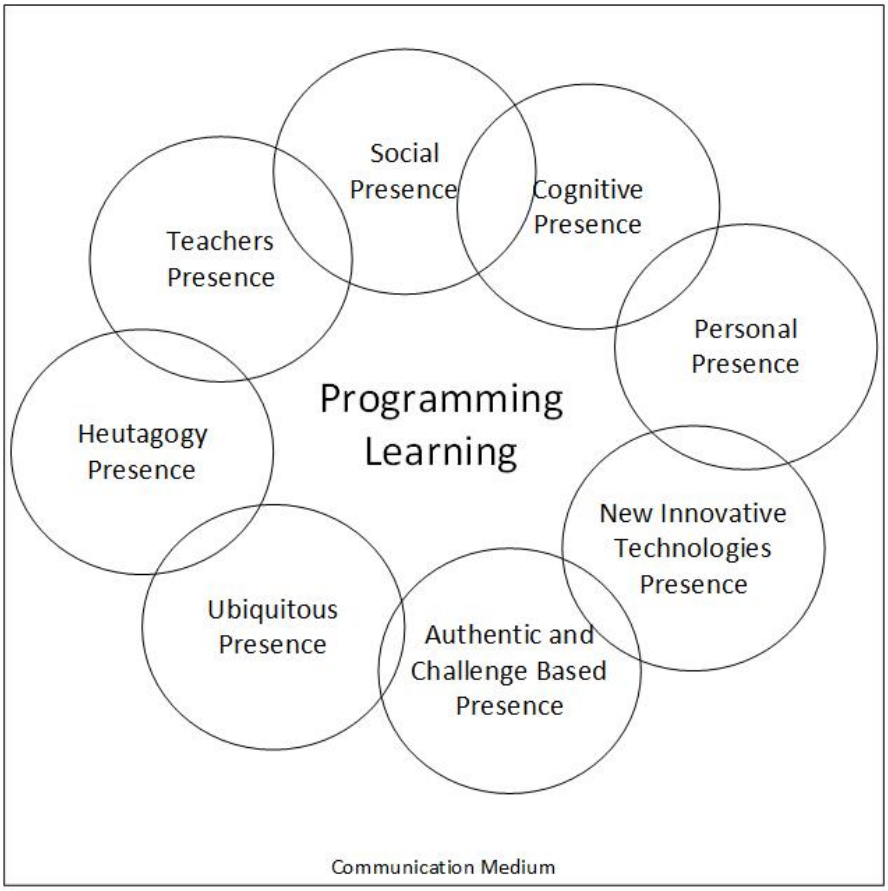


Figure 4.3 Proposed Programming Learning Conceptual Framework (Learner Centric)

The elements added include new innovative technology and all the learning approaches from the mobile learning model suggested by Ossiannilsson and Ioannides (2017). The key participants of the proposed conceptual framework are the students and the teachers. And similar to the Ossiannilsson and Ioannides's model (2017), there is no hierarchical relation between the learners and mentors. Both the teachers and students are producers and consumers of programming knowledge.

Technology presence includes the new innovative technology. Ubiquitous presence represents the learning that the social presence (teacher and student) gain from all sorts of online and offline social and academic interactions. Cognitive

presence represents the critical thinking skill acquired through the collaborative learning in the education environment.

Heutagogy presence represents the learning that is learner-centric rather than teacher- or curriculum-centric. The teacher does not set the course outline, but only defines assessments based on real life authentic problems. The learning is done by the student by exploring and utilizing his/her own sources.

Personal presence shows that the educational system only supports the learning but it does not provide it. The learner chooses what to learn, how to learn, and where to learn from. The educational system only acknowledges the student's learning (e.g. a certification etc.).

In challenge based presence, the students learn by researching and identifying problems themselves and then finding solutions themselves.

In authentic presence, the learner uses real world, open access authentic sources to learn from. Unlike the course or teacher centered education, the student knows about the sources of all the information sources utilized for the learning.

The underlying objective of the conceptual framework is to achieve cognitive, personal, ubiquitous, heutagogy, authentic and challenge-based learning through solving real world problems.

The proposed conceptual framework incorporates Lim's sociocultural perspective model in its design. Similar to Lim's model, the learning through the proposed conceptual framework can occur only if the teachers abandon their conventional teaching methods and also adopt the mobile/online as an active means of teaching and learning. Both the students and teachers are activity systems that are interdependent. Changes in either of the participants (activity systems) are reflected in the entire education system.

The proposed conceptual framework encompasses the McGill and Volet's (1997) learning knowledge assessment model within the element of teacher presence. The teacher is responsible for designing and formation of assessments that are based on the five categories of programming knowledge learning levels. Based on the knowledge level achieved by the students, they themselves can see how they need to modify their learning approach (e.g. visit libraries, take mentoring, learn concepts, language, etc.) to be able to make up for their knowledge shortcomings.

Based on the environmental factors and roles of participants in student pres-

ence, the indicators for the elements of the proposed learner centric conceptual framework are presented in Table 4.2 and 4.3.

Table 4.2 Proposed Learner Centric Conceptual Framework Indicators (1)

Elements	*Categories*	*Indicators (examples)*
Cognitive Presence	triggering event	recognizing the problem, feeling confused
	exploration	exchange information, discuss doubts
	integration	connect ideas, produce solutions
	resolution	apply new ideas, critically evaluate solutions
Social Presence	emotional expression	Emoticons, real-life descriptions
	open communication	Express freely, recognizing others
	group cohesion	Positive collaboration, help and support
Teaching Presence	instructional management	structure specific knowledge level authentic assessments, set up discussion topics, make groups
	Building understanding	Share personal values, show agreement, take opinions
	Direct instruction	Directing discussion, answering questions, making corrections, summarizing results/problems
Technology Presence	Adaptation	Increase or reduction in tool's features set
	Connectivity	Delayed learning due to network unavailability
Personal Presence	Preparedness	Having the required knowledge of some standard equivalence
Ubiquitous Presence	Skill acquisition	Learning a shortcut key combination for a tool function
	Innovative Ideas	Using an easier approach to solve a problem

Table 4.3 Proposed Learner Centric Conceptual Framework Indicators (2)

Heutagogy Presence	Knowledge discovery	Gaining mastery over language syntax
Authentic and Challenge Based Presence	New Problem identification	Finding a solution to an unresolved problem

4.3.1 Implementation

The conceptual framework can be implemented in an educational environment where the teacher does not offer any scope of knowledge to impart in a course. The knowledge and learning activities are planned and executed by the student. The teacher only prepares the assessments where each assessment determines the type of programming knowledge and cognitive skills acquired by the student. The assessments are designed to coincide with the programming concepts of a certain level (e.g. beginner, intermediate, expert) which the students have to acquire on their own.

To implement the conceptual framework, the target educational environment must facilitate and support the use of innovative technologies. For instance, the campus should have wireless network access across all the departments, so that the students can use their smart computing devices such as smart phones, tablets, etc. to continue gain and exchange of their programming knowledge online. The students and teachers can both participate in these activities. The social platforms (discussion forums, CMS, blackboards, etc.) offered by the educational institute must have responsive interface designs. Such interface designs allow the data to be presented in a user friendly way across a wide range of smart devices with different screen sizes. Use of collaboration tools can allow group work with full involvement teachers and well as the students.

4.3.2 Prototype System

The objective of the conceptual framework 2 is to learn cognitive skills alongside the mobile learning skills (ubiquitous, heutagogy, personal, authentic, challenge based learning). A theoretical prototype system (that can be based upon the conceptual framework) can be implemented as an educational event or a boot camp which provides programming learning by designing real life applications.

As the intent of the conceptual framework is to gain the cognitive skills of critical thinking, problem solving and independent learning, this can be achieved by conduction a programming learning activity based on acquiring expertise using the simplest of programming languages. For this, the simple languages of HTML and JavaScript could be used. Anyone who uses the internet uses websites which are the product of these two languages.

4.3.3 Vision for the Test Set-Up

The problem to solve in the boot camp could refer to a daily life activity on the internet e.g. registering to some online website. So, the assignment would be to create an online user registration form with form validation. The teacher would define an assessment criteria that would suggest the level of programming language of the students. Thereafter, the teacher would make student groups. Each group would comprise of to two students. The teacher would then assign the problem to the group. The teacher or student would also create a group on a mutually decided collaborative tool (such as JSFiddle) to work together online.

The students would not be provided with any sources to learn HTML and JavaScript from. They will learn on their own. Having access to library and internet sources, the students would explore the internet and search for the HTML and JavaScript learning tutorials themselves. For instance, they would land the HTML and JavaScript teaching site W3Schools.com on their own and work through the listed exercises and quizzes, attaining a certain amount of personal, heutagogy, authentic, challenge based and ubiquitous learning. Having learnt the required programming concepts and cognition, they would apply their learning and solve the real life web problem using the collaborative tool (JSFiddle).

During the problem solving process, they will share code, links, problems, make

suggestions and discussions, which would add to the learning of each of the students as well as the teacher. Any participant with better knowledge skills can lead towards the solution. The collaboration would increase not only their cognitive learning but also their challenge based learning. As it could happen that while searching for a JavaScript concept, the student is faced with another problem, which he/she bring back to the collaborative tool. The teacher may not have experienced the problem before. So, all the three participants can search online (increase their ubiquitous learning on the way) and then build a solution together, each adding to their knowledge personal knowledge of the programming language. In case a group does not meet the assessment criteria, the students can go back to their learning phase and seek authentic sources to increase their personal and heutagogy learning.

4.3.4 Technology Presence as a Key Element

For the proposed Learner Centric Conceptual Framework, a key component to make this conceptual framework improve the efficiency of programming learning is technology presence, as in the use of certain innovative technology to guide students in the learning process. The author designed a concept for a smart phone application that could be used for students to collaboratively edit their codes. The author chose a HTML text editor that runs on IOS. The application is called 'HTML ME'. IOS was chosen to emphasize the portability and improved accessibility of being able to access your own code on the go. This would increase the social online component for students and give them the feeling of being able to collaborate in real time. The author predicted that by enhancing the accessibility of code, students would naturally gain skills and find the coding more engaging and less confusing.

The designed mock-up screens can be seen in figure 4.4 and 4.5.

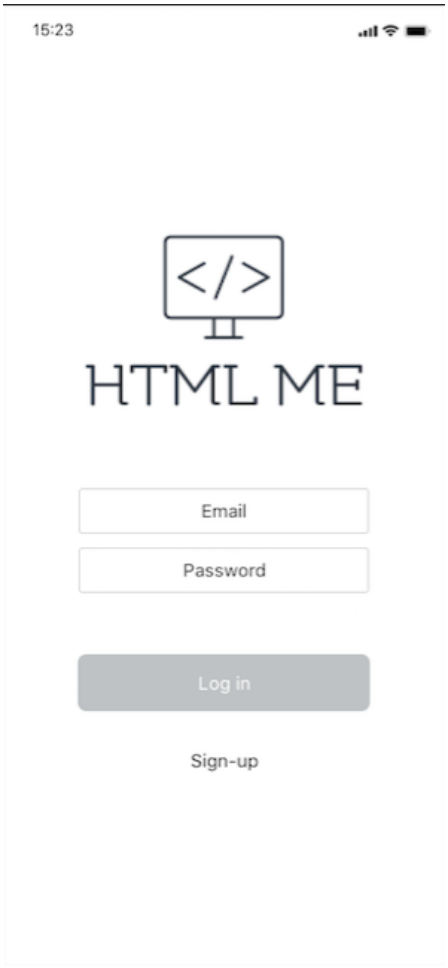


Figure 4.4 Screen Design One of 'HTML ME'

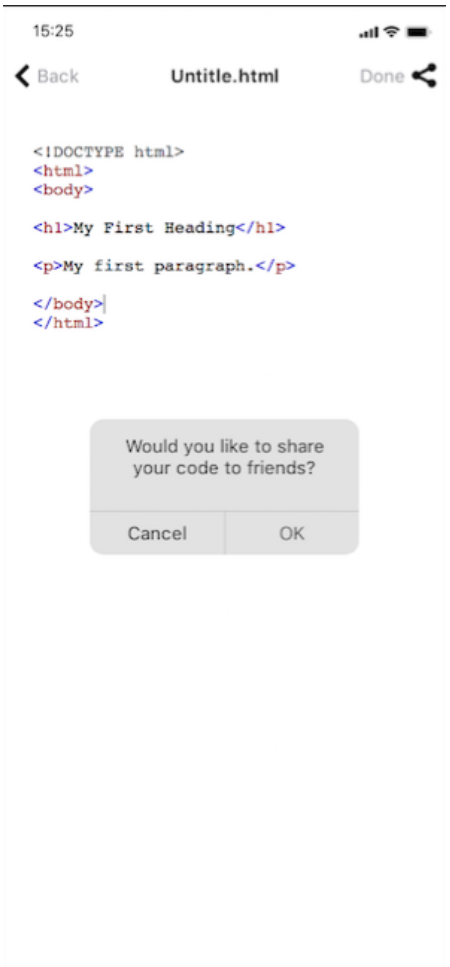


Figure 4.5 Screen Design Two of 'HTML ME'

4.3.5 Testing

Test Design Specifics

10 Japanese University students and 2 programming tutors would be summoned to a classroom. 5 students and 1 tutor would be team A, and the other half would be team B. Both teams would be given a task of making an online registration form by using HTML. For Team A, students would not be provided with any orders or instructions from the tutor but have access to internet and other learning sources and device such as books and smartphones. Students could use the conceptual IOS application 'HTML ME' to edit their code texts. Students could work individually or communicate with each other to work on the task together, and the tutor would give comments after they finish the tasks. Team B would follow the traditional learning method and would be given a paper handout with instructions and taught and guided by a tutor. At the end of the course, all the students from the two groups would be handed a questionnaires that asked them three questions:

1. How well were you able to understand the programming concept?
2. Did you find the course interesting?
3. How satisfied are you with your learning?

4.3.6 Test Limitations

In April 2020, Japan announced a complete lock-down of the country due to the global Covid-19 pandemic. The author was not allowed to leave the house for almost a month. Thereafter, social distancing measures to avoid the spread of the novel Coronavirus had immense implications for daily life in Japan, especially Tokyo. Even though the author had planned to once more collaborate with Life is Tech Inc., make use of the existing ressources at the company, collaborate with other tutors to make the application "HTML ME" and establish a second test at the Life is Tech Inc. Tokyo Office during Bootcamp School, which was supposed to take place again in May, 2020, the bootcamp was cancelled to ensure the safety of both students and tutors. The author was unable to rearrange the test, due to limited preparation time and the restrictions the government imposed on multiple

people being together in the same room for an extended amount of time. The overall economic uncertainty the parties involved were facing further contributed to the difficulties. Even though the author had already conceptualised the test-setup, the test itself was not possible. Therefore, the test design still remains on the conceptual level. The author hopes to conduct the test eventually at a later point in time.

Chapter 5
Conclusion

5.1. Summary

In 2020, Japan joins the list of countries that have programming learning part of its formal primary education. Japan, being one of the most massive economies world-wide, felt the need to reform its education. It is a historic step that needs to be investigated more in the future. When a trend is set by governments, it is usually a serious affair, directing towards a looming crisis that the masses are usually unaware of. In a lot of countries where programming learning was optional, the people took the initiative to learn voluntarily. This thesis work aimed to explore the context behind the global trend of studying programming as part of a formal education curriculum.

After the literature review and market research, the author created two conceptional frameworks to enhance the existing frameworks relevant for the current discourse introduced in Chapter 2. Conceptual framework 1 (Conceptual Framework for Children) was designed as a proposal to improve current frameworks in the related field. Teacher presence was added as a main component. A hallway test was conducted to test Conceptual Framework 1 and showed that the added teacher presence and the real-life interaction with objects was beneficial both for programming learning performance and the overall perceived motivation of students. The Conceptual Framework 2 (Learner Centric Framework) focused on real time collaboration and independent student work. It was designed but left in the test design stage due to difficulties related to Covid-19.

5.2. Related Works and Predictions

Computer Science is the second most sought after subject in 2020 according the QS World University Rankings (OCallaghan 2020). It is safe to say that learning programming will remain a trend for the foreseeable future. The research has confirmed that there is an unwavering high demand of IT workforce globally. So, there is a constant need to cultivate more IT talents to fill in the ever rising technical demand. In the present digital age, computer science has become a global medium of expression between the different industries (Jancheski, 2017; Schindler et al. 2017). So, unlike how it may seem, it is not merely the game testers, web designers, network engineers, software engineers and database managers who are required to learn computer science programming skills. In order to keep up with the fast evolving trends of the present digital world, attaining computer knowledge has become a necessity for every profession. Computer science should be considered as the new applied mathematics (Sterling 2016). Just as knowledge of mathematics get applied in different scientific subjects. Similarly, computational thinking will ultimately find application in all the physical, electronic, mechanical, scientific, financial, agricultural fields that require data or number analysis, result interpretation or a software to run. Such needs will have increased even more in future. Knowing how a computer works will increase the understanding of the workings of many systems of the society.

5.3. Future Works

The following section highlights future research questions and gives new topics and focal points to add to the existing research scope:

- **Testing of Conceptual Framework 2**
  A Collaborative Real-time Coding Platform for Young Adults
  In the future, the author plans to apply the test design for Conceptual Framework 2 introduced in Chapter 4. This will include an experiment with 10 Japanese University students as test participants and 2 programming tutors as test moderators. The students will be divided in Group A and Group B. The test task will be to create an online registration form by

using HTML. Group A will have access to the internet and other learning sources and devices such as books and smartphones. Group B will use the conceptual IOS application design 'HTML ME' introduced in Chapter 4, which will be turned into a real life application by then, to edit their code texts. The test will determine whether a collaborative real-time code platform has a positive influence on programming learning. Data will be collected by questionnaires.

- **Increase student motivation**
  The problem of uninterested or intimidated students can be nipped in the bud by making the introduction to programming easier, engaging and fun (Wilson and Mofat 2010). Future works could also focus more on exploring the factors that lead to an increased motivation in students and develop conceptual frameworks that maximise students engagement to make programming more fun.

- **Continuous teacher training**
  A concern of the present time is the teachers themselves lacking the coding skills. Teachers should make personal efforts to master the block based programming tools themselves so they are confident and able to teach the students (Sterling 2016). Collaborative teacher networks could ensure that teachers are able to continuously grow their own knowledge as they gain teaching experience in class. This could be done in the form of workshops and online platforms to help keep the teachers engaged and up-to-date with current standards in programming education.

- **Collaborative peer teaching**
  Future work could include a test design that lets older students be the teachers. This way, students could practice their obtained knowledge themselves while fortifying the social component and handing down information to their younger peers.

- **The effect of programming learning in other professional settings**
  Coding should be taught as an optional course in all professional areas. The cognitive, problem solving and computational thinking skills associated to

programming are life skills that can find their applications in all disciplines and professions. Many researchers are dedicating their time to further explore how coding can be applied to benefit society overall. To prove its need for medical professions, Morton et al. (2019) argue that doctors with coding skills can practice medicine as well as engage in creating useful, innovative technologies to increase efficiency and adaptability to the present modern medical world. Through a short coding course, medical students with no programming experience were able to build simple, clinical programs within two days of learning. The students acknowledged that the experience gained them an understanding of their own healthcare research and delivery projects. They welcomed the idea of introducing computing skills as an optional course alongside their medical studies. Future work could include the testing of other professions unrelated to programming specialists to see how coding can benefit them in their daily work.

- **Statistical analyses**
  The modern world cannot function without computing technology (Wilson and Moffat 2010). Future works could be more statistical analyses to gain knowledge on how to implement coding into the general skill set of a society, in order to stay competitive on the global market.

- **Testing in 3 different educational systems**
  During the literature review, it was deducted that a single programming learning framework cannot be applied to all types of educational institutes or learning programs. In order to further provide adequate conceptual educational frameworks and respond to the ongoing development of programming learning in the global society, three educational system scenarios should be subject to further testing in regard to this research:
  (1) Teacher centric environment: For the younger elementary, primary and secondary level learners, the mode of communication is generally face-to-face, the courses should be designed and conducted by teachers. More research needs to be done on which devices are best to be used in the classroom and how technology can aid teachers to improve their teaching.
  (2) Course centric environment: Working in groups improves cognitive

skills, while browsing and interacting through mobile devices helps to acquire ubiquitous learning. The mode of communication can be face-to-face, video, audio and textual. More research should be done on how to improve online collaboration programming tools such as JSFiddle, CodePen, etc.

(3) Learner-centric environment: The courses and content are not provided by the teacher. The students learn themselves, for example when obtaining knowledge certifications. The teacher mainly manages learning processes remotely and only supervises the course materials. More research should focus on how course materials can be improved to aid the students in learning self-sufficiently.

References

[1] AFP-JIJI. 2019. “Computer coding is child’s play in China“. *The Japan Times.* Dec 13, 2019. https://www.japantimes.co.jp/news/2019/12/13/business/tech/chinese-kids-young-adults-world-computer-coding-childs-play/

[2] Adán-Coello, Juan Manuel, Tobar, Carlos Faria, Eustáquio Menezes, Wiris Freitas, Ricardo. 2011. ”Forming Groups for Collaborative Learning of Introductory Computer Programming Based on Students’ Programming Skills and Learning Styles”. IJICTE. 7. 34-46. 10.4018/jicte.2011100104.

[3] Alsop, Thomas. 2020. “Share of households with a computer at home worldwide from 2005 to 2019”. *Statista.* Mar 2, 2020. https://www.statista.com/statistics/748551/worldwide-households-with-computer/

[4] Armstrong, Martin. 2018. “Education Struggling to Keep up with Digital Advances“. *Statista.* Feb 22, 2018. https://www.statista.com/chart/13017/education-struggling-to-keep-up-with-digital-advances/

[5] ARSCHOOL. ”Puroguramingu kyōiku no mondaiten. kodomo o nobasu puroguramingu no manabi-kata to wa” [Problems of programming education. What is the way to learn programming to grow children?]. Last modified March 3, 2020. https://arschool.co.jp/blog/archives/2880. (accessed July 25, 2020)

[6] Bergdahl, Nina. & Jalal Nouri & Uno Fors. 2020. “Disengagement, engagement and digital skills in technology-enhanced learning”. *Education and Information Technologies.*

[7] Bers, Marina. 2019. “Coding as another language: a pedagogical approach for teaching computer science in early childhood”. *Journal of Computers in Education*. 6.

[8] Chaffey, Dave. 2019. “Mobile marketing statistics compilation”. *Statista.* 11 Nov, 2019. https://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/

[9] Chenoy, Dilip., Shobha Mishra Ghosh and Shiv Kumar Shukla. 2019. “Skill development for accelerating the manufacturing sector: the role of ‘new-age’ skills for ‘Make in India”’. *International Journal of Training Research* 17: no.1: 112-130.

[10] Clements, Douglas H. 2002. “Computers in Early Childhood Mathematics.” *Contemporary Issues in Early Childhood* 3: 160 - 181.

[11] COETECO. “Puroguramingu kyōiku pōtarusābisu ‘koeteko byGMO’ “2020-nenban kodomo-muke puroguramingu kyōiku kanren sābisukaosumappu” o kōkai shōgakkō de no hisshū-ka ni awa sete sābisu e no san’nyū jigyō-sha mo zōka [”Ecoteco by GMO”, a programming education portal service, ”Chaosmap, a programming education-related service for children, released in 2020”]. Last modified May 8, 2020. https://coeteco.jp/articles/10743 (accessed July 25, 2020)

[12] Coughlan, Sean. 2015. “Computers ’do not improve’ pupil results, says OECD”. *BBC.* Sep 15, 2015. https://www.bbc.com/news/business-34174796

[13] Dressler, Ulf. 2017. “3 Ways to Make Japan’s Workforce Fit for the Future – Hays Survey”. Nov. 2, 2017. https://www.japanindustrynews.com/2017/11/3-ways-make-japans-workforce-fit-future-hays-survey/#lightbox/1/

[14] Ekene, Peter. 2019. “7 JavaScript Playgrounds to Use in 2019”. February 18, 2019. https://scotch.io/tutorials/7-javascript-playgrounds-to-use-in-2019

[15] Ester van Laar, Alexander J.A.M. van Deursen, Jan A.G.M. van Dijk, Jos de Haan. 2020. “Measuring the levels of 21st-century digital skills among

professionals working within the creative industries: A performance-based approach". *Poetics*. https://doi.org/10.1016/j.poetic.2020.101434.

[16] Fessakis, Georgios., Evangelia Gouli and Elisavet Mavroudi. 2012. "Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study", *Computers & Education* 63, no. 2013: 87-97.

[17] Garrison, D. Randy, Terry Anderson and William R. Archer. 1999. "Critical Inquiry in a Text-Based Environment: Computer Conferencing in Higher Education." *Internet and Higher Education* 2: 87-105.

[18] Ghose, Tia. 2016. "The Best Coding Toys for Kids". *Live Science press*. October 19, 2017. http://www.livescience.com/53957-best-coding-apps-and-toys.html

[19] Global Stats. 2020. "Browser Market Share Worldwide – May 2020". June, 2020. https://gs.statcounter.com/

[20] Help Net Security. "Number of connected devices reached 22 billion, where is the revenue?" June 26, 2020. https://www.helpnetsecurity.com/2019/05/23/connected-devices-growth/

[21] Internet World Stats. 2020. "Internet Usage Statistics. Japan". June 26, 2020. https://www.internetworldstats.com/asia/jp.htm

[22] Internet World Stats. 2020. "Internet Usage Statistics. The Internet big picture". June 12, 2020. https://www.internetworldstats.com/stats.htm

[23] ITU. 2019. "Measuring digital development: Facts and figures, 2019". https://www.itu.int/en/ITU-D/Statistics/Documents/facts/FactsFigures2019.pdf

[24] Jancheski, Metodija. 2017. "Improving Teaching and Learning Computer Programming in Schools through Educational Software". *Olympiads in Informatics* 11: 55–75.

[25] Kay, J.S. and J. G. Moss. 2012. "Using robots to teach programming to K-12 teachers," *2012 Frontiers in Education Conference Proceedings*, Seattle, WA, 1-6.

[26] Keen, Rachel. 2011. "The Development of Problem Solving in Young Children: A Critical Cognitive Skill". *Annual Review of Psychology* 62, no.1: 1-21.

[27] Kiss, Gabor. 2013. "Teaching Programming in the Higher Education not for Engineering Students". *Procedia - Social and Behavioral Sciences* 103: 922-927.

[28] Leahy, Denise and Diana Wilson. 2014. "Digital Skills for Employment". In: Passey D., Tatnall A. (eds) "Key Competencies in ICT and Informatics. Implications and Issues for Educational Professionals and Management". ITEM 2014. *IFIP Advances in Information and Communication Technology* 444.

[29] Lichtman, Karen. 2016. "Age and learning environment: Are children implicit second language learners?" *Journal of Child Lang.* Cambridge University Press. 43: 707-730.

[30] Lim, Cher Ping. 2002. "A theoretical framework for the study of ICT in schools: A proposal". *British Journal of Educational Technology.* 33: 411 - 421.

[31] Limelight Networks. 2018. "The State of Online gaming – 2018". *Limelight Networks.* https://www.limelight.com/resources/white-paper/state-of-online-gaming-2018/

[32] Liu, Xinya Kupsky. 2019. "What are the Most Popular College Majors in the World and Why? The Answers May Surprise you!" *Youth Time Magazine.* Oct 1, 2019. https://youth-time.eu/what-are-the-most-popular-college-majors-in-the-world-and-why-the-answers-may-surprise-you/

[33] Lujan, Heidi L, and Stephen E DiCarlo. 2006. "Too much teaching, not enough learning: what is the solution?" *Advances in physiology education* 30, no. 1: 17-22.

[34] McGill, Tanya & Simone Volet. 1997. "A Conceptual Framework for Analyzing Students' Knowledge of Programming". *Journal of Research on Computing in Education* 29, no.3: 276-297.

[35] Morton, Caroline E et al. 2019. “Computer Programming: Should Medical Students Be Learning It?” *JMIR medical education*, 5, no. 1.

[36] Myers, Brad A. and Andrew J. Ko. 2009. ”The Past, Present and Future of Programming in HCI”. *Human-Computer Interaction Consortium, Winter Park, CO.*

[37] Nouri, Jalal, Zhang, Lechen, Mannila, Linda and Norén, Eva. 2020. Development of computational thinking, digital competence and 21st century skills when learning programming in K-9, Education Inquiry, 11:1, 1-17.

[38] OCallaghan, Craig. 2020. “University Subject Rankings: Top Ten Most Viewed Subjects”. March 3, 2020. https://www.topuniversities.com/university-rankings-articles/university-subject-rankings/university-subject-rankings-top-ten-most-viewed-subjects

[39] Ossiannilsson, Ebba and Ioannides, Nicolas. 2017. “Towards a Framework and Learning Methodology for Innovative Mobile Learning: A Theoretical Approach”. *In Proceedings of the 16th World Conference on Mobile and Contextual Learning (mLearn 2017). Association for Computing Machinery,* Article 7, 1–8. New York, NY, USA, https://doi.org/10.1145/3136907.3136929

[40] Papavlasopoulou, Sofia., Michail N.Giannakos and Letizia Jaccheri. 2019. “Exploring children’s learning experience in constructionism-based coding activities through design-based research”. *Computers in Human Behavior.*

[41] Pirzada, Kashan and Fouzia Khan. 2013. “Measuring Relationship between Digital Skills and Employability”. *European Journal of Business and Management* 5, no. 24.

[42] Ross, Kelly Mae. 2016. “Study: Interest in STEM Fuels Growth in Number of International Students in U.S.” *US News.* Nov 14, 2016. https://www.usnews.com/education/best-colleges/articles/2016-11-14/study-interest-in-stem-fuels-growth-in-number-of-international-students-in-us

[43] Rosminah, Siti md derus, siti rosminah Mohamad Ali, Ahmad Zamzuri Ali, Mohamad. 2012. "Difficulties in learning programming: Views of students". 10.13140/2.1.1055.7441.

[44] Saeli, Mara, Jacob Perrenet, Wim Jochems and Bert Zwaneveld. 2011. "Teaching Programming in Secondary School: A Pedagogical Content Knowledge Perspective". *Informatics in Education*. 10: 73-88.

[45] Sano, Atsuko. 2019. "Coding will be mandatory in Japan's primary schools from 2020". *Nikkei Asian Review*. March 27, 2019. https://asia.nikkei.com/Economy/Coding-will-be-mandatory-in-Japan-s-primary-schools-from-2020

[46] Schindler, Laura, Gary Burkholder, Osama Morad and Craig Marsh. 2017. "Computer-based technology and student engagement: a critical review of the literature". *International Journal of Educational Technology in Higher Education* 14.

[47] Schmidt-Crawford, Denise A., Denise Lindstrom and Ann D. Thompson. 2018. "Coding for Teacher Education: A Recurring Theme that Requires Our Attention". *Journal of Digital Learning in Teacher Education* 34, no. 4: 198-200.

[48] Selby, Cynthia. "Four approaches to teaching programming". 2011. *Learning, Media and Technology: a doctoral research conference, United Kingdom* 9.

[49] Schwartz, Bennett L., and Anastasia Efklides. 2012. "Metamemory and Memory Efficiency: Implications for Student Learning." Journal of Applied Research in Memory and Cognition 1 (3): 145–51.

[50] Sellby, Cynthia C. 2011. "Four Approaches to Teaching Programming". *Learning, Media and Technology: a doctoral research conference* 04 July 2011, London

[51] Speranza, Anthony. 2018. "Programming with Scratch – An educator guide". May 1, 2018. https://anthsperanza.com/2018/05/01/scratch-educator-guide/

[52] Statista. 2020. "Number of connected wearable devices worldwide by region from 2015 to 2022". *Statista.* Feb 19, 2020. https://www.statista.com/statistics/490231/wearable-devices-worldwide-by-region/

[53] Sterling, Leon. 2016. Session L. "Coding in the curriculum: Fad or foundational?" [Paper presentation]. *Research Conference 2016 - Improving STEM Learning: What will it take?* https://research.acer.edu.au/research_conference/RC2016/9august/4

[54] Sun, Anna, & Xuifang Chen. 2016. "Online education and its effective practice: A research review". *Journal of Information Technology Education: Research* 15: 157-190.

[55] Tan, Phit Huan Yee, Ting Ling, Siew-Woei. 2009. "Learning Difficulties in Programming Courses: Undergraduates' Perspective and Perception". 42 - 46. 10.1109/ICCTD.2009.188.

[56] Uzunboylu, Hüseyin, Kinik E. and Kanbul, Sezer. 2017. "An analysis of countries which have integrated coding into their curricula and the content analysis of academic studies on coding training in Turkey". *TEM Journal* 6: 783-791.

[57] Wilson, Amanda and Moffat, David. 2010. "Evaluating Scratch to introduce younger schoolchildren to programming". *Proceedings of the 22nd Annual Workshop of the Psychology of Programming Interest Group*, Cambridge, UK.

[58] Yata, Chikahiko, Tadashi Ohtani and Masataka Isobe. 2020. "Conceptual framework of STEM based on Japanese subject principles". *IJ STEM Ed* **7,** no. 12.

[59] Yurt, Özlem and Nılgün Cevher-Kalburan. 2011. "Early childhood teachers' thoughts and practices about the use of computers in early childhood education". *Procedia Computer Science* 3: 1562-1570.