ISSN 2278-3091
Volume 8, No.1.6, 2019
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse6281.62019.pdf
https://doi.org/10.30534/ijatcse/2019/6281.62019

Learn Programming Framework for Malaysian Preschoolers

Laura P. Jack[1], Norazlina Khamis[2], Carolyn Salimun[3], Dinna M. Nizam [4], Zaidatol Haslinda[5], Aslina Baharum[6]
[1]Universiti Malaysia Sabah, Malaysia, laura@ums.edu.my
[2]Universiti Malaysia Sabah, Malaysia, norazlina@ums.edu.my
[3]Universiti Malaysia Sabah, Malaysia, carolyn@ums.edu.my
[4]Universiti Malaysia Sabah, Malaysia, dinna@ums.edu.my
[5]Universiti Malaysia Sabah, Malaysia, linda.sani@ums.edu.my
[6]Universiti Malaysia Sabah, Malaysia, aslina@ums.edu.my

ABSTRACT

Learning computer programming is a good way to develop Computational Thinking, but many students assumed that computer programming is a complicated subject, which leads toward their disinterest in computer science. We believe that this assumption can be overcome if the students were introduced to the concept of programming earlier during preschool years. The best programming method for children to learn programming is the tangible programming method but current Malaysian preschool syllabus does not contain a framework for preschoolers to learn tangible programming. This paper introduces the developmentally-appropriate Learn Programming Framework (LPF) for preschoolers to learn programming concepts. The structured syllabus in LPF involves a series of controlled-play activities to be used with the novel tangible board game. Once the framework has been successfully implemented, extensive evaluation of LPF will be done to test its effectiveness in teaching children about programming.

Key words : Computational Thinking, framework, learn programming, preschoolers.

1. INTRODUCTION

As the awareness of STEM education importance grow, schools have been integrating more technology and engineering modules into school curricula. In the Malaysia education context, the introduction of the Malaysia Education Blueprint (MEB) 2013-2025 saw the introduction of Information Technology and Communication (TMK) to students as young as 7 years old, through modules that let students use TMK in their learning process. One of the targets of the MEB is to prepare students with Higher Order Thinking Skills (HOTS) and Computational Thinking (CT). Computational Thinking (CT) is defined as the thought processes involved in understanding problems and formulating the solutions in ways that can be executed effectively by information-processing devices. CT is an important skill for a student to have in the current technological age, so they are prepared to face the technologically-filled world as an aware and critical student and eventually as an adult.

CT can be taught via many subjects, and one of the best methods to teach CT is through learning computer programming [1]. Computer programming allows whomever that learns it to become a problem solver and creator of technology. However, one of the issues with many students is their assumption that computer programming is complicated, which leads to their disinterest in computer science. We believe that this might be caused by the rather late introduction to computer programming for these students. While many developed countries such as the United Kingdom make it compulsory for primary school students (from 5 years old onwards) to start learning computer programming [2], the KSSR introduces the programming subject only to 12-year-olds.

We believe that children should be exposed to computer programming earlier, beginning with teaching programming concepts to preschoolers. Findings in early education research (covered in next section) points to the feasibilities of introducing computer programming to young children, and extensive research has shown tangible programming as the suitable method for children to use to learn programming. However, the Malaysian National Preschool Curriculum Standard (KSPK) [3] does not have any framework for young children to learn programming concepts using tangible method.

Our research is also motivated by i) our interest to bridge the digital education divide between urban and rural schools in Malaysia, and ii) make teaching programming introduction doable by any teacher. This paper introduces the Learn Programming Framework (LPF), a developmentally appropriate framework for Malaysian preschoolers to learn programming concepts using offline tangible method. The

design of the framework has been completed and nearing its implementation phase. Once it is successfully implemented, extensive evaluation of LPF will be done. The evaluation of LPF is designed to test the effectiveness of the LPF in making children learn about programming concepts as hypothesized in this paper.

2. LEARNING TO PROGRAM

During the early phase of computer usage in education, the focus has always been on using computer as a tool to help in the teaching and learning process. However, Papert's [4] groundbreaking research has provide insights and inspirations for researchers in educational technology to explore usage of computer beyond its purpose as a tool to help in reading, writing and mathematics. Papert's work also offered insights into how computers can help active learning and creation of knowledge for children, through its programmable and dynamic properties. Exploring computer through programming view requires children to use their intellectual resources while involved in processes such as designing, producing and using to create systems and structures [5]. Programming a computer allows a child to become creator and controller of technology, an empowering experience for a student.

Unfortunately, it became more apparent that existing interface for programming is not suitable for young children [6]. The early form of programming language was only in text format, meaning a programmer must be at least able to read and write, and understand syntax, if they want to program. It is unreasonable to expect these young children to cope with the symbolic systems required to successfully program a computer, when the child is just beginning to learn about natural language and its symbolizations [5]. The introduction of tangible programming goes way back in the mid-70s, when Radia Perlman designed programming interface with novel input devices [7]. Her work allows a child to physically show the computer what to do and make the computer memorize the sequence for playback. Fast forward to 1993, after many varieties of physical programming, Suzuki and Kato [8] developed AlgoBlock system to study collaborative problem solving. The system consists of large computational building blocks that children used to navigate a 'virtual submarine through a maze, on the computer screen'. AlgoBlocks and Tangible Programming Bricks by McNerney [9] in 2000 both offered platforms to explore tangible programming language, and conceptually a success as suitable programming tool for children, but the external appearance of the parts did not appeal to the children. Electronic Blocks by Wyeth and Purchase [10] is the first system that provides engaging and developmentally appropriate tool of programming for young children. Tangible user interfaces allow children to interact with physical objects, and let children write programs by assembling physical objects, making them the more appropriate tool for younger children [11], [12]. There are many other research [1], [13], [22]–[24], [14]–[21] conducted that further strengthen the position of tangible programming as the ideal programming language for young children.

Doubts have been raised on whether young children are capable of learning the concept of programming, mainly regarding whether children are developmentally capable to understand the complicated programming concepts such as abstract thinking and algorithms [25], [26]. Research by [27]–[31] showed that children as young as four can understand the basic concepts of computer programming and can build/program simple robotic projects. A study by Sullivan and Bers [31] show that beginning in pre-kindergarten, children were able to master basic robotics and programming skills.

2.1 Rural schools, teacher abilities and offline programming tool

Previous tangible programming research involves gadgets with sensitive and expensive parts/electronics. Such characteristics made these gadgets inaccessible to many of its target user. A gadget that is not robust enough to survive the rough handling of children will not be of much use. A high price tag also limits the number of gadgets that can be provided or bought for schools, and as such would hamper effort to provide equal opportunities for children from different levels of school to learn computer programming. Research by Ahmad [32]–[34] pointed that digital disparities between urban dan rural school are in terms of funding and access to technology. In some rural schools, the necessary resources needed to learn computer programming is considered a rare privilege due to unreliable power source, difficulties in getting technical support, and poor internet connection. Teacher competencies in computer programming is also a factor that hinders the success of computer programming education in Malaysian schools, but there is no empirical data to support this claim though. [32]–[34] did mention teacher lack of ICT skills as a factor that influence the success of ICT education in rural schools, but there is no specific mention of the teachers' computer programming skills.

Considering rural schools and teacher abilities, we figured that it would be useful to have programming tool that is robust and cheap. In 2013, programmer Dan Shapiro created a board game called Robot Turtle that teaches kids (age range of 3-8 years old) core computer programming concepts. Shapiro claimed that many children who played Robot Turtle showed an understanding of the core coding concepts and ability to apply them while playing the game, though there have been no empirical studies done yet to support that claim. Several other offline board games claim the same ability to teach young children concepts of computer programming (eg. ThinkFun Gravity Maze Marble Run Logic Game, and Thinkfun Code Master Programming Logic Game). Adopting the idea of a coding board game, we planned on the creation of

a learning programming framework that uses an offline coding board game as its programming tool.

2. DEVELOPING LEARN PROGRAMMING FRAMEWORK

An existing learning computer programming framework for young children that influenced our research a lot is the Tangiblek Robotics Program in Tufts University, Boston. Their work created novel human-computer interaction techniques to support learning with technology in early elementary school. They created a tangible-graphical hybrid programming language specifically for young children called CHERP and used it as the tool in the implementation of their TangibleK curriculum for learning programming and robotics. Results from their experiments showed that the children were able to learn and apply many aspects of robotics, programming, and computational thinking. Our work will differ with theirs from the theoretical aspect behind our framework, and the usage of offline tangible programming tool.

3.1 Theoretical framework of LPF

The theoretical approach used for designing the framework, which includes teaching the curriculum in several intervention sessions in pre-schools integrate elements from Lev Vygotsky's social constructivism theory, that emphasized a child's interaction with adults and capable peers, and usage of the cognitive tool will help a child learn skills otherwise unattainable without assistance. Vygotsky's social constructivism is rooted in Jean Piaget's (1954) constructivism theory, that says a child actively builds knowledge through experience. The framework also adopts the 'powerful ideas' from Papert's [4] constructionism. 'Powerful ideas' refers to fundamental concepts of a domain that are epistemological and personally useful, interconnected with other disciplines, and can be traced to a child's learned knowledge that has become instinctive [4], [35]. The curriculum for learning computer programming in the framework is composed of powerful ideas from the domain of computer science, where these ideas will be introduced in a context in which their use allows young children to solve problems. The seven ideas are Computer Programming, Command Sequences and Control Flow, Loops, Sensors, parameters, branches, and subroutines.

3.2 LPF design considerations

The Learn Programming Framework incorporates two types of design criteria: the framework design must be developmentally appropriate for young children, and at the same time it enables application of programming concepts. To satisfy the requirements of developmentally appropriate practices for early childhood education, the design of the curriculum activities in the framework must ensure:

1. Activities are open-ended and discovery-oriented. A child must be allowed to be actively involved in the learning process;
2. Interaction with the researcher is only to facilitate the play activity
3. Experiences must involve active manipulation of real tangible objects
4. The minimum requirement for entry-level knowledge and experience
5. Activities must be able to cater to children's varied skill and ability levels

To enable the application of programming concepts, the activities will use traditional games with non-tech tools. The design criteria for these traditional games are:

6. To have defined set of game rules
7. Gives control to the player
8. Be algorithmic – so that we can teach algorithms to the children
9. Be repetitive – to teach loops
10. To have conditions and trigger actions
11. To have subroutines
12. Could allow the player to choose the best option out of several solutions

These 12 design criteria become the foundation of the framework. The non-tech tools mentioned most likely will be like a board game, but no further explanations can be given about the details of the activities and traditional games due to both being still in the development process.

3.3 Learn Programming Framework

The LPF have been designed to be used as a fun learning programming session. It contains modules for a curriculum that spans 8 weeks long. Each module (labelled as Activity *n*) in Figure 1, is made up of activities to teach children each of the concepts in computer programming.

4. RESEARCH METHODOLOGY

This project uses a quasi-experimental research design with a pretest and post-test. Participants for the experiment will consist of preschoolers (aged 5 to 6 years old).

4.1 Demographic survey

During the implementation of the learn programming concept framework, we will survey the target participants in the selected pre-school. The survey will collect necessary demographic data (date of birth, gender, etc.) and data on the target participants' hours of experience with learning programming games/apps, experience with any robotics platforms, and expertise with programmable robots. The primary purpose of this survey is to assign each participant into groups based on their experience with learning computer programming.

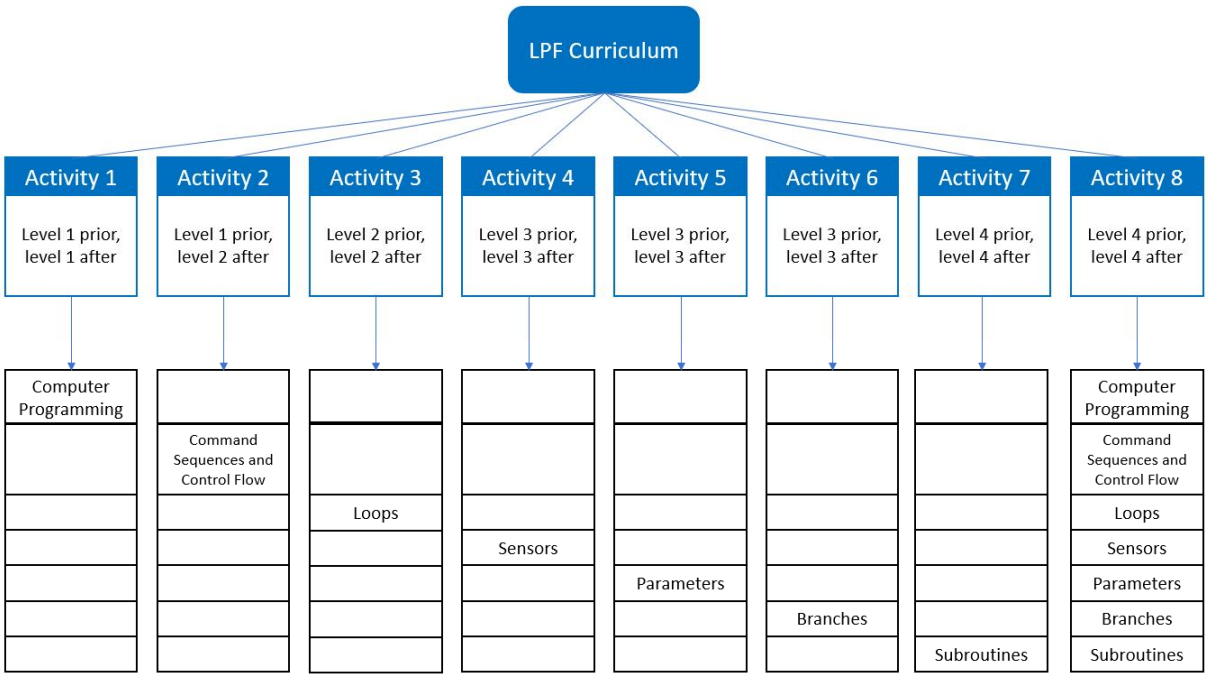

Figure 1: Learn Programming Framework

4.2 Experiments

Learning computer programming is one of the ways to develop CT. The purpose of this research is to develop a developmentally-appropriate framework for learning computer programming concepts. We could use a child's Computational Thinking skills as an indicator of the child's understanding of programming concepts. Participants will be divided into 2 groups (test group, and the control group).

A. Pre-test.

Both the test group and the control group will be tested on their current CT skills using an assessment tool adapted from TACTIC-KIBO research [30]. The test will be conducted in the selected pre-school, with only the participating child, teacher, and researcher present. Because of the prior survey conducted and the consequent group allocation based on survey results, we hypothesized that both groups will have similar CT level assessment results.

B. Treatment Phase.

The Treatment Phase will consist of intervention sessions conducted at the participating pre-schools. We plan to conduct 1 session every week (classroom-based setting) for the total duration of 8 weeks. In each of the intervention sessions, we will employ an offline play method using simple games and coding toys (board game) to teach one powerful concept of computer programming to the children. There are seven powerful concepts that we need to teach, hence the 8 weeks required duration for this Treatment Phase. We target that by the end of every intervention session, all the children in the Test group will have learned some, if not all, programming concepts.

C. Post-test.

After completion of the Treatment Phase, both the test group and control group will be tested again on their Computational Thinking skills using the adapted TACTIC-KIBO assessment tool. Similar to Pre-test, the test will be conducted in the pre-school, with only the participating child, teacher, and researcher present.

We hypothesized that due to the intervention of the Treatment Phase, the Test group will achieve higher scores in the assessment, compared to the Control group. If we can achieve this, it shows that the Learning Programming Framework is an effective framework for young children to learn computer programming concepts.

5. IMPLICATION OF THE STUDIES

The main contribution of this research will be the Learn Programming Framework. This framework will enable Malaysian preschools to introduce essential learning of computer programming concepts to its young students, without the burden of having to prepare expensive computers and the need to provide teachers trained in computer programming.

Looking at the bigger picture, it is also our target to close the gap of computer education between urban and rural schools. An offline or non-tech method that does not require expensive gadgets or qualified trainers will hopefully make learning computer programming easier or even achievable in rural areas.

The process of developing the framework also allows us to create an adaptation of the TACTIC-KIBO [14], which is a tool for assessing Computational Thinking in young learners. TACTIC-KIBO involves the usage of a robotic toy to engage the participants, while the adapted version will involve the usage of offline games. An offline assessment tool will be a valuable contribution to the computer education field.

REFERENCES

1. D. Wang, L. Zhang, C. Xu, H. Hu, and Y. Qi. **A tangible embedded programming system to convey event-handling concept**, in *TEI 2016 - Proc. 10th Anniv Conf. Tangible Embed Embodied Interact*, 2016, pp. 133–40. https://doi.org/10.1145/2839462.2839491
2. **Statutory framework for the early years foundation stage Setting the standards for learning, development and care for children from birth to five**, [Internet]. 2017 [cited 2019 Oct 31]. Available from: www.gov.uk/ofsted.
3. Kementerian Pendidikan Malaysia. **Kurikulum Standard Prasekolah Kebangsaan**, Kementerian Pendidikan Malaysia. 2016.
4. S. Papert. **Mindstorms: children, computers, and powerful ideas**, Basic Books, Inc., 1980.
5. P. Wyeth, and H.C. Purchase. **Programming without a computer: A new interface for children under eight**, in *Proc. - 1st Australasian User Interface Conf.*, 2000, pp. 141–8.
6. A. Cockburn, and A. Bryant. **Leogo: An equal opportunity user interface for programming**, *J. Vis. Lang. Comput.*, 1997, 8(5–6), pp. 601–19. https://doi.org/10.1006/jvlc.1997.0152
7. R. Perlman. **Using Computer Technology to Provide a Creative Learning Environment for Preschool Children**, 1976.
8. H. Suzuki, and H. Kato. **AlgoBlock: a tangible programming language, a tool for collaborative learning**, in *Proc. 4th Eur. Logo Conf.*, 1993, pp. 297–303.
9. T.S. McNerney. **Tangible Programming Bricks : An approach to making programming accessible to everyone**, Ph.D. dissertation, Massachusetts Institute of Technology, 1999.
10. P. Wyeth, and H.C. Purchase. **Tangible Programming Elements for Young Children**, in *CHI '02 Ext. Abstr. Human factors Comput. Syst.*, ACM, 2002, pp. 774-775.
11. T. Sapounidis, and S. Demetriadis. **Evaluating children performance with graphical and tangible robot programming tools,** *Pers. and Ubiquitous Comput*ing, vol 19, 2015, pp. 225-237. https://doi.org/10.1007/s00779-014-0774-3
12. S. Papavlasopoulou, M.N. Giannakos, and L. Jaccheri. **Reviewing the affordances of tangible programming languages: Implications for design and practice**, *IEEE Global Engineering Education Conference, EDUCON*, 2017, pp. 1811-1816.
13. H.S. Raffle, A.J. Parkes, and H. Ishii. **Topobo: A constructive assembly system with kinetic memory**. in *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 2004, pp. 647-654.
14. M. Gordon, E. Ackermann, and C. Breazeal. **Social Robot Toolkit: Tangible Programming for Young Children**, in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts*, ACM, 2015, pp. 67-68.
15. K. Tada, and J. Tanaka. **Tangible Programming Environment Using Paper Cards as Command Objects**, *Procedia Manufacturing*, 2015, Vol.3, pp. 5482–5489.
16. Q. Jin, D. Wang, X. Deng, N. Zheng, and S. Chiu. **AR-Maze: A Tangible Programming Tool for Children Based on AR Technology**, in *Proceedings of the 17th ACM Conference on Interaction Design and Children*, ACM, 2018, pp. 611-616. https://doi.org/10.1145/3202185.3210784
17. M.S. Horn, and Jacob RJKK. **Designing Tangible Programming Languages for Classroom Use**, in *Proceedings of the 1st international conference on Tangible and embedded interaction*, ACM, 2007, pp. 159-162.
18. D. Wang, C. Zhang, and H. Wang. **T-Maze: A Tangible Programming Tool for Children**, in *IDC2011 The 10th International Proceedings on Interaction Design and Children*, Ann Arbor, USA, ACM, 2011, pp. 127-135.
19. A. Sipitakiat, and N. Nusen. **Robo-Blocks: designing debugging abilities in a tangible programming system for early primary school children**, in *Proceedings of the 11th International Conference on Interaction Design and Children*, ACM, 2012, pp. 98-105.
20. D.Y. Kwon, H.S. Kim, J.K. Shim, and W.G. Lee. **Algorithmic Bricks: A tangible robot programming tool for elementary school students**, *IEEE Transactions on Education*, vol. 55, no. 4, 2012, pp. 474-479.
21. F. Scharf, Thomas Winkler, Claudia Hahn, Christian Wolters, and Michael Herczeg. **Tangicons 3.0: An Educational Non-Competitive Collaborative Game**, in *The 11th International Proceedings on Interaction Design and Children*, Bremen, Germany, 2012, pp. 144-151. https://doi.org/10.1145/2307096.2307113
22. K. Chawla, M. Chiou, A. Sandes, and P. Blikstein. **Dr. Wagon: A "stretchable" toolkit for tangible computer programming**. in *IDC'13 Proceedings of the 12th ACM Conference on Interaction Design and Children*, ACM, 2013, pp. 561-564.
23. D. Wang, Y. Zhang, and S. Chen. **E-block: A tangible programming tool with graphical blocks**. *Mathematical Problems in Engineering*, 2013.

24. P.M.Y. Lam, C.K.H. Lai, Y.T. Choi, B.J. Huxtable, and J.R. Castro, A. Hawryshkewich, and C. Neustaedter. **Loopo: a tangible programming game for kids**, in *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, ACM, 2014, pp. 179-180.
25. J. Piaget. **Part I: Cognitive development in children: Piaget development and learning**, *Journal of Research in Science Teaching*. Vol. 2. 1964, pp. 176–86.
26. E. Thelen, G. Schoner, C. Scheier, and L.B. Smith. **The dynamics of embodiment: A field theory of infant perseverative reaching**, *Behaviorial and Brain Sciences*, Vol. 24, 2001, pp. 1-34.
27. M.U. Bers. **Blocks to Robots Learning with Technology in the Early Childhood Classroom**, *Teachers College Press*, 2008.
28. E. Kazakoff, and M. Bers. **Programming in a Robotics Context in the Kindergarten Classroom: The Impact on Sequencing Skills**, *Journal of Educational Multimedia and Hypermedia*, Vol. 21, 2012, pp. 371-397.
29. L.P. Flannery, E. Kazakoff, P. Bontá, B. Silverman, M.U. Bers, and M. Resnick. **Designing ScratchJr: Support for Early Childhood Learning Through Computer Programming**, in *Proceedings of the 12th International Conference on Interaction Design and Children*, ACM, 2013, pp. 1-10. https://doi.org/10.1145/2485760.2485785
30. E. Relkin. **Assessing young children's computational thinking abilities**, Ph.D. dissertation, Tufts University, 2018.
31. A. Sullivan, and M.U. Bers. **Robotics in the early childhood classroom: learning outcomes from an 8-week robotics curriculum in pre-kindergarten through second grade**. Int. Journal of Technology and Design Education, Vol. 26, 2016, pp. 3-20;
32. Khairani A.Z. **Assessing Urban and Rural Teachers' Competencies in STEM Integrated Education in Malaysia**, *MATEC Web of Conferences*, vol. 87, EDP Sciences, 2017. https://doi.org/10.1051/matecconf/20178704004
33. Siti Norazlina Kamisan. **Halangan terhadap penggunaan Komputer dan ICT di dalam pengajaran dan pembelajaran (P&P) di kalangan guru di sekolah menengah kebangsaan luar bandar di daerah Kulai Jaya, Johor**, Ph.D. dissertation, Universiti Teknologi Malaysia, 2008.
34. Maizura Mohd Yunus, and Mahyuddin Arsat. **Satu Kajian Cabaran Guru di Sekolah Pedalaman Daerah Muar, Johor**, Ph.D. dissertation, Universiti Teknologi Malaysia, 2008.
35. M. Bers, I. Ponte, K. Juelich, A. Vera, and J. Schenker. **Teachers as Designers: Integrating Robotics in Early Childhood Education**. *Information technology in childhood education annual*, Vol. 1, 2002, 123-145.