MyJICT
Malaysian Journal of Information and Communication Technology
EISSN 0127-7812

Vol. 9, Issue 1, pp. 11-16 | JUNE 2024
DOI: https://doi.org/10.53840/myjict9-1-174

ANALYSIS OF SELF-REGULATED LEARNING ELEMENTS IN INTRODUCTORY COMPUTER PROGRAMMING

[i*]Siti Noor Ahmad, [ii]Juzlinda Mohd Ghazali, [iii]Farhana Abdullah Asuhaimi, [iv]Helyawati Baharuddin

Faculty of Creative Multimedia and Computing
Universiti Islam Selangor (UIS)

*(Corresponding author) e-mail: sitinoor@uis.edu.my

Article history:
Submission date: 24 March 2024
Received in revised form: 27 April 2024
Acceptance date: 27 June 2024
Available online: 30 June 2024

Keywords:
Self-Regulated Learning, introductory computer programming, programming

Funding:
This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Competing interest:
The author(s) have declared that no competing interests exist.

Cite as:
Ahmad, S. N., Mohd Ghazali, J., Abdullah Asuhaimi, F., & Baharuddin, H. (2024). Analysis of Self-Regulated Learning Elements in Introductory Computer Programming. Malaysian Journal of Information and Communication Technology (MyJICT), 9(1), 11-16.
https://doi.org/10.53840/myjict9-1-174

ABSTRACT

This paper presents analysis for self-regulated learning (SRL) element in the context of introductory computer programming courses. SRL is a critical skill for students to develop as it enables them to take control of their learning process and become more effective learners. The proposed elements are based on insights from educators and experts in the field. The findings of this analysis provide valuable guidance for educators looking to integrate SRL principles and elements into introductory computer programming courses. Additionally, the analysis highlights the importance of considering the diverse needs and backgrounds of students in the design of SRL interventions.

Introduction

Introductory Computer Programming is introduced as early as first semester by introducing the problem-solving concept, pre-programming tools, algorithm and pseudo code. The subject continues to semester 2 by introducing the concept in programming, variables, datatypes, operations, operators (math, logical, Boolean) and basic control structure. Learning programming continues in the other semester for more complex concept like object-oriented programming, mobile programming, web programming and visual programming.

The platform or the medium of learning has changed from traditional face to face to fully online (blended, synchronous or asynchronous platform). Students are not only learning programming during lecture and lab session but they are also required to learn by themselves as they need to plan and organize their study plan with minimal supervision from their lecturer. Learning programming is not easy and learning it online do not make it better.

Self-regulated learning (SRL) is a core conceptual framework that cover both cognitive and non-cognitive aspect of learners involving their cognitive, motivational and emotion. According to (Loksa & Ko, 2016), Self-regulation is defined as the ability to identify and organise your thought and planning and to assess them toward a specific goal. Self-Regulated Learning referring to the ability of a person to understand and control his own learning environment. Self-regulation abilities include goal setting, self-monitoring, self-direction, and self-reinforcement.

Self-regulated computer programming learning is crucial and beneficial to students because it enables students to be more independence (Zimmerman, B. J. (2000)), personalized (Boekaerts, M., & Corno, L., 2005), adaptable (Tsai, C. C. (2005) to learning environment. This paper will present the elements analysis of self-regulated computer learning based on literature review and experts' insight to be included in the proposed Self-Regulated Computer Programming Learning Model.

Literature Review

Self-regulated learning (SRL) empowering students to take charge of their learning process. According to (Puustinen & Pulkkinen, 2001), there are five popular Self-Regulated Learning theories; Boekaerts, Borkowski, Pintrich, Winne and Zimmerman. The definition of SRL among these theories are mostly similar. (Puustinen & Pulkkinen, 2001) has divided the definition of SRL into two group based on the theory. Boekaerts, Pintrich, and Zimmerman define SRL as a goal-oriented process. They emphasise the constructive or self-generated nature of SRL and agree that monitoring, regulating and controlling one's own learning includes cognitive but also motivational, emotional and social factors. Borkowski and Winne define SRL as a metacognitively governed process aimed at adapting the use of cognitive tactics and strategies to tasks. They do not include goal orientations in their definitions, they both assume self-regulated learners (or good information users, as Borkowski puts it) to be intrinsically motivated and goal-oriented.

Zimmerman has developed three models of SRL that evolve from Triadic Analysis to Cyclical Phases. Zimmerman's SRL model covers Metacognition, Motivational and Emotion factors. These factors are represented in three phases, which are forethought phase, performance and self-reflection. In forethought phase, students will plan, set the goal, analyze the task analysis and how to reach self- motivation in their learning strategies. Students will execute the task as their plan and monitor their progress using self-control strategy in order to maintain their engagement in term of cognitive and motivation to finish the task in performance phase. Lastly, the self-reflection phase is where the students assess their performance and generate self-reaction that can positively or negatively influence how they can improve their performance in solving next task. SRL by Zimmerman is cyclical phases, the nature of self- regulation, self-reflection further influences forethought processes and improve students' performance and become expert in the next task.

Self-Regulated Learning helps understanding the programming and improving the student' performance and motivation although not many works focusing the role of self-regulation in

programming and problem solving. According to (R. Garcia et al., 2018), SRL approaches have been used in early programming research on mental models and learning techniques, the relationship between SRL and the performance of introductory programming, particularly the employment of metacognitive strategies, and find that students who perform well on programming assignments also frequently utilise metacognitive strategies.

The challenges faced by students especially at the introductory level are the ability to understand the problem and to be able to solve the problem either the problem is given by their lecturer or otherwise. Problem solving skill is important and necessary for the student to be able to identify, tackle and solve the problem and transform it into algorithmic solution. Lack with problem solving skill will turn down the learner's motivation in learning programming.

Student's motivation and interest are key factors in learning programming successfully (Cheah, 2020; Medeiros et al., 2019; Rahim et al., 2018; Salleh et al., 2013b). The weak foundation of problem-solving skills and reasoning leads the difficulties in interpreting the problem. Reading, understanding, and learning from other's people coding examples is preferred. It is also depending on knowledge base regarding the general, domain and related programming languages. Students' engagement, perception, confident, poor time management, minimal work habit is contributing factors impacting the student's motivation.

Previous works have shown that there are several issues in programming tried to be solved with Self- Regulated learning. Four categories of studies based on issues on SRL and programming have been identified. The first category is SRL approach on student motivation and performance in problem solving (Loksa & Ko, 2016), programming (Cheng et al., 2019; M. B. Garcia, 2021; Lishinski & Yadav, 2021; Song et al., 2021; Tsai, 2019). The second category is the SRL adoption or approach on learning technology such as e-learning (Song et al., 2021), flipped classroom (Çakiroğlu & Öztürk, 2017), blended learning (Cigdem, 2020) and digitized technologies (R. Garcia et al., 2018). The third category on SRL relationship on students' coding (Castellanos et al., 2017; Song et al., 2021) and the fourth category proposing SRL Framework to help novices transition from entry-level to advance computer programmer (expert)(Loksa & Ko, 2016; Pedrosa et al., 2016).

Most of SRL in programming studies based on Bandura, Zimmerman and Pintrich SRL model to investigate of relationship student performance and motivation. The empirical study by (Cheng et al., 2019) and (Castellanos et al., 2017) using the MSLQ instrument by Pintrich. (Loksa & Ko, 2016) and (Pedrosa et al., 2016) on the other hand not only investigate the relationship between SRL and programming problem solving but proposing the framework the role of self-regulation in problem solving and programming.

Methodology

In this paper we perform an interview the educators using semi structured interview to identify the self- regulated learning elements for introductory computer programming. Semi-structured interview protocol is used to gather the information regarding self-regulated learning for introductory and to identify the requirements of Self-Regulated learning elements that are suitable for students in learning programming from lecturer's perspective. Three (3) experts/lecturers who has more that 5-years' experience will be interviewed. Criteria of experts are as follows based on (A.Aziz & Siraj, 2015; Ariffin et al., 2020):

i. Expert must in the field of programming learning and teaching. In this study, experts are referring to lecturers from Higher Learning Institute (IHL).
ii. Lecturers who have at least Master Degree in related Information Technology or Computer Science.
iii. Lectures who have at least five years' experience in teaching programming at introductory and intermediate level

Result and Discussion

Three lecturers from Higher Learning Institute were selected and the detailed of respondents R1, R2, and R3 namely are as Table 1.

Table 1: Lecturers from Higher Learning Institute

Participant	Gender	Subject Taught	Teaching Experience
R1	Female	C++, Problem Solving	24 years
R2	Male	C++, Java, Problem Solving	10 years
R3	Female	Java, VB	16 years

The key themes were constructed based on the interview analysis:

i. Strategies for effective planning in programming tasks
ii. Self-Control and Strategies in programming
iii. Time management in programming
iv. Self-Reflection in programming

Finding for theme 1: Strategies for effective planning in programming task

Respondent R1, R2 and R3 agreed that planning and goal setting is important elements in programming task.

R1" …. *Help students to learn the basic concept of programming. input, process, output and also coding….*"

R2 "*.. due to time constraint, self-regulated learning helps students to plan and learn according to their pace and initial knowledge based on topic with guidance from lecturers.*"

R3 "*Planning involves understanding problems, choosing strategies like flow charts or pseudocode, and gathering resources from documentation and online aids. Goal setting requires specific, measurable, attainable, relevant, and time-bound objectives, breaking down problems into smaller tasks, and monitoring progress regularly*."
Goal setting and planning is important in programming task and it will be difficult for novice or beginners, R2 suggested there should be guidance from lecturer to help the students. R3 suggested to use pre-programming tools like flowchart, pseudocode dan break the problem into smaller tasks.

Finding for theme 2: Self-control and Strategies in programming

Student can become effective programmers when they have self-control and strategies in programming. Students will refer to internet, friends and lecturers in order to complete programming task. Strategies in programming are study group, divide and conquer, discussion with peers and lecturers.
R1 *" … when students are given tasks, they will form a study group, ask friend and lecturers to guide them…. they will share their solution and explain it to others…*"
R2 *"… coding, normally they refer to friend first, then they will ask lecturers…"*

However, not all students will seek help by asking their peers and lecturers based on R3 respond.
R3 *"Yes and No,Not all my student employs the strategies discuss problems, and seek help from classmates and lecturers when producing code.Based on the skills and abilities of the students, whether they like to explore or not."*

Finding for theme 3: Time Management in programming

Students are having problem to complete the coding within the set time and re-evaluate their coding. This statement is supported by R1, R2 and R3.
R1 *"..lab test, if they understand they will answer… they will wait others, if their friend can answer the question, they will follow.."*
R2".. *some will do faster, some are not… have to push.."*

R3 " *There are students who are able to complete the given code within the allotted time, but there are also students who are unable to finish it within the given time frame."*

Finding for theme 4: Self-Reflection in programming

Self-reflection in programming is necessary, not only to complete the coding but also reflection of relationship with God, friends and lecturers and supported by R1, R2 and R3.
R1 ".. *students nowadays always find others fault…"*

R2 " … *budak-budak surau, excellence ..when it comes to relationship with GOD , friends and lectures*…"

R3 "*Yes of course, reflecting on programming problems in the context of relationships with God, friends, and lecturers encourages a holistic approach to learning. It fosters ethical behaviour, enhances collaboration, and promotes a respectful and productive educational environment*."

Conclusion

The analysis of the interviews highlights several critical elements in the context of programming education: effective planning, self-control and strategies, time management, and self-reflection.

Effective Planning in Programming Tasks:
All respondents emphasized the importance of planning and goal setting in programming. Effective planning involves understanding problems, selecting appropriate strategies such as flowcharts and pseudocode, and breaking down tasks into manageable components. Goal setting needs to be specific, measurable, attainable, relevant, and time-bound. While these skills are essential, novice programmers often require guidance from lecturers to develop and refine them.

Self-Control and Strategies in Programming:
Self-control and strategic approaches are vital for becoming effective programmers. Students often rely on a mix of resources including the internet, peers, and lecturers. Common strategies include forming study groups, employing the divide-and-conquer method, and engaging in discussions. However, the use of these strategies varies among students, with some being more proactive than others in seeking help and collaborating.

Time Management in Programming:
Time management emerged as a significant challenge, with students often struggling to complete coding tasks within set time frames. The disparity in students' abilities to manage time and complete tasks suggests a need for tailored support and perhaps more practice-oriented teaching methods to enhance their efficiency and effectiveness.

Self-Reflection in Programming:
Self-reflection is considered crucial not only for improving coding skills but also for fostering a well- rounded development that includes ethical behaviour and collaboration. Reflecting on one's relationship with God, friends, and lecturers was highlighted as a component of a holistic learning approach, promoting a respectful and productive educational environment.

In conclusion, the findings underscore the need for a multifaceted approach to programming education that includes structured planning, strategic learning, effective time management, and continuous self- reflection. Providing students with guidance and fostering a supportive environment will help them develop the necessary skills and mindset to excel in programming.

Acknowledgement

The research presented in this paper is part of my PhD thesis at Universiti Islam Selangor (UIS) I would like to thank my supervisor, Dr Juzlinda Mohd Ghazali, for her invaluable guidance and support throughout this research. I also acknowledge the contributions of my colleagues in the FMKK, whose feedback and assistance have been crucial in the development of this work.

References

Çakiroğlu, Ü., & Öztürk, M. (2017). Flipped classroom with problem-based activities: Exploring self- regulated learning in a programming language course. *Educational Technology and Society*, *20*(1), 337– 349.

Castellanos, H., Restrepo-Calle, F., González, F. A., & Echeverry, J. J. R. (2017). Understanding the relationships between self-regulated learning and students source code in a computer programming course. *Proceedings - Frontiers in Education Conference, FIE*, *2017-Octob*(April 2020), 1–9. https://doi.org/10.1109/FIE.2017.8190467

Cheng, G., Poon, L. K. M., Lau, W. W. F., & Zhou, R. C. (2019). *Exploring the Relationship Between Self-Regulated Learning Strategies And Computer Programming Achievement In Higher Education*. *5*, 67–74. https://doi.org/10.17501/24246700.2019.5108

Cigdem, H. (2020). How Does Self-Regulation Affect Computer-Programming Achievement in a Blended Context? *Contemporary Educational Technology*, *6*(1), 19–37. https://doi.org/10.30935/cedtech/6137

Garcia, M. B. (2021). Cooperative learning in computer programming: A quasi-experimental evaluation of Jigsaw teaching strategy with novice programmers. *Education and Information Technologies*, *26*(4). https://doi.org/10.1007/s10639-021-10502-6

Garcia, R., Falkner, K., & Vivian, R. (2018). Systematic literature review: Self-Regulated Learning strategies using e-learning tools for Computer Science. *Computers and Education*, *123*, 150–163. https://doi.org/10.1016/j.compedu.2018.05.006

Lishinski, A., & Yadav, A. (2021). Self-evaluation Interventions: Impact on Self-efficacy. *ACM Transactions on Computing Education*, *21*(3).

Loksa, D., & Ko, A. J. (2016). The role of self-regulation in programming problem solving process and success. *ICER 2016 - Proceedings of the 2016 ACM Conference on International Computing Education Research*. https://doi.org/10.1145/2960310.2960334

Pedrosa, D., Cravino, J., Morgado, L., Barreira, C., Nunes, R. R., Martins, P., & Paredes, H. (2016). Simprogramming: the Development of an Integrated Teaching Approach for Computer Programming in Higher Education. *INTED2016 Proceedings*, *1*(April), 7162–7172. https://doi.org/10.21125/inted.2016.0699

Puustinen, M., & Pulkkinen, L. (2001). Models of Self-regulated Learning: A review. *Scandinavian Journal of Educational Research*, *45*(3), 269–286. https://doi.org/10.1080/00313830120074206

Song, D., Hong, H., & Oh, E. Y. (2021). Applying computational analysis of novice learners' computer programming patterns to reveal self-regulated learning, computational thinking, and learning performance. *Computers in Human Behavior*, *120*, 106746.

Tsai, C. Y. (2019). Improving students' understanding of basic programming concepts through visual programming language: The role of self-efficacy. *Computers in Human Behavior*, *95*, 224–232. https://doi.org/10.1016/j.chb.2018.11.038