Journal of Infrastructure, Policy and Development 2024, 8(16), 9252.
https://doi.org/10.24294/jipd9252

Article

Self-efficacy as students' motivation factor in learning programming

Rajermani Thinakaran[1,*], Suriayati Chuprat[2], Vijayakumar Varadarajan[1], Madina Yussubaliyeva[3], Muazzam Maqsood[4]

[1] Faculty of Data Science and Information Technology, INTI International University, Nilai 71800, Malaysia
[2] Razak Faculty of Technology and Informatics, Universiti Teknologi Malaysia, Kuala Lumpur 54100, Malaysia
[3] International Research Projects Office, D. Serikbayev East Kazakhstan State Technical University, Oskemen 070000, Kazakhstan
[4] Department of Computer Science, COMSATS University Islamabad, Attock Campus, Attock, Pakistan
*** Corresponding author:** Rajermani Thinakaran, rajermani.thina@newinti.edu.my

CITATION

Thinakaran R, Chuprat S, Varadarajan V, et al. (2024). Self-efficacy as students' motivation factor in learning programming. Journal of Infrastructure, Policy and Development. 8(16): 9252. https://doi.org/10.24294/jipd9252

ARTICLE INFO

Received: 9 January 2024
Accepted: 4 March 2024
Available online: 18 December 2024

Abstract: In learning, one of the fundamental motivating factors is self-efficacy. Therefore, it is crucial to understand the level of students' self-efficacy in learning programming. This article presents a quantitative study on undergraduate students' perceived programming self-efficacy. 110 undergraduate computing students took part in this survey to assess programming self-efficacy. Before being given to the respondents, the survey instrument, which included a 28-item self-efficacy assessment and 30 multiple-choice programming questions, was pilot-tested. The survey instrument had a reliability of 0.755. The study results show that the students' self-efficacy was low when they solved complex programming tasks independently. However, they felt confident when there was an assistant to guide them through the tasks. From this study, it could be concluded that self-efficacy is an essential achievement component in programming courses and can avoid education dropouts.

Keywords: motivation; programming; programming difficulties; self-efficacy

1. Introduction

Within Revolution 4.0 (IR 4.0), the demands for IT employees have been continuously on the rise across the globe and in Malaysia (Bujang et al., 2020). With business organization shifting their focus to adapt to the technological disruptions brought forward by the global Corona Virus Disease (COVID-19) (Mustaffa et al., 2022), the demands for competent programmers to cater to these changes become more eminent. Therefore, to meet the organizations' needs, programming courses have become an important part of today's curriculum.

On the other hand, programming subjects seem to drop their attractiveness due to the nature of the subject. In computing education, the subject has been classified as one of the seven grand challenges (McGettrick et al., 2005). Students find this subject hard to understand and the passing rate is low not only worldwide (Zhang et al., 2022) but also in Malaysia (Thinakaran et al., 2023).

In Malaysia, there are many studies have been carried out in teaching and learning programming. Based on former studies from the year 2010 to 2022, the most common difficulties faced by students are understanding the programming concepts and applying them to a specific problem (Aris, 2011; Derus and Ali, 2012; Gila et al., 2017; Hui and Umar, 2010; Ismail et al., 2010; Osman et al., 2012; Rahmat et al., 2012; Rosminah and Ali, 2017; Rum and Ismail, 2017; Sabjan et al., 2021; Saharudin et al., 2018; Yacob and Saman, 2012; Yong and Tiong, 2022). This problem occurs because these students lack logical thinking skills (Hooshyar et al., 2016; Jono et al., 2015;

Rahman et al., 2013) such as how to write programming language syntax, how to use a program development environment and also how to find bugs in the written program (Cheah, 2020; Derus and Ali, 2012).

The above outcomes, disclose that two aspects cause difficulty in learning programming, which are cognitive skills and basic skills. The conscious is where students feel demotivated to learn programming which eventually leads to a number of students dropping out from the course. These claims are further supported by Bakar et al. (2020) and also Kanaparan et al. (2019).

In the learning process, motivation plays an important role since motivation is considered one of the key factors that leads to excellent student performance (Kanaparan et al., 2019; Thinakaran et al., 2017). Even though there are many research has been carried out on the causes of high failing rates in programming subjects, however only a few researches could be found in the literature on students' programming motivation in the Malaysian context (Bakar et al., 2020; Kanaparan et al., 2019). Therefore, this study asks the following question: "To what extent are students motivated to learn programming?"

2. Literature review

Motivation is a fundamental component of the learning process. It is understood to be an enabler for learning programming and academic achievement (Kanaparan et al., 2019). Enthusiasm to learn programming is affected by a set of motivating factors (Thinakaran et al., 2018). The motivating factors can be classified into two forms which are extrinsic and intrinsic. Extrinsic factors comprise clear direction, reward and recognition, punishment, social pressure and competition (Rayan and Deci, 2000). While intrinsic factors are associated with the student's values, interests, needs and attitudes (Rayan and Deci, 2000). De Vicente (2023) states that motivation can be subject to student characteristics, traits and permanent. Trait consists of control, challenge, independence and fantasy characteristics, while permanent characteristics consist of confidence, sensory interest, cognitive interest, effort and satisfaction.

Another essential factor that drives students' performance is their self-efficacy. It has long been identified that self-efficacy is one of the important motivational factors for learning (Zimmerman, 2000). The theory of self-efficacy was presented by Albert Bandura (1986), a Canadian psychologist. Bandura (1986) characterized self-efficacy as "people's judgments of their capabilities to organize and execute courses of action required to attain designated types of performances." Bandura (1986) also believes that self-efficacy is influenced in four different ways which are i) the selection of activities in which a student participates; ii) the level of student effort in performing an activity; iii) persistence in facing difficulties to complete an activity; iv) student performance in an activity.

Because of the subject complexity, student self-efficacy as intrinsic motivation is very important. Thus, it is vital to know the level of student self-efficacy in learning programming. Kanaparan et al., (2019) demonstrated that students' beliefs about their self-efficacy in programming could be used to determine how well the students are doing in their programming subject. Additionally, it has long been known that self-efficacy is a fundamental motivation for learning (Zimmerman, 1999).

Self-efficacy

Students tend to have some self-efficacy beliefs where they hold some opinions about their ability about the specific learning domain. They also hold some outcome expectations about the success or failure of specific actions. For example, a student might approach a programming question with the view that: "I tend to find programming difficult (self-efficacy belief) so I am likely to need a lot of help to complete the task (outcome expectation)". These views tend to act as a frame of reference that leads students' thinking, feelings and actions in a learning situation (Thinakaran et al., 2023).

Through literature on self-efficacy, many scholars have demonstrated their relationship in learning programming (Abdunabi et al., 2019; Kanaparan et al., 2019; Kittur, 2020). While Kanaparan et al. (2019) state that individuals with a high perception of self-efficacy in a particular situation strive to accomplish a task. They do not easily give up and are persistent and patient. In addition, from 800 meta-analyses by Hattie(2008), the researcher has identified self-efficacy as the strongest predictor of educational achievement.

In conclusion, self-efficacy is an important phenomenon that needs to be focused on. Based on the empirical findings gathered by previous researchers, this study therefore used self-efficacy to evaluate students' motivation level in programming.

3. Methods

For this study, a deductive approach has been adopted also known as the top-down approach. The approach starts with a theory followed by a hypothesis, data collection and finally confirmation. **Figure 1** illustrates the deductive approach used in this study in which the theory used is Self-efficacy as student motivation. Meanwhile, the hypothesis used in this study is: "If students' self-efficacy as students' motivation is low in programming subject, then the passing rate in programming subject is also low". In this study, the students' motivation level was identified by using an adapted self-efficacy questionnaire. Finally, a confirmation of the hypothesis was also made.

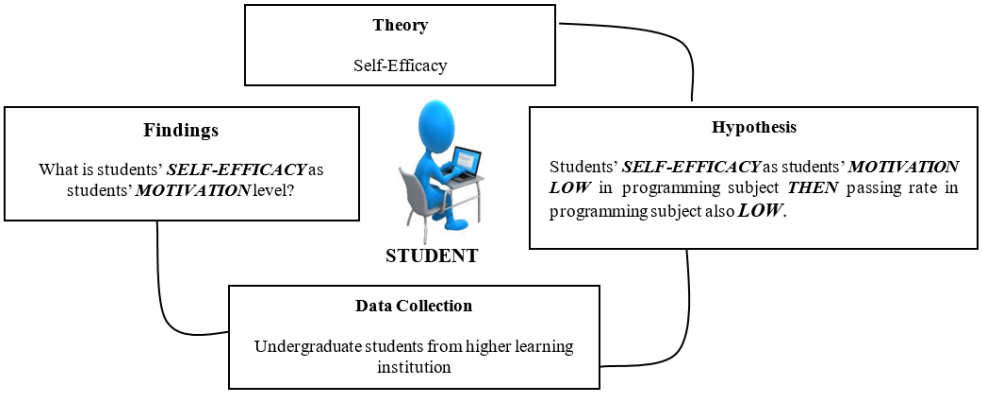

Figure 1. Deductive approach of this study.

3.1. Questionnaire design

The study was conducted using an adapted Self-Efficacy Scale for computer programming questionnaire which has been developed by Ramalingam and Wiedenbeck (1998). This questionnaire was used previously by many researchers (Atiq, 2017; Owolabi and Adegoke, 2014; Özyurt and Özyurt, 2015; Saleh et al., 2018) in their respective studies.

In this study, the questionnaire was further developed in three parts. The first part requires the respondents to provide their demographic information. The second part consisted of 32 statements (items) which ascertained the respondents perceive self-efficacy as students' motivation level. The strength of self-efficacy is measured by the responses on a 5-point Likert-type scale ranging from 1 (not confident at all) to 5 (absolutely confident). The third part is a standardized test consisting of 30 objective questions to assess respondents' knowledge of Java programming.

The standardized test was designed to justify students' self-efficacy results that would be used to correlate with students' standardized test results. For example, if students' self-efficacy is low, the total score on the standardized test should be poor. If students' motivation levels show average or high but their standardized test shows low, it can be concluded that students did not answer genuinely.

3.2. Questionnaire validity and reliability

The designed questionnaire was validated using the content validation criteria (**Table 1**) and face validation criteria (**Table 2**). The validation criteria would be able to inform about how well the individual item in the tool corresponds to the concept of what is being examined (Lun et al.,2022; Mohammadzadeh et al., 2017).

For content validation, under normal circumstances, the use of three experts is acceptable. However, a panel of 5–10 experts is preferred while the use of > 10 experts is probably unnecessary (Lynn, 1986). The designed questionnaire was sent to 9 reviewers, 2 industry experts (programmers in Java Language) and 7 lecturers (teaching OOP using Java).

Table 1. Content validity index—questionnaire.

Content Validation Criteria	IE1	IE2	L1	L2	L3	L4	L5	L6	L7	CVR
(1) The objective of the instrument is stated clearly.	5	5	5	5	5	5	5	5	5	1.00
(2) The format is appropriate.	5	5	5	5	5	5	5	5	5	1.00
(3) The meaning of every item is clear.	4	4	3	3	3	3	3	3	3	−0.56
(4) The instruction is clear.	5	5	5	5	5	5	5	5	5	1.00
(5) The questions fully represent Object Oriented Programming using Java.	2	3	1	1	1	1	1	1	1	−1.00
(6) The answer for each item is appropriate.	4	4	3	3	3	3	3	3	3	−0.56
CVI										0.15

IE—Industry Expert, L—Lecturer.

In this study, the Content Validity Ratio (CVR) and Content Validity Index (CVI) were used to quantify the validity of the questionnaire. CVR is an item statistic that is useful in the rejection or retention of individual items. On the other hand, CVI is the mean for CVR for all the items included in the final instrument (Gilbert and Prion,

2016). The numeric value of CVR is determined by the Lawshe Table (Lawshe, 1975). **Table 1** shows the results of CVR by nine experts according to each validation criteria. CVI is calculated by the sum of CVR value which is divided by the number of items (Gilbert and Prion, 2016). The numeric value of CVI is 0.80 or greater is acceptable (Gilbert and Prion, 2016) to indicate an excellent questionnaire.

According to the experts' review (**Table 1**), the CVR for criteria 3, 5 and 6 was found to be < 7.8 and CVI is 0.15. This indicates that the questionnaire needs further improvements. To further improve the questionnaire, industry experts and lecturers provided some comments to improve the appropriateness of the questionnaire designed. Some of the comments are:

3.2.1. For part B

- Change the word "could" to "can".
- Remove statements 2, 4, 5, 6, 7 8 and 16.
- Rephrase statement 11 to "I can write a long and complex Java program to solve any given problem".
- Rephrase statement 12 to "I can organize and design my program in an object-oriented manner".
- Rephrase statement 18 to "I can understand a long, complex multi-file program.".
- Rephrase statement 32 to "I can write a program that someone else could comprehend and add features to it at a later date.
- Statement 14, break it into 4 different statements: I can identify the objects in the problem domain.; I can define the objects in the problem domain.; I can write object declaration in the problem domain.; and I can use the object declaration in the problem domain.
- Rearrange the statement: 13,12, 1, 3, 9, 10, 11, 14, 15, 17, 18, 20, 21, 22, 23, 24, 25 and 26.

3.2.2. For part C

- Revise the questions to the OOP introduction chapter containing object, method and classes. These comments were given by 4 particular lecturers (L2, L4, L6 and L7) due to their experiences. The majority of students struggle to understand basic concepts such as methods, classes and objects. This leads the students cannot understand the following chapters which apply the basic concepts and consequently the students fail the subject.
- The experts and lecturers agree on Java language due to the nature of the language as object-oriented.

The questionnaire has thus been revised based on the comments given by industry experts and lecturers. The revised questionnaire is still comprised of three parts. However, the second part was revised to 28 questions (items). In the third part, 30 objective questions were revised and the questions only focused on object, method and class in OOP to assess students' knowledge. The revised questionnaire was validated again by the experts and lecturers. All the items were rated as "Strongly Agreed" and the CVR and CVI which yielded a score of 1.00 show the high validity of the questionnaire according to the study's objectives.

The questionnaire was face validated using a face validation form by 29 undergraduate computing students from a private university according to the

validation criteria shown in **Table 2** representing the item statistics rating on face validation of each item of the instrument. All the items in the instrument were relevant to the content of the study due to the reliability coefficient yielded $\alpha = 0.755$ through Cronbach's alpha (Cronbach, 1951). Reliability refers to how well a test measures what it should, while Cronbach's alpha tests were used to see if the survey instrument is reliable. The revised and validated questionnaire is available in the following google link https://drive.google.com/file/d/1x4CPSakHaHZJoiWeuz_DeaSJbwxW-UHF/view?usp=sharing.

Table 2. Face validation item statistics—questionnaire.

Face Validation Criteria	Mean	Std. deviation	*N*
(1) The instruction is clear.	4.52	0.574	29
(2) The wording of the questions is easy to understand.	4.31	0.604	29
(3) The flow of the questions is easy to follow.	4.52	0.634	29
(4) The meaning of every item is clear.	4.24	0.689	29
(5) The format is appropriate.	4.31	0.660	29

3.3. Study group and data collection

This study utilized a cluster sampling technique to determine the participants involved to obtain the required data. The cluster sampling technique has been frequently applied for data collection (Jackson, 2011). A cluster involves a group of participants that represents the population identified and included in the sample. In this study, the cluster study groups were undergraduate students who were enrolled in OOP subjects using Java language. The author approached 13 private colleges and universities by sending an official letter to obtain consent to conduct the survey.

Only 8 private colleges and universities have responded to participate in this study. The data collection activities are based on the adapted questionnaire using Google Forms through Microsoft Teams. A 15-minute briefing section was conducted to explain the purpose and how the survey will be conducted. Students were asked to complete the questionnaire during online class time to secure a high response rate. A token of RM5.00 was given to each participant as a token of appreciation.

4. Results and discussion

The distribution of students representing the respective universities and colleges who took part in the survey is summarized in **Figure 2**. A total of 116 students participated in the survey, where the sample size was in the range according to Roscoe (1975). Refer to **Figure 2**, college and university names were not revealed due to confidentiality issues outlined in the Malaysian Personal Data Protection Act (2010). Among 116 respondents, 110 students identified as Malaysian nationality and only 10 students had programming experience before enrolling in the computing programme.

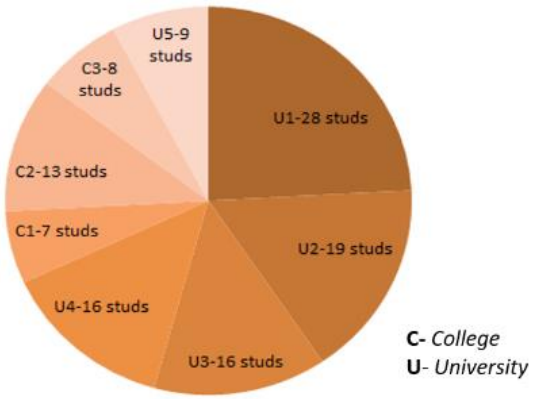

Figure 2. Number of student distribution among higher learning institutions.

The indication of students' motivation level is categorized as low, average and high as shown in **Table 3**. In analyzing motivation level, raw scores were calculated by summing up the answers given to the 5-step Likert-type scale (Korkmaz, 2013). While the lowest point that could be attained from the scale was 28, the highest was 140. While the prospective students who attained scores between 28 and 65 points were assessed as having low motivation, those who attained a score between 66 and 103 points were assessed as having average motivation and the ones who scored more than 104 points were assessed as having high motivation. In the standardized test, they are identified as poor, average and good respectively by summing up the total scores where the lowest scores are 0, the highest is 30 as shown in **Table 3**. The prospective students who attained a score between 0 and 9 points were assessed as having poor knowledge, those who attained a score between 10 and 19 points were assessed as having average knowledge and the ones who scored more than 20 points were assessed as having good knowledge in OOP Introduction topic.

Table 3. Students' motivation level and standardized test range.

Motivation Level			**Standardize Test**		
Low	Average	High	Poor	Average	Good
28–65	66–103	104–140	0–9	10–19	20–30

The findings with eight different private higher institutions consisting of 5 universities and 3 colleges are shown in **Table 4** were available at the following link https://docs.google.com/document/d/1re8 UWIJYCITIRKuqNAQAZrXUwZKK5Lea/edit?usp=sharing&ouid=106402626769 666511970&rtpof=true&sd=true. **Table 4** indicates students' motivation level and the standardized test scores based on **Table 3** Students' Motivation Level and Standardize Test Range. The authors identified that 4 results are mismatched because the self-efficacy result didn't correlate with students' standardized test scores. So, the following results are removed from the table.

- Student U1S7 where the motivation level was average (78) but standardized test scores were low (8).
- Student U2S7 where motivation level was average (78) but standardized test scores were low (7).

- Student U1S7 where motivation level was high (128) but standardized test scores were low (9).
- Student C1S6 where motivation level was average (84) but standardized test scores were low (9).

Table 4. Students' motivation level and standardized test results in OOP.

Motivation Level			**Standardized Test Scores**		
Low	Average	High	Poor	Average	Good
28–65	66–103	104–140	0–9	10–19	20–30
55% (58 students)	31% (33 students)	14% (15 students)	55% (58 students)	31% (33 students)	14% (15 students)

After removing the mismatched results stated above, the student's motivation level along with their OOP standardized test score can thus be presented as shown in **Table 4**. The findings indicate that out of 106 students involved in the survey, 55% (58 students) of the students are found to achieve low motivation and poor scores on the OOP standardized test, while 31% (33 students) were found to achieve an average motivation level and OOP standardized test score. Only 14% (15 students) were recorded to achieve a high motivation level and a good score on the OOP standardized test. Overall, it can be seen that students' motivation level and their OOP standardized test scores correlated with each other.

Refer to the hypothesis used in this study "If students' self-efficacy as students' motivation is low in programming subject, then the passing rate in programming subject is also low'. From the results of the study (**Table 4**), the hypothesis was accepted because more than half of the participants (58 out of 106 students) motivation levels are considered to be low and they failed to perform the fundamental concepts of OOP tested in the standardized test. According to the findings, it shows that students' low motivation was correlated with students' poor standardized test scores which is 55%. The result was aligned with Özyurt and Özyurt (2015) and Yukselturk and Altiok (2017) assumption that students may fail in their programming subject as they have low self-efficacy perceptions in programming.

- Q13. I can debug (correct all the errors) a long and complex program that I had written and make work.
- Q14. I can understand a long, complex multi-file program.
- Q21. While working on a programming project, if I got stuck at a point, I could find ways of overcoming the problem.
- Q24. I can mentally trace through the execution of a long, complex multi-file program given to me.
- Q25. I can rewrite lengthy and confusing portions of code to be more readable and clearer.

86% of the students' ratings for the above stated question were "not confident at all" and "slightly confident" which it showed their self-efficacy was low. This outcome is consistent with the findings reported by Çoklar and Akçay (2018).

Anyhow, for the following individual question, 92% of students' ratings are "moderately confident" and "mostly confident", while their self-efficacy level is

average. Looking at the questions, it can be concluded that the students are more confident performing the programming task when there is an assistant.

- Q12. I can make use of a pre-written method, given a clearly labeled declaration of the method.
- Q15. I can complete a programming project if someone shows me how to solve the problem first.
- Q16. I can complete a programming project if I had only the language reference manual for help.
- Q17. I can complete a programming project if I can call someone for help if I get stuck.
- Q18. I can complete a programming project once someone else helps me get started.
- Q20. I can complete a programming project if I just had the built-in help facility for assistance.

5. Conclusion

Studies conducted by educators and researchers show that self-efficacy directly affects the learning process. This study aimed to identify students' self-efficacy as a measure of student's motivation level in learning programming mainly Malaysian students. From the findings, the overall students' self-efficacy level in programming is mostly low. This finding is consistent with international literature (Avcu and Ayverdi, 2020; Durak et al., 2019; Erol, 2020). A closer look at each student's rating indicates a lack of self-efficacy when it comes to complex programming tasks. On the other hand, the students' self-efficacy is much more promising when there is guidance during their programming activity. As programming knowledge and skills are an important part of students' careers, it can be concluded that the level of students' self-efficacy as an indicator of their motivation is not sufficient, and they lack confidence in developing a complete program solution to a given task by themselves.

Author contributions: Conceptualized the research framework, designed the methodology, led the data analysis, RT and SC; conducted experiments, collected data, contributed to manuscript drafting, MY and MM; provided critical revisions, supervised the project, secured funding, RT and VV. All authors reviewed and approved the final manuscript.

Conflict of interest: The authors declare no conflict of interest.

References

Abdunabi R, Hbaci I, Ku HY. Towards Enhancing Programming Self-Efficacy Perceptions among Undergraduate Information Systems Students. Journal of Information Technology Education 2019 Jun 1; 18. doi: 10.28945/4308.

Aris, TNM. Object-oriented programming semantics representation utilizing agents. Journal of Theoretical and Applied Information Technology 2011; 31(1): 10-20.

Atiq SZ. Board# 40: The Relationship between Engineering Students' Self-efficacy Beliefs and Their Experience Learning Computer Programming: A Sequential Explanatory Mixed-Methods Investigation. In2017 ASEE Annual Conference & Exposition 2017 Jun 24. Doi: 10.18260/1-2--27849

Avcu YE, Ayverdi L. Examination of the Computer Programming Self-Efficacy's Prediction towards the Computational Thinking Skills of the Gifted and Talented Students. International Journal of Educational Methodology 2020; 6(2): 259-70. doi: 10.12973/ijem.6.2.259

Bakar MA, Mukhtar M, Khalid F. The Effect of Turtle Graphics Approach on Students' Motivation to Learn Programming: A Case Study in a Malaysian University. International Journal of Information and Education Technology. 2020 Apr; 10(4): 290-7. doi: 10.18178/ijiet.2020.10.4.1378

Bandura A. Social foundations of thought and action. Englewood Cliffs, NJ. 1986; 23-28.

Bujang SDA, Selamat A, Krejcar O, et al. Digital Learning Demand for Future Education 4.0—Case Studies at Malaysia Education Institutions. In Informatics 2020; 7(2):13.

Cheah CS. Factors contributing to the difficulties in teaching and learning of computer programming: A literature review. Contemporary Educational Technology 2020 May 8; 12(2): 272. doi: 10.30935/cedtech/8247

Çoklar AN, Akçay A. Evaluating Programming Self-Efficacy in the Context of Inquiry Skills and Problem-Solving Skills: A Perspective from Teacher Education. World Journal on Educational Technology: Current Issues 2018; 10(3): 153-64.

Cronbach LJ. Coefficient alpha and the internal structure of tests. Psychometrika 1951 Sep; 16(3): 297-334.

De Vicente A. Towards tutoring systems that detect students' motivation: an investigation; 2023.

Derus SR, Ali AM. Difficulties in learning programming: Views of students. In1st International Conference on Current Issues in Education (ICCIE 2012) 2012 Sep; 74-79.

Durak HY, Yilmaz FG, Yilmaz R. Computational thinking, programming self-efficacy, problem solving and experiences in the programming process conducted with robotic activities. Contemporary Educational Technology 2019 Apr 1; 10(2): 173-97. doi: 10.30935/cet.554493

Erol O. How Do Students' Attitudes towards Programming and Self-Efficacy in Programming Change in the Robotic Programming Process? International Journal of Progressive Education 2020; 16(4): 13-26. doi: 10.29329/ijpe.2020. 268.2

Gilal AR, Jaafar J, Omar M, et al. Suitable personality traits for learning programming subjects: a rough-fuzzy model. International Journal of Advanced Computer Science and Applications 2017; 8(8).

Gilbert GE, Prion S. Making sense of methods and measurement: Lawshe's content validity index. Clinical Simulation in Nursing 2016 Dec 1; 12(12): 530-1. Doi: 10.1016/j.ecns.2016.08.002

Hattie J. Visible learning: A synthesis of over 800 meta-analyses relating to achievement. Routledge 2008 Nov 19.

Hooshyar D, Ahmad RB, Yousefi M, et al. Applying an online game-based formative assessment in a flowchart-based intelligent tutoring system for improving problem-solving skills. Computers & Education 2016 Mar 1; 94:18-36. doi: 10.1016/j.compedu.2015.10.013

Hui TH, Umar IN. Metaphors and pair programming as a constructivist strategy in computing education: A literature review. Emerging Trends in Higher Education Learning and Teaching 2010; 62.

Ismail MN, Ngah NA, Umar IN. Instructional strategy in the teaching of computer programming: a need assessment analyses. The Turkish Online Journal of Educational Technology 2010 Apr 1; 9(2):125-31.

Jackson, S. L. (2011). Research methods and statistics: A critical approach (4th ed.). Cengage Learning.

Jono MN, Hasanordin R, Salleh S, et al. Measuring of Effectiveness of Courseware Content Using Learning Theory for a Programming Subject. InEnvisioning the Future of Online Learning: Selected Papers from the International Conference on e-Learning 2015; 193-202. doi: 10.1007/978-981-10-0954-9_17

Kanaparan G, Cullen R, Mason D. Effect of self-efficacy and emotional engagement on introductory programming students. Australasian Journal of Information Systems. 2019 Jul 8; 23. doi: 10.3127/ajis.v23i0.1825.

Kittur J. Measuring the programming self-efficacy of Electrical and Electronics Engineering students. IEEE Transactions on Education 2020 Mar 10; 63(3): 216-23.

Korkmaz O. Prospective CITE teachers' self-efficacy perceptions on programming. Procedia-Social and Behavioral Sciences 2013 Jul 4; 83: 639-43. doi: 10.1016/j.sbspro.2013.06.121

Lawshe CH. A quantitative approach to content validity. Personnel psychology 1975 Dec 1; 28(4): 563-75.

Lun CC, Rong TH, Seng LK, et al. A Case Study on the Impact of Video Games Towards Malaysian Youth. Journal of Theoretical and Applied Information Technology 2022 Oct 15; 100(19).

Lynn, M. Determination and quantification of content validity. Nursing Research 1986; 35(6): 382-385.

Malaysian Personal Data Protection Act 2010. Available at https://www.kkmm.gov.my/pdf/Personal%20Data%20Protection%20Act%202010.pdf

McGettrick A, Boyle R, Ibbett R, Lloyd, et al. Grand challenges in computing: Education—a summary. The Computer Journal 2005; 48(1), 42-48. doi: 10.1093/comjnl/bxh064

Mohammadzadeh M, Awang H, Ismail S, et al. Establishing content and face validity of a developed educational module: Life skill-based education for improving emotional health and coping mechanisms among adolescents in Malaysian Orphanages. Journal of Community Health Research 2017 Dec 10; 6(4): 223-8.

Mustaffa A, Lajuma S, Wider W. Employee engagement during COVID-19 in Malaysia. Front. Sociol 2022; 7(9):76966. doi: 10.3389/fsoc.2022.976966

Osman MA, Loke SP, Zakaria MN, et al. Secondary students' perfectionism and their response to different programming learning tools. In2012 IEEE Colloquium on Humanities, Science and Engineering (CHUSER) 2012 Dec 3; 584-588. doi. 10.1109/CHUSER.2012.6504380.

Owolabi J, Adegoke BA. Multilevel analysis of factors predicting self efficacy in computer programming. International Journal on Integrating Technology in Education (IJITE) 2014; 3(2): 19-29. doi: 10.5121/ijite.2014.3203

Özyurt Ö, Özyurt H. A study for determining computer programming students' attitudes towards programming and their programming self-efficacy/Bilgisayar programcılığı öğrencilerinin programlamaya karşı tutum ve programlama öz-yeterliklerinin belirlenmesine yönelik bir çalı. Eğitimde Kuram ve Uygulama 2015 Jan 2; 11(1): 51-67.

Rahman FA, Baidowi ZMP, Ahmad J. Learning Introductory C Programming: Relevant Exercises Based On Student Difficulties Factors. In 11th International and National Conference on Engineering Education (INCEE-11) 2013 May; 1(1): 5-10.

Rahmat M, Ahmad K, Idris S, et al. Relationship between employability and graduates' skill. Procedia-Social and Behavioral Sciences 2012 Oct 17; 59: 591-7. doi: 10.1016/j.sbspro.2012.09.318

Ramalingam V, Wiedenbeck S. Development and validation of scores on a computer programming self-efficacy scale and group analyses of novice programmer self-efficacy. Journal of Educational Computing Research. 1998 Dec; 19(4): 367-81. doi: 10.2190/C670-Y3C8-LTJ1-CT3P

Roscoe, J. T. (1975). Fundamental research statistics for the behavioral sciences.

Rosminah S, Ali AZ. Development of hardware-interfacing learning kit for novice learning programming. International Journal of Information and Education Technology 2016 Aug 1; 6(8): 647. doi: 10.7763/IJIET.2016.V6.767

Rum SN, Ismail MA. Metocognitive support accelerates computer assisted learning for novice programmers. Journal of Educational Technology & Society 2017 Jul 1; 20(3):170-81.

Ryan RM, Deci EL. Intrinsic and extrinsic motivations: Classic definitions and new directions. Contemporary educational psychology. 2000 Jan 1; 25(1): 54-67. doi: 10.1006/ceps.1999.1020.

Sabjan A, Abd Wahab A, Ahmad A, et al. MOOC Quality Design Criteria for Programming and Non-Programming Students. Asian Journal of University Education 2021 Jan 24; 16(4): 61-70. doi: 10.24191/ajue.v16i4.11941

Saharudin AM, Yusoff M, Haron H, et al. An Analysis of Factors Influencing Students Performance in Programming Assessment. Advanced Science Letters 2018 Nov 1; 24(11): 8182-5. doi: 10.1166/asl.2018.12519

Salleh N, Abdullahi MS, Nordin A, et al. Cloud-Based Learning System for Improving Programming Skills And Self-Efficacy. Journal of Information and Communication Technology 2018 Oct 1; 17(4): 629-51. doi: 0.32890/jict2018.17.4.2642

Thinakaran R, Ali R, Husin WNAAW. A Case Study of Undergraduate Students Computer Self-Efficacy from Rural Areas. Int. J. Eng. Technol 2018; 7: 270. doi: 10.14419/ijet.v7i3.20.19164

Thinakaran R, Ali R, Morina Abdullah KRS, et al. Motivation Assessment Model using Fuzzy Logic in Programming Tutoring System. Indian Journal of Science and Technology 2017; 10(48): 48. doi: 10.17485/ijst/2017/v10i48/120769

Thinakaran R, Ali R. Programming Tutoring Systems and Motivation Assessment Model. Advanced Science Letters 2017 Apr 1; 23(4): 2709-12. doi: 10.1166/asl.2017.7702

Thinakaran R, Chupra S, Batumalay M. Motivation assessment model for intelligent tutoring system based on mamdani inference system. IAES International Journal of Artificial Intelligence 2023 Mar 1; 12(1):189.

Yacob A, Saman MY. Assessing level of motivation in learning programming among engineering students. InThe International Conference on Informatics and Applications (ICIA2012) 2012 Jun; 425-432.

Yong ST, Tiong KM. A Blended Learning Approach: Motivation and Difficulties in Learning Programming. International Journal of Information and Communication Technology Education (IJICTE) 2022 Jan 1; 18(1): 1-6. Doi: 10.4018/IJICTE.301276

Yukselturk E, Altiok S. An investigation of the effects of programming with Scratch on the preservice IT teachers' self-efficacy perceptions and attitudes towards computer programming. British Journal of Educational Technology 2017 May; 48(3): 789-801. doi: 10.1111/bjet.12453

Zhang W, Zeng X, Wang J, et al. An analysis of learners' programming skills through data mining. Education and Information Technologies 2022; 1-19. doi: 10.1007/s10639-022-11079-4.

Zimmerman BJ. Self-efficacy: An essential motive to learn. Contemporary educational psychology 2000 Jan 1; 25(1): 82-91. doi: 10.1006/ceps.1999.1016.