International Journal of Engineering & Technology, 7 (4.38) (2018) 788-794

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Metacognitive Strategies in Teaching and Learning Computer Programming

Siti Nurulain Mohd Rum[1], Maslina Zolkepli[2]

[1,2]Department of Computer Science, Faculty of Computer Science & Information System, Universiti Putra Malaysia, 43330, Serdang, Malaysia

Abstract

It has been noted that teaching and learning programming is challenging in computer science education and that this is a universal problem. To understand and to code programs are perceived as being very challenging in computer science education. This is due to the demand for practical ability rather than theory alone. Studies have revealed that students with metacognitive management skills perform well in programming compared to lower-performing students. The more difficult the programming activity, the greater the need for the programmer to own metacognitive control skills. The cognitive processes in learning computer programming require a novice programmer to develop metacognitive skills. The main objective of this research work is to identify the metacognitive strategies in teaching and learning programming. An exploratory study was setup to identify the level of metacognition awareness of novice programmers using the MAI instrument. Interview sessions with expert lecturers were also conducted to identify the metacognitive approaches and the pedagogical method applied in the teaching and learning activities. The learning behaviours of novices were also identified through the interviewing sessions. It can be concluded that there is a correlation between the metacognitive awareness level of an individual and their academic achievement.

Keywords: *Computer Programming, Metacognitive Strategies, Novice Programmer, Computer Science Education*

1. Introduction

In general, Malaysian students have claimed that the skills required for computer programming are difficult to acquire. Studies have shown that analytical skills and problem-solving are the two major problems in computer programming [1-4]. N. Ismail et al. [5] claimed that many students are “woefully inadequate” in terms of problem-solving skills. Soloway & Spohere [1] agreed that the major weaknesses of students in computer science are analytical thinking and problem-solving techniques, and that these major issues and concerns of computer science programs should emphasize these skills. Problem-solving in computer programming studies involves cognitive skills that require the students to work in a methodical manner and build representations [6], which create an environment that is suitable for developing metacognition skills and awareness. Flavell [10] defined metacognition as thinking about thinking or cognition (e.g., memory, reasoning, perception, etc.). Volet [11] described “metacognitive strategies” in teaching and learning as the use of metacognition elements to aid learning, including checking, planning, selecting, monitoring, evaluating and revising. These strategies are parts of important components for self-regulated learning models in the classroom [12]. Studies by [13] showed that the application of such strategies is positively correlated with academic achievement. The activities in programming foster students to evaluate their thinking process. These cognitive activities facilitate the process of applying a newly acquired problem to novel problem situations [6]. Regardless of what problem-solving approaches are used, it is always the first step before coding the program. Trial-and-error is fundamental to problem-solving, and contributes to the development of knowledge [7]. The competency of programmers can be developed through meticulously organized programming knowledge and skills [3] that demand metacognitive skills. The role of metacognition in solving programming problems is imperative. Bergin et al. [8] found that students who possess metacognitive management skills and strategies perform well in programming compared to those in whom the skills are lacking. Moreover, the more difficult the problem in programming, the higher the demand for metacognitive control, positive feedback, and purposeful reflection [9].

2. Difficulties encountered by novice programmers

Learning programming is very challenging, and has resulted in high dropout rates at universities. [1, 14]. Typically, it takes about 10 years for a novice to become an expert [1]. Many attempts have been made by other researchers to identify the attributes of novices, their learning behaviours, and the connection with the different aspects of programming. The process of learning can be described as the progress of the learner from novice level to expert level. In terms of the differences between novices and experts in respect of programming knowledge; the novices are battling with syntactic knowledge [15], while in the context of semantic knowledge the expert possesses effective notions or a virtual mental model. In terms of schematic knowledge, experts use a structural approach to categorize programs based on required routines. In contrast, novices use superficial features as they are unskilled with problem decomposition and tend to use low-level plans. The experts decompose problems into manageable sub-problems where they keep the overall view of the problem in mind and will consider many alternative solutions compared to novices [15, 16].

Therefore, teaching and learning strategies are needed in overcoming the challenges faced by novice programmers [1]. The central development competency of a programmer involves problem-solving skills [1], and yet, these skills appear inadequate. Soloway & Spohere [1] pointed out that analytical thinking and problem-solving are major weaknesses in programming. Metacognition skills and awareness have been identified as important keys to success for problem-solving across domains [17, 18]. Novices are also identified as having metacognitive deficiencies regardless of their age [19, 20]. In addition, they fail to reflect on what they have learned [21]. Thus, to be a lifelong learner and become skilful in problem-solving, it is necessary to develop articulation and reflection in novice programmers. The role of metacognition in problem-solving in computer programming is imperative. Bergin et al. [8] found that students who possessed metacognitive management skills and strategies perform very well in programming compared to lower-performing students. Interestingly, the more complex a programming problem is, the greater the demand for metacognitive control, purposeful reflection, and positive feedback [22]. To judge how clearly and effectively a programmer understands a problem in programming, it is necessary for them to apply in-depth reading skills and meta-comprehension skills. In this research work, a questionnaire survey and interview sessions were conducted with three objectives; firstly, to understand the level of metacognitive awareness of novices, secondly, to identify the type of pedagogical approach that assists them during computer programming, and, thirdly, to understand their learning behaviour. To support these objectives, interview sessions with expert lecturers were also conducted to explore the metacognitive application in teaching and learning computer programming at the university level.

3. Novices' metacognitive awareness

The Metacognitive Awareness Inventory [23] was used in this study to measure the novice programmers' awareness of metacognition in learning computer programming at university. This inventory consists of 52 self-report items that are broadly divided into two components of metacognition: metacognitive knowledge and metacognitive regulation. The knowledge of cognition depicts the general knowledge about an individual's own learning process. Whereby, the regulation of cognition is associated with the knowledge concerning the way students plan, implement strategies, monitor, correct errors, and evaluate their learning. In this study, we define the characteristics of novice programmers, as someone with insufficiencies in knowledge and the skills of programming. Therefore, the target respondents in this study are undergraduate students of computer science that have just completed the introductory computer programming at the university.

3.1. Subjects

This study involved 164 undergraduate computer science students (102 females and 62 males). The target sample size was 400, and the response rate was 41%. Conventionally, for an online survey, a 40% response rate is considered to be a good response rate [24]. All the respondents were invited through email as well as a Facebook group.

3.1.1 Findings

The correlation coefficient of r for all variables is presented in Table 1 and Table 2. The results indicate that there is a positive linear correlation between the GPA and MAI score with a correlation coefficient of r = 0.8226 and significant at the 1% level. The findings confirmed that metacognition has a positive effect on students learning success in the Introductory Computer Programming course at the university, as when the MAI score increases, the GPA tends to increase as well, and vice versa. The variations in MAI explained about 67.67% of the variation in GPA (r2 = 0.6767 with n=164), indicating that there is a possibility of 32% of other factors influencing or affecting student learning success in Computer Programming. The results also demonstrated a strong correlation between the GPA and Knowledge about Cognition (KC) with r = 0.7483, GPA with Regulation of Cognition (RC) with r = 0.8224, GPA and Procedural Knowledge (P) with r = 0.4387, GPA and Declarative Knowledge (DL) r = 0.7358, and GPA and Conditional Knowledge (CDL) with r = 0.6134. These findings corroborate that students' performance achievement in computer programming learning has a strong correlation with metacognitive awareness.

Table 1: Correlation Coefficient between GPA and other sub-components of knowledge Cognition components in MAI

Y	Grade Point Accumulative (GPA)				
X	Knowledge Cognition	MAI	Procedural Knowledge (P)	Declarative Knowledge (DL)	Conditional Knowledge (CDL)
r	0.7483	0.8226	0.4387	0.7358	0.6134
r2	0.5599	0.6767	0.1925	0.5413	0.3762
t	14.3562	18.4136	6.2140	13.8274	9.8843
Pr(>|t|)	0.0000	0.0000	0.0000	0.0000	0.0000

Table 2: Correlation Coefficient between GPA and other sub-components of Regulation Cognition components in MAI

Y	Grade Point Accumulative (GPA)					
X	Regulation Cognition	Planning	Information Management System (IMS)	Comprehension Monitoring (CM)	Debugging (DBG)	Evaluation (EVL)
r	0.8224	0.7061	0.6882	0.7025	0.6023	0.5679
r2	0.6763	0.4986	0.4737	0.4935	0.3627	0.3225
t	18.3987	12.6917	12.0747	12.5629	9.6027	8.7816
Pr(>|t|)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

The results presented in Table 2 are consistent with the findings of other researchers [25-29] who found that the metacognition of learners possibly impacts the process and outcome of their learning. The studies by [25-29] explored the metacognitive awareness in a particular domain of knowledge to see the effect on learning strategies, listening performance, reading comprehension performance, and attitude towards academic achievement. A number of attempts have also been made by others to embed metacognitive elements in traditional teaching and learning as well as in computer-based support systems, such as [30-33]. With the results obtained, it is feasible for us to investigate further the application of metacognitive strategies in teaching and learning computer programming.

4. Classroom learning

The goal of this study is to identify the pedagogical approaches and metacognitive strategies used to assist students during the computer programming class. The observational study was conducted in the classroom during the Introductory Computer Programming class session for which the instructors had been informed in advance. This study was conducted in the Faculty of Computer Science in a public university in Malaysia. A letter requesting consent to conduct the survey was given in advance to the instructors. Positive feedback was received from the instructors about their willingness to give full support. The observational study was carried out in the lab during a normal class session. Prior to the data collection, a face-to-face meeting with the instructors was conducted to fully explain this research work. The observations and the recording activities took place during the teaching and learning activities in the class session for further analysis. To have a full view of the activities in the lab, an appropriate place was located for the researchers. The questions for the observational study were partly derived from the motivating factors that metacognition skills require for teaching and learning computer programming at the university. This observational study sought to answer the following questions:

1. What kind of learning approaches are applied in the computer programming class session?
2. What kind of metacognitive strategies are performed during each learning approach?
3. What are the advantages of each learning strategy?

4.1. Subjects

This study involved five different Computer Programming class groups with different instructors and sessions. The respondents in this study were students who had been taught by the instructors. An observational study was conducted during a normal lesson as stipulated in the faculty timetable. Five sessions were involved in this observational study. Each class had an average of 20 students aged between 20 years and 25 years old. Most of the students in the classes were those who had never performed programming and had no programming knowledge. The students included were first-year undergraduates taking Introductory Computer Programming.

4.2. Findings

The results were analysed and summarized as presented in Table 3. The observation was done quantitatively based on the data collection during the survey. The five observation sessions were termed CSO1, CSO2, CSO3, CSO4, and CSO5. The first question of the observational study related to how the instructor begins the class session. To gain the students' attention, the findings showed that the instructor introduced the learning topics in CSO1. This was carried out by asking the students a question at the end of the class. While, in CSO2, the instructor/lecturer provided a brief overview of the content through handouts and outline. During CSO3, the relevance of learning through the lab exercises was explained. Whereby in CSO4 and CSO5, the instructor/lecturer related the learning content of previous class materials. The second observation question related to how the instructor/lecturer teaches during the computer programming class. All of the instructors/ lecturers used the same approach by asking students specific questions. At the end of the class sessions, a small quiz was conducted to determine the effectiveness of each learning approach as well as the strategies applied throughout the learning sessions. The students were given 15 minutes to answer the five questions that were devised with the aid of an instructor.

Table 3: The approach to pedagogy in teaching and learning computer programming at the university

How did students learn during class?	Metacognitive Strategies Applied	Advantages	Class Session
Collaboration	Planning & Organizing -Preview Assignments given to decide how to approach them -Create a timeline to divide big tasks into manageable chunks between group members Monitoring -Check their progress against a timeline -Troubleshoot issues -Have peers monitor and observe each other	Working and sharing ideas with others that were on the same level Learning and get assistance from others Discuss the development of thinking skills; thus, promote metacognitive reflecting strategies	CSO1, CSO3
Discussion and dialogue	Connecting -Link new learning to prior experience and knowledge -Use familiar and easy to relate to examples	Students feeling comfortable voicing various opinions and viewpoints Students become responsible for the class and their own learning Build self-confidence More interactive Promote metacognitive reflecting	CSO2, CSO5
Repetition	Self-Reflect -Self-assess about knowledge understanding	Repeating lecture gives the students an opportunity to assess their ability as well as what they did and did not understand, which, indirectly, builds the metacognitive skills of the students	CSO4

The findings of the survey showed that three types of learning approaches were commonly used by instructors to teach computer programming: cooperative learning, discussion and dialogue, and repetition. It has been proven in many other studies [34, 35] that learning through collaboration is an effective way of learning as it encourages the learning process itself and fosters respect as well as friendship among diversified groups of students. Several studies demonstrated that cooperative learning involves metacognitive thoughts [34, 35]. Cooperative learning covers a range of group-based learning approaches [36]. It is a set of different methods and students were encouraged by the teacher to cooperate in learning. It encourages the interaction between students. The cooperative learning activity engages the students in the learning process and seeks to improve the critical thinking, reasoning, and problem-solving skills of the learner, which, eventually, promotes the metacognitive learning atmosphere. Several metacognitive strategies were found during cooperative learning, such as planning and organizing, as well as monitoring. During the planning stage, all the assignments were reviewed by all the students in the respective groups to decide the methods to approach them. Timelines were developed to ensure that the assignments were manageable and fulfilled the requirements. These kind of activities have the potential to develop their critical thinking skills and time management skills. The second learning approach was through discussion and dialogue between the students and the lecturer. Meaningful discussion that facilitates reflective thinking can be initiated when learners raise thoughtful questions or provide critical feedback. Lecturers ask questions to help students discover what has been taught, to comprehensively explore the subject matter, and to generate discussion and peer-to-peer interaction. Student-initiated questions increase higher-order learning skills by requiring them to analyse information, connect seemingly disparate concepts, and articulate their thoughts [37]. The benefits of this learning approach are that it enables students to consider various viewpoints and to be responsible for the class and their own learning. In addition, this approach makes the students feel comfortable enough to speak, which, indirectly, could help students to build their self-confidence and produce a more interactive learning environment. One activity that was related to metacognitive strategy was connecting new information to former knowledge during dialogue and discussion. The more connections they make, the better the learning comprehension [38]. The third approach was through repetition. Simple repetition can have a powerful impact on learning. This approach provides students with the opportunity to assess their ability concerning what they understood; hence, indirectly developing the self-reflective thinking of students. [39] suggests that reflective thinking is an active, persistent, and careful consideration process that leads to a deeper understanding. Students who actively participate in reflective thinking and are aware of and control their learning, have the ability to assess what they know, what they need to know, and how they bridge that gap during learning situations [40]. When students encounter complex problems, reflective thinking helps them to understand their progress in learning, explore the appropriate strategies for a problem, and identify appropriate solutions to solve the problem [40]. The results from the quiz revealed that 85% of students from CSO1, CSO4, and CSO5 were able to answer all the questions. Whereas, 75% of the students from class CSO2 and CS03 obtained all the correct answers. As shown in Table 3, the metacognitive strategies underlying each learning approach had the greatest impact on the learning success of students in teaching and learning computer programming in the classroom.

5. Novices' learning behaviour

The objective of this study was to understand the learning behaviour of the novice programmers in learning computer programming at the university. Seven questions were formulated based on three types of metacognition knowledge, as categorized by [10]. Person variable – which refers to what one knows and recognizes about his or her strengths and weaknesses in learning and processing information [10]. Task variable – which refers to what one knows or can figure out about the nature of a task and the demands required for completing the task [10]. Strategy variable – which refers to the strategies a person has "at the ready" to apply in a flexible way to successfully accomplish a task [10]. Livingston [41] provided all three variables in metacognition: "I'm aware that I (person variable) have difficulty with programming the logic problem (task variable), so I will sketch a pseudocode to describe the algorithm first and later translate it into the programming language (strategy variable)." The questions are also designed based on the analysed data extracted from the interviewing sessions with the expert lecturers, as follows:

1. Do you know your own strengths and weaknesses in learning programming?
2. How do you motivate yourself to learn programming?
3. What are the types of resource you usually refer to?
4. What techniques do you use to process information in learning programming?
5. What kind of help-seeking method do you use to make you understand while learning programming?
6. Do you track the amount of time spent on solving problems in programming?
7. How do you evaluate your own knowledge performance and understanding?

5.1. Participants

The respondents were the same students who participated in the survey. They were invited again by email for the interview sessions. One-hundred emails were randomly sent out to respondents; however, only ten participants responded to the email and agreed to take part in this study. As shown in Table 4, the respondents were aged between 19 and 23 years old. Most of the respondents had insufficient knowledge and skills about programming thereby meeting the definition of a novice.

5.2. Findings

The recorded results were transcribed verbatim and analysed. For the first question, most of the respondents failed to recognize their own strengths and weaknesses in programming. Only respondents R4 and R6 were aware of their strengths and weaknesses. The respondents' knowledge relates to the declarative knowledge as well as the 'Person Variables'. [10] indicates that it is important for one to be able to process or use his/her critical thinking that is related to learning. The second question related to the respondents' motivation for learning programming. R1, R2, R5, and R10 responded to 'cooperative learning', such as knowledge sharing with peers to make them motivated to learn programming. However, the R3, R7, R8, and R9 claimed that programming is a difficult subject and that they were not motivated to learn programming at all. In contrast, R4 and R6 were self-motivated and enjoyed learning programming. Question three was about the main references to learning programming. The Internet was the main source of reference for all the respondents to learn programming. Apart from the Internet, books and materials given by the instructors were also the resources used by novices as a reference. Question four related to the information management strategies to process information. It also related to the 'Strategies Variable', which [10] described as the strategies applied to understanding the knowledge. Respondents R1, R6, R7, and R8 translated the information obtained into words that that they could understand, whereas R3 and R10 used the note summarization technique to process information. Respondents R2, R4, and R9 used pictures and diagrams to represent the information or knowledge obtained. However, respondent R5 used the organizational structure to represent the information. Question five related to the Strategies Variable; R1 used social media like forums and Facebook as the medium to seek help while learning; R2, R3, R4, R5, and R8 preferred to seek help from the instructor or someone possessing good knowledge about programming; and others preferred to study with friends and compare answers from the quizzes, tests, and assignments given by the lecturer, find similar problem solutions from the Internet, and by trial and error. The sixth question referred to the time consumed in learning computer programming individually. Most of the respondents did not record the time taken to learn programming individually. Only R1 responded, stating that he spent about two to four hours for learning programming individually. Question seven was about how students evaluate their own knowledge and understanding. In response to knowing his strengths and weaknesses, R1 responded that he often communicated with the lecturer. R2 familiarized himself with the exercises and sought help from the instructor when he was confused. Both R3 and R4 had similar answers – check quizzes, tutorials, and test papers and check with the instructor or others. Others used the Internet to check the answers for the tests, exam papers or quizzes for similar problems.

6. Metacognitive in teaching programming

The main objective of the interviews with expert lecturers was to explore the metacognitive implementation and metacognition awareness in computer programming courses at universities. The interview sessions were conducted at the faculty. The participation was voluntary in nature, and each session lasted around 45 minutes. The interview sessions were conducted during the mid-term of the semester. Each interview session was conducted individually. The questions asked during the interview sessions are:

1. Have you ever heard of metacognition?

2. Can you describe what metacognition is? (After reading the definition)
3. How important is 'metacognition' for a lecturer in computer programming?
4. Do you really think that metacognition is important in computer science education? Why?
5. How are 'metacognitive skills' taught in your class?
6. How are 'metacognition skills' applied in your teaching?
7. What is important to you when you are teaching?
8. What things are in your mind when you are teaching?
9. What did you do before the teaching session?
10. What is your preparation before the teaching session?
11. What do you usually do after you teach?

6.1. Participants

Ten (10) expert lecturers were selected from the faculty for the interview sessions. The selection process was based on the years of experience in teaching courses in computer programming and their years of experience in related research projects and consultancy. Three of them were PhD holders, and the others were master degree holders. All the participants in this study had teaching experience of more than five years in various types of programming languages (e.g., object-oriented, structured programming, JAVA programming, C, and C++ programming and assembly languages).

6.2. Findings

The data from the interview sessions were transcribed verbatim and analysed as explained in the following sections.
Expert Lecturer Definition of Metacognition.
At the beginning of the interview session, all the expert lecturers were asked whether they had heard the term 'metacognition'. All the expert lecturers responded that they had never heard of 'metacognition'. A piece of paper presenting the definition of 'metacognition' was provided to each of them. The definition of 'metacognition' is presented as follows:
"Metacognition is a form of critical thinking that allows one to understand, to analyse and control one's cognition, especially when engaging in teaching and learning. It can be in many forms that include knowledge about how to apply particular strategies for learning as well as for problem-solving. It is about 'cognition about cognition' or 'knowing about knowing' [42].
After reading the definition of metacognition, they were asked again about the terms. One of them responded to "applying learning theory" while others answered that metacognition is "the way learners learn and develop knowledge," "Awareness of Cognition" with only one respondent giving an almost complete answer; that is, "Awareness of Cognition and Monitoring of Cognition." The results indicate that most of the participants responded to "Monitoring Cognition" and "Awareness of cognition". It is feasible to discuss the appropriate definition of metacognition with expert lecturers.

Expert Definition of Metacognition.
All the participants agreed that metacognition is an important factor for successful teaching. Respondent one, who has been teaching C++ programming for almost 15 years, said that "Lecturers must think hard, as it is important for effective learning." While other responses were as follows, "It is important, and lecturers must think about their own thinking," "Lecturers must think hard, it is important for effective learning," "Educators must understand their own teaching processes," "Lecturers must be aware of their own emotions, thoughts, and behaviours."
Metacognition application before teaching
Computer Programming.
All the participants responded to "Planning". The answers given by the respondents to the questions are "Design and plan lessons," "lesson-plans," "Prepare for class the day before and arrive in the classroom a few minutes early," "Analyse the teaching material and confirm the content of the lesson," "write outlines for discussion in the classroom," and "make notes and review activities." All the activities stated above are the activities that take place.
Metacognition application during teaching
Computer Programming.
All the participants responded that the 'student' is the important variable in teaching. The examples of respondents were as follows, "prepared well before teaching, monitor myself during teaching to ensure message can be conveyed effectively to the student," "listening to student and trying to answer any questions in the simplest way that can be easily understood," "interaction with student" "writing," "discussing," "questioning and thinking with students," "problem-solving," "book and journal reading," "spotting obstacles," "mapping concept," "modelling," "direct instruction," "presenting," "thinking aloud," "reciprocal teaching."
Metacognition application after teaching Computer
Programming.
The highest response from all the participants was "Evaluating" or "Assessing". The examples from the respondents were as follows "Prepare teaching content for the following class session and try to revise content from previous session that was not sufficiently provided in the previous session", "Evaluating the strength and weakness of one's own teaching", "Assessing/evaluating student learning by giving short quizzes", "assessing teaching and planning for next round teaching" and "remains in the classroom for questioning and answering session for 15 or 30 minutes at the end of the class session". One of the respondents indicated that he reflects by asking himself, "Am I really satisfied with my teaching session today? How do I need to revise and improve the lesson plan for the next teaching session?" Another respondent said that he categorized teaching into three different stages, before entering the classroom and starting to teach the instructor must plan the lesson, during teaching the instructor must act and think, and, after teaching, the instructor must reflect.

7. Discussion

The findings of the survey showed that there is a relationship between metacognitive skills and academic achievement. Three types of learning approach that are commonly used by instructors in teaching computer programming consist of cooperative learning, discussion, and dialogue as well as the repetition lecturing approach. Cooperative learning in computer programming is an approach to study and work in a group that minimizes the occurrence of unpleasant situations and maximizes the learning process. There are many reasons why cooperative learning works well in learning programming, as they learn more hands-on activities than when listening and watching. Moreover, cooperative learning in problem-solving can enhance the metacognitive knowledge and strategies of the individual. However, working and learning in a group can often cause situations where some of the students in the group are left behind because of different levels of understanding and ability. Although it allows work and assignments to be completed, not everyone understands what has been done. The discussion and dialogue is another approach to teaching and learning that involves two-way communication between the students and the lecturer. A well-planned discussion with students in the classroom encourages students to speak and stimulate their thinking process and add variety to the learning session. An effective discussion can be a powerful medium for encouraging the learning atmosphere in the classroom. However, passive students who are too shy to answer questions and raise an opinion are left behind in a large-group context. Repeating lectures helps students to build metacognitive skills; however, with this teaching approach, students are too dependent on the lecturer. Based on three types of metacognitive skills, as categorized by [10], person variable, task variable and strategy variable, the findings showed that most students own metacognitive knowledge albeit most of them do not recognize their strengths and weakness. If the students understand their abilities in learning programming, they could be able to recognize their strengths and weaknesses. There are many ways to recognize strengths and weakness in learning programming, such as get feedback from lecturers and engage in computer programming activities. Expert lecturers can identify strengths and weaknesses in students by administering verbal quizzes, tests, hands-on projects, and written assessments. This research work also revealed that expert lecturers acknowledge and recognize the role of metacognition and apply it in teaching and learning in computer programming. Fourteen metacognition strategies that were employed during teaching were identified. These include coaching, questioning, mapping concept, problem-solving, reciprocal teaching, thinking aloud, reading books, modelling, journal, presenting, direct instruction, writing, asking to think, and discussing. An efficient lecturer must understand himself, must be aware of each student, and know the content of the knowledge as well as the pedagogical approach applied during teaching; this is to ensure that the knowledge can be conveyed effectively.

8. Conclusion

Metacognitive strategies are associated with successful learning. This has been proven by a number of studies. This study was designed to identify the pedagogical approach assisting students in learning computer programming. The MAI instrument was used to measure the students’ metacognitive awareness level with their performance in the Introductory computer programming subject (GPA) at university. Interestingly, this study demonstrated the association between student achievement and their metacognitive awareness level. This research work also revealed that metacognitive awareness, knowledge, and strategies have been practiced throughout the teaching and learning activities. These give opportunities for the students to increase their ability to transfer the knowledge as well as provide the knowledge concerning how to extend their abilities. The absence of metacognition will make the learners unable to recognize their abilities [43]. It is suggested that an effective plan could be developed by instructors for the students to improve their learning if they are able to apply metacognitive strategies in teaching and learning activities [44].

References

[1] E. Soloway and J. C. Spohrer, Studying the novice programmer: Psychology Press, 2013.

[2] R. Poli and J. Koza, Genetic Programming: Springer, 2014.

[3] M. C. Linn and M. J. Clancy, "The case for case studies of programming problems," Communications of the ACM, vol. 35, pp. 121-132, 1992.

[4] P. Henderson, "Modern introductory computer science," ACM SIGCSE Bulletin, vol. 19, pp. 183-190, 1987.

[5] M. N. Ismail, N. Azilah, U. Naufal, and U. T. M. C. Kelantan, "Instructional strategy in the teaching of computer programming: a need assessment analyses," TOJET, vol. 9, pp. 125-131, 2010.

[6] R. E. Mayer, Applying the science of learning: Pearson/Allyn & Bacon Boston, 2011.

[7] R. J. Sternberg and P. A. Frensch, Complex problem solving: Principles and mechanisms: Psychology Press, 2014.

[8] S. Bergin, R. Reilly, and D. Traynor, "Examining the role of self-regulated learning on introductory programming performance," in Proceedings of the first international workshop on Computing education research, 2005, pp. 81-86.

[9] M. Havenga, "Problem-solving processes in computer programming: a case study," in Southern African Computer Lecturers’ Association (SACLA) Conference Proceedings, 2011, pp. 91-99.

[10] J. H. Flavell, "Metacognition and cognitive monitoring: A new area of cognitive–developmental inquiry," American psychologist, vol. 34, p. 906, 1979.

[11] S. E. Volet, "Modelling and coaching of relevant metacognitive strategies for enhancing university students' learning," Learning and Instruction, vol. 1, pp. 319-336, 1991.

[12] P. R. Pintrich, "A conceptual framework for assessing motivation and self-regulated learning in college students," Educational psychology review, vol. 16, pp. 385-407, 2004.

[13] M. Richardson, C. Abraham, and R. Bond, "Psychological correlates of university students' academic performance: a systematic review and meta-analysis," Psychological bulletin, vol. 138, p. 353, 2012.

[14] E. Soloway and J. C. Spohrer, Studying the novice programmer: Lawrence Erlbaum Hillsdale, NJ, 1989.

[15] E. Lahtinen, K. Ala-Mutka, and H. M. Järvinen, "A study of the difficulties of novice programmers," in ACM SIGCSE Bulletin, 2005, pp. 14-18.

[16] M. Hu, M. Winikoff, and S. Cranefield, "Teaching novice programming using goals and plans in a visual notation," in Proceedings of the Fourteenth Australasian Computing Education Conference-Volume 123, 2012, pp. 43-52.

[17] J. E. Davidson, R. Deuser, and R. J. Sternberg, "The role of metacognition in problem solving," 1994.

[18] R. Azevedo and V. A. Aleven, International handbook of metacognition and learning technologies vol. 26: Springer, 2013.

[19] A. L. Brown, "Metacognitive development and reading," Theoretical issues in reading comprehension, pp. 453-481, 1980.

[20] J. D. Bransford, A. L. Brown, and R. R. Cocking, "How people learn," ed: Washington, DC: National Academy Press, 2000.

[21] D. C. Berliner and R. C. Calfee, Handbook of educational psychology: Macmillan Library Reference USA, Simon & Schuster Macmillan, 1996.

[22] M. Havenga, B. Breed, E. Mentz, D. Govender, I. Govender, F. Dignum, et al., "Metacognitive and problem-solving skills to promote self-directed learning in computer programming: teachers’ experiences," Sa-educ Journal, vol. 10, pp. 1-14, 2013.

[23] G. Schraw and R. S. Dennison, "Assessing metacognitive awareness," Contemporary educational psychology, vol. 19, pp. 460-475, 1994.

[24] E. Deutskens, K. De Ruyter, M. Wetzels, and P. Oosterveld, "Response rate and response quality of internet-based surveys: An experimental study," Marketing letters, vol. 15, pp. 21-36, 2004.

[25] J. McCabe, "Metacognitive awareness of learning strategies in undergraduates," Memory & Cognition, vol. 39, pp. 462-476, 2011.

[26] H. Tok, H. Özgan, and B. Döş, "Assessing Metacognitive Awareness And Learning Strategies As Positive Predictors For Success In A Distance Learning Class/Uzaktan Eğitim Sınıfında Başarının Pozitif Yordayıcısı Olarak Bilişötesi Farkındalık Stratejisi Ve Öğrenme Stratejilerinin Değerlend," Mustafa Kemal Üniversitesi Sosyal Bilimler Enstitüsü Dergisi, vol. 7, 2010.

[27] C. C. Goh and G. Hu, "Exploring the relationship between metacognitive awareness and listening performance with questionnaire data," Language Awareness, vol. 23, pp. 255-274, 2014.

[28] F. Takallou, "The effect of metacognitive strategy instruction on EFL learners’ reading comprehension performance and metacognitive awareness," Asian EFL Journal, vol. 13, 2011.

[29] C. Tosun and E. Senocak, "The effects of problem-based learning on metacognitive awareness and attitudes toward chemistry of prospective teachers with different academic backgrounds," Australian Journal of Teacher Education, vol. 38, p. 4, 2013.

[30] M. Tiantong and S. Teemuangsai, "The four scaffolding modules for collaborative problem-based learning through the computer network on moodle lms for the computer programming course," International Education Studies, vol. 6, p. 47, 2013.

[31] V. A. Aleven and K. R. Koedinger, "An effective metacognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor," Cognitive science, vol. 26, pp. 147-179, 2002.

[32] R. Azevedo, "Beyond intelligent tutoring systems: Using computers as METAcognitive tools to enhance learning?," Instructional Science, vol. 30, pp. 31-45, 2002.

[33] I. Roll, V. Aleven, B. M. McLaren, and K. R. Koedinger, "Designing for metacognition—applying cognitive tutor principles to the tutoring of help seeking," Metacognition and Learning, vol. 2, pp. 125-140, 2007.

[34] M. V. Veenman, B. H. Van Hout-Wolters, and P. Afflerbach, "Metacognition and learning: Conceptual and methodological considerations," Metacognition and learning, vol. 1, pp. 3-14, 2006.

[35] D. W. Johnson, R. T. Johnson, and K. A. Smith, "Cooperative learning: Improving university instruction by basing practice on

validated theory," Journal on Excellence in University Teaching, vol. 25, pp. 1-26, 2014.

[36] R. Hertz-Lazarowitz, S. Kagan, S. Sharan, R. Slavin, and C. Webb, Learning to cooperate, cooperating to learn: Springer Science & Business Media, 2013.

[37] T. Tofade, J. Elsner, and S. T. Haines, "Best practice strategies for effective use of questions as a teaching tool," American journal of pharmaceutical education, vol. 77, p. 155, 2013.

[38] .D. L. Driscoll, "Building connections and transferring knowledge: The benefits of a peer tutoring course beyond the writing center," The Writing Center Journal, pp. 153-181, 2015.

[39] R. H. Ennis, "Critical thinking: A streamlined conception," in The Palgrave handbook of critical thinking in higher education, ed: Springer, 2015, pp. 31-47.

[40] A. Ghanizadeh, "The interplay between reflective thinking, critical thinking, self-monitoring, and academic achievement in higher education," Higher Education, pp. 1-14, 2016.

[41] J. A. Livingston. (1996). Effects of metacognitive instruction on strategy use of college students. Available: http://www.all-about-psychology.com/biological-psychology.html

[42] J. E. Metcalfe and A. P. Shimamura, Metacognition: Knowing about knowing: The MIT Press, 1994.

[43] D. Dunning, K. Johnson, J. Ehrlinger, and J. Kruger, "Why people fail to recognize their own incompetence," Current directions in psychological science, vol. 12, pp. 83-87, 2003.

[44] C. B. McCormick, C. Dimmitt, and F. R. Sullivan, "Metacognition, learning, and instruction," Handbook of psychology, vol. 7, pp. 69-97, 2013.