The Effect of Turtle Graphics Approach on Students' Motivation to Learn Programming: A Case Study in a Malaysian University

Marini Abu Bakar, Muriati Mukhtar, and Fariza Khalid

***Abstract*—Computer programming is not an easy subject to learn or teach, particularly to first-year students in higher learning institutions. Numerous past studies have shown that students faced difficulties in comprehending and applying the programming concept when writing programs to solve problems. A slight negative experience at the initial stage of the studies is enough to disappoint the students. This then resulted in lowering the motivation of students to learn programming which impacted the students' performance in the programming course. Thus, an approach must be developed so that the motivation to learn and practice programming remains high. This article proposes the use of Turtle Graphics at the initial stage of Java Programming course in Universiti Kebangsaan Malaysia. Via this approach, by writing suitable programs, the students are able to produce animations and graphics output. Students' motivation levels are then measured via the ARCS model, which measure the students' motivation based on four aspects, namely attention, relevance, confidence, and satisfaction. The results of this study found that the students demonstrate high motivation in all four motivational aspects.**

***Index Terms*—ARCS model, course content, graphics library, novice programming, objects-first.**

I. Introduction

Specifically, computer programming is a fundamental element for a world that is currently facing the 4th Industrial Revolution (4IR). Computer programming is no longer limited to university students undertaking Computer Science courses, but everyone needs to have the ability to do programming [1], [2]. This situation resulted in the necessity to re-evaluate the approach towards teaching and learning in higher education.

Programming is the foundation for the Computer Science and Information Technology curriculum, and generally, for the STEM (Science, Technology, Engineering, and Mathematics) fields. Literature study reports almost everyone agreed that teaching and learning of programming, particularly, the first programming course usually known as CS1 is tough and challenging for the students [3]–[7] and failure rate is high [8]–[10]. Lecturers also found that it is quite challenging to teach the programming course [6], [11], [12]. This is supported by the finding of a study on the programming lab tests performance in comparison with the lab exercises, which showed that the average percentage of all correct answers (full marks) in the lab exercises was 69.8% compared to only 8.9% in the lab tests [13]. The CS1 programming course is an important foundation for other courses in the Computer Science field.

Many factors contribute to the challenges in the teaching and learning of programming courses. For instance, the features and concepts involved in the programming subject, do raise various problems [10]. The programming concept is difficult to comprehend, particularly by new students [4], [14]. This causes many students to admit that they dislike programming because they have to work hard to understand the concepts. The difficulties in understanding a programming concept are even more challenging for object-oriented programming languages [5], [14], [15]. The programming characteristic that builds one concept over another different concept also renders difficulties to the students at the initial phase, which led to the complication of moving to the next phase of learning. Besides the concepts, students also face difficulty in handling the complex syntax of the programming language [16]. They face challenges in converting the algorithm to a program using the correct syntax. Programming also requires the ability to think abstractly, make generalizations, and think critically [17]. A programmer must imagine and understand many abstract terms, which do not have any similarities with real-life. These and many other situational problems make CS1 a very challenging course both for lecturers and students in a higher learning institution. In addition, [17] succinctly compiled the problems into a) the teaching strategies that did not support the students' various styles of learning, b) a dynamic teaching concept that used static resource materials, c) lecturers focused more on teaching about the programming language instead of the method to solve teaching problems, d) students lack the motivation to learn programming, e) programming requires an abstract thinking capacity, f) students lack sufficient logical and mathematical knowledge, g) programming language has a complex syntax, h) students were introduced to programming at a difficult stage of their life.

The problems described inadvertently will impact upon the motivation of the students to learn programming, which will in turn has an impact on the students' performance in a programming course. A slight negative experience at the

Manuscript received November 14, 2019; revised March 13, 2020. This work was supported by the Ministry of Education of Malaysia under research grant FRGS/1/2016/ICT01/UKM/02/3.

Marini Abu Bakar and Muriati Mukhtar are with the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia (e-mail: marini@ukm.edu.my, muriati@ukm.edu.my).

Fariza Khalid is with the Faculty of Education, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia (e-mail: fariza.khalid@ukm.edu.my).

initial stage of learning is enough to discourage the students [4]. This will prove detrimental in the long run because, in order to succeed at programming, students need to put in a lot of effort in doing exercises, and students would not be able to go through them all unless they are highly motivated [18]. Thus, increasing or retaining students' motivation at certain levels are necessary in their learning process [19]. Therefore, an approach is needed to enhance the students’ motivation in learning programming. This proposed approach will also have to consider the needs of the current generation that are now entering the institutions of higher learning. At the present time, the students are from the Z generation who are inclined towards the visual learning style [20]. For the visually inclined students, it is difficult to understand an abstract concept, more so with static resource material.

The following section will discuss the concept of learning motivation and the ARCS model which is used to measure motivation. Section III briefly reviews the graphical approaches used in the teaching of programming. Section IV outlines the research method. This is followed by sections V and VI which respectively describe the case study and the results obtained. Section VII presents the conclusion of the study and future works.

II. Motivation to Learn

Reference [21] defines motivation as a process that energizes, directs, and sustains behavior. Motivation influences students' focus and effort in completing an activity or achieving a learning goal. Self-determination theory represents a framework for the study of motivation and individual personality that emphasizes internal resource for personality development, attitude and behavior [22]. There are two types of motivation related to self-determination theory, namely intrinsic and extrinsic motivation. Intrinsic motivation refers to the desire to perform something due to interest and self-satisfaction. On the other hand, extrinsic motivation is based on external goals, such as to gain certain rewards or to avoid punishment.

According to [22], intrinsic motivation can be enhanced when a student feels that he or she has competence, autonomy, and relatedness. Competent refers to someone who feels that he or she performed a task efficiently. The sense of competence can only be achieved if a person has the autonomy, that is when someone can control his or her action. Meanwhile, relatedness refers to a situation in which someone wishes to interact meaningfully with other people. Reference [23] classified extrinsic motivation into four categories, namely external regulation, introjected regulation, identified regulation, and integrated regulation. Extrinsic motivation with the lowest autonomy is in the external regulation in which it happens due to conformity to external needs, rewards, or punishment. Typically, an action is taken in a coerced or desperate situation. Extrinsic behavior with the most autonomy is in integrated regulation, which happened self-consciously.

A. *ARCS Motivational Framework Model*

ARCS is a motivational design model that was developed by Keller [24] to enhance and retain students’ motivation to learning. Motivational framework refers to the layout process of the resources and procedure to improve the motivation for learning. The purpose is to increase students’ participation in learning activities. ARCS model comprises four factors, namely attention, relevance, confidence, and satisfaction.

1) *Attention*: Related to learning activities that are essential to gain and sustain students' attention. To stimulate students’ motivation to learn, the lecturer needs to get their attention at the initial learning process with something unexpected. This attention must be retained during the learning process using a different approach.
2) *Relevance*: It is important to ensure that the student believes that the learning experience meets their personal goals. Students should clearly understand that the learning activity is relevant to their needs to effect a positive attitude.
3) *Confidence*: The main reason for students to be less motivated is due to fear of failure or too much expectancy for success. The lecturer needs to design the learning activities so that the students believe that they can learn the content and will succeed in an assignment. Confidence is imperative to ascertain success.
4) *Satisfaction*: For students to have desire to continue learning, they must be satisfied with the result or process of learning. Satisfaction can result from extrinsic and intrinsic factors. Extrinsic factors are external factors such as grades or material rewards. Intrinsic factors can be very powerful, such as feelings of accomplishments and feelings of competent. Students’ satisfaction will sustain motivation in learning.

These four components of motivation are based on a learning-related motivational theory that was introduced by Keller [24] in 1987. The ARCS model has been applied and validated in multiple research in various fields [25]–[29]. The ARCS model combined the theories of learning behavior, cognitive, and effective learning in a framework to measure motivation and learning achievement [26].

Keller [24] divided each of the four categories into three subcategories based on the motivation variables listed in the categories, as shown in Table I.

TABLE I: Keller' ARCS Model

Category	Subcategory
Attention	A1 Perception arousal
	A2 Inquiry arousal
	A3 Variability
Relevance	R1 Goal orientation
	R2 Motive matching
	R3 Familiarity
Confidence	C1 Learning requirements
	C2 Success opportunity
	C3 Personal control
Satisfaction	S1 Natural consequence
	S2 Positive consequence
	S3 Equity

B. *Related Works*

Several previous studies employed the ARCS model in learning of programming. Initially, [30] used the ARCS model to measure the programming internal training motivation for the ICT industry in Japan. Next, [25] applied the ARCS model in testing the suitability of using the textual programming language (TPL) for primary school students. Later, [31] compared the application of textual programming language (TPL) and visual programming language (VPL) towards primary school students. The three studies employed a 12-item questionnaire based on the subcategories in Table I.

Reference [19] applied the ARCS model in programming learning at Qatar University by developing an e-learning module, including classroom interaction. He used a 5-item questionnaire to assess the approach's effectiveness.

Reference [26] used the ARCS model to measure students' motivation using blended learning for the Science Computer students at one of the universities in Saudi Arabia. They employed the Instructional Materials Motivational Survey (IMMS) questionnaire, which has 36 items that measured the four categories of the ARCS model.

Reference [32] integrated the ARCS model and problem-based learning (PBL) with a flipped classroom for learning of programming. They used a 21-item questionnaire based on the four categories of the ARCS model.

III. Graphics Library Approach in CS1

Most of the animation and graphics library approach used in learning programming is based on the LOGO programming language. LOGO was created by Seymour Papert and several fellow researchers at Massachusetts Institute of Technology (MIT) in 1967 [33]. LOGO is a language to teach children the basics of programming. The turtle in LOGO is a pointer object that can be operated and moved based on simple instructions. Lines are drawn following the turtle movement. When the LOGO language was first introduced, the instructions controlled a physical device in the shape of a turtle. With the development of the computer monitor technology, the turtle was depicted as a screen cursor as seen in the current computer. Among the instructions that can be given to the turtle are FORWARD n for moving n steps forward and RIGHT n for turning n degrees to the right.

The application of the graphics libraries in the teaching of programming in the universities has been reported several decades ago. Reference [34] applied the Karel the Robot libraries in the programming course. Karel the Robot was developed in 1981 for the introduction of the Structured Programming course, using Pascal. Karel the Robot continued development, and there are several popular versions, such as Robot [35] and Karel J Robot [36], which have been widely used in the teaching of the basic programming course.

Reference [37] developed the graphics libraries in ANSI C before converting it to Java. They encountered several problems while developing the libraries because the graphics library in the early version of Java was not stable. The Turtle package was developed by [33] using the Java language to be used in the CS1 course teaching. They employed the inverted curriculum approach, which was introduced by [38]. With the approach, the important concepts were introduced first, followed by the explanation later. The Turtle package was used in every topic of the course. A positive effect was reported, but no analysis was conducted to show the efficacy.

Reference [39] developed Traffic libraries using the Eiffel programming language. The Traffic libraries provide a large scale of interactive multimedia and graphics environment that allow the students to produce game programs in the basic programming course. The libraries which have nearly 150,000 lines of codes and 750 class allows the students to write programs to manipulate an application domain, which in this case is the traffic system of a city. Without using the libraries, it is impossible for the students to produce such programs in the basic programming course. They designed an introductory programming course based on the inverted curriculum approach in which students used the software libraries without the need to know the actual implementation of the libraries. This approach is based on the object-first approach, which was first introduced in the Computing Curricula 2001 document. With this approach, students can produce large programs, which are almost similar to the programs made by professionals, based on the capacity of the libraries.

IV. Method

This study proposed a new course structure for Computer Programming Course at the Faculty of Information Science and Technology (FTSM), Universiti Kebangsaan Malaysia (UKM), to replace the traditional approach. In the new course structure, Turtle Graphics approach that implements the object-first paradigm is applied in the development of new course materials to introduce the basic concept of programming.

The participants of this study consist of 167 students who enrolled for Computer Programming Course during 2017/2018 academic session. At the end of week four of the course, students' motivation is measured using the ARCS model.

TABLE II: ARCS Model Questionnaire Used in This Study

Attention	
A1	*Perception arousal*: I enjoy learning TurtleGraphics in the programming course.
A2	*Inquiry arousal*: I want to learn more about TurtleGraphics
A3	*Variability*: I am fascinated with the various problems that programming can solve using TurtleGraphics.
Relevence	
R1	*Goal orientation*: TurtleGraphics approach helps me to understand the basic concept of programming.
R2	Motive matching: TurtleGraphics is a suitable choice for me to learn a basic programming concept.
R3	Familiarity: I am familiar with the basic concept of programming that was introduced by TurtleGraphics.
Confidence	
C1	Learning requirement: I am clear about the purpose of using TurtleGraphics in the programming course.
C2	Success opportunity: I can write a programme well using TurtleGraphics.
C3	Personal Control: I am confident to write a programme to solve a problem using TurtleGraphics.
Satisfaction	
S1	Natural consequences: Solving a programming assignment using TurtleGraphics give me satisfaction (e.g., in producing an artwork assignment).
S2	Positive consequences: I am happy to be able to produce a good programme using TurtleGraphics.
S3	Equity: I am satisfied with the assessment method of the TurtleGraphics programme that I produce.
Motivation	
M	Overall, using TurtleGraphics motivate me to learn programming.

The measuring instrument is adapted from the questionnaire that has been verified in the studies by Tsukamoto *et al*. [25], [30], [31], which is based on Keller's [24] 12 categories ARCS model. Table II shows the questionnaire that has been adapted and verified by an expert. The 13th item is to measure the overall level of motivation. The questionnaire is measured using 5-point Likert Scale (from 1=strongly disagree to 5=strongly agree).

The questionnaire reliability level analysis, with a Cronbach Alpha value of 0.973, indicates that the level of reliability of the questionnaire items is very good [40].

V. A Case Study at Universiti Kebangsaan Malaysia

In the Faculty of Information Science and Technology (FTSM), Universiti Kebangsaan Malaysia (UKM), the traditional approach has been used since the CS1 programming course was first introduced in the 1980s. Programming languages used in the 1980s and early 1990s were FORTRAN and Pascal using the structured programming approach. Around 1994, the programming language was changed to C language and later to C++ in the early 2000s. Even though the C++ language is an object-oriented language, the approach used is still the structured programming approach.

Computer Programming is a core course for all programmes of FTSM, namely Computer Science Programme, Information Technology Programme, and the Software Engineering Programme. It is compulsory for every student of FTSM to take the Computer Programming course during the first semester of their study, followed by two or more other programming courses in the following semester according to their chosen programme. The Computer Programming course is conducted through lectures, tutorials, and laboratory sessions. Students' performance is assessed through laboratory tests, which requires the students to solve problems with hands-on coding. The first-year students who enrolled in FTSM came from various backgrounds. About half of them have learned basic programming at the pre-university level, either in Matriculation Programmes, Foundation Programmes or in High Schools. The other half of the students do not have any formal knowledge of programming before entering FTSM.

The faculty management decided to use the Java language in 2015 as the CS1 programming language replacing the C/C++ language, which has been used since 1994. The management also stipulated the objects-first approach is used by introducing the object-oriented programming concept at the beginning of the course.

A holistic effort to design a new teaching module and approach for the Computer Programming course was conducted at the beginning of the first semester for the 2015/2016 academic session. A study was performed for similar courses in other universities, both locally and abroad, and followed by several discussion sessions with the faculty. Three main aspects were emphasized; first, the programming language to be used, second, the teaching methodology, and third, the content material and supporting technology [41].

For the first aspect, it imposed by the faculty that the programming language to be used is the Java language. Java language was chosen because it has several advantages compared to C/C++ used earlier. Java is widely used in universities [42], [43] and industries [44]. Besides, Java supported extensive class libraries, including the well-established graphics libraries that can be utilized to develop software libraries to allow students to write programs with animation and graphics output as early as the first laboratory session in the CS1 course.

The second aspect involved the selection of teaching methodology, whether applying the objects-first approach, or the structured approach. The objects-first approach was selected as Java is an object-oriented language. Furthermore, the introduction of objects-first approach was made easy by using the libraries developed, such as the turtle graphics libraries.

The third aspect involved identifying the content and the supporting technology. Research was conducted on several technologies and software libraries used by other institutions, such as the turtle graphics concept that was introduced by Seymour Papert in 1967 [33], Karel the Robot [34], Alice, BlueJ and Greenfoot environment. The approach selected for implementation in the first semester of the 2015-2016 academic session was to develop a turtle graphics library based on the turtle graphics concept. The changes made are summarized in Table III.

TABLE III: Changes Made in the Study

	Prior to the 2015/2016 session	2015/2016 session onwards
Programming language	C++	Java
Programming paradigm	Structured	Object-oriented, objects-first
Teaching methodology	Traditional (textual)	Using turtle graphics library (graphical and animated output)

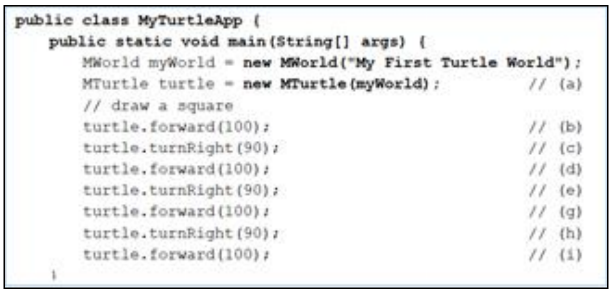
Table IV listed the topics for the proposed new course. Topic 1 is the Introduction to programming and Java which discussed the overview of programming and Java, and the history of Java.

TABLE IV: Topics of the Proposed Course Structure

Week	Content
1	1. Introduction to programming & Java
2	2. Objects: using, creating and communicating 3. Input-Output
3	4. Repetition structure: for 5. Conditional structure: if, switch
4	6. Data types and operators 7. Problem-solving I
5	8. Repetition structure: while, do-while, nested loop
6	9. Array & array processing (1-dimensional array)
7	9. Array & array processing (1-dimensional array)
8	10. String and string processing
9	11. Problem solving II
10	12. Static variables and methods
11	13. Array and array processing (2-dimensional array)
12	13. Array and array processing (2-dimensional array)
13	14. Classes
14	Revision

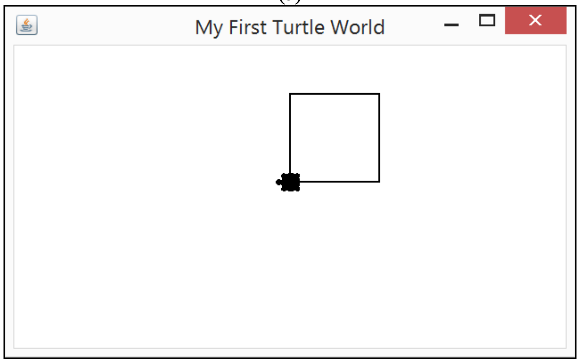
Topic 2 introduced the basic concept of object-oriented. Firstly, a few examples of real-world objects are used to discuss the concept of state, behavior and identity of an object. This followed by a discussion on how objects are created and used in programs, and how to communicate with the objects. To facilitate students' understanding of writing object-oriented programs, the TurtleGraphics library is used. Topic 2 asserted that the objects-first approach is employed.

The example used in Topic 2 is to draw a square, which also introduces the concept of sequential structure (Fig. 1). In Topic 3, input statement and variables are introduced that enable students to write a program that can draw square of different sizes.



```
public class MyTurtleApp {
    public static void main(String[] args) {
        MWorld myWorld = new MWorld("My First Turtle World");
        MTurtle turtle = new MTurtle(myWorld);          // (a)
        // draw a square
        turtle.forward(100);                            // (b)
        turtle.turnRight(90);                           // (c)
        turtle.forward(100);                            // (d)
        turtle.turnRight(90);                           // (e)
        turtle.forward(100);                            // (g)
        turtle.turnRight(90);                           // (h)
        turtle.forward(100);                            // (i)
    }
```

(a)



(b)

Fig. 1. Sequential structure: (a) program, (b) output.

Topic 4 and 5 introduced the concept of repetition structure and selection structure respectively. The example discussed in Topic 4 is to draw 10 squares in a row using for statement. The output is shown in Fig. 2.

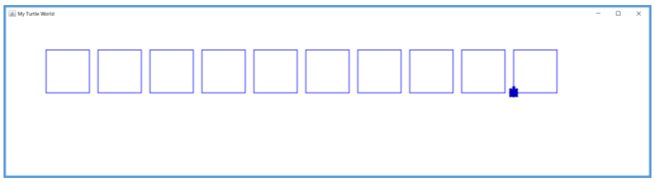
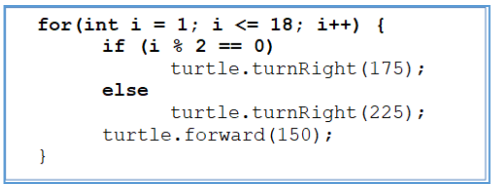


Fig. 2. Output of repetition structure program.

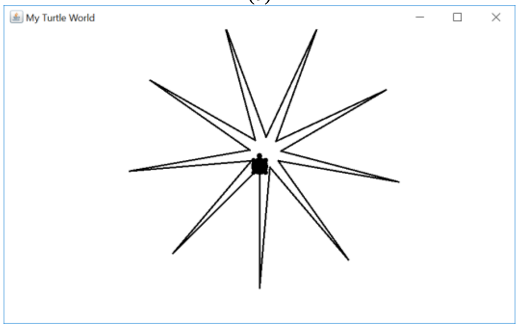
For the selection structure, the example shows a condition to check the odd or even number to determine the degree of angle that the turtle turns. The program segment with the output is shown in Fig. 3.

The teaching material was delivered as presentation slides during lectures. Follows, students are required to practice the learned concept and hands-on during the laboratory session. Students are required to write Java programs using the TurtleGraphics library to solve the given problems. On week 4, students' ability to use the Eclipse environment to write programs to solve problems using the TurtleGraphics library was evaluated through a laboratory test. Three simple problems are given and students need to complete them in two hours.



```
for(int i = 1; i <= 18; i++) {
    if (i % 2 == 0)
        turtle.turnRight(175);
    else
        turtle.turnRight(225);
    turtle.forward(150);
}
```

(a)



(b)

Fig. 3. Selection structure: (a) code segment, (b) output.

Students' performance in the laboratory test is encouraging. The majority of students managed to solve the problems given in the test and 87.5% scored full marks as compared to only 48% of students obtained full marks in the earlier session, which used the traditional approach.

In order to gauge students' motivation and skill in using TurtleGraphics, they were asked to write a program that draws a creative artwork using the TurtleGraphics library as an assignment. They were given two weeks to submit their work. Fig. 4 shows some of the creative artwork returned.

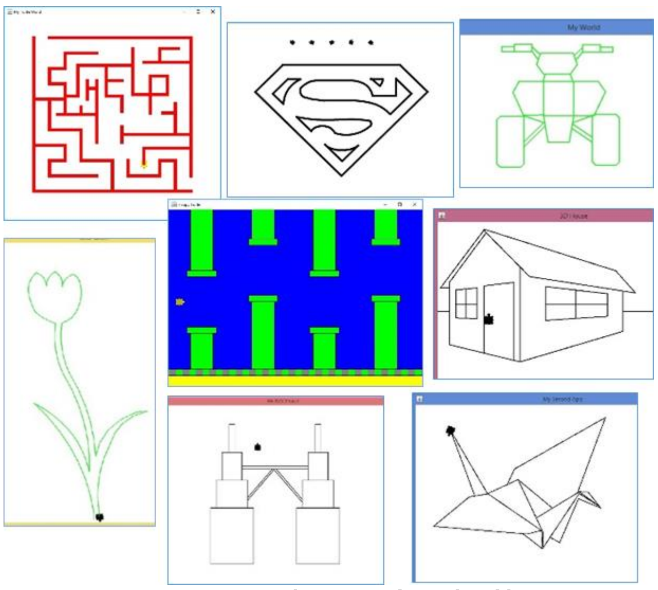


Fig. 4. Creative artwork using TurtleGraphics library.

VI. Motivation Analysis Using the ARCS Model

Students' motivation towards programming using the TurtleGraphics approach was evaluated using the ARCS motivational model introduced by Keller [24]. The number of respondents for this study was 169 students who attended the Computer Programming course during semester 1, 2017/2018 session. Table V shows the result of this study.

The students' level of motivation toward using TurtleGraphics at the start of the programming course was measured using the 5-point Likert scale (1=strongly disagree, 2=disagree, 3=moderately agree, 4=agree, 5=strongly agree). A descriptive analysis of the min score reading using the indicators are divided into three levels, namely low (1.00 – 2.33), medium (2.34 – 3.66), and high (3.67 – 5.00).

TABLE V: STUDENTS' MOTIVATION SCORE USING THE ARCS MODEL

No	Items	Mean Score
A1	*Perception arousal*: I enjoy learning TurtleGraphics in the programming course.	4.4734
A2	*Inquiry arousal*: I wish to learn more about TurtleGraphics.	4.3609
A3	*Variability*: I am fascinated by the variety of problems that can be solved using TurtleGraphics.	4.2899
R1	*Goal orientation*: TurtleGraphics approach helps me to understand a basic programming concept.	4.3669
R2	*Motive matching*: TurtleGraphics is a suitable choice for me to learn a basic programming concept.	4.3905
R3	*Familiarity*: I am familiar with the programming basic concept introduced by TurtleGraphics.	4.0769
C1	*Learning requirements*: I am clear about the objective of TurtleGraphics application in the programming course.	4.2308
C2	*Success opportunity*: I could write a programme well using TurtleGraphis.	4.1775
C3	*Personal control*: I am confident to write a programme for solving problems using TurtleGraphics.	4.1006
S1	*Natural consequence*: Completing a programming assignment using TurtleGraphics gives me satisfaction (for example in producing an artwork assignment)	4.2722
S2	*Positive consequence*: I am happy when I can produce a good programme using TurtleGraphics.	4.4201
S3	*Equity*: I am satisfied with the assessment method of the TurtleGraphics programme that I produced.	4.2840
M	Overall, using TurtleGraphics motivates me to learn programming	4.3136

Fig. 5 shows the findings from Table V in a bar chart form. Based on the mean score, all the respondents give a high response for all items. This indicates that the use of TurtleGraphics is effective to enhance students' motivation in learning programming. The highest item is A1 (4.4734), namely perception arousal, which indicates that the students enjoy using TurtleGraphics to learn programming. The lowest item is R3 (4.0769), namely familiarity, which means the students are less familiar with the programming basic concept. This is due to nearly half of students who took the course have never learned programming before entering university, and also due to some slow learners who found programming is difficult.

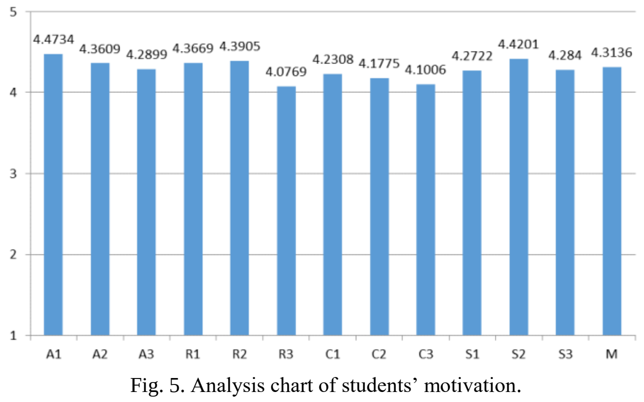


Fig. 5. Analysis chart of students' motivation.

Further analysis is based on four motivational elements that were outlined in the ARCS model, namely attention (A), relevance (R), confidence (C), and satisfaction (S), is presented in Table VI. The mean score is high for all the four motivational elements. The highest element is attention (A) at 4.3748, which shows that the TurtleGraphics approach in the programming course can attract the students' attention. The lowest element among the four elements is confidence (C) at 4.1696, which could be caused by several students who have never learned programming before, and the students who are weak in understanding the programming concept. These students might need a longer time to understand the required concepts.

TABLE VI: ANALYSIS BASED ON FOUR MOTIVATION ELEMENTS OF ARCS

Motivation Element	Mean Score	Std. Deviation
Attention	4.3748	0.84452
Relevence	4.2781	0.81786
Confidence	4.1696	0.83896
Satisfaction	4.3254	0.85446
Mean score	4.2870	0.78742

The mean average score for the overall motivational study is 4.2870, which shows that the TurtleGraphics approach in the basic programming course can retain the students' motivation effectively in learning programming.

VII. CONCLUSION

This paper articulates the results of a research that was done to develop an approach for the teaching and learning of programming at a local university in Malaysia. The main concerns when developing the approach was that, this approach must be able to help students to retain the level of motivation needed in learning programming and, it must also cater to the needs of the new generation of students entering the higher institutions of learning. The proposed approach is the graphics output approach using the TurtleGraphics library which formed part of the introductory CS1 course at FTSM, UKM. The purpose is to make sure that the students remain motivated by reducing students' frustration and anxiousness, particularly during the early stage of the programming course. The results indicate that students' motivation from the four aspects of the ARCS model was high, with the overall motivation value of 4.2870 (out of the maximum score of five). This indicates that the approach has the potential to be used in the teaching and learning of programming in other institutions of higher learning.

The authors are currently exploring the potential of using the graphics library approach by designing and developing suitable new graphics libraries to be applied on different topics in Computer Programming course. This will be discussed in another article. Another plan is to conduct the discussed approach to students of different levels such as pre-university and high school.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Author Contributions

Marini conducted the research, analyzed the data and wrote the first draft; Muriati and Fariza supervised the research, review and improved the article; all authors had approved the final version.

Acknowledgment

The authors would like to show our gratitude and sincere thanks to Computer Programming teaching team Rodziah Latih, Masura Rahmat, Noor Faridatul Ainun Zainal, Shahrina Shahrani, Rohizah Abd Rahman, Azura Ishak (Universiti Kebangsaan Malaysia) for supporting the research.

References

[1] A. Mohd *et al.*, "Pendidikan dan Pendekatan Pengaturcaraan (PEARL)," *Penyelidikan Komputeran dalam Era Revolusi Industri 4.0*, 2017, pp. 139–146.

[2] S. Combéfis, G. Beresnevicius, and V. Dagiene, "Learning programming through games and contests: Overview, characterisation and discussion," *Olympiads in Informatics*, vol. 10, no. 1, pp. 39–60, 2016.

[3] S. M. M. Rubiano, O. Lopez-Cruz, and E. G. Soto, "Teaching computer programming: Practices, difficulties and opportunities," *2015 IEEE Frontiers in Education Conference (FIE)*, 2015, pp. 1–9.

[4] E. Dunican, "Making the analogy : Alternative delivery techniques for first year programming courses," *14th Work. Psychol. Program. Interes. Group, Brunel Univ.*, no. June, pp. 89–99, 2002.

[5] N. Thota and R. Whitfield, "Holistic approach to learning and teaching introductory object-oriented programming," *Comput. Sci. Educ.*, vol. 20, no. 2, pp. 103–127, 2010.

[6] G. M. M. Bashir and A. S. M. L. Hoque, "An effective learning and teaching model for programming languages," *J. Comput. Educ.*, vol. 3, no. 4, pp. 413–437, 2016.

[7] M. A. Bakar, M. I. Esa, N. Jailani, M. Mukhtar, R. Latih, and A. M. Zin, "Auto-marking system: A support tool for learning of programming," *Int. J. Adv. Sci. Eng. Inf. Technol.*, 2018.

[8] C. Watson and F. W. B. Li, "Failure rates in introductory programming revisited," in *Proc. the 2014 Conference on Innovation & Technology in Computer Science Education - ITiCSE '14*, 2014, pp. 39–44.

[9] J. Bennedsen and M. E. Caspersen, "Failure rates in introductory programming," *ACM SIGCSE Bull.*, vol. 39, no. 2, p. 32, 2007.

[10] A. Vihavainen, J. Airaksinen, and C. Watson, "A systematic review of approaches for teaching introductory programming and their influence on success," in *Proc. the Tenth Annual Conference on International Computing Education Research*, 2014, pp. 19–26.

[11] T. Koulouri, S. Lauria, and R. D. Macredie, "Teaching introductory programming: A quantitative evaluation of different approaches," *Trans. Comput. Educ.*, vol. 14, no. 4, pp. 26:1–26:28, 2014.

[12] R. Latih, M. Abu Bakar, N. Jailani, N. M. Ali, S. M. Salleh, and A. M. Zin, "A design for challenge-based learning of programming," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 8, no. 5, pp. 1912–1918, 2018.

[13] M. Bakar *et al.*, "Penilaian latihan makmal pengaturcaraan menggunakan PC2," *Teknologi Komputeran Generasi-Z*, Penerbit UKM, 2019, pp. 140–152.

[14] M. Kölling and J. Rosenberg, "Guidelines for teaching object orientation with Java," *ITiCSE '01 Proc. 6th Annu. Conf. Innov. Technol. Comput. Sci. Educ.*, vol. 33, no. 3, pp. 33–36, 2001.

[15] E. J. Johan, S. Idris, M. A. Bakar, and M. Mukhtar, "Persuasive object oriented programming lab assignment framework," *Int. J. Technol. Incl. Educ.*, vol. 4, no. 1, pp. 557–565, 2015.

[16] R. M. Kaplan, "Using problem-based learning in a CS1 course -Tales from the trenches," in *Proc. the International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, 2015, pp. 86–90.

[17] A. Gomes and A. J. N. Mendes, "Learning to program-difficulties and solutions," *Int. Conf. Eng. Educ.*, pp. 1–5, 2007.

[18] T. Jenkins, "The motivation of students of programming," *ACM SIGCSE Bull.*, vol. 33, no. 3, pp. 53–56, 2001.

[19] S. Alhazbi, "ARCS-based tactics to improve students' motivation in computer programming course," in *Proc. 10th International Conference on Computer Science and Education, ICCSE 2015*, 2015, pp. 317–321.

[20] E. J. Cilliers, "The challenge of teaching generation Z," *PEOPLE Int. J. Soc. Sci.*, vol. 3, no. 1, pp. 188–198, 2017.

[21] J. W. Santrock, *Educational Psychology*, McGraw-Hill, 2011.

[22] R. M. Ryan and E. L. Deci, "Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being," *Am. Psychol.*, vol. 55, no. 1, pp. 68–78, 2000.

[23] R. M. Ryan and E. L. Deci, "Intrinsic and extrinsic motivations: Classic definitions and new directions," *Contemp. Educ. Psychol.*, 2000.

[24] J. M. Keller, *Motivational Design for Learning and Performance: The ARCS Model Approach*, Springer, 2010.

[25] H. Tsukamoto, Y. Takemura, H. Nagumo, I. Ikeda, A. Monden, and K. Matsumoto, "Programming education for primary school children using a textual programming language," in *Proc. 2015 IEEE Frontiers in Education Conference (FIE)*, 2015, pp. 1–7.

[26] S. M. Bin-jomman and M. Al-Khattabi, "Measuring the effect of use web 2.0 technology on saudi students' motivation to learn in a blended learning environment," *Int. J. Adv. Comput. Sci. Appl.*, 2018.

[27] H. Tsukamoto, Y. Takemura, H. Nagumo, A. Monden, and K. I. Matsumoto, "Prediction of the change of learners' motivation in programming education for non-computing majors," in *Proc. Frontiers in Education Conference, FIE*, 2015.

[28] B. Huang and K. F. Hew, "Measuring learners' motivation level in massive open online courses," *Int. J. Inf. Educ. Technol.*, vol. 6, no. 10, pp. 759–764, 2016.

[29] M.-H. Ying and K.-T. Yang, "A game-based learning system using the ARCS model and fuzzy logic," *J. Softw.*, vol. 8, no. 9, pp. 2155–2162, 2013.

[30] H. Tsukamoto, Y. Takemura, H. Nagumo, and K. Matsumoto, "Analysis of the motivation of learners in the in-house training of programming in Japanese ICT industries," in *Proc. 2011 24th IEEE-CS Conference on Software Engineering Education and Training, CSEE and T 2011*, 2011, pp. 121–128.

[31] H. Tsukamoto *et al.*, "Textual vs. visual programming languages in programming education for primary schoolchildren," in *Proc. Frontiers in Education Conference, FIE*, 2016x.

[32] Y. H. Chang, A. C. Song, and R. J. Fang, "Integrating ARCS model of motivation and PBL in flipped classroom: A case study on a programming language," *Eurasia J. Math. Sci. Technol. Educ.*, 2018.

[33] M. E. Caspersen and H. B. Christensen, "Here, there and everywhere - on the recurring use of turtle graphics in CS1," in *Proc. the Australasian Conference on Computing Education*, 2000, pp. 34–40.

[34] B. W. Becker, "Teaching CS1 with karel the robot in java," in *Proc. Thirty-Second SIGCSE Tech. Symp. Comput. Sci. Educ. (SIGCSE '01)*, pp. 50–54, 2001.

[35] B. W. Becker *et al.*, *Java: Learning to Program with Robots*, 2007.

[36] J. Bergin, M. Stehlik, J. Roberts, and R. Pattis, *Karel J Robot : A Gentle Introduction to the Art of Object-Oriented Programming in Java*, Dream Songs Press, 2013.

[37] E. Roberts and A. Picard, "Designing a Java graphics library for CS 1," *ACM SIGCSE Bull.*, vol. 30, no. 3, pp. 213–218, 1998.

[38] B. Meyer, "Towards an object-oriented curriculum," *J. Object-Oriented Program.*, vol. 6, no. 2, pp. 76–81, 1993.

[39] M. Pedroni and B. Meyer, "The inverted curriculum in practice," *ACM SIGCSE Bull.*, vol. 38, no. 1, p. 481, 2006.

[40] J. Pallant, *SPSS Survival Manual: A Step by Step Guide to Data Analysisusing*, 2016.

[41] A. Pears *et al.*, "A survey of literature on the teaching of introductory programming," *SIGCSE Bull.*, vol. 39, no. 4, pp. 204–223, 2007.

[42] Simon, R. Mason, T. Crick, J. H. Davenport, and E. Murphy, "Language choice in introductory programming courses at australasian and UK universities," in *SIGCSE 2018 – Proc. the 49th ACM Technical Symposium on Computer Science Education*, 2018.

[43] E. Murphy, T. Crick, and J. H. Davenport, "An analysis of introductory programming courses at UK universities," *Art, Sci. Eng. Program*, 2017.

[44] TIOBE Index. (2019). TIOBE programming community index definition. [Online]. Available: https://www.tiobe.com/tiobe-index/

Marini Abu Bakar is a senior lecturer in the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. Her research interests include programming education, computer graphics computing and software technology.

Muriati Mukhtar is an associate professor at the Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia. Her research interests include e-service design and service science.

Fariza Khalid is a senior lecturer at the Faculty of Education, Universiti Kebangsaan Malaysia. Her research interests include e-learning, online communities of practice and emerging technologies for educational purposes.