e-ISBN : 978-967-25608-3-8

SIG : e-Learning@CS
https://appspenang.uitm.edu.my/sigcs/
Publication Date : 15 – Apr - 2022

PROGRAMMING RELATED FACTORS INFLUENCING IN LEARNING A PROGRAMMING: A STUDY ON ENGINEERING STUDENTS IN UITM PULAU PINANG

*Rozita Kadar[1], Jamal Othman[2], Naemah Abdul Wahab[3],
Maisurah Shamsuddin[4] and Siti Balqis Mahlan[5]
** rozita231@uitm.edu.my[1], jamalothman@uitm.edu.my[2], naema586@uitm.edu.my[3], maisurah025@uitm.edu.my[4], sitibalqis026@uitm.edu.my[5]*

[1,2,3,4,5]Faculty of Computer and Mathematical Sciences
Universiti Teknologi MARA, Cawangan Pulau Pinang, Malaysia

ABSTRACT

In response to a current need, higher education institutions have begun to provide programming courses not only to computer science students, but also to non-computer science students. This is because, in recent years, understanding of computer technology and programming in a variety of areas has become necessary to meet the industry's demand for information and communication competency. The issue is that teaching and learning computer programming languages, particularly for non-computer science students, can be difficult. Educators face challenges in getting students to comprehend programming principles and develop outstanding programming abilities in order to tackle real-world problems. Students' lack of logical, creative, and critical thinking leads to inadequacies in problem-based learning implementation. Several factors have been identified as contributing to programming language problems, including educators, students' abilities, and programming nature, and this study investigates a programming-related factor that contributes to computer programming learning challenges, as well as the students' background knowledge. A study was conducted on engineering students in Universiti Technologi MARA in a total of 241 students involves in this study. According to the results of a mean and standard deviation study, it is found that programming-related factor were particularly effective in learning a programming. With these findings, it can be a guideline for educators in dealing with problems in learning a programming.

Keywords: Programming Problem, Programming Nature, Program Structure, Engineering Student.

Introduction

Nowadays, having a good understanding of computer technology, as well as programming skills, is required to satisfy industrial demands (Siti Rosminah & Ahmad Zamzuri, 2012). For many students, particularly those with a non-computer science background, the current demand causes challenges and presents a considerable obstacle. According to (Moström, 2011), novice students must understand the problem, develop a solution using normal problem-solving methodologies, and then write down the solution in a programming language in such a way that a computer can understand the instructions.

According to the existing study, traditional teaching practices, as well as students' study methods and attitudes, must be improved. Instructional methods such as hands-on programming practise and cutting-edge teaching and learning tactics must be utilised to increase students' interest in computer education. Educators are faced with new challenges, demanding the creation of new

instructional tools. Students' background knowledge and attitudes, teaching and learning methods, and social context are all aspects that contribute to learning obstacles in computer programming, according to (Gomes et al., 2012). Educators must also address gaps in students' knowledge backgrounds, making large-scale student management more difficult (Ahmad & Ghazali, 2020).

Therefore, the purpose of this study is to identify the challenges that students have when learning a programming language by investigating the programming-related factors that influence students' programming language learning and proposing ways to overcome these concerns. This research is intended to help computer science educators improve their teaching methods for basic programming courses, as well as improve students' interest in and performance in programming courses.

Programming Nature in Learning a Program

Learning any programming languages is far more complicated, as it necessitates other abilities such as algorithm design, programming writing, and syntax understanding (Baist & Pamungkas, 2017). It is not easy to write a program code, according to (Moström, 2011), novice students must comprehend the problem, design a solution using standard problem-solving approaches, then write down the solution in a programming language in a way that a computer can follow the instructions.

The problem arise among students starting at the beginning is related to the understanding of programming environment, which causes students to see programming as something difficult (Ahmad & Ghazali, 2020). Also, the inability of students to reason logically and their lack of problem-solving skills are two major factors that contribute to programming inefficiency. Although different programming methods and approaches have been developed to assist students in learning programming, not all of them focus on the programming stages of problem resolution (Yusoff et al., 2020). Similarly, despite the existence of a variety of learning aids and teaching strategies, such as teaching by doing, using relevant examples, demonstrations, direct examples, and trail-guided teaching approaches, teaching issues and programming learning remain unresolved (Cheah, 2020).

One of the issues discussed regarding the effectiveness of students mastering a programming language is the nature of programming, which plays an important role in determining the effectiveness of students' ability to master a programming language. Pears et al. (2007) reported that most institutions use an object-oriented language, but many use Java, C and C ++, languages to teach procedural programming, whereas less than 10% of institutions teach functional programming.

Despite the popularity of such languages, there has been much debate about the suitability of these languages for education, especially when introducing programming to novices. These languages are not designed specifically for educational purposes, in contrast to others designed with this specific purpose (such as Python, Logo, Eiffel, and Pascal).

There are interrelated types of programming's' nature of difficulties while learning to program as stated in the previous studies. (Siti Rosminah & Ahmad Zamzuri, 2012) identified three issues that should be addressed by educators, which are: the lack of understanding of the basic concepts of programming structure; problem in designing a program to complete a specific task and; inability to identify the syntax of programming languages. This is in contrast to (Bosse & Gerosa, 2017), which is more focused on the ability of students in using computers and performing system development tasks. The present discussion focuses more on the opinions outlined by (Xinogalos, 2016), which has outlined five problems faced by students related to programming nature, namely: developing an algorithm, transferring an algorithm to a programming language, programming structures, modularisation, and; testing and debugging. Guided by (Xinogalos, 2016), these five issues were discussed based on the study on previous work as well as observations of more than 10 years in the world of programming education. The findings of the present observation are stated in the summary as shown in Table 1.

The following section will discuss the survey conducted on non-computer science students that focuses on the factors that effect on learning a program. This focused factor is related to the nature of the program which will show the environment of a programming language in giving effect in the programming learning process.

Table 1 Programming-Related Factors

Problem	Author(s)	Descriptions
Basic knowledge of Programming	(Chan Mow, 2008; Costa et al., 2012; Lahtinen et al., 2005; M, 2014; Xinogalos, 2016)	• Unable to transform the problem into a programming instruction. Most students may understand the syntax and semantics of individual statements, but they have no idea how to put them together into legitimate programmes. • Difficulties with language libraries, such as looking through them, finding the right function, and correctly using it in a programme. • Inability to combine syntax, logic, and concepts. Insufficient ability to translate problems into a charitable action plan. • A lack of understanding of effective instructional methods. • Difficulties in representing a program using notation. The symbols of a programming language, as well as the grammatical rules for assembling them into a programme, are referred to as notation.
Understanding the Structure of Programming	(Bosse & Gerosa, 2017; Chan Mow, 2008; Lahtinen et al., 2005; Qian & Lehman, 2017; Swidan et al., 2018; Wittie et al., 2017; Xinogalos, 2016)	• Failure to recognise that each command is carried out in the state generated by the preceding ones. • The complexity of comprehending the order of statements, the value of a variable, and the interactivity of an input action. • The most challenging programming concepts are pointers, arrays, and data structures.
Module Structure of Programming	(Bosse & Gerosa, 2017; Xinogalos, 2016)	• Students' difficulties in working with functions. • A lack of understanding of the scope of variables and why passing and returning arguments is required.
Testing and debugging	(Bosse & Gerosa, 2017; Chan Mow, 2008; Pears et al., 2007; Qian & Lehman, 2017; Siti Rosminah & Ahmad Zamzuri, 2012)	• Incapable of mastering the compiler, as well as error and warning messages • One of the difficulties is dealing with syntax mistakes, with the most typical issue being a lack of ability to discover faults. • Inability to visualise the status of the program during code execution. • Missing semicolons, mismatched parentheses, brackets, or quotation marks, and employing the illegal start of expressions.

Methodology

This study involved a total of 241 students who took programming courses at UiTM Cawangan Pulau Pinang. It consists of diploma and degree students from the Faculty of Mechanical Engineering (FKM) and the Faculty of Civil Engineering (FKA) as shows in Figure 1. Students are required to answer all questionnaires related to this course after they have completed the 14-week lecture.

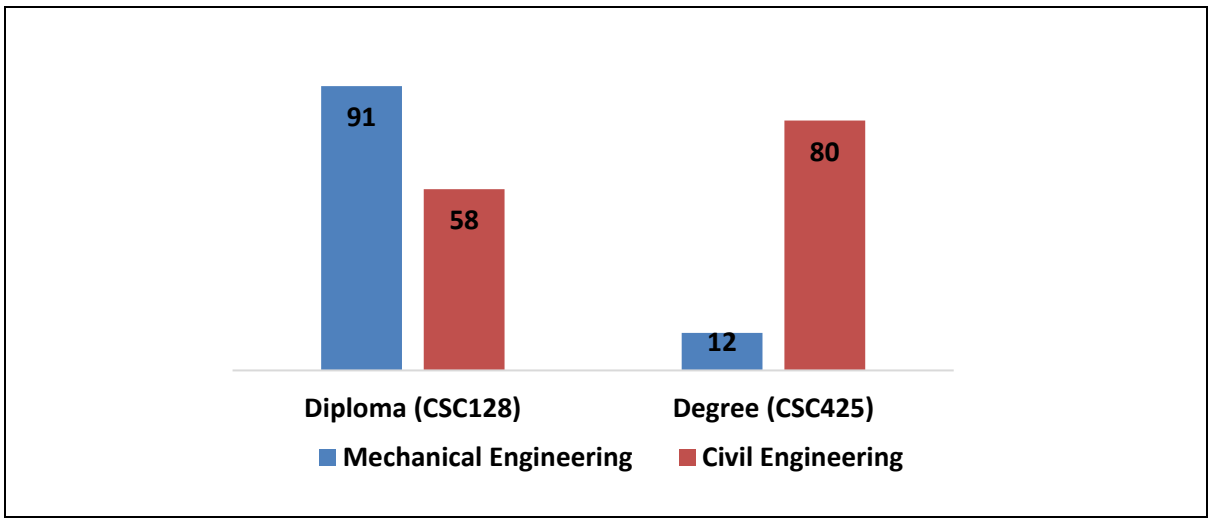

Figure 1. Number of Students in the Study

The questionnaire for the study consists of 2 parts as shows in Table 2 and Table 3. The first part is about the course information taken by students and the second part focuses on the factors related to the students on the programming course. All questions contain 15 items that focused more on Basic Element of Programming (4 items), Control Structure of Programming (5 items) and the Module Structure of Programming (4 items). These questionnaires use the five-point Likert scale. the programming ability-related factors used range from 5-Strongly Agree, 4-Agree, 3-Natural, 2-Disagree, and 1-Strongly disagree that values greater than 3 are positive and values less than 3 are negative statements.