International Journal of Engineering & Technology, 7 (4.33) (2018) 14-16

SPC

International Journal of Engineering & Technology

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

Research paper

Assisting Students' Understanding of Memory Location Concept through Visualization

Itaza Afiani Mohtar*, Normah Ahmad, Puteri Nor Hashimah Megat Abdul Rahman, Bohari Wahijan

Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Perak Branch, Tapah Campus, Tapah Road, 35400 Perak, Malaysia
**Corresponding author E-mail: itaza328@perak.uitm.edu.my*

Abstract

Learning programming for the first time is very difficult to many students. This difficulty negatively influences the students' interest in learning programming thus poses a challenge to the lecturers to maintain students' active involvement in learning. Students find it difficult to grasp the abstract concept of memory location, thus affects their understanding in writing programs. A memory location simulation program (MeLSim) is proposed to assist students with a realistic and visual experience of the abstract memory location concept. The objectives of this research are to develop a memory location simulation program and to determine students' understanding of the memory location concept after using the simulation. The students were given a pre-test and then required to use MeLSim for two weeks. They were then given a post-test. It was found that, there is significant difference on median total scores before and after using MeLSim. From the results, it can be concluded that students' using MeLSim improved their test scores. This research provides evidence that visualization can assist students in achieving better understanding of the lessons taught which in turn positively influence their test results.

Keywords: *algorithm visualization; novice programmers; memory location concept.*

1. Introduction

First year introductory programming students have difficulty understanding certain fundamental concepts in programming thus are quick to label them difficult [1-3]. Some of the difficulties which arose from programming complexity and ambiguity eventually contributed to the lack of motivation among them [4], and disengagement [5].

A study by [6-8] listed common syntax errors done by novice programmers when programming. Among them were 'type mismatch', 'cannot resolve identifier', 'missing;' and 'confusing the assignment operator (=) with the comparison operator (==)'.

In [9] suggested that syntax error concerning the assignment operator (=) may be attributed to the students' confusion with mathematical techniques. This error must be addressed because it involves the concept of memory location.

In every programming language, once a variable is declared, it will automatically be given a memory location with its address. Where and how to retrieve these data will immensely depend on the concept of memory location. In computing, memory location is associated to the memory address used at various levels by software and hardware. Memory address is unique for every data and it is determined by the computer's memory management and the operating system. Hence, each address holds the location of a particular data.

This conceptual knowledge is very important especially for novices in structured programming. This is because correct variable assignments depend on the students' conceptual understanding of how data is stored and manipulated in the computer memory.

2. Related Works

Visualizing the algorithm can be very helpful and useful in teaching programming. A study on the effectiveness of algorithm visualization (AV) [10] has been carried out onto five scenarios, namely AV for lectures, AV for study, AV for assignments, AV for class discussion, AV for labs, AV for office hours and AV for tests. The AV was achieved by using graphic representations and has proven that AV technique is able to engage the students in learning programming [11].

Similarly, interactive visualization [12] is a powerful education tool where it uses the Boolean model and the vector space model that allows the students to explore the processes step-by-step and with different parameters, thus gaining a deep understanding of the processes as well as the concepts behind it.

The scope of visualization in algorithms as well as the processes allows students to focus on the flow of the algorithm and its logic. It is essential and proven to be beneficial in the process of learning for the students, but it may not be for all levels of students. Some students can grasp the knowledge from the top-down approach, while some prefer otherwise.

In fundamental programming concepts, it can also be delivered to students via game development using Flash and ActionScript [13]. The "Game First" approach to C++ will help students to solidify the concepts easily. Hence, researchers can develop software applications such as a tool in learning programming. There are also several simulations done to make the students able to have a pictorial approach to programming.

Understanding the logic alone is insufficient in programming but one must also know what happens to the instructions in the computer memory. A robotic simulation done by [14] is said to make

students learn programming faster. It is called as Robot Virtual Worlds (RVW) and students are allowed to interact with it in understanding the algorithm, for example, "given the program, the robot will do _____", as well as in general programming, for example, "if the condition of an if statement is true, then all the of the codes inside its curly brackets will run: true/false". In other words, the robot will guide the students in learning programming. This is in line with the findings by [15], where game-based assignments will motivate students to create game strategy via programming for a challenging strategy-type game. As a matter of fact, learning to control robots will also motivate students to apply programming concepts learnt, thus making learning programming interesting and fruitful.

In line with increased handheld devices usage, visualization of algorithm has also moved into mobile territory [16].

Nevertheless, either learning programming via games or robotic simulation or visualization, it is undeniable that the role of concepts in program comprehension is very important. For novice programmers, it is essential to understand the concept of memory location in the computer in parallel to the processes of the algorithm.

3. Memory Location Simulation

In view of the positive impact of visualization in increasing novices' understanding, a memory location simulation program or MeLSim was developed to assist novice programmers in understanding the memory location concept.

MeLSim's window is divided into three panels. The leftmost panel (code panel) is used by the students to type source codes regarding variable assignments and mathematical operations which will change the memory content.

The rightmost panel (Memory Location panel) will display the content of the variable as the student traverse the codes, line by line.

The middle panel (Code Evaluation panel) enables the students to observe the changes to the contents of the variable in the Memory Location panel by clicking the 'Next' button. Each time the students' click the 'Next' button, the program will evaluate the next line of code. Figure 1 shows the screenshot of MeLSim.

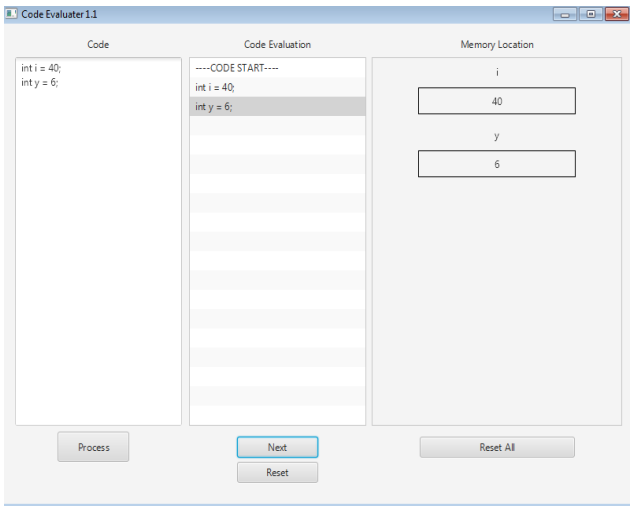

Fig. 1: Screenshot of MeLSim

The source code that can be entered in the code panel can include instructions like:

- Assignments
- Input
- Basic mathematical operations (+, -, *, /)

Variables can either be numeric or strings.

4. Methodology

An experiment was conducted involving 8 study groups with a total of 159 students for the subject Fundamentals of Computer Problem Solving, taught in the first year of diploma program. All the groups were given standard explanation on the topic regarding the memory location concept.

They were then given a pre-test which consists of 5 questions testing their understanding of the memory concept. The questions were adapted from [17]. Sample of the questions is shown in Figure 2.

Determine the contents of the variables after the execution of the following statements. 1. int a = 78; a = 100;	Answer:

2. int x = 60;
int y = 70;
x = x + y;	

Fig. 2: Sample questions for pre-test

After the pre-test, they were given instructions on how to use MeLSim. They were given 2 weeks to try out the program. After the two weeks ended, they were given a post-test with the same number of questions to the pre-test.

5. Results and Discussion

The median total score before (pre-test) and after (post-test) the students used MeLSim were then analyzed to see the difference. Table 1 shows an excerpt of the results obtained pre and post-test.

Table 1: Excerpt of the total score for pre and post-test

Group	Students	Total Score	
		Pre-Test	Post-Test
1	19	73	82
2	22	52	80
3	27	34	77

The differences between pre-test and post-test are not normally distributed (Figure 3), thus Wilcoxon Signed Rank Test was applied.

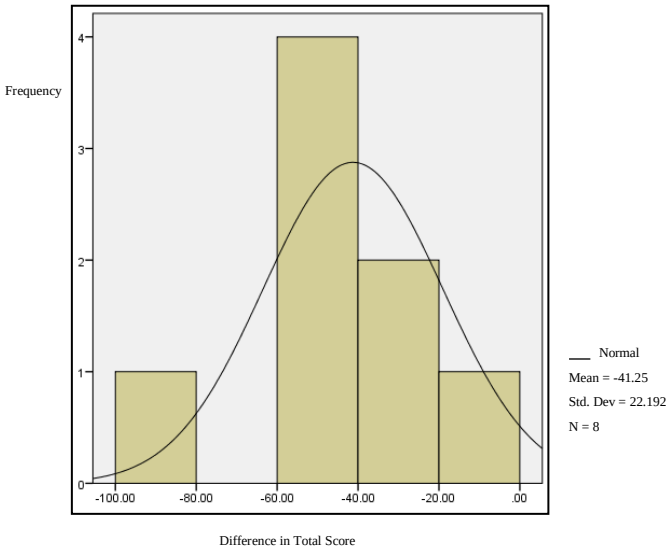

Fig. 3: Data distribution for the differences between pre and post-test

Based on Table 2, the total score of 8 groups were taken before and after using MeLSim. It was found that, there is significant difference on median total scores before and after using MeLSim with p-value 0.012 [z-statistic: -2.521]. Furthermore, it was found that the total score after using MeLSim was higher (median (iqr): 81.00 (41)) compared to total score before using MeLSim (median (iqr): 43.00 (55)).

Table 2: Median and IQR of total score

Variable	Median (IQR)		Z-Statistic	p-Value
	Before	After		
Total Score	43.00 (55)	81.00 (41)	-2.521	.012

6. Conclusion

From the results, it can be concluded that students' using MeLSim improved their test scores. The simple setting of MeLSim and the ease in which the students can self-explore the program may have contributed to this. It may also point to the possibility that the students' experience meaningful learning and engagement when using MeLSim. However, this is to be explored further in another study.

This study provides evidence that visualization can assist students in achieving better understanding of the lessons taught which in turn positively influence their test results.

MeLsim which is a simple tool to assist students' understanding in memory location concept, can further be expanded to include other basic programming structures. With this aid, it is hoped that students' will find that programming is not quite as complex and difficult as it seems.

Acknowledgement

This research was financially supported by UiTM Aras Grant, 600-IRMI/DANA 5/3/ARAS (0084/2016). The authors would also like to acknowledge the lecturers and students from the Faculty of Computer and Mathematical Sciences involved in the study.

References

[1] Guzdial M (2015), Learner-centered design of computing education: Research on computing for everyone. *Synthesis Lectures on Human-centered Informatics* 8(6), 1-65.

[2] Quintana C, Krajcik J, Soloway E, Fishman LC & O'Connor-Divelbiss S (2013), Exploring a structured definition for learner-centered design. *Proceedings of the 4th International Conference of the Learning Sciences*, pp. 256-263.

[3] Jenkins T (2002), On the difficulty of learning to program. *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences* 4, pp. 53-58.

[4] Kinnunen P & Malmi, L (2006), Why students drop out CS1 course? *Proceedings of the 2nd International Workshop on Computing Education Research*, pp. 97-108.

[5] Isiaq SO & Jamil MG (2018), Enhancing student engagement through simulation in programming sessions. *International Journal of Information and Learning Technology* 35(2), 105-117.

[6] Denny P, Luxton-Reilly A & Tempero E (2012), All syntax errors are not equal. *Proceedings of the 17th ACM Annual Conference on Innovation and Technology in Computer Science Education*, pp. 75-80.

[7] Hristova M, Misra A, Rutter M & Mercuri R (2003), Identifying and correcting Java programming errors for introductory computer science students. *ACM SIGCSE Bulletin* 35(1), 153-156.

[8] Gobil AR, Shukor Z & Mohtar IA (2009), Novice difficulties in selection structure. *Proceedings of the International Conference in Electrical Engineering and Informatics* 2, pp. 351-356.

[9] Kohn T (2017), Variable evaluation: An exploration of novice programmers' understanding and common misconceptions. *Proceedings of the ACM SIGCSE Technical Symposium on Computer Science Education*, pp. 345-350.

[10] Hundhausen CD, Douglas SA & Stasko JD (2002), A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages and Computing* 13(3), 259-290.

[11] Shaffer CA, Cooper ML, Alon AJ, Akbar M, Stewart M, Ponce S & Edwards SH (2010), Algorithm visualization: The state of the field. *ACM Transactions on Computing Education* 10(3), 1-22.

[12] Brusilovsky P, Ahn JW, Rasmussen E (2010), Teaching information retrieval with web-based interactive visualization. *Journal of Education for Library and Information Science* 15(3), 187-200.

[13] Leutenegger S & Edgington J (2007), A games first approach to teaching introductory programming. *ACM SIGCSE Bulletin* 39(1), 115-118.

[14] Liu A, Newsom J, Schum C & Shoop R (2013), Students learn programming faster through robotic simulation. *Tech Directions* 72(8), 16-19.

[15] Jiau HC, Chen JC & Ssu KF (2009), Enhancing self-motivation in learning programming using game-based simulation and metrics. *IEEE Transactions on Education* 52(4), 555-562.

[16] Marcelino M, Mihaylov T & Mendes A (2008), H-SICAS, A handheld algorithm animation and simulation tool to support initial programming learning. *Proceedings of the 38th ASEE/IEE Frontiers in Education Conference*, pp. 7-12.

[17] Dehnadi S (2006), Testing programming aptitude. *Proceedings of the 18th Annual Workshop of the Psychology of Programming Interest Group*, pp. 22-37