ASEAN Journal of Educational Research and Technology

Journal homepage: https://ejournal.bumipublikasinusantara.id/index.php/ajert

Review on Utilization of Visual Basic Software for Improving Students' Computer Programming Performances

Chuckee B. Gatchalian, Joy Edilaine R. Handig, Fioglo Baluyot*

National University Philippines, the Philippines
Correspondence: E-mail: reyes.joyedilaine@gmail.com

ABSTRACT

The main objective of this study is to review on utilization of visual basic software for improving students' computer ability. This study is a literature survey. We focused on two aspects: (i) utilization of visual basic software and (ii) computer programming performance. This study provided important information and benefits to the school, administrators, teachers, students, and researchers.

ARTICLE INFO

Article History:
Submitted/Received 23 Nov 2023
First Revised 16 Dec 2023
Accepted 20 Feb 2024
First Available online 21 Feb 2024
Publication Date 01 Dec 2024

Keyword:
Computer programming,
Visual basic.

1. INTRODUCTION

To improve teaching and learning, many applications have been added to the educational process. These apps have a significant impact on how well pupils learn the material they are studying. Numerous software tools are available for teaching and studying programming (Radošević *et al*., 2009). These software programs are necessary because programming environments and software are interdependent, and computers are needed as a platform to implement and validate programming languages (Amoako *et al*., 2013). The process of programming includes several steps, including designing, planning, testing, and debugging. Students must comprehend programming language syntax to learn how to create programs. Anger and a lack of desire to study programming are frequently caused by the intricacy of programming and the challenging logic of programs.

These problems play a part in the high dropout rate in programming classes that exist in most institutions, universities, and senior high schools today. Even in advanced programming classes, many students find programming to be challenging and discouraging, especially in the beginning. These days, senior high schools must use the right methods for teaching computer programming so that students can study using software. This software tool can help students learn computer programming more easily on their own.

Here, this paper is to review on utilization of visual basic software for improving students' computers. This study provided important information and benefits to the school, administrators, teachers, students, and researchers.

2. RESULTS AND DISCUSSION

This section presents the review of the related literature and study to evaluative report of information found in the literature related to the selected area of this study. This review should describe, summarize, evaluate, and clarify the literature and studies.

2.1. Utilization of Visual Basic Software

The expectations from academics and the industry, to have students and employees who are independent and capable of quickly writing code to resolve work-related issues, are growing high. However, teaching and learning visual basic programming software is certainly not easy and very challenging. The literature shows that a lot of work has been done to improve this. Nonetheless, little effect of this work has had an impact on the actual practice of teaching and learning software development programming skills. This gap has been addressed in this paper to enhance the teaching and learning process of programming to students. Furthermore, teaching programming literature research has been classified into 3 categories; teaching approach, teaching model, and teaching tool.

Hattie (2013) showed the meta-analyses of visual basic software applications in schools indicate that are used effectively when there is a diversity of teaching strategies when there is pre-training in the use of visual basic software applications as teaching and learning tools, when there are multiple learning opportunities (e.g. deliberative practice, increasing time on task), when a student, not teacher, is in "control" of learning, when peer learning is optimized and when feedback is optimized. In other words, Hattie claimed that the following conditions should be fulfilled to integrate visual basic software applications into the classroom; namely the role of the teacher, the need for professionalization, and the need for adapted teaching and learning approaches.

Yelland (2006) stated learning with a visual basic software application needs more than making learning activities digital, it is also about creating 'contexts for authentic learning that

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

use new technologies in integrated and meaningful ways to enhance the production of knowledge and the communication and dissemination of ideas. Obviously, concerning integrating visual basic software applications into the classroom setting, it is the teacher's main responsibility to facilitate this educational innovation.

Integrating innovative computer programs during classroom practices inevitably demands teachers to acquire new technological and pedagogical skills. The didactical use of computer programs is decisive for the learning process. Teachers need skills to be able to transform the learning content, the so-called Technological Pedagogical Content Knowledge (TPACK). However, it is known that most of the teachers integrate technology to provide content digitally, instead of using them to enhance learner-centered approaches.

Davies (1993) described the stages that reflect the level of visual basic application program integration among teachers. The five stages are familiarization, utilization, integration, reorientation, and revolution. While familiarization entails the stage where the teacher becomes aware of computer programs and their potential uses, (r) evolution refers to the stage where the visual basic application program is a learning tool that is seamlessly woven into the teaching and learning process. In other words, computer application software can be integrated from enhancing learning to transforming learning.

Giannakos, & Chorianopoulos (2015) concluded that despite challenges, using visual basic application software could be an effective and enjoyable learning experience and that there was no one pedagogical solution for all classes. The authors cited a range of instructional approaches used in the reviewed studies, including using visual basic programming languages to introduce young learners to programming and the use of authentic text-based languages. Game-making and physical computing were mentioned to motivate learners and use physical computing to explore concepts. Further, they suggested making creative products with much scaffolding to start with, combining kinesthetic activities with programming activities and the careful selection of educational contexts.

According to Du Boulay (1986) using visual basic software applications in teaching is a collaborative approach where two people work at one computer to complete a single design, algorithm, coding, or testing task. One person takes the role of the driver, having control over the keyboard and mouse, and the second person is the navigator or observer. The navigator constantly reviews the code written, keeps track of progress against the design, and continuously collaborates. While working on a task, the driver and navigator swap roles after a certain period, and the code is only changed with the agreement of both parties.

George (2002) concurred that there was evidence that visual basic software applications led to improved learning for both task outcomes and code quality in a university setting. They recommended further study of partner compatibility and investigation of the detailed information regarding how and why pair programming works or does not work. They suggested broadening research to investigate teaching programming in education of less strictly defined collaborative strategies.

Falkner & Vivian, (2015) concluded that using programmable robots to teach programming is not new. However, the emergence of the maker community and the development of low-cost educational microcontrollers and block-based programming languages have created renewed interest and new opportunities for teachers to consider. In line with these recent changes to the physical computing landscape, research has started to emerge, but it is fragmented and limited are starting to develop physical computing frameworks and approaches that can be built upon. However, further work is needed to validate these approaches in class settings.

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

According to the study of Radošević *et al*. (2009) from the Faculty of Organization and Informatics, University of Zagreb in Europe, it is recorded that Verificator is one of the best Educational Tools for Learning Visual Basic Programming. In teaching programming some specific problems concerning the teaching it as well as the organization of the teaching process need to be considered. One of the biggest problems is that students tend to adopt certain bad programming habits in their attempt to more easily deal with their examinations, such as trying to write programs without any syntax and logical checking. It is very hard to help them correct those errors once they are deeply rooted. Our students' web questionnaire and its results show that the majority of problems in learning programming among our students arise from the gap between the understanding of programming language syntax and problem-solving algorithms. The verification and proposed visual basic software application are the same in purpose to prevent students from making a lot of errors they are likely to make in learning programming and help them to learn visual basic programming language syntax and adopt good programming habits.

Based on the study of Esteves *et al*. (2011) from the Polytechnic Institute of Leiria in Portugal, it is written regarding how to improve the teaching and learning process on computer programming through the use of the Second Life virtual world. The emergence of new technologies such as three-dimensional virtual worlds brings new opportunities for teaching and learning. They conducted an action research approach to the analysis of how teaching and learning computer programming at the university level could be developed within the Second Life virtual world. Its results support the notion that it is possible to use this environment for better effectiveness in the learning of programming. The main results are the identification of problems hampering the teacher's intervention in this virtual world and the detection of solutions for those problems that were found effective in the success of using this environment for teaching/learning computer programming. Study observing students' apprenticeship in the Second Life virtual world and teachers' experience, to analyze how the processes of teaching and learning computer programming occur within this virtual world.

Kessler and Anderson (1986) investigated the use of Pencil Code, a hybrid visual basic software application, where learners can switch between block and text modes. They concluded that novice programmers used both modalities throughout their programming experience. They also reported that all learners started with the block-based mode first; with some quickly moving to text while others stayed in block-based mode, and all learners completed the tasks given. Winthrop and Holbert suggested that the opportunity to switch between modalities provided a means by which all learners could participate while keeping the more experienced programmers engaged.

A study by Kurland *et al*. (2013), a small-scale evaluation of Stride, a visual basic programming software application, concluded that learners using Stride completed tasks set more quickly than those writing Java using a traditional editor. They also reported that Stride users spent less time on syntactic it's to their code and significantly less time with non-compellable code. The study was limited due to sample size, sample selection, and the short-term nature of the out-of-school study. The authors called for further research on the relationship between block, frame, and text editors, how the transition might be mediated, with a focus on long-term effects and impact on learning gains, as well as a more in-depth investigation of learner perception of the editors.

From all the aforementioned examples it can be concluded that visual basic software applications are very useful in teaching programming to novice students, primarily because they can show and explain programming concepts in a very simple way. However, their main

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

disadvantage is that the majority of them are focused on only one programming language and the simplest program constructs. Furthermore, the aim of teaching programming is that students understand basic programming concepts, and they become able to apply those concepts during the problem-solving process, regardless of the implemented programming language. Unfortunately, most of these tools are too focused on the visualization component and less on learning the syntax and semantics of the selected programming language.

Therefore, the visualization tools are more appropriate for teaching programming in elementary schools, but not at universities where the student needs to learn algorithmic approaches to solve given problems. Finally, the results of research that has followed the use of most of the aforementioned tools have not revealed any significantly better results and students still experience the same difficulties when learning basic programming. Therefore, the researcher developed a proposed visual basic software application to determine the effectiveness and efficiency for the students, an environment in which students will learn strategies they will use in solving problems and developing the necessary concepts and skills to create computer programs.

Hassan *et al*. (2006) investigated the application of Visual Basic Computer Programming Language to simulate some secondary school Mathematics which is computational by nature. The significance of Visual Basic as a Programming Language and the headache faced when solving secondary school Mathematics analytically led to the design and implementation of this system. The structure of examination in Ghana especially, the objective test organized by West African Examination Council (WAEC) is mostly computational. Therefore, students need programming skills to be able to compute solutions faster. The latest technologies developed from landmark inventions have made our life quite easy, especially computers remain the pioneer in this regard. Advancements in computing architecture, applications, and programming languages helped in solving ample numerical problems.

Hohenwarter *et al*. (2009) studied and introduced dynamic mathematics visual basic software to secondary school teachers stating that Visual Basic as a programming language is mostly applied to mathematics to solve real-life problems. Mathematical programming plays a prominent role in applications and is an area in which two separate disciplines have arisen. First, there is the algorithm and mathematical properties discipline. People working in this area are interested in the theoretical and computational properties of mathematical programming solution techniques. Computer program-associated forms are frequently used by several simulation models to execute logical operations in a sequential manner that is scheduled by the program. Hence, secondary school students can now apply their analytical and algebraic techniques alongside computation to solve mathematical problems.

A study showed learning challenges faced by novice programming students studying high-level and low-feedback concepts. Visual basic programming is one of the fundamental skills to be adopted by students of information sciences at the end of the first year of their study.

It is also expected that they will know basic programming concepts and have developed an algorithmic approach to solving problems. Visual basic programming concepts include the knowledge of control structures (sequence, selection, and iteration), mechanisms of aggregation (array, structure, union), pointers, functions, etc. When teaching visual programming, we aim to teach novice students the basic principles of using programming concepts regardless of the programming language. However, goals are sometimes hard to meet in a real-life context.

The research which was conducted at the multi-national level has shown those novice students, after completing and passing their courses in which they learn the basic concepts of programming, still encounter problems creating simple program solutions. This is probably

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

accounted for by the fact that novice programmers do not have enough experience or skills regarding which programming constructs to use or where and how to use them. Furthermore, students are often faced with syntax errors. It is common for a small set of identical mistakes to occur regularly with all novice students (Morgan, 2017). To help students in their first steps in learning programming, many visualization and animation software systems are developed.

The development of mathematics teaching materials in the material system of two-variable linear equations using Microsoft Visual Basic media was developed by Widayanti, Yuberti, Irwandani, and Hamid in 2018 taking into account the Standard of Competence due to mathematics learning in the two-variable linear equation system material for eighth-grade students of the middle school. Teaching materials developed have gone through the validation stage by material experts, media experts, and in trials on students at Public Middle School 12 Bandar Lampung.

The quality of teaching materials has reached the feasibility standard of teaching materials from the results of the assessment of material experts, media experts, educational practitioners, and students. Researchers are trying to develop computerized-based teaching materials designed as attractive as possible. In supporting the development of teaching materials, Widayanti, Yuberti, Irwandani & Hamid used an application Visual Basic 6. It is one of the software applications in Windows. In developing applications, Visual Basic uses a visual approach to design user interfaces of forms, while coding uses basic languages that tend to be easily understood following mathematics learning in the two-variable linear equation system material for eighth-grade students of Middle School. These teaching materials have a validation stage by material experts, media experts, and in trials on students at Bandar Lampung 12 Public Middle School. The quality of teaching materials has reached the feasibility standard of teaching materials from the results of the assessment of material experts, media experts, educational practitioners, and students. It can be concluded that mathematics teaching materials using visual basic media that have been developed in this study are worthy of being used as supporting teaching materials in the teaching and learning process.

Corman *et al*. (2015) stated that a better understanding of cognitive style and cerebral dominance provides for greater productive information systems. Hudak and Anderson (2011) studied information regarding computer science courses and emphasized the need to examine students' cognitive maturity and learning style factors often ignored in research aimed at ascertaining the reasons for academic success at the college level. The study also highlighted the need to examine both cognitive maturity and learning style in the studies of academic success at the college level (Hudak & Anderson, 2011). Such research enhances industry training and academic teaching. Prior cognitive research has been with procedural and object-oriented languages, such as Basic, Pascal, C++, and Java. This research focuses on the cognitive style that is involved with the programming aspects of Visual Basic.

The application of Visual Basic Computer Programming Language to Simulate Numerical Iterations, the merit of Visual Basic as a Programming Language, and the difficulties faced when solving numerical iterations analytically, this research paper encourages the use of Computer Programming methods for the execution of numerical iterations and finally fashion out and develop a reliable solution using Visual Basic package to write a program for some selected iteration problems.

A study investigated the cognitive characteristics of learning a visual basic software. It replicated an earlier study by White (2012), it used visual basic software, an object-oriented programming language. The two cognitive characteristics investigated in this research with visual basic were 1 cognitive development, as measured by the proposition logic test, and 2 cognitive hemispheric style, as measured by the Hemispheric Mode Indicator (HMI). Prior

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

research has shown that object-oriented and procedural programming involves a high level of cognitive development and that procedural programmers are left-brain hemispheric-style thinkers. The findings from this study using a visual basic software, contradicts prior research with other programming paradigms. This study found that visual basic software was being left cognitive hemispheric style, just like procedural programming. However, cognitive development was unimportant. While procedural and object-oriented languages require a high cognitive development level, visual basic software requires a lower level. This supports the theory that different programming language paradigms require different cognitive characteristics.

Visual basic software, the “project development approach” is adopted to design teaching and the teaching order is adjusted to implement the principle of “problem-guide, project-driven, practice-based, stressing training combined theory enough for the degree”. The results of enhancing real combat training, diluting the proof-of-experiments, and strengthening the design-based experiments are making students master the basic method of programming design through practice project training and familiar with the basic software development process. Furthermore, the students' analyzing and solving problem skills have been enhanced and the students' innovative practice ability has been improved.

The findings and conclusions from this study establish a foundation in the research of programming languages' influences on cognitive style. This study indicates students need to have the correct cognitive style to succeed in a VB programming course. Such a course does not change cognition to the correct thinking style. To argue that allowing any student into programming, because they will develop the cognitive style needed, is a mistake. Students must already have the needed cognitive style to succeed in programming. Students placed in classes that best fit their cognitive style have a higher probability of success. As stated by White (2012), it is well known regarding the implication in making programming courses. It needs prerequisites.

2.2. Computer Programming Performance

Programming is a fundamental concept in computer science and a good understanding and mastery of programming is fundamental to becoming successful in studying any computing discipline. Most importantly, a practical knowledge of computer programming is a prerequisite for higher-level computing courses. Such practical knowledge coupled with other knowledge and skills is also required by industry for employment. Nowadays, even university students enrolled in non-computing programs are also interested in taking programming courses as digital competence becomes a necessity in all disciplines.

Despite its importance, the literature shows a high failure rate in introductory programming courses Gomes & Mendes (2016) and the university where this study took place is no exception. Consequently, researchers have been examining the factors related to success in programming. For example, Bennedsen & Caspersen (2014) investigated potential success factors for an introductory programming course in Denmark and found that mathematics grade from high school was one indicator. In their study in Ireland, Bergin & Reilly (2011) found that mathematics and science scores from high school have a strong correlation with performance in programming.

The predictors of success such as prior programming experience, mathematical ability, academic and psychological variables, gender, comfort, and student effort for an object-first programming course. The findings showed that prior programming experience, mathematical ability, and gender were not predictors of success, whereas student effort and comfort level were found to be the strongest predictors of success.

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

Sharma & Shen (2018) conducted a comparative study between two universities in India and Australia to investigate whether education culture factors affect performance in programming. Their findings showed that prior programming experience, gender, the reason for studying programming, attendance, and revision had different effects on the performance of students in the two universities. For example, for Australian university students, those who had prior programming experience performed better than those who did not. For the Indian university students, prior programming experience did not show any effect on the student's performance.

Similarly, for the Australian university students, there was no difference in programming performance between males and females, whereas for the Indian university students, there was a statistically significant difference between males and females. On the other hand, factors such as activities performed in the lecture theatre and preparation before lecture and laboratory had the same effect for both universities.

Their findings give some insight into whether some factors could be associated with education culture. Education culture in their study refers to parameters such as teaching methodology, assessment criteria, attendance criteria, laboratory setup, and exam structure.

Even though many studies investigated factors affecting programming performance, the findings show diverse factors and, in some cases, even different results for the same factors. This difference could be attributed to many factors such as environment, culture, teaching methodology, course structure, type of assessment, instructor, student, and external factors.

This makes it difficult to apply the results of one study in other contexts. For example, prior experience in programming is widely cited as a predictor of success in university-level programming courses. However, the programming experience assumed is usually a high school programming experience that varies considerably between the West and developing countries. Observing this situation, Bennedsen & Caspersen (2014) indicated that factors that affect performance in programming courses vary from institution to institution.

This variation includes the difference in programming styles used in senior secondary schools and introductory programming courses at the university level. With this in mind, Apiola & Tedre (2012) advocated for more research investigating factors affecting students' performance in programming courses in developing countries as the factors that affect performance in introductory programming could vary between different educational contexts.

The background of students, their fields of study, and learning approaches applied to the study of programming courses. It is worth considering as a major factor and necessary to research the causes of failure of students in programming courses from the learner's perspective. Programming courses form part of the core concentration areas for students especially those studying Information Technology (IT) and Computer Science as well as those other fields of study sandwiched with IT in an undergraduate degree program. Through the use of questionnaires, interviews, and focused groups, a survey was conducted using one hundred (100) students at the middle and end of the semester. The responses from the three groups of students were compared. Their opinions on the usefulness of their background, field of study, and learning approaches toward programming courses were investigated. The needs and concerns about these key factors are highlighted in the survey and discussed thereby leading to the inferences made and then proposed recommendations on the learning approach concerning the background and field of study of students in computer programming courses to improve understanding of programming by students, hence, reducing failure rates.

The impact of the difference between the mother tongue as first language and the medium of instruction, English in this case on programming performance. Their findings show that

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

students whose first language was the same as the language of instruction performed better than those students whose mother tongue was not English. A study by Butler and Morgan (2017) found a correlation between students' ability in English and performance in programming did not show any significant correlation between English, foreign language, and programming result. We can observe that findings about correlations between the English language and performance in programming are not in agreement. In addition, there are no adequate justifications provided as to why there is a correlation or not.

Another factor extensively studied in the programming performance is gender. Gender is one factor that can predict success in introductory programming courses in the USA found that gender did not affect student success in introductory programming. On the other hand, Lau & Yuen (2011) found that gender, learning style, and mental model have an influence in programming performance. A comparative study by Sharma & Shen (2018) between an Indian and an Australian university revealed that, while there was no gender difference in the performance of Australian university students, a statistically significant difference between male and female students in the Indian university emerged. Gender as a success factor appears to be inconclusive as some studies found correlations while others did not.

In a study conducted to investigate the correlation between prior academic performance and performance in the first year of programming, a strong correlation between science examination scores and performance in programming. In their analysis, they used the highest score from physics, chemistry, and general science subjects as a science score since the number of students who had scores in the individual subjects was small. Their explanation for the strong correlation between science and programming scores is that both subjects are laboratory-based and an experience in science subjects might have helped in the programming course. In a similar study, Bergin & Reilly (2011) found a strong correlation between high school leaving examination results in physics and biology and first year programming results. However, no correlation was found between chemistry and programming scores and the authors indicated the need for further research to understand the reason behind the lack of correlation between chemistry and programming.

As computing as a subject is increasingly being offered at the high school level, studies have attempted to see to what extent such experience helps in introductory programming courses. Several studies have demonstrated a positive relationship between previous computer experience, especially programming, and success in a programming course. Moreover, studies have shown that students who have had prior experience using a computer before doing a programming course have successfully executed programming.

Some investigated whether South African students' grade 12 information technology performance can be a predictor of performance in computer programming at the university level. The results showed a moderate correlation between the information technology marks in Grade 12 and university-level programming course grades. Similarly, Morrison and Newman (2015) found that students who had prior programming experience, in their case in languages such as Pascal and Microsoft Visual Basic, performed better in their university-level programming than students with no prior computer programming experience. They concluded that a course that emphasizes procedural programming aspects may work well as a preparation for introductory programming focusing on procedural programming. A study by Watson and Godwin (2014) also found a significant difference in programming performance between students who had prior programming experience and those who did not have any prior programming experience. The programming language used in the introductory programming was indicated to be Java together with BLueJ and the students who had prior experience in programming had used Java too.

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

On the other hand, the prior programming experience in imperative programming has no influence on university-level object-oriented programming performance using Java. The introductory programming course used in the study focused on graphical design-centric objects-first programming. The discrepancy is attributed to the mismatch between the programming approaches used at secondary school and those at the university level. They concluded that imperative programming and object-oriented programming have different predictors of success. Similarly, no correlation between prior programming experience in imperative languages and university-level object-oriented programming in Java. They consequently suggested that the traditional notion of success factors for object-oriented programming needs to be reassessed. Based on their study of the cognitive shifts from procedural programming to object-oriented programming. The assumption that results of procedural programming education apply to the learning of object-oriented programming is not founded on research results. They called for systematic research into the fundamental cognitive and educational issues in learning object-oriented programming.

3. CONCLUSION

This study's primary goal is to review the use of Visual Basic software to enhance students' computers. This study is a survey of the literature with an emphasis on two areas: (i) computer programming performance and (ii) the use of Visual Basic software. The school, administrators, teachers, students, and researchers all benefited from the significant information and advantages this study offered.

4. AUTHORS' NOTE

The authors declare that there is no conflict of interest regarding the publication of this article. The authors confirmed that the paper was free of plagiarism.

5. REFERENCES

Amoako, P. Y. O., Sarpong, K. A. M., Arthur, J. K., and Adjetey, C. (2013). Performance of students in computer programming: Background, field of study and learning approach paradigm. *International Journal of Computer Applications*, *77*(12), 17-21.

Apiola, M., and Tedre, M. (2012). New perspectives on the pedagogy of programming in a developing country context. *Computer Science Education*, *22*(3), 285–313.

Bennedsen, J., and Caspersen, M.E. (2014). Failure rates in introductory programming. *SIGCSE Bulletin*, *39*(2), 32–36.

Bergin, S., and Reilly, R. (2011). Programming: Factors that influence success. *SIGCSE Bulletin*, *37*(1), 411–415.

Butler, M., and Morgan, M. (2007). Learning challenges faced by novice programming students studying high level and low feedback concepts. *Proceedings Ascilite Singapore*, *1*, 99-107.

Corman, L. and Guynes, C. and Vanecek, M (2015). A study of laterality and cognitive style in information systems and liberal arts students. *Journal of Computer Information Systems*, *35*(2), 26-34.

Davies, S. P. (1993). Models and theories of programming strategy. *International Journal of Man-Machine Studies*, *39*(2), 237-267.

Du Boulay, B. (1986). Some difficulties of learning to program. *Journal of Educational Computing Research*, *2*(1), 57-73.

Esteves, M., Fonseca, B., Morgado, L., and Martins, P. (2011). Improving teaching and learning of computer programming through the use of the second Life virtual world. *British Journal of Educational Technology*, *42*(4), 624-637.

Falkner and Vivian, (2015). Problems-based learning for foundation computer science courses. *Computer Science Education*, *10*(2), 109-128.

George, C. E. (2002). Using visualization to aid program construction tasks. *ACM SIGCSE Bulletin*, *34*(1), 191-195.

Giannakos, and Chorianopoulos (2015). Pedagogical changes in the delivery of the first-course in computer science: Problem solving, then programming. *Journal of Engineering Education*, *87*, 313-320.

Gomes, A.J., and Mendes, A. (2016). A study on student performance in first year CS courses. Paper presented at the 15th annual conference on Innovation and technology in computer science education, Bilkent, Ankara, Turkey. *South Africa Computer Journal*, *46*, 14–23.

Hassan, A. B., Abolarin, M. S., and Onawola, H. J. (2006). The application of Visual Basic computer programming language to simulate numerical iterations. *Leonardo Journal of Sciences*, *2006*(9), 125-136.

Hattie J. (2013). Design principles for authoring dynamic, reusable learning objects. *Australian Journal of Educational Technology*, *19*(1), 46-58

Hohenwarter, J., Hohenwarter, M., and Lavicza, Z. (2009). Introducing dynamic mathematics software to secondary school teachers: The case of GeoGebra. *Journal of Computers in Mathematics and Science Teaching*, *28*(2), 135-146.

Hudak, M. A. and Anderson, D. E. (2011) Formal operations and learning style predict success in statistics and computer science courses. *Teaching of Psychology*, *17*(4) 231- 234.

Kessler, C. M., and Anderson, J. R. (1986). Learning flow of control: Recursive and iterative procedures. *Human-Computer Interaction*, *2*(2), 135-166.

Kurland, D. M., Pea, R. D., Clement, C., and Mawby, R. (2013). A study of the development of programming ability and thinking skills in high school students. *Studying the Novice Programmer*, *2013*, 83-112.

Lau and Yuen (2011). A multi-national, multi-institutional study of assessment of programming skills of first-year CS students. *SIGCSE Bulletin*, *33*(4), 125-180.

Mayer, R. E., Dyck, J. L., and Vilberg, W. (1986). Learning to program and learning to think: What's the connection?. *Communications of the ACM*, *29*(7), 605-610.

Morgan M. (2017). Teaching eiffel as a first language. *Journal of Object-Oriented Programming*, *9*, 30-41.

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860

Morrison and Newman (2015). Difficulties in learning and teaching programming: Views of Students and Tutors. *Education and Information Technologies*, *7*(1), 55-66.

Radošević, D., Orehovački, T., and Lovrenčić, A. (2009). Verificator: Educational tool for learning programming. Radošević, D., Orehovački, T., Lovrenčić, A:" Verificator: Educational Tool for Learning Programming", *Informatics in Education*, *8*(2), 261-280.

Sharma, R., and Shen, H. (2018). Does education culture influence factors in learning programming: A comparative study between two universities across continents. *International Journal of Learning, Teaching and Educational Research*, *17*(2), 1–24.

Watson, Li and Godwin (2014). Exploring the role of visualization and engagement in computer science education. *SIGCSE Bulletin*, *35*(2), 131-152.

White, G. (2012). Visual basic programming impact on cognitive style of college students: need for prerequisites. *Information Systems Education Journal*, *10*(4), 74-83.

Yelland (2006). Lifting the hood of the computer: Program animation with the Teaching Machine. *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, *2*, 831-835.

DOI:
p- ISSN: 2828-4887 e- ISSN: 2828-4860