# The moderating effect of logic in the learning of C++ computer programming using screencasting

**Chin-Soon Cheah [1], Lai-Mei Leong [2]**

[1] ccs.cscheah@gmail.com
School of Educational Studies, Universiti Sains Malaysia, Malaysia

[2] lmleong@usm.my
School of Educational Studies, Universiti Sains Malaysia, Malaysia

http://dx.doi.org/10.17220/mojet.2019.02.006

## ABSTRACT

Difficulties in the learning of computer programming have been a universal problem. One of the main contributors of this problem is due to the teaching material used in guiding the students. Traditional teaching method using books and PowerPoint slides are not effective enough to support the dynamic nature of computer programming. Hence, a more effective teaching material such as screencasting is required to support the learning of computer programming. This study examined the moderating effect of logic in the learning of C++ computer programming using screencasting among undergraduates. A true experimental pre-test and post-test research design was conducted involving 65 first-year undergraduate students (aged 19 – 22) who have never attended any formal computer programming course prior to the study. These students were randomly assigned to two groups: the screencast and narration (SN) mode group and the screencast, text, and narration (STN) mode group. Results showed that the different levels of logic amongst the students in the two treatment modes did not have any effect in the learning of C++ computer programming.

**Keywords:** *C++ computer programming, logic, screencast, multimedia*

## INTRODUCTION

Globally, learning computer programming has been a universal problem in the computer science curriculum as it has high failure rates and withdrawals during the initial stage of introductory computer programming courses (Butler & Morgan, 2007; Robins, Rountree, & Rountree, 2003). There are many contemporary static teaching materials available but none of them is effective in explaining the dynamic nature of computer programming. Moreover, logic has been recognized as one of the important cognitive aspects in understanding computer programming (Galton, 1992; Gibbs & Tucker, 1986; Gomes & Mendes, 2007; G. L. White & Sivitanides, 2002). Past researches have shown that the development of higher cognitive abilities in terms of logical reasoning is important in understanding computer programming (Eckerdal, Thuné, & Berglund, 2005; Fletcher, 1984; Hudak & Anderson, 1990; Little, 1984; G. White & Sivitanides, 2003). The usage of programming controls, such as the selection

structure statement called the "if" statement, determine an action to be executed when a condition is met. This requires biconditional reasoning which is a precondition to formal operational reasoning (Gomes & Mendes, 2007; Ismail, Ngah, & Umar, 2010b; Lawson, 1983). The ability of operational reasoning is further needed in developing the understanding of propositional logic (Brainerd, 1978; Enyeart, 1983; Irons, 1982). Propositional logic involves the logic of truth table and this has a direct analogy in computer programming (Galton, 1992; Little, 1984). Besides that, the ability to write subprocedures for a computer program correlates significantly with analogical reasoning (Clement, Kurland, Mawby, & Pea, 1986). Therefore, a more comprehensive and effective material that can explain effectively the concepts of programming is needed to overcome the problem (Jenkins, 2002).

The purpose of this study is to examine the moderating effect of logic in the learning of C++ computer programming using screencasting. This study also looked into the effectiveness of using screencasting in learning C++ computer programming.

## LITERATURE REVIEW

Logic has been one of the important aspects in understanding computer programming and past researches have shown that there are positive correlations between computer programming and cognitive abilities such as general reasoning (Hudak & Anderson, 1990; Wegerif, 2002; G. L. White & Sivitanides, 2002). Logic is needed to support the mapping of various logical elements and explanation of programming concepts within the program control in computer programming structure. In a programming control such as "if" statements, it requires biconditional reasoning such as "if and only if " logic which is a precondition to formal operational reasoning (Gomes & Mendes, 2007; Ismail, Ngah, & Umar, 2010a). Furthermore, most of the programming concepts require spatial visualisation to understand how the program flows and leads to the final output of the overall program. In addition to that, programming requires high cognitive abilities to support problem solving skills by determining the logic flow of a program (Dalbey & Linn, 1985; Hudak & Anderson, 1990).

According to Piaget's Theory of Cognitive Development (Epstein, 1990; Piaget, 1972), logical thinking skills are developed in the formal operation level. In this level, an individual develops abilities to deal with abstraction, solve problems systematically, form hypotheses and engage in mental manipulations (Bichler & Snowman, 1986). Hence, researchers like Eckerdal et al. (2005) and Galton (1992) find that the ability in logic reasoning is essential in understanding computer programming. Moreover, the development of appropriate truth table requires the ability of propositional logic that has a direct analogy in computer programming (Folk, 1994; Galton, 1992).

Based on the plethora of research available, teaching materials containing multimedia elements which are able to explain the dynamic nature of computer programming in terms of logic and support spatial visualisation are needed (Berk, 2009; Evans & Gibbons, 2007; Fluck, 2001; Jonassen, 2004; Malik & Agarwal, 2012; Mayer, Dow, & Mayer, 2003). Thus, screencasting would be ideal as it is able to embed multimedia elements as well as provide dynamic elements to explain the nature of computer programming (Carter, 2012; Murphy & Wolff, 2009). Moreover, it has the ability to provide visual and audio explanation simultaneously. For instance, animated handwritten step-by-step solutions involving complex calculation and equation that determine the logic decision can be shown to support the spatial visualisation elements (Mohorovičić & Tijan, 2011). Besides that, screencasting can be used as an additional teaching material to complement the existing teaching material such as books and written instructions. This will not only increase students' understanding, but also enhance the learning experience when it comes to a subject like computer programming

which requires visualization and spatial imagination (Mohorovičić, 2012). In a research conducted by Mohorovičić and Tijan (2011) at University of Rijeka, screencasting has been shown to produce substantial positive results in the learning of 'Algorithm and data structures', which are the basic building blocks of creating the logic flow of computer programs, in programming courses. Students expressed their satisfaction and highlighted that the video explanations of the concepts, syntax and algorithms of C++ increased their learning pace and understanding. The nature of screencasting enables the explanation of problem-solving skills and the demonstration of problem solving process. Furthermore, the on screen spatial queue and audio narration can overcome the weakness of the conventional static text base material of a book.

To further enhance the effectiveness of the screencasting, developers can opt to follow Mayer's Cognitive Theory of Multimedia Learning (Figure 1) when designing the screencast.

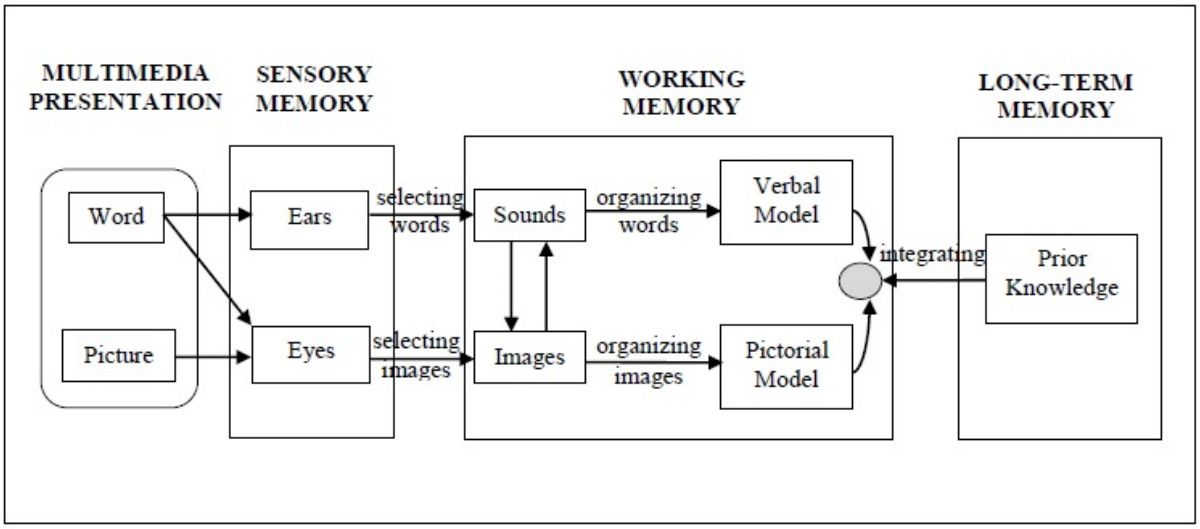


***Figure 1*** Cognitive Theory of Multimedia Learning (Mayer, 2002)

According to Mayer (2005), in implementing a multimedia learning environment, a learner must engage in five cognitive processes: (i) selecting related words for processing in verbal working memory, (ii) selecting related images for processing in visual working memory, (iii) organising the selected words into a verbal model, (iv) organising the selected images into a pictorial model, and (v) integrating the verbal and pictorial representations with prior knowledge. Logic, which is located in the prior knowledge, is integrated with the verbal and pictorial model for effective learning.

Therefore, screencasting would be an ideal teaching material in the subject of computer programming. This is because it has all the multimedia elements to support dynamic elements which are used to explain the nature of computer programming. Furthermore, concurrent visual and audio explanations are able to enhance the learning experience by preventing the overloading of either one of the visual or audio channel. For instance, animated handwritten step-by-step solutions involving complex calculation and equation that determine the logic decision can be shown to support the spatial visualisation elements. In addition, screencasting can be used to complement the existing static teaching materials such as books and written instructions. To further increase the effectiveness of the screencasting, Mayer's Cognitive Theory of Multimedia Learning (Figure 1) can be used as a guideline

when designing the screencast to prevent the overloading of channels, and effectively engaging in the five cognitive processes.

## METHODOLOGY

The purpose of this study is to evaluate the effectiveness of screencasting on the learning of C++ computer programming and to determine whether logic has a moderating effect. The participants consisted of 65 first-year undergraduate students (19 – 22 years old) who have never attended any formal computer programming course before. They were randomly assigned to one of two types of learning modes, namely, screen and narration (SN) mode, and screen, text and narration (STN) mode. The SN mode presented the information as screen recording and narration, while the STN mode presented the information as screen recording, text, and narration. The main difference between these two learning modes was that the STN mode provided the narration in text form as well which appeared at the bottom of the screen.

Before the participants were allowed to view the screencast, they answered a C++ computer programming pre-test and the Group Assessment of Logical Thinking (GALT) test. The pre-test consists of 30 multiple-choice questions based on the topic of Introduction to C++ Computer Programming. The GALT test was adopted from Roadrangka, Yeany, and Padilla (1983) and was used to determine the level of logical thinking of each respondent and to classify the respondent into high and low logic groups. Based on the participants' logic levels, they were then randomly assigned to either the SN or STN mode so that each mode contained both high and low logic participants. Table 1 shows the distribution of the participants in each mode. They viewed the screencast one hour a week, for five consecutive weeks. Each week, a new C++ computer programming topic was introduced. The viewing session for the SN and STN modes were isolated to prevent the participants from communicating. At the end of the final week, a C++ computer programming post-test was conducted to assess the students' performance. The questions in the post-test were the same as the pre-test, except for the sequence of the questions which were jumbled up. This was to prevent the students from memorising the answers from the pre-test. The test had a Kuder-Richardson, KR20, of 0.78, indicating that the internal reliability of the test was at an acceptable level.

**Table 1**

*Distribution of Low/High logic groups based on learning modes*

| Mode | Total | Low logic | High logic |
|---|---|---|---|
| | | n (%) | n (%) |
| **SN** | 33 | 14 (42.42) | 19 (57.57) |
| **STN** | 32 | 13 (40.62) | 19 (59.37) |

**Data Analysis and Findings**

The mean and standard deviation of the C++ computer programming pre-test and post-test scores of the participants in the SN and STN modes are shown in Table 2. It clearly shows both the pre-test and post-test mean scores in the SN mode are higher than the STN mode.

**Table 2**

*Mean and Standard Deviation of the C++ Computer Programming Pre-test and Post-test Scores*

| | | C++ Computer Programming Pre-test | | C++ Computer Programming Post-test | |
|---|---|---|---|---|---|
| **Mode** | **N** | **Mean** | **Std. Deviation** | **Mean** | **Std. Deviation** |
| **SN** | 33 | 14.24 | 4.98 | 24.27 | 2.60 |
| **STN** | 32 | 14.03 | 5.63 | 23.63 | 3.09 |

A two-way ANCOVA was conducted to determine any significant difference in the participants' achievement in learning C++ computer programming, as measured by their post-test scores, between those using SN and STN modes. The pre-test scores were set as the covariate. The results, as shown in Table 3, indicate that there is a significant difference in the participants' achievement in learning C++ computer programming between those using SN and STN modes, $F(1,60) = 19.28$, $p = .00$, $\eta^2 = .24$. Since the post-test mean score of the SN mode is higher than the STN mode, as can be seen in Table 2, it can be concluded that the participants in the SN mode learned C++ computer programming more effectively than those in the STN mode.

**Table 3**

*Two-Way ANCOVA of C++ Computer Programming Post-test scores of participants in SN and STN modes*

| **Source** | **Type III Sum of Squares** | **df** | **Mean Square** | **F** | **Sig.** | **Partial Eta Squared** |
|---|---|---|---|---|---|---|
| **Corrected Model** | 300.047[a] | 4 | 75.012 | 12.158 | .000 | .448 |
| **Intercept** | 3192.356 | 1 | 3192.356 | 517.401 | .000 | .896 |
| **Mode** | 118.984 | 1 | 118.984 | 19.284 | .000 | .243 |
| **C++ Computer Programming Pretest** | 90.414 | 1 | 90.414 | 14.654 | .000 | .196 |
| **Error** | 370.199 | 60 | 6.170 | | | |
| **Total** | 35378.000 | 65 | | | | |
| **Corrected Total** | 670.246 | 64 | | | | |

a. R Squared = .448 (Adjusted R Squared = .411)

Table 4 shows the mean and standard deviation of the participants' post-test scores according to their logic levels (low and high) and learning modes (SN and STN). It is evident that participants with high logic level in both learning modes outperformed those with low logic level in the post-test. This might be due to the fact that logic is an important aspects in understanding computer programming and supported the outcome of past researches.

**Table 4**

*Mean and Standard Deviation of C++ Computer Programming Post-test scores according to logic levels and learning modes*

| | Low logic | | | High logic | | |
|---|---|---|---|---|---|---|
| **Mode** | **n** | **Mean** | **SD** | **n** | **Mean** | **SD** |
| **SN** | 14 | 23.07 | 2.46 | 19 | 25.21 | 2.25 |
| **STN** | 13 | 20.00 | 245 | 19 | 23.16 | 3.48 |

A two-way ANCOVA analysis was conducted to ascertain any significant interaction effect between the two levels of logic and the two learning modes on the post-test scores, with the pre-test scores set as the covariate. The results, as shown in Table 5, indicate that there is no significant interaction effect, $F(1,60) = 1.43$, $p = 0.24$.

**Table 5**

*Two-Way ANCOVA for different levels of Logic among students using different modes of Screencast (SN & STN)*

| Source | Type III Sum of Squares | df | Mean Square | F | Sig. | Partial Eta Squared |
|---|---|---|---|---|---|---|
| **Corrected Model** | 300.047[a] | 4 | 75.012 | 12.158 | .000 | .448 |
| **Intercept** | 3192.356 | 1 | 3192.356 | 517.401 | .000 | .896 |
| **Mode** | 118.984 | 1 | 118.984 | 19.284 | .000 | .243 |
| **Logic** | 105.824 | 1 | 105.824 | 17.151 | .000 | .222 |
| **C++ Computer Programming Pretest** | 90.414 | 1 | 90.414 | 14.654 | .000 | .196 |
| **Mode * Logic** | 8.839 | 1 | 8.839 | 1.433 | .236 | .023 |
| **Error** | 370.199 | 60 | 6.170 | | | |
| **Total** | 35378.000 | 65 | | | | |
| **Corrected Total** | 670.246 | 64 | | | | |

a. R Squared = .448 (Adjusted R Squared = .411)

## DISCUSSION

The effectiveness of the screencast maybe due to its ability to support the dynamic nature of C++ computer programming in terms of logic and spatial visualisation. Computer programming subjects consists of many concepts that require logic reasoning and spatial visualisation such as biconditional, selection, and looping to understand the programming flows (Berk, 2009; Evans & Gibbons, 2007; Gomes & Mendes, 2007; Malik & Agarwal, 2012). During the initial stage of learning computer programming, it is vital that students have an effective learning material that supports the logic reasoning and flow of a computer program. Understand how the program flows is important as it leads to the final output of the overall program in decision making. Moreover, screencast is able to present visual and audio simultaneously which further enhanced the learning experience (Mohorovičić & Tijan, 2011). Besides that, spatial visualisation elements such as animated step-by-step solutions that appear on screen further assist students in solving complex calculation and equation (Mohorovičić & Tijan, 2011).

In terms of interaction effect between the students' logic levels and their learning modes on learning C++ computer programming, the results showed that there was no significant interaction effect in both of the learning modes. This means that learning C++ computer programming via the SN mode, which has been shown to be more effective than that of the STN mode, is not moderated by the students' logic levels. Even though having a non-significant result, the mean for C++ Computer Programming Posttest score for the screencast and narration (SN) group, regardless of the students' level of logic is higher compared to the mean for the screencast, text, and narration (STN) group. Moreover, there is an overall increase of mean in both of the learning modes regardless of the students' level of logic. The outcome might be due to the ability of screencasting in supporting the understanding of logic reasoning in learning computer programming flow. Furthermore, reason behind of the higher mean achieved by the students' in the SN mode probably due to the effectiveness of Mayer's Cognitive Theory of Multimedia Learning design in balancing the working memory between audio and visual channel.

Every research has its limitations and also potential improvements. Future researches may look into different types of moderating variables such as gender, learning style, and psychological aptitudes. Besides that, qualitative data collection such as interviews and observations can be included to gather in-depth information regarding the preferred learning modes and the reasons for their choice.

## CONCLUSION

Based on the overall increase of mean values between the C++ computer programming pre-test and post-test scores in both of the learning modes, it shows that screencasting indeed supports the learning of C++ computer programming. The success of the screencasting might probably be the effectiveness of multimedia element in explaining the dynamic nature of computer programming and support spatial visualisation. Besides that, it might be due to the possibility that the nature of screencasting multimedia elements were able to support the explanation of the programming logic regardless of the learners' level of logic capability. Furthermore, the results show that the screencast and narration (SN) modes is a much more effective learning mode compare to the screencast, text, and narration (STN) mode in the teaching of C++ computer programming.

## REFERENCES


Berk, R. A. (2009). Multimedia teaching with video clips: TV, movies, YouTube, and mtvU in the college classroom. *International Journal of Technology in Teaching and Learning, 5*(1), 1-21.

Bichler, R., & Snowman, J. (1986). psychology Applied to Teaching Houghton Miffin Company: Boston.

Brainerd, C. J. (1978). *Piaget's theory of intelligence*: Prentice Hall.

Butler, M., & Morgan, M. (2007). *Learning challenges faced by novice programming students studying high level and low feedback concepts.* Paper presented at the Proceedings of Ascilite, Singapore.

Carter, P. (2012). *An experience report: on the use of multimedia pre-instruction and just-in-time teaching in a CS1 course.* Paper presented at the Proceedings of the 43rd ACM technical symposium on Computer Science Education.

Clement, C. A., Kurland, D. M., Mawby, R., & Pea, R. D. (1986). Analogical reasoning and computer programming. *Journal of Educational Computing Research, 2*(4), 473-486.

Dalbey, J., & Linn, M. C. (1985). The demands and requirements of computer programming: A literature review. *Journal of Educational Computing Research, 1*(3), 253-274.

Eckerdal, A., Thuné, M., & Berglund, A. (2005). *What does it take to learn'programming thinking'?* Paper presented at the Proceedings of the first international workshop on Computing education research.

Enyeart, M. A. (1983). *Relationships among propositional logic, analogical reasoning, and Piagetian level*: University Microfilms International.

Epstein, H. T. (1990). Stages in human mental growth. *Journal of educational psychology, 82(4)*, 876-880.

Evans, C., & Gibbons, N. J. (2007). The interactivity effect in multimedia learning. *Computers & Education, 49*(4), 1147-1160.

Fletcher, S. H. (1984). *Cognitive Abilities and Computer Programming*. Retrieved from https://files.eric.ed.gov/fulltext/ED259700.pdf

Fluck, A. (2001). The rise and rise of computers in education. *Children's Ways of Knowing: Learning through experience*, 144.

Folk, M. J. (1994). INFLUENCES OF DEVELOPMENTAL LEVEL ON A CHILD'S ABILITY TO LEARN CONCEPTS OF COMPUTER PROGRAMMING.

Galton, A. (1992). Logic as a formal method. *The Computer Journal, 35*(5), 431-440.

Gibbs, N. E., & Tucker, A. B. (1986). A model curriculum for a liberal arts degree in computer science. *Communications of the ACM, 29*(3), 202-210.

Gomes, A., & Mendes, A. J. (2007). *Learning to program-difficulties and solutions.* Paper presented at the International Conference on Engineering Education–ICEE.

Hudak, M. A., & Anderson, D. E. (1990). Formal operations and learning style predict success in statistics and computer science courses. *Teaching of Psychology, 17*(4), 231-234.

Irons, D. M. (1982). *Predicting programming performance in novice programmers by measures of cognitive abilities.* Texas Christian University.

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010a). The effects of mind mapping with cooperative learning on programming performance, problem solving skill and metacognitive knowledge among computer science students. *Journal of Educational Computing Research, 42*(1), 35-61.

Ismail, M. N., Ngah, N. A., & Umar, I. N. (2010b). Instructional strategy in the teaching of computer programming: A need assessment analyses. *TOJET, 9*(2), 125-131.

Jenkins, T. (2002). *On the difficulty of learning to program.* Paper presented at the Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences.

Jonassen, D. H. (2004). *Handbook of research on educational communications and technology*: Taylor & Francis.

Lawson, A. E. (1983). The acquisition of formal operational schemata during adolescence: The role of the biconditional. *Journal of Research in Science Teaching, 20*(4), 347-356.

Little, L. F. (1984). *The influence of structured programming, gender, cognitive development and engagement on the computer programming achievement and logical thinking skills of secondary students*: University Microfilms.

Malik, S., & Agarwal, A. (2012). Use of multimedia as a new educational technology tool-A study. *International Journal of Information and Education Technology, 2*(5), 468.

Mayer, R. E. (2002). Multimedia learning. *Psychology of Learning and Motivation, 41*, 85-139.

Mayer, R. E. (2005). *Cognitive theory of multimedia learning (pp. 31-48)*. Santa Barbaras, California: University of California.

Mayer, R. E., Dow, G. T., & Mayer, S. (2003). Multimedia learning in an interactive self-explaining environment: What works in the design of agent-based microworlds? *Journal of Educational Psychology, 95*(4), 806.

Mohorovičić, S. (2012). *Creation and use of screencasts in higher education.* Paper presented at the MIPRO, 2012 Proceedings of the 35th International Convention, Opatija, Croatia.

Mohorovičić, S., & Tijan, E. (2011). *Using Screencasts in Computer Programming Courses.* Paper presented at the Proceedings of the 22nd EAEEIE Annual Conference, Maribor, Slovenia.

Murphy, L., & Wolff, D. (2009). Creating video podcasts for CS1: lessons learned. *Journal of Computing Sciences in Colleges, 25*(1), 152-158.

Piaget, J. (1972). Intellectual evolution from adolescence to adulthood. *Human development, 15*(1), 1-12.

Roadrangka, V., Yeany, R., & Padilla, M. (1983). *The construction of a group assessment of logical thinking (GALT).* Paper presented at the th annual meeting of the National Association for Research in Science Teaching. Dallas, Texas, April.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and Teaching Programming: A Review and Discussion. *Computer Science Education, 13*(2), 137-172. doi:10.1076/csed.13.2.137.14200

Wegerif, R. (2002). Literature review in thinking skills, technology and learning. Retrieved from https://telearn.archives-ouvertes.fr/hal-00190219/document

White, G., & Sivitanides, M. (2003). An empirical investigation of the relationship between success in mathematics and visual programming courses. *Journal of Information Systems Education, 14*(4), 409.

White, G. L., & Sivitanides, M. P. (2002). A theory of the relationship between cognitive requirements of computer programming languages and programmers' cognitive characteristics. *Journal of Information Systems Education, 13*(1), 59.