Journal of Education and Practice
ISSN 2222-1735 (Paper) ISSN 2222-288X (Online)
Vol.8, No.6, 2017

The Effect of Employing Self-Explanation Strategy with Worked Examples on Acquiring Computer Programing Skills

Riyadh Alhassan
P.O. Box 92782 – Riyadh 11663 – Saudi Arabia
College of Education, King Saud University, Saudi Arabia

Abstract
The purpose of this study was to examine the effect of employing self-explanation learning strategy supported with Worked Examples on acquiring computer programing skills among freshmen high school students. The study adopted a quasi-experimental method, where an experimental group (n = 33) used the self-explanation strategy supported with worked examples in learning programing, and a control group (n = 31) learned to do programing using the learning method defined in the National Guidelines for teaching computer curriculum. The results of the study showed that students in the experimental group achieved significantly better in programing knowledge and skills compared to the control group. The study recommended to include the self-explanation strategy in computer programing courses in high school computer literacy textbooks**.**
Keywords: Computer programing, Self-explanation, Worked examples, Computer education

1. Introduction

Computer Programing has been considered a complex and difficult process and requires a mindset of high capacities (Alhassan, 2014; Shamma, 2014; Yousif, Zahran, & Metwally, 2015; Ala-Mutka, 2004; Major, 2010) and Dijkstra (1989) views learning programing as the most difficult topic that can be taught to students. Some estimations indicate that about 40 to 50 percent of students who study programing in the first year of their Computer Science courses get a score of "good" or less, or withdraw the programing course (Schuyler, 2011). Thus, exploring effective instructional strategies by specialists in the teaching of computer science holds a critical importance (Kert & Kurt, 2012; Bucks, 2012).

Specialists in teaching computer science believe that it is crucial to help novice learners of computer science to learn programing languages (Al-Najar, 2014; Mohamed, Metwally, & Ali, 2014; Lee, 2014). Most of the new programing learners come to this field with a desire to write programs to help themselves and others find solutions or facilitate some of the problems of life or work, but they do not have any previous background on the concept of programing and the required mental abilities. In the Saudi context, the learners in public intermediate and high schools are introduced to the concept of programing for the first time and start to acquire some misconceptions about programing. Students learn "Scratch" language which is directed towards young people at intermediate levels (Al-Attas, 2014). Similarly, in high schools, they start to identify the higher-level programing languages such as Visual Basic, and programing of smart devices (Al-Ghamdi & Musa, 2014; Waziri, Khaddran, & Mustafa, 2014).

Notably, the difficulties experienced by students during their learning of programing in high schools, cannot be compared with what some of them face while studying computer science at the college level (Dijkstra, 1989). In fact, students in high schools learn only few principles of programing, as well as a limited number of commands and functions. However, learning Visual Basic Language does not resemble learning of any other subject already learned successfully in previous grades. Such a course is seen as a challenge to what the students might previously have as a stereotype of the difficulty of learning computer programing (Muhyiddin, and Banna, 2008).

In the current study, learners were trained to use self-explanation strategy that is supported with worked examples, via the use of educational materials that are consistent with the objectives of computer and information technology course. This strategy is used to overcome the difficulties and challenges that the students faced while learning programing of computers and smart devices. The learners directly interacted with the educational materials that have been designed according to the principles of instructional design.

Self-explanation strategy occurs when learners try to explain and clarify concepts for themselves, and check their own understanding of those concepts, which in turn generates mental coherence and help achieve better learning (Ali, Al-Qandil, & Baltia, 2015; Lee, 2014; Kalyuga, 2009; Vav Merrienboer & Sluijsmans, 2009). Many of the previous studies regarding self-explanation in Physics and Math showed positive effects on students' achievement and their understanding of the difficult concepts (Fukaya, 2011; Nokes, Hausman, VanLehn, & Gershman, 2011; Durkin, 2011). However, studies on the use of "self-explanation" strategy in the field of computer Programing are scarce (Lee, 2014) as there is only a set of studies conducted in the nineties of the previous century about learning "Lisp" programing language, which showed that the learners who received intensive training on self-explanation strategy have excelled academically in comparison with their peers who did not receive such training (Bielaczyc, 1995; Bielaczyc & Pirolli, 1995; Pirolli & Recker, 1994). A similar

study showed consistent results when learning SQL databases programing language (Yuasa, 1994). In the same line, Kwon and Jonassen found positive results when students learn "JavaScript" programing language (2011).

The self-explanation strategy is one of the general learning strategies that are based on the Constructivist Theory of learning, which aims to draw the attention of learners to educational materials while making sure of their understanding of the concepts presented to them (Roy & Chi, 2005). Processes followed in that strategy proved its ability to help learners to understand vague concepts (McNamara, 2009; McNamara & Magliano, 2009). One previous study indicated that the use of self-explanation strategy when learning computer programing concepts has shown positive results of learners with prior low or high knowledge on the learning subject (Kwon & Jonassen, 2011). However, the impact of "self-explanation" strategy on academic achievement when learning Computer Programing is rarely studied (Lee, 2014), Thus, the present study sought to bridge the gap in the literature by replicating the studies on self-explanation strategy to learn computer language programing in light of different variables such as the programing language, nature of the sample, grade level of the students, and the integration of worked examples within the self-explanation strategy.

The self-explanation strategy can be implemented through several patterns or styles. In speaking style, the learner thinks aloud about issues (McNamara, 2009); while in the writing style, the learner prints ideas that come to his or her mind during the problem-solving process (Munoz, Magliano, Shridan, & McNamara, 2006). For learners who face difficulty in intensive reading, the idea or thought writing style is considered more suitable for them, especially when solving scientific problems that require higher mental processes, such as learning programing. On the other hand, learners who are proficient in intensive reading get better gains through the speaking style, especially when learning theoretical topics (Munzo et al., 2006).

Renkl (1997) explains the potential impact of integrating self-explanation strategy with worked examples as the learners use self-explanation in order to explain for themselves the steps followed in the worked examples. Renkl believes that the use of self-explanation strategy leads students to activate and use their previous experiences to interpret the examples provided to them. In the case of taking a step in the worked example provided to learners without offering a convincing explanation for that step, they resort interpretation for themselves about what is learned from that step. Furthermore, students learn self-explanation strategy from early ages when they imitate the answers given in the worked examples. Thus, it helps them to develop problem-solving skills in the early stages of learning (Calin-Jageman & Ranter, 2005).

A number of studies indicates that supporting the self-explanation strategy with certain types of teaching, such as the provision of worked examples helps to transfer the learning effect to new educational settings (Kalyuga, 2009). For example, the study of Rittle-Johanson (2006) showed that strengthening the direct teaching with self-explanation strategy is more effective than independent teaching, and helps learners to transfer learning to new topics, and retain learning over time (Rittle-Johanson, 2006). The use of worked examples in teaching helps to reduce the time spent in learning, and the transfer of experiences to new topics (Cooper & Sweller, 1987). Several studies also indicate that students ignore texts that explain complex processes, and instead they learn from worked examples because of the ease of tracking and understanding rules via them (Clark & Mayer, 2008; Mayer, 2011; Moreno & Mayer, 2007).

1.1 Statement of the Problem

Through working in the undergraduate Computer Education program that prepares Computer Science teachers, and meetings with supervisors of computer science teachers in the Ministry of Education, the researcher has noticed a large number of questions risen by students whom the researcher taught in the computer education major regarding the problems their students faced at high schools while learning modern programming languages, and the continuous complaints about the rigidity of topics and the difficulty to understand them, as well as the low academic achievement. This may be due to several reasons, which may include the weakness of the instructional design of the computer courses, and the lack of strategies to facilitate students' learning of these concepts of programing. Some previous studies have indicated similar cases in the field of programming (Bilal, 1997; Al-Bassiouni, 2012; Al-Bassiouni, 1991; Joseph, Zahran, and Metwally, 2015; Vaziri, 2014). Reviewing the units that deal with the principles of programming and especially those that focus on the concepts of programing, the researcher noticed that they tend to be verbal and symbolic that students may feel that programming is too abstract to apply. In addition, most of the computer teachers in the field have not learned during the academic preparation the modern programming languages that are presented in the current computer courses, which may reflect on their performance in teaching due to their lack of the adequate training to teach these modern programming languages. Accordingly, there is an urgent need to examine new teaching strategies that may facilitate the learning and teaching of computer programming, which may help in teaching new programming concepts.

1.2 Purpose of the Study

This study aimed to examine the relative effect of using the self-explanation strategy coupled with worked

examples on students' achievement in computer programming skills at the high school level. For that purpose, several instructional units in computer programming in Visual Basic Studio were designed with the self-explanation strategy and worked examples built into them.

1.3 Research Questions

This study addressed the following research questions:

1. Is there a statistically significant difference in academic achievement in the cognitive skills of computer programming, at a level of 0.05, among the students who studied computer programming using self-explanation strategy supported by worked examples and those who learned computer programming via the traditional method?
2. Is there a statistically significant difference in academic achievement in the technical skills of the computer programming, at a level of 0.05, among the students who studied computer programming using self-explanation strategy supported by worked examples and those who learned computer programming via the traditional method?

1.4 Limitations of the Study

This study was limited to a sample of male students from the first year (freshmen) of a high school in Riyadh, Saudi Arabia. In addition, the study was limited to computer programming and information technology course taught in Saudi Arabian high schools.

1.5 Significance of the Study

Learning computer programming during the k-12 education helps students develop problem-solving skills (Alhassan, 2014, Shamma, 2014; Yousif, et. al, 2015; Ala-Mutka, 2004; Major, 2010; Lee, 2014) and giving them an understanding of how computers and smart devices' software works. In addition, it helps them to know more about the field of computer science before moving on to the university in order to make an informed decision while choosing a university major. This study contributes to identifying the impact of the self-explanation strategy which was proved to be effective in learning Science, Mathematics when learning computer programming.

In addition, the significance of this study stems from the scarcity of contemporary studies on the use of self-explanation strategy in learning computer programming. The present study intended to examine the effect of such a strategy on relatively modern learning programming languages, and identify the impact of supporting that strategy with worked examples.

Moreover, the educational materials that were designed in this study may help computer and IT curriculum designers identify the most appropriate approaches to include self-explanation strategy supported with worked examples, especially in programming courses. This, in turn, may contribute to overcoming the educational problems faced by students and teachers alike in teaching and learning modern programing languages in public education.

1.6 Definition of terms

1.6.1 Self-explanation: It is the process carried out by learners in order to assess the degree of understanding of the material in order to reach a higher degree of understanding. It is a cognitive process in which learners engage while reading the text or the given issue, and explain to themselves their own understanding of what they have read in a written or spoken manner (McNamara & Magliano, 2009). In the current study, students conducted the self-explanation process by writing directly what they understood about the task of programing, and worked examples of programming for a deeper understanding of the problem.

1.6.2 Worked examples: Worked examples provide detailed step by step instructions about solving an issue, and require less mental processing of the solution to the problems of conventional process (Greeno, 1980). When designing instructional units, the worked examples consist of a problem followed by a number of steps to solve the problem to teach novice students the steps that will be conducted by an expert or a specialist in the field to solve that problem (R. C. Atkinson, Renkl, Merrill, 2003). Worked examples also offer a detailed solution of an issue, and provide learners with a structure to understand how to reach a desired solution (Crippen & Earl, 2007). In this study, the provision of several worked programing examples, with explanations of each step written in the submitted program.

1.6.3 Computer programing: Computer programing involves variables, commands and instructions (called code or program) entered by a programmer to a computer using a programing language, according to a specific sequence for a human to help delivering a high degree of speed, accuracy and proficiency, so that a written program gives the required correct results (Vaziri et al., 2014). In this study, computer programing refers to the students use of the Visual Basic Studio language for writing a software code to correctly implement the given tasks needed in the software issue.

2. Theoretical Background and Previous Studies

2.1 Studying Computer Programming

Several specialists in the field of teaching and learning computers believe that the real value of computers lies in the ability of humans to direct computers to do the operations that may take the individual a lot of time and effort (Al-Mohammadi, 2015; Vaziri et al., 2014). The only way to direct a computer to carry out the required operations is by programing it commands similar to human language, called programing language. Programing languages are a mediator between a user and a computer. Thus, there is a need to include computer programing topics in the curricula of the public education (Mohammad Metwally, Ali, 2015).

There are common characteristics and similarities between most modern programing languages. Some of the aspects of similarity in the general skills of programing are program planning, writing algorithms, drawing flowcharts, dealing with variables and constants, dealing with conditional sentences, dealing with loops, and dealing with arrays and functions (Waziri, 2014). It could be said that studying programing skills of any programing language (Visual Basic Studio, Scratch, Java, etc.) is to learn the skills of programing in general, but through specific languages. This way a learner or a programmer can learn programing skills in a particular language and when he or she has a desire to learn a more recent language, the learning process will be easier than learning full programing skills because of the identical basic programing skills in most modern languages (Lee, 2014).

With regard to the methods of assessing students learning programing skills, there are two main methods of assessment in the field of programing (Mohammed, Metwally, & Ali, 2015): First, assessing the cognitive side: it is the side that handles the information and knowledge for learning programing, and it is evaluated using various kinds of achievement tests. Second, assessing the technical side: this side assesses the student's ability to write software code within the required speed and accuracy. This aspect is often assessed by asking a learner to write a software code, to implement and to check its validity and outcomes.

2.2 Self-explanation Strategy

It is relatively similar to self-questioning (Al-Rougi, & Aldhamana, 2014; Al-Authiaqi, 2015; Ali, Qandil, & Baltia, 2015; Hafez, 2015); and is an effective educational strategy that generates acceptable amount of mental load (Lee 0.2014); and used by learners in order to help them improve and enhance their understanding of the concepts during the learning process (Crippen & Earl, 2007). McNamara and Magliano, (2009) define the process of self-explanation as a mental process in which learners engage when they read the educational material. They believe it is a necessary process to understand the complex and the new educational material for learners. Meanwhile, Roy and Chi, (2005) argue that self-explanation is a structural process that can be used in several areas to ensure that learners are paying attention to a scientific subject while they reflect on their new understanding. One study showed that students in a physics course at the university level who have used self-explanation strategy by trying to think of the justifications for the interpretation of steps to solve physics problems in the examples provided to them in the textbook, may learn better than students who did not try to interpret the steps to solve examples provided to them (Chi, Bassok, Lewis, Reimann, & Glaser, 1989).

Previous studies have also shown that students' meditation and reflection about what they have previously learned in the instructional materials is considered one of the most important processes in learning (Al-Authiaqi, 2015; Davis, 2003). Dewey (as cited in Lee, 2014) interprets reflection on learning as thinking about what is learned with a goal of acquiring what is being learned. Dewey explains that reflecting on learning is a process during which a learner rethinks and examines the current understanding of an issue, and then finds a relationship between what is learned and what has been learned previously, and then generates new knowledge and understanding. Dillenbpur and Self (1992) believe that the process of reflection during learning is a dialogue between the learner and himself. They view this dialogue as a discourse between an individual and another, where the individual is trying to confirm, deny, or search for suitable alternatives about the concepts or the previous understanding. These assumptions of Dewey about reflection during the learning process indicate that the process of reflection plays a vital role in comprehending new knowledge. Student reflection is a mental process through which he or she learns through the process of planning, and by guiding learners to use self-explanation strategy, teachers ensure that students might possibly reflect on what is being learned. It was found that those learners who were asked to reflect on their learning through self-explanation have demonstrated better academic performance than their colleagues who were not directed to use the self-explanation strategy when learning (McNamara, 2009).

Several studies have shown that guiding learners to use the self-explanation strategy helped them to understand new instructional material deeper than without such a strategy, especially when supporting such strategy by some of other educational strategies (Roy & Chi, 2005; McNamara, 2009). The process of self-explanation can spontaneously occur outside the context of the learning process without the need for previous training to use them. When individuals have a difficulty in understanding something, they spontaneously may stop for a moment in order to explain to themselves the difficulty they face, and when they feel satisfied with the

answers that they reached, they then feel comfortable and continue the job that they are doing. Thus, the process of self-explanation can be regarded as a normal human activity with a visible and clear purpose in order to understand the materials that are learned (froth, 2015; McNamara, 2009).

2.3 The effect of self-explanation strategy on learning

Some of the previous studies have shown a positive effect of the use of self-explanation strategy on learning. In a number of studies carried out by Chi (1996) and Chi & VanLehn, (1991), in order to study the impact of such strategy on learning physics and mechanical concepts, findings have shown a positive impact of the strategy on those students who have been encouraged to use the strategy, compared to those who did not use that strategy. The findings also showed that supporting the strategy with worked examples was more effective than the use of the mere strategy. The researchers indicate that the positive effect stems from the fact that the strategy helps learners to control their learning and understanding of the material and the new concepts.

In a study by Kalyuga (2009), the findings indicated that self-explanation strategy should be supported by the guidance of the teacher and that this might lead to improvement of the transfer of learning experiences to new educational settings and real life situations. Similarly, Rittle-Johnson (2006) assures that the self-explanation strategy appears to be more effective when combined with direct instruction, where teachers encourage students to use self-explanation strategy during the learning process.

In the same vein, a group of studies conducted in the mid-nineties showed that students who received direct training and guidance about the use of the self-explanation strategy significantly outperformed their peers who did not use such a strategy in learning computer programing languages (Bielaczyc, 1995; Pirolli & Recker, 1994; Bielaczyc & Pirolli, 1995). These studies show that the application of that strategy in learning computer programing skills may improve students' learning. However, there is an obvious need for more recent studies in this area and with more contemporary programing languages, which is what the current study sought to provide.

2.3.1 Self-explanation strategy patterns

The literature shows that there are three modes or methods that can be applied to achieve the desired benefits of the self-explaining strategy, which are as follows:

First, the pattern of silent reflection with oneself: In this mode, a learner thinks silently without being felt by those around him. It is considered a silent and subtle way to implement this strategy, and often this process occurs more frequently than any other form of self-explanation strategy in various life situations. The core of this process is that the individual thinks deeply about what he studies, and tries to explain new concepts and processes in order to reach a deeper understanding (Lee, 2014). It is quite remarkable that there is a scarcity of studies on the use of this type of self-explanation strategy in learning, due to the difficulty of monitoring the implementation of the learners for this strategy.

Second, the thinking out loudly method: According to Auster (Oster, 2001), the method of audible thinking requires students to describe orally their ideas and strategies that they employ to understand the subject matter that they learn. This, in turn, pushes to enhance metacognitive awareness of the learner, which may improve the ability to understand the material. Moreover, previous studies have indicated that employing this style of self-explanation strategy patterns in learning has led to a significant improvement in students' understanding of a scientific material (Oster, 2001). Furthermore, in a number of studies in the field of learning to solve mathematical problems using the worked examples and the application of self-explanation strategy through the audible thinking pattern, there was a significantly enhancement in the performance of participating students more than their peers who did not apply this pattern (Chi & VanLehn, 1991; Chi, 1996).

Third, the writing of thoughts method: According to Lee (2014), activating the writing method when applying the self-explanation strategy, requires the teacher to encourage learners to write the sequence of their thinking and their reflections on the learned concepts and scientific material. This method is more effective in learning science subjects, compared to the study of theoretical materials, especially among students who find difficulty to express their thinking in an audible manner (McNamara, 2009; Munoz et al., 2006).

2.4 Worked examples

Many studies have showed that teaching and learning that is supported with worked examples during the early stages of learning results in a more significant positive impact than learning that did not use that kind of examples, with more positive effect being in scientific subjects such as Mathematics (Van Gog et al., 2006; Zhang, 2001), natural sciences (Crippen & Earl, 2007; Richey & Nokes-Malach, 2013), engineering (Pollock et al., 2002), and computer programing (Atkinson et al., 2000; Garner, 2002; Murphy & Wolff, 2009).

When studying computer programing, which represents a high mentally burden, the use of worked examples may help lessen this overload (Garner, 2002). Typically, worked examples in programing are a software code that has detailed observations and comments for each step justifying the reason for choosing a specific programming statements, or a particular algorithm (Crippen & Earl, 2004, 2007; Hohn & Moraes, 1997). Many of the previous studies on the use of worked examples in education showed that learners prefer learning

directly through these examples, rather than reading long texts provided in educational books for the purpose of explaining complex concepts. Researchers found that students go straight towards the worked examples when studying scientific materials (Math, Science, Engineering, etc.), and ignore lengthy texts that explain the rules because of the attractiveness of these examples and the effortlessness of learning from them, especially if they have detailed explanations of each step of the solution (Clark & Mayer, 2008; Mayer, 2011). This phenomenon illustrates the human extreme desire towards the easiest path of learning, which is provided by worked examples (Lee, 2014).

2.5 The application of self-explanation strategy with worked examples

Renkl (1997) describes the self-explanation strategy in the context of learning through worked examples as learners' attempt to clarify the steps for themselves to solve the issue described in the example. The results of the study carried out by Renkl showed that successful learners who achieved the highest academic performance spend more time in the study of worked examples and generate many new ideas as they attempt to solve the self-explanation examples. When students apply self-explanation strategy with worked examples, they use and employ their previous experience and knowledge to explain and clarify the given issue, which in turn leads to the generation and acquisition of new knowledge (Pirolli & Recker, 1994).

Although there are available steps to resolve the issue in the worked examples, the burden of clarifying these steps during the implementation of self-explanation strategy is the responsibility of the learner if there are no sufficient details about the steps to resolve the issue, or if there are steps that have not been clarified (Lee, 2014). The self-explanation strategy is often stimulated internally through the mental awareness of the learner if he or she feels the need for further clarification for the steps to resolve the problem. In addition, the positive effect of the strategy of self-explanation with worked examples was significant with learners of varying ages (Calin-Jageman, 2005).

In a series of studies conducted by Pirolli and Recker, (1994), the results showed a significantly positive impact on writing of correct programming code, and the ability to find errors in (debug) a given program, when students learn programing using the self-explanation strategy supported by a number worked examples. Thus, the current study provided the students with a number of worked examples covering several programing topics and problems to imitate them.

3. Methodology

This study intended to identify the effect of using self-explanation strategy with worked examples on student learning of computer programing skills. The quasi-experimental methodology was employed with an experimental group that was taught computer programming using the self-explanation strategy with worked examples, and a control group which was taught using the traditional instruction methodology highlighted in the teacher guide the teachers received from the ministry of education and the official national curriculum document.

3.1 Population and the Sample of the Study

The study population consisted of all first year (freshmen) high school students who were studying the Computer and Information Technology subject matter in Riyadh, Saudi Arabia. A purposive sample of sixty-four (64) students was selected from one public high school in the city of Riyadh. The sample was divided over two classes, with one of them being selected randomly to be the experimental group (n = 33), and the other one as the control group (n = 31).

3.2 Instruments of the Study

3.2.1 The educational materials: The educational materials were designed specifically for the experimental group and included content that encouraged students to use the self-explanation strategy in addition to several worked examples in the topics of programing covered in the syllabus. The content was arranged in modules for the experimental group according to that of the official syllabus. The design of these educational materials followed the model of instructional design proposed by Dick, Carey, and Carey (2005).

3.2.2 The pretest and posttest: Among the steps involved in the Dick, et. al (2005) model of instructional design is the building of evaluation tools. In this step, the pretest and posttest were built. The pretest measured the knowledge and skills in the field of programing in order to ensure equivalence of the experimental and control groups. The test consisted of ten questions of multiple choice and completion types. The internal consistency of the test as measured by Cronbach's alpha coefficient was 0.71, which is within the acceptable range of validity of the internal consistency (Tuckman, 1999).

Regarding the posttest, its purpose was to identify how much students have learned Computer Programing concepts, knowledge, and skills during their study of the instructional units. The test contained two types of questions:

1- Multiple choice and completion questions were used to measure the theoretical knowledge in the field of

programing.
2- Authentic assessment tasks (questions) asking learners to write programming code in Visual Basic in order to check students' skills in programing and understanding of the programming commands. The software code was evaluated based on a five-point scale (1 = totally wrong, 2 = partially correct, 3 = half of the code is correct, 4 = mostly correct, 5 = totally correct).
The test was then reviewed by four of the supervisors of computers courses in the Ministry of Education, five teachers of the course, and two faculty members from the College of Computer Science in order to ensure consistency of the test with its objectives and to check the facial validity of the test. Accordingly, several clauses of the test have been edited in order to improve clarity and ease of reading. The internal consistency of the test as measured by Cronbach's alpha coefficient was 0.74, which is within the acceptable range of the test validity (Tuckman, 1999).

3.3 Procedures
After obtaining approval to carry out the experimental study, the pretests were administered to the two groups before embarking on programing lessons. The "T." test of two independent groups showed that there are no statistically significant differences between the average scores of the experimental and control groups in the pretest, which assures equality of the two groups with respect to previous experiences in the field of computer programing.

Then, the experimental and control groups were taught by the computer teacher at the school for 24 classes, after he has been trained in the use of the modules, and the use of the self-explanation strategy, as well as in how to motivate students to use the modules and to write their self-explanations. To ensure that students write their self-explanations, they have been notified that these self-explanations will be graded.

On the other hand, the control group was taught in the computer lab, according to the curricula of the computer course. The teacher used the principles of programing steps presented in the textbook to write program codes, as the teacher performs the steps and displays them in front of all the students. Later, the students were requested to do the same steps outlined in the textbook and complete the programs mentioned in the book.

4. Results

This study aimed to identify the effect of self-explanation strategy supported with worked examples in acquiring computer programing skills. The following is a review of the findings of the study, after collecting achievement tests data.

4.1 Groups equivalence in the pretests
The achievement pretests were administered to all study sample before the experiment, in order to check the equivalence of the control and experimental groups. The "t" test for independent samples has been used.
Table 1. T-test results for group equivalence

Group	N	Mean	Std. Deviation	t-value	Sig. (Two-Tailed)
Control	31	3.43	1.025	1.013	0.315
Experimental	31	3.03	0.881		

It could be seen from table 1 that there were no statistically significant differences between the control and experimental groups in the pretest, which indicates that the two groups were equivalent.

The first question of the study was to identify if there is a statistically significant difference in academic achievement in the knowledge of the computer programing, at a level of 0.05, among the students who studied computer programing using self-explanation strategy supported by worked examples and those who learned computer programing via the traditional method. To answer the question, the two independent samples t-test was used to compare the two groups of the study. Eita square (ɳ2) was used as well to measure the effect size.
Table 2. Independent samples t-test results for the post-tests of the control and experimental groups in the knowledge of computer programing.

Group	N	Mean	Std. Deviation	t-value	Sig.	η^2	Effect size
Control	31	6.23	1.203	8.051	0.000	0.458	High
Experimental	333	3.03	0.996				

It is evident from table 2 that there is a statistically significant difference at the level of 0.000 in the mean scores of achievement tests in the knowledge of computer programing skills between the control and experimental groups in favor of the experimental group. The better achievement of the experimental group can be attributable to the use of self-explanation strategy in the teaching of the knowledge related to computer programing.

To find out the effect size of these statistically significant effects, ETA square (ɳ2) was calculated. It could be seen from Table 2 that the square value of the ETA (ɳ2) equals 0.458, indicating an impact of the independent variable, which is the use of self-explanation strategy supported with worked examples, is in the

high range according to Cohen (1988).

The second question of the current study was to explore if there is a statistically significant difference in academic achievement in the technical skills of computer programing, at a level of 0.05, among the students who studied computer programing using self-explanation strategy supported by worked examples and those who learned computer programing via the traditional method.

Table 3. Independent samples t- test results for the posttests of the control and experimental groups in the achievement test of programing skills.

Group	N	Mean	Std. Deviation	t-value	Sig.	η^2	Effect size
Control	31	22.00	4.250	8.836	0.000	0.416	High
Experimental	333	34.73	3.302				

Table 3 shows that there was a statistically significant difference between the control (M=22.0) and experimental groups (M=34.73) in the mean scores of the achievement posttest of programing skills where t value equals 8.836 at the significance level of 0.00. It can be noticed that the experimental group outperformed the control group in the achievement test of programing skills which in turn indicates a positive effect of the use of self-explanation strategy supported with worked examples on the learning of computer programing skills

Similarly, it could be seen from table 2 that the square value of ETA (η2) equals 0.416, which is greater than the value 0.14 set by Cohen (1988) for the high effect size, indicating high impact of the independent variable, which is the use of self-explanation strategy supported with worked examples on the academic achievement of the computer programing skill.

5. The Discussion

Many previous studies have found a positive effect of the use of self-explanation strategy on the students' learning in a number of subject matters (Lee, 2014; Kwon & Jonassen, 2011; Yuasa, 1994; Bielaczyc, 1995). The findings of the current study are in line with these findings in the field of computer programing learning, both the knowledge and skills. The results of the present study are consistent with the results of a number of old studies conducted by Pirolli and Recker, (1994), as the results showed a positive effect for the use of self-explanation strategy supported with worked examples when learning topics related to programing. This positive effect was obvious via the students' ability to write correct programing code and statements, and the ability to find errors in a given software.

Equally important, the present study showed a positive impact of supporting the self-explanation strategy by using worked examples that make it easier for students to learn new programing concepts, which may be viewed as a complicated issue by a number of new learners. This positive impact for the integration of these two strategies could be attributed to their role in lessening the mental effort of the learner, and systematically guiding him or her to the best way to solve the programming problems. The self-explanation strategy contributes to the student's learning by challenging the limits of his or her thinking through asking a lot of questions about what is being learned (Lee, 2014).

A further possible justification for these positive findings in achievement in programming knowledge and skills might be due to the use of self-explanation strategy when learning programing as this strategy helps to understand complex concepts more deeply. A number of studies showed that guiding learners to use self-explanation strategy helps them to understand new subject matter with greater depth of learning than that in the case of a traditional method of teaching, especially when the strategy is supported with worked examples (Roy & Chi, 2005; McNamara, 2009). The self-explanation strategy may also help students reach positive results in terms of information retrieval and the application of the previous skills in new teaching contexts, and make students more self-reliant in the construction of meaning through their exploration of the mistakes and correcting them, which may lead to the retention of gained concepts and skills for a longer time (Lee, 2014; Kalyuga, 2009; Vav Merrienboer & Sluijsmans, 2009).

Providing students with worked examples helps them overcome the common mistakes committed by new learners in computer programing. Such examples offer the correct use of the programming commands in multiple contexts, which enables them to write new programming code with minimal errors (Kwon & Jonassen, 2011). In addition, when learning computer programing, in particular, which involves a mental burden (Cognitive Load), the use of worked examples may help relieve this mental overload (Garner, 2002).

5.1 Recommendations

In light of the findings of the current study, the following recommendations could be suggested: First, due to the relatively positive effect of self-explanation strategy with worked examples in improving the academic achievement and learning of programing skills, it is recommended to include this strategy in the textbooks of the computer and information technology curriculum, especially in topics oriented towards the teaching of programing. Second, there is an obvious need to train in-work computer teachers, as well as student teachers who are majoring in computer education in colleges of education on the use of self-explanation strategy

supported with worked examples.

Furthermore, worked examples showed an ability to improve and facilitate students' learning; therefore, it is recommended to incorporate a large number of worked examples within the computer curriculum textbooks when introducing programming principles. It is also suggested to include worked examples when teaching computer applications where such examples may be useful such as database design or spreadsheets.

5.2 Suggestions for Future Studies

Based on the findings of the present study, a replication of this study with a sample of intermediate level learners who study Scratch programing language is suggested. It is also suggested to conduct a study to explore the impact of self-explanation strategy on the computer courses that require a great deal of higher order thinking skills such as designing databases and spreadsheets, and programing of robots.

References

Ala-Mutka, K. (2004). Problems in Learning and Teaching Programing - A literature study for developing visualizations in the Codewitz-Minerva project [Electronic Version]. Retrieved Mars. 23, 2016 from http://www.cs.tut.fi/~codewitz/literature_study.pdf.

Al-Attas, A. (2014). Scratch programing language in the education. *Knowledge Journal, Ministry of Education, Saudi Arabia,* 234, 86 - 89.

Al-Authiaqi, Y. (2015). The effectiveness of self-questioning strategy in the development of some of reading comprehension skills for the first-grade high school students. *Journal of the College of Education in Assiut -Egypt, 31 (3),* 268.

Al-Bassiouni, M. (1991). The impact of students' specialization on the achievement on a computers and language programing course and their attitudes towards the use of computers in education. *Journal of the Faculty of Education in Damietta -Egypt*, *15, (26*), 77 - 126.

Al-Bassiouni, M. (2012). Developing a learning environment in the light of the constructivist theories of learning to develop programing skills of computer teacher students. *Journal of the College of Education in Mansoura -Egypt, 78, (2),* 293 - 371.

Al-Ghamdi, A. & Moses, M. (2014). The impact of the interaction between the pattern of educational control in the multimedia software and the cognitive method of learners on high school students programing skills. *(Unpublished Master's Thesis).* Al-Baha University, Al-Baha.

Alhassan, R. (2014). The effect of Teaching problem-solving non-Mathematic problems on self-efficacy and performance in the Introduction to Programing course. *International Journal of Educational Research - UAE, 35*, 62 - 93.

Ali, N. Al-Qandil, A. & Baltia, H. (2015). The effectiveness of the use of self-questioning strategy in the enhancing verbal mathematical problem solving skills among primary school students. *Journal of Mathematics Educations -Egypt, 18,* (6),189.

Al-Mohammadi, N. (2015). The effectiveness of a proposed educational software on the achievement of a computer course of first-grade high school students in Jeddah. *Arab Studies in Education and Psychology - Saudi Arabia,* 62, 305 - 327.

Al-Najar, M. (2013). A proposed strategy of Web 2.0 on the development of the computer programing skills of computer teachers class. *Egypt Educational Sciences, 21*, (4), 245 - 281.

Al-Rougi, T. & Al-Dahmani, D. (2014). The effectiveness of the of self-questioning and activating prior knowledge strategies on the development of critical reading skills among the first-grade high school students and their attitudes towards reading. *(Unpublished PhD Dissertation*). Umm Al-Qura University, Makkah.

Atkinson, R. C., Renkl, A., & Merrill, M. M. (2003). Transitioning from Studying Examples to Solving Problems: Effects of Self-explanation Prompts and Fading Worked-Out Steps. *Journal of Educational Psychology, 95*(4), 774-783.

Atkinson, R. K., Derry, S. J., Renkl, A., & Wortham, D. (2000). Learning from Examples: Instructional Principles from the Worked Examples Research. *Review of Educational Research, 70*(2), 181-214.

Bielaczyc, K. (1995). *Learning through student-generated explanations: Investigating the effects of individual and collaborative explanation strategies and metacognition on the acquisition of knowledge and skills for computer programing.*, U California, Berkeley, Berkeley

Bielaczyc, K., & Pirolli, P. (1995). Training in Self-explanation and Self-Regulation Strategies: Investigating the Effects of Knowledge Acquisition Activities on Problem Solving. *Cognition and Instruction, 13*(2), 221-252.

Bilal, E. (1997). Difficulties encountering second-grade high school students in their studies of the introduction to the BASIC programing language course and their impact on their achievement. *Journal of the College of Education, Aswan - Egypt, 12*, 326 - 376.

Bucks, G. W. (2010). *A Phenomenographic Study of the Ways of Understanding Conditional and Repetition Structures in Computer Programing Languages.* Purdue University.

Calin-Jageman, R. J., & Ratner, H. H. (2005). The Role of Encoding in the Self-explanation Effect. *Cognition and Instruction, 23*(4), 523-543.

Chi, M. T. H. (1996). Constructing Self-explanations and Scaffolded Explanations in Tutoring. *Applied Cognitive Psychology, 10*, S33-S49.

Chi, M. T. H., & VanLehn, K. A. (1991). The content of physics self-explanations. *Journal of the Learning Sciences, 1*, 69-106.

Chi, M. T. H., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science: A Multidisciplinary Journal, 13*(2), 38.

Clark, R. C., & Mayer, R. E. (2008). E-learning and the science of instruction (2nd ed). In. San Francisco: Jossey-Bass.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences* (2nd ed.). Hillsdale, NJ: Lawrence Erlbaum Associates.

Cooper, G., & Sweller, J. (1987). Effects of Schema Acquisition and Rule Automation on Mathematical Problem-Solving Transfer. *Journal of Educational Psychology, 79*(4), 347-362.

Crippen, K. J., & Earl, B. L. (2004). Considering the Efficacy of Web-based Worked Examples in Introductory Chemistry. *Journal of Computers in Mathematics and Science Teaching, 23*(2), 151-167.

Crippen, K. J., & Earl, B. L. (2007). The impact of web-based worked examples and self-explanation on performance, problem solving, and self-efficacy *Computers & Education, 49*, 809-821.

Dick, W., Carey, L., & Carey, J. O. (2005). *The Systematic Design of Instruction* (6th ed.). Boston, MA: Allyn and Bacon.

Dijkstra, E. (1989). On the Cruelty of Really Teaching Computing Science. *Communications of the ACM, 32*(12), 1398-1414.

Dillenbourg, P., & Self, J. A. (1992). A Computational Approach to Socially Distributed Cognition. *European Journal of Psychology of Education, 7*(4), 352-373.

Durkin, K. (2011). The Self-explanation Effect when Learning Mathematics: A MetaAnalysis. *Society for Research on Educational Effectiveness*, 10.

Fukaya, T. (2011). Effects of self-explanation training on learning scientific concepts: Intervention based on SBT theory. *Japanese Journal of Educational Psychology, 59*(3), 13.

Garner, S. (2002). *Reducing the Cognitive Load on Novice Programmers.* Paper presented at the Ed-Media 2002 World Conference on Educational Multimedia, Hypermedia & Telecommunications.

Greeno, J. (1980). Some examples of cognitive task analysis with instructional implications. In R. Snow, P. Federico & W. Montague (Eds.), *Aptitude, learning and instruction* (Vol. 2, pp. 1-21). Hillsdale, NJ: Lawrence Erlbaum Associates.

Hafez, M. (2015). The effectiveness of self-questioning strategy on the achievement of the third-grade students of teacher training institutes and decision-making in chemistry course. *Educational and Psychological Research Journal -Iraq*, 44, 312.

Hohn, R. L., & Moraes, I. (1997). Use of rule-based elaboration of worked examples to promote the acquisition of programing plans. *Journal of Computer Information Systems, 38*(2), 35-40.

Kalyuga, S. (2009). Instructional designs for the development of transferable knowledge and skills: A cognitive load perspective. *Computers in Human Behavior, 25*(2), 332- 338.

Kert, S. B., & Kurt, A. A. (2012). The Effect of Electronic Performance Support Systems on Self-Regulated Learning Skills. *Interactive Learning Environments, 20*(6), 485- 500.

Kwon, K., & Jonassen, D. (2011). The Influence of Reflective Self-explanations on Problem-solving Performance. *Journal of Educational Computing Research, 44*(3), 247-263.

Lee, L. (2014). The Effect of self-explanation and Reading Questions and answers on Learning Computer Programing. (Unpublished Doctoral Dissertation). University of Nevada [UMI number: 3613893].

Major, L. (2010). Systematic Literature Review Protocol: Teaching Novices Programing Using Robots. *Online Submission*.

Mayer, R. E. (2011). Applying the science of learning to multimedia instruction. In J. Mestre & B. Ross (Eds.), *The psychology of learning and motivation: Cognition in education* (Vol. 55, pp. 77-108). San Diego, CA: Elsevier Academic Press.

McNamara, D. S. (2009). The Importance of Teaching Reading Strategies. *Perspectives on Language and Literacy*, 34-40.

McNamara, D. S., & Magliano, J. P. (2009). Self-explanation and metacognition: The dynamics of reading. In D. J. Hacker, A. C. Graesser & J. Dunlosky (Eds.), *Handbook of metacognition in education* (pp. 60-81). Mahwah, NJ: Erlbaum.

Mohammed, M., Metwally, A., Ali, N. (2015). The effectiveness of cognitive trips via the Web in the development of programing skills for the third preparatory grade students. *Journal of the College of Education, Banha University - Egypt, 26*, (101), 235 - 263.

Moreno, R., & Mayer, R. E. (2007). Interactive multimodal learning environments. *Educational Psychology Review, 19*, 309-326.

Muhyiddin, S., & Al-Banna, S. (2008). An interactive learning of Visual Basic language. *Al-Rafidin Development, 30,* (92), 131 - 149.

Munoz, B., Magliano, J. P., Sheridan, R., & McNamara, D. S. (2006). Typing versus thinking aloud when reading: Implications for computer-based assessment and training tools. *Behavior Research Methods, 38*(2), 211-217.

Murphy, L., & Wolff, D. (2009). Scaffolding active programing instruction with theoretically grounded screencasts and annotated worked examples (pp. 5). Pacific Lutheran University, Tacoma, WA 98447.

Nokes, T., Hausmann, R., VanLehn, K. A., & Gershman, S. (2011). Testing the Instructional Fit Hypothesis: The Case of Self-explanation Prompts. *Instructional Science: An International Journal of the Learning Sciences, 39*(5), 645-666.

Pirolli, P., & Recker, M. M. (1994). Learning Strategies and Transfer in the Domain of Programing. *Cognition and Instruction, 12*(3), 235-275.

Pollock, E., Chandler, P., & Sweller, J. (2002). Assimilating complex information. *Learning and Instruction, 12*(1), 61-86.

Renkl, A. (1997). *Intrinsic Motivation, Self-explanations, and Transfer*. Paper presented at the Annual Meeting of the American Educational Research Association.

Richey, J. E., & Nokes-Malach, T. J. (2013). How much is too much? Learning and motivation effects of adding instructional explanations to worked examples. *Learning and Instruction, 25*, 104-124.

Rittle-Johnson, B. (2006). Promoting Transfer: Effects of self-explanation and Direct Instruction. *Child Development, 77*(1), 15.

Roy, M., & Chi, M. T. H. (2005). The Self-explanation Principle in Multimedia Learning. In R. E. Mayer (Ed.), *The Cambridge Handbook of Multimedia Learning* (pp. 271- 286). New York, NY: Cambridge University Press.

Schuyler, S. T. (2011). Using problematizing ability to predict student performance in a first course in computer programing. *Dissertation Abstracts International Section A: Humanities and Social Sciences, 71*(7-A), 2432.

Shamma, M. (2014). A proposed strategy for the use of Facebook in education for enhancing educational programing skills of students of the Information Systems class. *The fourth scientific conference: educational IT and online electronical training and the ambitions of modernization in the Arab world* Egypt, Cairo: Egyptian Society for Technology in Education and the Faculty of Education - Al-Azhar University - Egypt .247 -316.

Tuckman, B. (1999). Conducting Educational Research. Harcourt Brace College Publishers.

Van Gog, T., Paas, F. G. W. C., & van Merrienboer, J. J. G. (2006). Effects of Process Oriented Worked Examples on Troubleshooting Transfer Performance. *Learning and Instruction, 16*(2), 154-164

Van Merrienboer, J. J. G., & Sluijsmans, D. M. A. (2009). Toward a Synthesis of cognitive Load Theory, Four-Component Instructional Design, and Self-Directed Learning. *Educational Psychology Review, 21*(1), 55-66.

Waziri H., Khadhran, N., & Mustafa, A. (2014). The effectiveness of a proposed multimedia interactive program using Flash software on the development of some programing skills of preparatory school students. *Reading and Knowledge Journal. Egypt, 149*, 117 - 138.

Yousif, A., Zahran, A., & Metwally, A. (2015). The effect of using a educational electronic forum on the development of some of the object-oriented programing skills using Visual Basic .Net language at the preparatory grade students. *Journal of the College of Education Banha University, Egypt, 26*, (103), 225 - 246.

Yuasa, M. (1994). *The effects of active learning exercises on the acquisition of SQL query writing procedures.* Georgia Inst of Technology.

Zhang, C. (2001). Effects of worked examples and exercise time on problem-solving transfer. *Acta Psychologica Sinica, 33*(2), 170-175.