*Journal of Learning and Teaching in Digital Age, 2018, **3**(2), 42-50*

Comparison of Three Instructional Strategies in Teaching Programming: Restudying Material, Testing and Worked Example

Mustafa Tepgeç
Hacettepe University, Turkey
mustafatepgec@gmail.com

Yasemin Demiraslan Çevik
Hacettepe University, Turkey
yasminey13@gmail.com

ABSTRACT

The aim of the study is to determine the effects of different instructional strategies on retention performance and cognitive load in teaching programming. The study also aimed to compare these strategies in terms of instructional efficiency. The study group consisted of 106 students enrolled in the first grade at a high school. Instructional strategies used in the study are testing (n=38), restudying material (n=31) and studying worked example with self-explanation prompts (n=37). In the intervention process, the study booklet was first presented to all groups. The booklet prepared for this study covers topics such as variable identification, decision structures, pseudo-codes and flowcharts in teaching programming basics. The booklet was presented to the restudying group for three times and they were expected to study material in depth for each session. Subsequently, isomorphic problems were presented for the testing group. In the other group, worked examples were presented and learners were expected to comprehend the logic underlying the problems. Immediately after the intervention, the first retention test and the cognitive load scale were applied. The final retention test was conducted three weeks later the first retention test was implemented. The study concluded that worked example with self-explanation prompts is more efficient than the other two strategies in teaching programming basics in terms of instructional efficiency. In addition, the fact that testing has increased the long-term retention of knowledge has been confirmed. However, when cognitive load levels were taken into account, there was no difference between the testing and the restudying material strategies. It is expected that the study will contribute to the literature due to the findings in regard to pedagogy of programming.

Keywords: *testing effect, teaching programming, worked example, self-explanation, programming pedagogy, cognitive load*

INTRODUCTION

In the educational context, tests are commonly used to assess the degree to which targeted behaviors have been achieved. In educational environments, evaluation through tests can be used to recognize learners or identify deficiencies within the learning process. It can also be used to reveal the situation at the end of the learning process. It is stated, however, that testing can be used not only to assess the extent to which targeted behaviors are achieved but also to increase the likelihood of recalling the information expected to be learned at the same time (Demiraslan Çevik & Çoban, 2016). This phenomenon is called "testing effect" in the literature. The testing effect is based on the assumption that after an initial study, testing increases long-term retention compared to restudying the material or not testing (Roediger & Karpicke, 2006a). There may be no difference in regard to the performances between the group that was tested and the group that was restudying material in the measurements conducted immediately after the learning period, and the restudying group may even perform better (Van Gog, Kester, Dirkx, Hoogerheide, Boerboom & Verkoeijen, 2015). However, in the measurements conducted one week after the learning process, the group that was tested performed better retention performance than the restudying group (Roediger & Karpicke, 2006b). Retention performance was generally considered in the studies related to testing effect. In addition, commonly the testing and the restudying material group were compared. In studies investigating the testing effect, participants are exposed to the study material twice or more times and they are tested (Terry, 2011). While restudying group studies at least twice on the material, the testing group studies the material once and then take at least one test.

Testing effect has been demonstrated with a variety of learning materials, such as word lists (Wheeler et al., 2003), facts (Carpenter et al., 2008), prose passages (Roediger & Karpicke, 2006b), symbol–word pairs (Coppens et al., 2011), videotaped lectures (Butler & Roediger, 2007), visuospatial materials such as maps

***Correspondence to*:** *Mustafa Tepgeç, Hacettepe University, Turkey, E-mail: mustafatepgec@gmail.com*

(Carpenter & Pashler, 2007), numerical materials such as functions (Kang et al., 2011), and multimedia materials such as animations (Johnson & Mayer, 2009). Roediger and Karpicke (2006) examined the testing effect in the acquisition and retention of knowledge in their study carried out with university students. In one condition, students study the learning material once and then restudy three more times, in the other condition they study the material once and were tested three times. The retention test was presented one week after the learning process. As a result, it was concluded that the testing as the learning strategy showed higher retention performance. In the study conducted by Uçar and Demiraslan Çevik (2018) with secondary school students, the strategies of restudying the material and the testing on the "Safe Internet Usage" are compared. As a result of the study, it was seen that the testing group was more successful in short-term retention (5 minutes after the learning process) than the group that was restudying the material. Nevertheless, it has been demonstrated that there is no difference between the groups in the long-term retention (a week after the learning process). It was also found that the group subjected to more tests did not perform better than the other testing group. Researchers have argued that this result may be due to the lack of feedback for wrong answers. Therefore, it can be stated that the effect of the testing on the retention performance varies according to the context of the learning process. However, the findings of the research on the effects of testing on retention in areas such as mathematics, programming, and physics, which require high-level problem-solving skills, are very limited. For this reason, it is important to compare the efficiency of the testing strategy, which improves retention performance, and the worked examples, a proven strategy to be effective especially in the domains requiring problem solving skills.

Worked example is an instructional strategy in which the solution steps given to the learner to complete a problem or a task are presented to guide learners. Research shows that providing worked examples to novice learners is a more efficient strategy than problem solving (Atkinson et al., 2000; Renkl, 2014; Van Gog & Rummel, 2010). This finding stems from the fact that learners without sufficient prior knowledge on the subject tend to have inefficient searching strategies during problem solving. Therefore, the cognitive load of learners increases in this situation. Cognitive load theory is based on the fact that the working memory resources have limited capacity. The basic principle the theory advocates is to eliminate the components that negatively affect learners' cognitive resources when designing instruction, thus ensuring that the working memory resources are used efficiently. Cognitive load or, in other words, working memory load is differentiated into three types: the intrinsic load, the extraneous load and the germane load. The intrinsic load is caused by the interaction of learners' prior knowledge and task complexity (Sweller, 2010). The instrinsic load can not be manipulated without changing the nature of the task or the learners' prior knowledge (Paas & Sweller, 2014). The extraneous load is caused by poor or inappropriate instructional design (Likourezos & Kalyuga, 2017). The extraneous load is completely under the manipulation of instructional designers (Sweller, van Merrienboer & Paas, 1998). Excessive cognitive load occurs when the sum of the resources allocated to the intrinsic load and the extraneous load exceeds the working memory capacity. In cases where cognitive overload does not occurs, the difference between the intrinsic load and the extraneous load corresponds to the germane load (Paas & Sweller, 2014). The germane load revealed in the later years of the theory refers to the use of working memory resources associated with learning. Worked examples is a strategy put forward against excessive cognitive load resulting from the problem solving process (Sweller, 1989). In addition, it is also important how the worked examples are designed. Various design forms have been put forward on how to use worked examples in the literature.

The self-explanation principle is concerned with ensuring that learners experience a more active process while studying examples by explaining the solution steps, rather than being a passive recipient of information. Chi, Bassok, Lewis, Reimann and Glaser (1989) were the pioneers to employ the self-explanation principle. Sweller (2010) stated that learners use the cognitive resources more effectively in the self-explanation process. Sweller also claimed that it is a strategy to help optimize the cognitive load of the self-explanation process.

In the learning process, the strategy of studying with worked examples is not completely irrelevant to the strategies of testing or restudying the material. According to the design of the worked example, this strategy may actually be similar to the testing strategy. In other words, problem solving is expected at the same time from the learners when the worked examples are supported with self-explanations. However, it should be noted that the testing effect is not the same as the problem solving strategy that is often used in the worked example literature. Because the focus during problem solving is not only to retain the knowledge but also to produce and construct it (Van Gog et al., 2015). In the restudying material strategy, the study material related to the topic is usually presented to the learners. Here, learners are expected to examine the material in depth. In the case of worked examples, it is generally expected that learners will understand the solution steps of the problem by studying the examples. In both cases, learners can passively study the material. However, the focus is more on the solution steps in worked examples.

In the next section, programming instruction and the use of instructional strategies mentioned in this section in programming instruction are examined.

Teaching Programming Basics with Algorithms

Programming is the expression of procedural steps for solving a problem or a task through programming language platforms (Vihavainen, Airaksinen & Watson, 2014). Algorithms are used in teaching programming logic, which is the basis of programming. The algorithms are a sequence of procedural steps followed to solve a problem (Gal-Ezer, Vilner & Zur, 2014). One of the most important reasons why algorithms are used so often in teaching programming basics is to embody the programming process in real life situations. In the teaching of algorithms, it is aimed to embody the concepts such as variables, decision structures, and loops used in computer programming.

The most common methods used in teaching algorithm are the expression of the solutions step-by-step and the flowcharts. The step-by-step expression of the algorithm is the numbering of the steps and writing them line by line. In flowcharts, sub-processes such as start/end, decision making, process and input/output are shown with a specific symbol. The main purpose of the flowchart is to visualize the solution steps of the problem to create a more efficient program draft. However, the step-by-step expression of the algorithm and the flowcharts are insufficient to transition to a real programming platform. In these cases, the pseudo-codes are often used. In pseudo-code, the algorithms are still expressed in text, but commands like "Print Result" are used instead of "The result obtained after the calculation is displayed on the screen". In this way, algorithms are created with similar commands used in real programming.

In the literature, many factors influencing programming performance have been put forward. Robins, Rountree and Rountree (2003) stated that one of the most important factors influencing programming performance is the instructional strategies implemented. Similarly, Margulieux, Catrambone and Guzdial (2016) have emphasized the need for programming educators to focus on instructional strategies rather than programming platforms used. As mentioned, the testing effect in the field of cognitive psychology is often used to learn and retain the concepts. However, in recent years studies have shown that the use of tests as study material is also effective in domains where problem solving skills are required (Hoogerheide, Renkl, Fiorella, Paas & Van Gog, 2018; Leahy, Hanham & Sweller, 2015; Van Gog & Kester, 2012). Therefore, it is important to examine the efficiency of this strategy in programming teaching, which requires problem-solving skills. However, Mayer (2013) stated that in programming teaching, strategies that provide guidance to novice learners should be applied rather than discovery-based instructional strategies such as problem solving. It has been demonstrated in various studies that worked examples are an effective strategy in teaching programming to novice learners (Abdul-rahman and Boulay, 2014; Lee, 2013; Margulieux, Catrambone and Guzdial, 2016; Margulieux and Catrambone, 2016; Si, Kim and Na, 2014). Therefore, it is important to compare the effectiveness of the teaching strategies mentioned in this section on novice learners. It is predicted that the study will contribute to the literature on the pedagogy of programming.

Aim of the Study

The purpose of the study is to determine the effects of different instructional strategies (testing, restudying material, and studying worked example) on initial and final retention performance and cognitive load in teaching programming to high school students. The study also aims to compare instructional strategies in terms of instructional efficiency scores. In this context, answers to the following questions were sought:

1. How do the different instructional strategies affect participants' initial retention test (after 10 minutes) performance?

2. How do the different instructional strategies affect participants' final retention test (after 3 weeks) performance?

3. How do the different instructional strategies affect participants' cognitive load?

4. Is there a significant difference between the instructional strategies in terms of instructional efficiency?

METHOD

Participants and Design

The study group consisted of 106 students (49 female and 57 male) enrolled in the first grade at a high school in Ankara in the 2017-2018 academic year. This is a quasi-experimental study conducted in three classes based on three instructional strategies (testing, restudying, studying worked example). Participants were not randomly assigned to groups, but it was randomly determined which instructional strategies would be applied to which groups. There are 37 learners in worked example group (Study-Worked example-Worked example), 38 learners in testing group (Study-Test-Test) and 31 learners in restudying material group (Study-Study-Study). Participants have indicated that they have not taken any programming courses before.

© 2018, *Journal of Learning and Teaching in Digital Age*, *3*(2), 42-50

Materials

The booklet prepared in the research is also presented to three groups in the same format and content at the beginning of the process. This booklet developed based on the book in the curriculum "Secondary School Computer Science" and consists of 4 pages. The restudying material group deeply reviewed this booklet throughout the process.

After the booklet is given in the worked example group, two isomorphic problems are given together with the solution steps. It is given the same problems with worked example and testing group. But as different from the worked example group, the solution steps were not given to testing group. In both groups, learners were expected to solve the problem using both flowcharts and pseudo-codes.

Retention tests consist of 3 open-ended problems which include variable identification, decision structures, pseudo-code and flowcharts in teaching algorithm. This test, which was also presented as a long-term retention test, was reapplied after 3 weeks from the initial retention test. The learners are expected to write the solution steps of each problem with both pseudo codes and flowcharts. The learners were given 30 minutes for both initial and final retention tests. The retention tests were applied as a paper-pencil exam.

A 20-item rubric was prepared by the researchers to assess the responses of the learners. Then, the rubric was presented to the expert opinions. Eight items were removed from the rubrics and 2 items were added in the direction of the expert feedback. In the last case of rubric, there are 14 items in total. Each item was rated on 2 points, so each problem was rated on 28 points. The maximum score for a student can get from the test is 84 points for three questions. In order to examine the consistency of these scores, the inter-rater reliability coefficient was calculated. In this study, Cohen's Kappa coefficient was used because both rubrics were categorical and 2 raters evaluated. As the result of the analysis, the Kappa coefficient was calculated as 0.681. The Kappa coefficient is interpreted as perfect fit over 0.75 (sometimes 0.80), poor fit below 0.40, and good fit between 0.40 and 0.75 (Krippendorff, 1980, Neuendorf, 2002). Accordingly, it can be said that the reliability between the raters is good.

In this study, a single-item 9-point rating scale, developed by Paas and van Merrienboer (1993) and adapted to Turkish by Kılıç and Karadeniz (2004), was given to the students immediately after the retention test. On the scale, students were asked "How much effort did it take to perform this task?" and asked to mark the effort they perceive, 1 "not at all", 9 "too much". Kılıç ve Karadeniz (2004) calculated the Cronbach Alpha internal consistency coefficient as 0.78.

With the purpose of calculating instructional efficiency, Paas & van Merrienboer (1993) developed a formula by considering the cognitive load imposed on while completing the test, as well as the performance in the retention test. Figure 1 shows the way how to calculate instructional efficiency.

$$Instructional\ Efficiency = \frac{Z_{Performance} - Z_{Cognitive\ Load\ During\ Learning}}{\sqrt{2}}$$

Figure 1. Instructional Efficiency Formula

According to this formula, the cognitive load scores and retention test scores handing out the students after retention test were first standardized. Then the efficiency of each instructional strategy is calculated according to this formula.

Procedure

In order to determine the effectiveness of the different instructional strategies used in teaching programming, the basic structure of programming and algorithm types were explained to all learners in the first week. In the second week, materials were handed out to the groups according to the instructional strategy they were exposed to. In the first 10 minutes of the first lesson, programming basics booklet was handed out to all groups. In the second and third 10-minutes periods for the restudy group, this booklet was re-presented and expected to be studied in depth from the learners. Between these 10 minutes periods, distraction tasks for 3 minutes were given to all groups. The learners in the testing group performed the solution of the problem consisting of a question after the first distraction task with flowchart and pseudo code. A similar problem has been presented after the second distraction task. A problem was handed out to learners in worked example group as with testing group for 10 minutes periods. However, unlike the testing group, the solution steps of the problem were specified and the learners were expected to understand the logic of the problem. In the second lesson, initial retention test and cognitive load scale were conducted for 30 minutes. After 3 weeks from the initial retention test, final retention test was conducted. All materials were handed out as print to participants. Figure 2 indicates the procedure.

© 2018, *Journal of Learning and Teaching in Digital Age*, *3*(2), 42-50

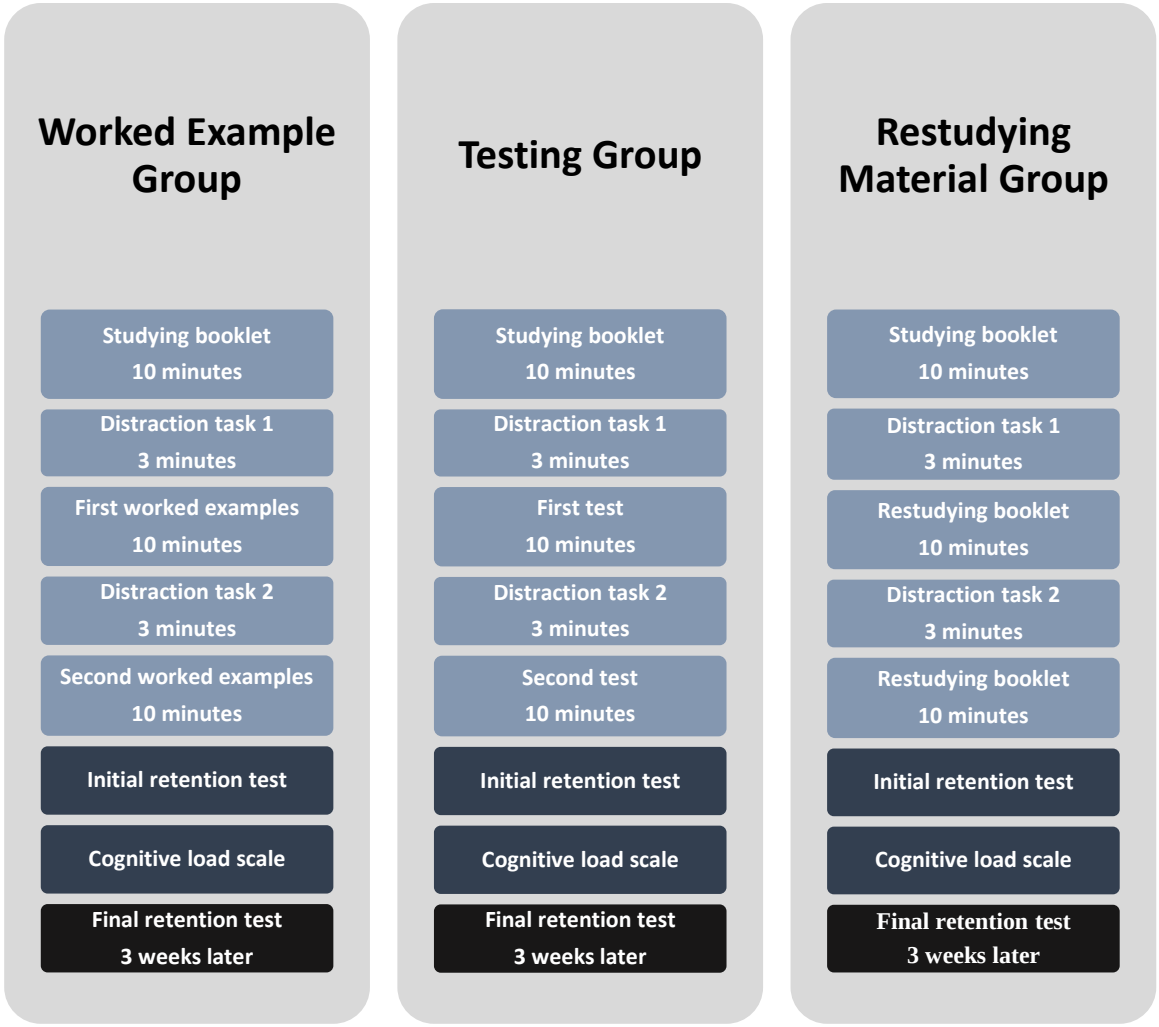


Figure 2. Intervention Procedure

FINDINGS

Initial Retention Test Performances

To answer the first problem of the study, which is "How do the different instructional strategies affect participants' initial retention test (after 10 minutes) performance?" ANOVA was performed. According to the initial retention test scores, the worked example group (n=37, m=32,73, sd=12,92) outperformed the other groups. Also, testing group (n=38, m=29,13, sd=13,61) was superior to the restudying group (n=31, m=25,67, sd=9,35). Table 1 presents the findings on the ANOVA, which was performed to find out whether the results were statistically significant or not.

As seen in Table 1, there was no significant difference between the instructional strategies in terms of the initial retention test performances.

Table 1. ANOVA findings on initial retention test scores

Group	Sum of Squares	df	Mean Square	F	p
Between Groups	842,92	2	421,24	2,802	0,065
Within Groups	15486,41	103	150,35		
Total	16328,90	105			

Final Retention Test Performances

ANOVA was performed to explore the second problem of the research which is "How do the different instructional strategies affect participants' final retention test (after 3 weeks) performance?" Based on the mean values, the worked example group (n=32, m=30,75, sd=11,48) outperformed the other groups. Also, testing group (n=37, m=28,89, sd=9,36) was superior to the restudying group (n=31, m=22,74, sd=9,72). ANOVA results are shown in Table 2.

© 2018, *Journal of Learning and Teaching in Digital Age*, *3*(2), 42-50

Table 2. ANOVA findings on final retention test scores

Group	Sum of Squares	df	Mean Square	F	p
Between Groups	1110,857	2	555,428	5,346	0,006
Within Groups	10077,503	97	103,892		
Total	11188,360	99			

Table 2 shows that there are significant differences according to instructional strategies in the scores of final retention test. Post-hoc analysis was conducted to determine which groups caused this difference. The results are presented in Table 3.

Table 3. Tukey test findings on retention test scores

Instructional Strategy(I)	Instructional Strategy (II)	Mean Difference (I-II)	p
Worked Example	Testing	1,85811	0,731
	Restudying	8,00806	0,007
Testing	Worked Example	-1,85811	0,731
	Restudying	6,14996	0,039
Restudying	Worked Example	-8,00806	0,007
	Testing	-6,14996	0,039

Table 3 demonstrates that worked examples and testing strategies are more superior to restudying strategy in terms of final retention performance. In other words, studying worked examples and testing strategies contribute to permanent learning. On the other hand, there is no significant difference between worked example and testing strategies.

Cognitive Load Levels

ANOVA was performed in order to explore the third research problem which is "How do the different instructional strategies affect participants' cognitive load?" Based on the mean values, testing strategy (n=37, m=6,87, sd=1,47) caused more cognitive load than the other two groups. In addition, cognitive load scores of the worked example group (n=32, m=6,19, sd=1,52) is higher than restudying group (n=31, m=5,90, sd=1,49). Table 4 shows the results of ANOVA analysis to determine whether these findings are significant or not.

Table 4. ANOVA findings on cognitive load levels

Group	Sum of Squares	df	Mean Square	F	p
Between Groups	17,357	2	8,679	3,874	0,024
Within Groups	230,727	103	2,240		
Total	248,085	105			

Table 4 shows that there is a significant difference in cognitive load levels of learners in terms of instructional strategy used. The Tukey test from Post hoc analysis was conducted to determine from which groups this difference originate. The findings are presented in Table 5.

Table 5. Tukey test findings on cognitive load levels

Instructional Strategy(I)	Instructional Strategy (II)	Mean Difference (I-II)	p
Worked Example	Testing	-0,67923	0,126
	Restudying	0,28596	0,713
Testing	Worked Example	0,67923	0,126
	Restudying	0,96520	0,024
Restudying	Worked Example	-0,28596	0,713
	Testing	-0,96520	0,024

Table 5 indicates that there is a significant difference between testing and restudying groups with regards to cognitive load levels. In other words, the testing group's cognitive load level is higher than restudying group. Nevertheless, there was no significant difference when comparing the other groups with each other.

Instructional Efficiency

To explore the fourth problem of the study, which is "Is there a significant difference between the instructional strategies in terms of instructional efficiency?" ANOVA was performed. According to instructional efficiency scores, worked example (m=0,264, sd=1,01) is more efficient strategy that the other strategies in terms of instructional efficiency. Also it was found that restudying (m=-0,005, sd=0,68) is superior to testing strategy (m=-0,253, sd=0,82). Table 6 demonstrates the ANOVA results conducted to explore whether these findings are significant or not.

© 2018,

Table 6. ANOVA findings on instructional efficiency

Group	Sum of Squares	df	Mean Square	F	p
Between Groups	5,002	2	2,501	3,408	0,037
Within Groups	75,581	103	0,734		
Total	80,583	105			

Table 6 shows that there is a significant difference in instructional efficiency in terms of instructional strategy used. The Tukey test from Post hoc analysis was conducted to determine from which instructional strategy this difference originates. The findings on Tukey test are presented in Table 7.

Table 7. Tukey test findings on instructional efficiency

Instructional Strategy(I)	Instructional Strategy(II)	Mean Difference (I-II)	p
Worked Example	Testing	0,51649	0,028
	Restudying	0,26833	0,406
Testing	Worked Example	-0,51649	0,028
	Restudying	-0,24815	0,458
Restudying	Worked Example	-0,26833	0,406
	Testing	0,24815	0,458

Table 7 demonstrates that worked example strategy is more efficient than testing with regard to instructional efficiency. Nevertheless, there was no significant difference when comparing the other groups with each other.

DISCUSSION AND CONCLUSION

The purpose of this study is to determine the effects of different instructional strategies (testing, restudying material and studying worked example) on initial and final retention, scores, cognitive load, and to compare these strategies in terms of instructional efficiency in teaching programming to high school students. As a result of the research, it was seen that the applied instructional strategies did not make a difference in the achievement test immediately after the learning process. This result supports the findings in the literature (Demiraslan Çevik & Çoban, 2016; Van Gog et al., 2015). However, the use of the tests as a study material generally influences long-term retention performance (Roediger & Karpicke, 2006a).

When final retention test results are taken into account, testing and studying worked examples are more effective strategies than restudying material. It is consistent with the findings in the literature that studying the test is superior to the restudying material in the context of final retention (Butler & Roediger, 2007; Roediger & Karpicke, 2006b). So this is an expected outcome, and the strategy that needs to be focused on here should be studying worked example. In both the studying worked example and the restudying the material strategies, the learners study on the material given to them. However, since worked examples are supported by self-explanations, learners may have had a more focused and more active process to understand the underlying logic of the problem. Thus, why the studying worked example leads to more permanent learning than restudying material can be explained in this way.

When the degree of the applied instructional strategies' influence on the cognitive load levels of the learners was examined, it was determined that the testing strategy causes more cognitive load than the restudying strategy. This result may be due to the fact that the learners' cognitive resources have been directed to inefficient search strategies during testing. It can be said that problem-solving and testing strategies are not the same strategies but they can affect the cognitive load in a similar way. Studies show that the problem-solving method leads to excessive cognitive load (Renkl, 2014; van Merrienboer & Sweller, 2010). Therefore, this result can be explained by the findings that the testing strategy causes more cognitive load than the restudying strategy.

The main purpose of this study is to compare the efficiency of different instructional strategies in programming instruction. Performance scores and cognitive load levels of learners were used to determine instructional efficiency. As a result, it is concluded that the worked example with self-explanation prompts is the most superior strategy in terms of instructional efficiency. This result proved to be consistent with the findings that the strategy of worked examples is a more efficient one than the traditional problem solving strategies for novice learners (Atkinson et al., 2000; Renkl, 2014). One of the reasons for this result may be that the worked example is an effective strategy in optimizing the cognitive load. Therefore, the fact that learners show similar performance with less cognitive effort proves that the worked example strategy is efficient. In addition, supporting worked examples with self-explanation prompts may have provided learners with a more active learning process.

In sum, when the literature is examined, it is seen that testing and restudying material strategies are generally compared in studies related to testing effect. In this study, worked examples as a different strategy has been tested and it has been determined that it is more efficient than the other two strategies for teaching programming basics. In addition, the fact that testing has increased the long-term retention of knowledge has been confirmed. However, when cognitive load levels were taken into

account, there was no difference between the testing and the restudying material strategies.

The scope of this research is limited to teaching algorithm and basic programming concepts. Strategies used in this study can be investigated with more complex concepts, such as loops, classes, and functions, which require more advanced programming skills. Thus, the efficiency of studying worked example can be tested in cases where high level programming knowledge and skills are required. In this study, a subjective scale consisting of a single item was used in determining the cognitive load. The reliability of this scale, developed by Paas (1992), has been tested with various studies. However, the level of cognitive activity in the learning process can be determined with objective measures such as electroencephalography (EEG), eye tracking, etc., to ensure more reliable results. In subsequent studies, one of the objective measurements can be used in addition to the cognitive load rating scale.

REFERENCES

Abdul-Rahman, S. S., & Du Boulay, B. (2014). Learning programming via worked-examples: Relation of learning styles to cognitive load. *Computers in Human Behavior*, *30*, 286-298.

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.

Bergersen, G. R., & Gustafsson, J. E. (2011). Programming skill, knowledge, and working memory among professional software developers from an investment theory perspective. *Journal of Individual Differences.*

Butler, A. C., & Roediger, H. L. (2007). Testing improves long-term retention in a simulated classroom setting. *European Journal of Cognitive Psychology*, *19(4-5)*, 514-527.

Chi, M. T., 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*, *13*(2), 145-182.

Demiraslan Çevik, Y., & Çoban, T. (2016). Testing effect in learning digital property and cyber ethics. *SDU International Journal of Educational Studies, 3(1)*, 84-99.

Hoogerheide, V., Renkl, A., Fiorella, L., Paas, F., & Van Gog, T. (2018). Enhancing example-based learning: Teaching on video increases arousal and improves problem-solving performance. *Journal of Educational Psychology*.

Johnson, C. I., & Mayer, R. E. (2009). A testing effect with multimedia learning. *Journal of Educational Psychology*, *101*(3), 621.

Kılıç, E., & Karadeniz, Ş. (2004). Hiper ortamlarda öğrencilerin bilişsel yüklenme ve kaybolma düzeylerinin belirlenmesi. *Kuram ve Uygulamada Egitim Yönetimi Dergisi*, *10*(4), 562-579.

Leahy, W., Hanham, J., & Sweller, J. (2015). High element interactivity information during problem solving may lead to failure to obtain the testing effect. *Educational Psychology Review*, *27*(2), 291-304.

Lee, N. (2013). *The Effects of Self-Explanation and Reading Questions and Answers on Learning Computer Programming Language (*Unpublished Doctoral Dissertation). University of Nevada, Las Vegas.

Likourezos, V., & Kalyuga, S. (2017). Instruction-first and problem-solving-first approaches: alternative pathways to learning complex tasks. *Instructional Science*, *45*(2), 195-219.

Margulieux, L. E., Catrambone, R., & Guzdial, M. (2016). Employing subgoals in computer programming education. *Computer Science Education*, *26*(1), 44-67.

Margulieux, L. E., & Catrambone, R. (2016). Improving problem solving with subgoal labels in expository text and worked examples. *Learning and Instruction*, *42*, 58-71.

Mayer, R. E. (2013). *Teaching and learning computer programming: Multiple research perspectives.* Routledge.

Paas, F. G., & Van Merriënboer, J. J. (1993). The efficiency of instructional conditions: An approach to combine mental effort and performance measures. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, *35*(4), 737-743.

Renkl, A. (2014). Toward an instructionally oriented theory of example-based learning. *Cognitive science*, *38*(1), 1-37.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, *13*(2), 137-172.

Roediger, H. L., & Karpicke, J. D. (2006a). The power of testing memory: Basic research and implications for educational practice. *Perspectives on Psychological Science, 1*, 181–210.

Roediger, H. L., & Karpicke, J. D. (2006b). Test-enhanced learning – Taking memory tests improves long-term retention. *Pyschological Science, 17*(3), 249-255. Doi: 10.1111/j.1467-9280.2006.01693.x

Si, J., Kim, D., & Na, C. (2014). Adaptive Instruction to Learner Expertise with Bimodal Process-oriented Worked-out Examples. *Educational Technology & Society*, *17*(1), 259-27

Skinner, B. F. (2016). *The technology of teaching.* BF Skinner Foundation.

Sweller, J. (1989). Cognitive technology: Some procedures for facilitating learning and problem solving in mathematics and science. *Journal of educational psychology*, *81*(4), 457.

Sweller, J. (2010). Element interactivity and intrinsic, extraneous, and germane cognitive load. *Educational psychology review*, *22*(2), 123-138.

Terry, W. S. (2011). Öğrenme & Bellek: Temel İlkeler, Süreçler ve İşlemler. Çev., Banu Cangöz. Ankara: Anı Yayınları.

Uçar, B. & Demiraslan Çevik, Y. (2018). Test Etkisinde Farklı Öğrenme Koşullarının Etkisi: Güvenli İnternet Kullanımı Konusu Örneği. *Bartın Üniversitesi Eğitim Fakültesi Dergisi*, *7*(1), 29-66.

Van Gog, T., & Kester, L. (2012). A test of the testing effect: acquiring problem-solving skills from worked examples. *Cognitive Science*, *36*(8), 1532-1541.

Van Gog, T., Kester, L., Dirkx, K., Hoogerheide, V., Boerboom, J., & Verkoeijen, P. P. (2015). Testing after worked example study does not enhance delayed problem-solving performance compared to restudy. *Educational Psychology Review*, *27*(2), 265-289.

Van Gog, T., & Rummel, N. (2010). Example-based learning: Integrating cognitive and social-cognitive research perspectives. *Educational Psychology Review*, *22*(2), 155-174.

Vihavainen, A., Airaksinen, J., & Watson, C. (2014, July). A systematic review of approaches for teaching introductory programming and their influence on success. In *Proceedings of the tenth annual conference on International computing education research* (pp. 19-26). ACM.