International Journal of Technology in Education

www.ijte.net

e-Learning Objects Designing Approach for Programming-based Problem Solving

Abdullah Basuhail
Faculty of Computing and Information Technology, King Abdulaziz University, KSA

To cite this article:

Basuhail, A. (2019). e-Learning objects designing approach for programming-based problem solving. *International Journal of Technology in Education (IJTE), 2*(1), 32-41.

International Journal of Technology in Education (IJTE) is a peer-reviewed scholarly online journal. This article may be used for research, teaching, and private study purposes. Authors alone are responsible for the contents of their articles. The journal owns the copyright of the articles. The publisher shall not be liable for any loss, actions, claims, proceedings, demand, or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of the research material. All authors are requested to disclose any actual or potential conflict of interest including any financial, personal or other relationships with other people or organizations regarding the submitted work.

IJTE International Journal of Technology in Education
Volume 2, Issue 1, Fall 2019

e-Learning Objects Designing Approach for Programming-based Problem Solving

Abdullah Basuhail

Article Info

Article History

Received:
15 August 2019

Accepted:
19 October 2019

Keywords

Learning objects
Problem-solving
Programming
Animation and graphics
e-Learning
Educational technology

Abstract

Problem solving techniques are one of the primary topics introduced to the computing and information technology students in tertiary education. In general, those techniques are presented using a collection of traditional tools, such as texts, graphs, illustrations, procedural steps, pseudocodes, algorithms, and flowcharts. In many cases, there is no direct link between problem-solving from one side and the techniques and the programming structures used from the other side. This paper presents an approach that exploits computer interactive animations and graphics to implement learning objects for teaching and learning problem-solving based on computer programming. The main feature of this approach is its smoothness in associating the programming concepts and structures provided to the solution of a designated problem. Moreover, the developed learning objects are featured by the characteristic of reusability and possibility of inclusion in e-learning systems. The designed learning object can function as a workable hands-on model for the learners to get experience in programming. The technique we developed for the design and implementation of the learning objects for programming-based problem-solving can be utilized in other problem-solving disciplines.

Introduction

Information technology centered learning is very essential in contemporary tertiary education and it became increasingly pervasive. Its inclusion in education can assist the learners to realize information and concepts in an effective and a timely manner. Furthermore, it can save times that can be exploited towards more discoveries of the topic matters. E-learning objects are considered among the technological tools that can be exploited in modern educational systems. A learning object (LO) is an integration of digital information and/or practical units that are all incorporated to emphasize a single educational objective. It plays a significant role in the applied experience of the learners. The Institute of Electrical and Electronics Engineers (IEEE) defines a learning object as “any entity, digital or non-digital, that may be used for learning, education or training” (Standard for learning object metadata, 2002).

The use of the controlled learning objects can support the learners with the capability to explore the instructional concepts on their own. Furthermore, those learning objects can make the instructional context to be intuitively perceived. The use of the learning objects supplemented with animations and graphics are considered as superior and powerful tools to be incorporated in various education disciplines to present theories and concepts. They offer techniques and tools that can be applied to show temporal changes and sequences of the topic matters. In general, the mechanism of problem-solving using computer programming-based explanation is illustrated by the means and support of procedural steps, pseudocodes, sequences, algorithms, flowcharts and processes. In this paper, we suggest implementation of programming-based problem-solving using electronic learning objects supported with computer animations and graphics.

The use of technology-based techniques and tools, such as computer animations, graphics, and/or learning objects, and its impact in the tertiary education has been pointed out in many studies and researches. For example, Henderson (1994) applied animated models for teaching notions of computer systems organization. Smith and Escott (2004) reported a noticed improvement in the students’ achievements using animations in teaching computing concepts. Osman and Elmusharaf (2014) demonstrated the effectiveness of integrating algorithm and program animation in teaching data structure techniques. Some researchers directed analysis for using instructional animation and they found significant benefit of its use over the use of solely static elements (Hoffler & Leutner, 2007). Hwang, Tam, Lam, and Lam (2012) studied animations and indicated their effectiveness in the learning. Lowe (2004) illustrated that animation is an interesting tool to be used for the learners. Falvo (2008) and Rotbain, Marbach, and Stavy (2008) indicated that the use of animations gives an

accurate and rich representation for topics which are very hard to grasp from solely text-based approach. The use of computer animation and graphics increases the learner confidence in learning programming as reported by Shindo (2000).

Specifically, the use of the learning objects in the teaching and learning programming and its effectiveness and challenges has been addressed in many research studies (Narasimhamurthy & Al Shawkani, 2009). Researchers concluded that the use of the learning objects contributes significantly to the teaching-learning process of programming (Adamchik & Gunawardena, 2003; Begosso, Begosso, Begosso, Ribeiro, & Martins, 2015; Narasimhamurthy & Al Shawkani, 2009). Some authors (Halverson, Wolfenstein, Williams, & Rockman, 2009) showed that the design of digital learning objects can spark professional learning. Interactive learning object are appreciated by many instructors in the search for new methods and support for novice programming students (Matthiasdottir, 2006). Several studies showed that the use of the learning objects contributes significantly to the improvement of computer science teaching-learning process (Andreza & Magalhães, 2019; Bastos & Francisco, 2019; Begosso, Begosso, & Begosso, 2016; Cavus & Ibrahim, 2004; Luna-Ramírez & Jaimez-González, 2014; Jaimez-González, García-Mendoza, Luna-Ramírez, Nápoles-Duarte, & Vargas-Vargas, 2018; Villalobos & Jiménez, 2009).

This paper proposes a technique that utilizes electronic learning objects supported with computer animations and graphics to teach and learn programming concepts and structures designed for problem-solving. It demonstrates an approach for tracking programs' execution mechanism in correspondence with the problem-solving. The technique can be as well moderated to fit and adopted in various educational disciplines.

Programming-based Problem Solving

Problem-solving techniques are considered as a supporting pillar of computer programming in tertiary education. In general, computer programs are written based on problem-solving. Many techniques, structures and concepts are devoted during the teaching of programming to assist the process of providing solutions to those problems. The programming structures and concepts include a collection of numbers, constants, variables, initialization, manipulation, assignments, conditions, loops, arrays, and data structures. Instructors dedicate times and exert efforts to transfer the knowledge about these notions to the learners. Along with the verbal words and gesture signs, they may use some other supporting tools, such as text labels, markings, drawings, …etc., to illustrate the logical sequence of execution of the developed solutions. However, these tools are static in their nature, and they may require the excretion of extra efforts to stimulate the learners to understand the method applied for fashioning a certain programming problem solution. Furthermore, using these solutions in e-learning and self-learning environments, require extra support towards the learners to understand the notions crafted for the problem.

The application of interactive learning objects in knowledge transfer can support the learning of the programming notions applied to the problem-solving techniques. Moreover, using this technique during the learning can save significant efforts that can be devoted towards attaining additional practices and experiences of the topic matter. The model proposed in this paper provides a design for learning objects that integrates programming-based concepts and structures in association with the problem-solving techniques.

Method

Learning objects for programming-based problem solving can be developed based on a systematic methodology. The design can illustrate the concepts and structures of the program codes implemented for problem-solving. Figure 1 shows a proposed model for this approach. The model consists of six stages. In the following sections the stages of this model will be highlighted.

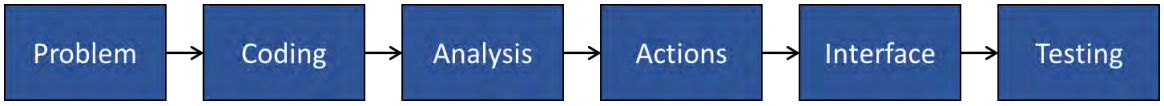

Figure 1. Learning Object Design Model for Problem-solving

Problem Description

The design methodology of a learning object for problem-solving starts with the setting of well documented description of the problem under consideration. The description could be expressed in textual, pseudocode, algorithmic, or graphical format. The output of this stage is in the form of procedural steps, algorithm or pseudocode that fully describes the designated problem.

Coding

The next step in the proposed approach is to translate the problem description from the previous stage into program codes, using a programming language of preference, that precisely realize a complete programming solution to the problem. The output of this stage can be used as a basis to implement the learning object for the operation and functioning of the problem's solution code.

Code Analysis

The written program code from the coding stage is partitioned into small segments that complement each other. An individual code segment is analyzed into a sequence of steps. In general, the code segments consist of collection of programming structures and concepts, such as different data types, variables, arrays, indices, initializations, conditions, loops, iterations, assignment operations, manipulations, etc. These components are clearly singled out for later transformation to sequence of actions. By the completion of this stage, an analysis table that describes the program code by a sequence of steps will be generated.

Interface

In the next step, an interface to the learning object is designed. The designed interface is based on the nature of the problem and the output of the analysis from the previous stage. Related components such as textboxes, labels, and graphics are integrated to the learning object display as the need arises. This will support in the clarification of the problem-solving topic matter to the learners. Furthermore, control elements such as command buttons, radio buttons, and check boxes are incorporated to facilitate the interaction of the user with the learning object. The use of these controls can assist to steer the sequences of the actions in accordance with the logical order of the program code provided for the problem under consideration. It also can let the user to interact with the learning object and observe the consequences.

Actions Implementation

Afterwards, the actions are imitated with the support of computer graphics and animations. The flowing of the actions through the entire lifecycle of the learning object design is the most critical phase. In general, it consumes most of the time and effort in the entire development process. Superimposing cascaded animations over the different object's components is performed in this stage. It is done by applying transitions steered by the problem under consideration. Assorted animations are applied carefully to imitate the intended learning objectives in accordance with the mechanism of the problem-solving. An assortment of animations, such as e*ntrance*, *emphasis*, *exit*, and *motion path* categories with multiple effects, can be cascaded over the integrated components of the learning object. The animation transitions and processions are coded to be triggered either by mouse clicks, or by setting timers using the control elements that have been incorporated in the learning object interface.

Testing

In general, the implemented programming-based problem-solving learning objects are intended for use in e-learning systems or might be integrated to function with other learning objects. Furthermore, the object could also be integrated in a session for the use by instructors during teaching sessions, or it could be shared with the learners using electronic learning management systems (LMS) or over the Internet. The correct functioning and the precise sequence of proceeding must be tested to overcome any confusion that may arise from the use of the

learning object. A thoroughly testing procedure can be applied to ensure accurate and precise operation of the overall created learning object and to capture confusions.

In the case of any operational inconsistency or malfunction of the learning object, the design procedure of this approach is revisited until the final learning object outcome satisfies the desired functional operation. The model described here has been applied to implement learning objects for numerous structures and concepts used in programming-based problem-solving. In the next section, an application will be demonstrated to illustrate the procedural steps of the proposed model.

Results

In this section we will illustrate the application of the proposed methodology to implement a learning object for programming-based problem solving of a certain problem. The application is a problem in probability and statistics, specifically, collecting and analyzing survey results.

Application

The problem which is considered here is stated as follows: a survey is conducted to collect the opinions of students towards the service provided by the student affairs office (SAO) at an academic institution. A sample of students of size *n* are to be enquired to rate the quality of the service provided by SAO on a scale from 1 to 10, with 1 being *unsatisfied* and 10 being *excellent*. It is intended to conduct the survey by collecting the responses of the students and then determine the frequency of each rating for further data analysis.

In order to provide a programming-based solution for the given problem, a program code is written. Here, it is intended to simulate the survey problem using computer programming. The code simulates the experiment of inputting responses of *n* students. The code reads in the students' responses and then it determines the frequency of each rating by examining the value of the response. Figure 2 lists a code-segment written in the C# high-level programming language that is implemented for this purpose. An array that consists of ten (number of different ratings) counters are used to tally the occurrences of student responses. Based on the response outcome, the corresponding counter is incremented.

```
1. int[] frequency = new int[11];
2.  int student, response;
3.  for (student=1; student<=n; ++student) {
4.        response = System.in.read ();
5.  frequency[ response ]++; }
```

Figure 2. Code-segment for the Student's Survey Problem

The code-segment comprises several programming structures and concepts. Specifically, the code includes, array creation and initialization, variables declaration, loop iteration, range, condition, array manipulation, index entry, assignment, and increment. A one-dimensional array is used to tally the number of times each possible response appears. Based on the survey problem, Table 1 lists the concepts and structures used in this programming code.

Table 1. Programming Concepts and Structures used in the Survey Problem

Line	Programming Concept/Structure
1	array creation and initialization
2	variables declaration
3	loop, control variable, iteration, initial value, condition, increment
4	input student's response
5	assignment operation, array-index entry, array manipulation, increment

Table 2 lists the analysis steps of the code for the actions to be performed in the learning object based on the structures and concepts intended to be learned and grasped from the programming code of the problem-solving.

Table 2. Analysis of Program Code

Program Code	Concept/Structure Description
`int[] frequency = new int[11];`	1. use array of 11 elements 2. initialize all the array elements by value of 0 3. array index 0 entry is never used
`int student, response;`	4. declare and initialize two variables
`for (student=1; student<=12; ++student)`	5. iterate for 12 times 6. use *student* as loop control variable 7. initialize *student* by 1 8. increment *student* variable 9. check *student* does not exceed 12 (stopping condition)
`response = System.in.read ();`	10. read-in student's input 11. assign input to *response*
`frequency[response]++;`	12. use *response* value as an entry to the *frequency* counters 13. increment the corresponding frequency counter by 1

The design of the interface for problem-solving learning object can be modeled using integrated components. Figure 3 demonstrates an exhibition that constructs an interface for the learning object for the survey problem. The exhibit comprises of seven elements, specifically, *response* callout that displays the student rating, 12 *faces* that represents the students who will participate in the survey, *student* that displays the loop iteration variable, seven-boxes that represent the *frequency* counters each of which will hold the tally of the occurrences of the corresponding student response, *next* control button for steering the sequence of the actions of the learning object, *textbox* for brief textual description of the problem, and *code* box to facilitate association between the code and the performed actions in the visualization of the learning object.

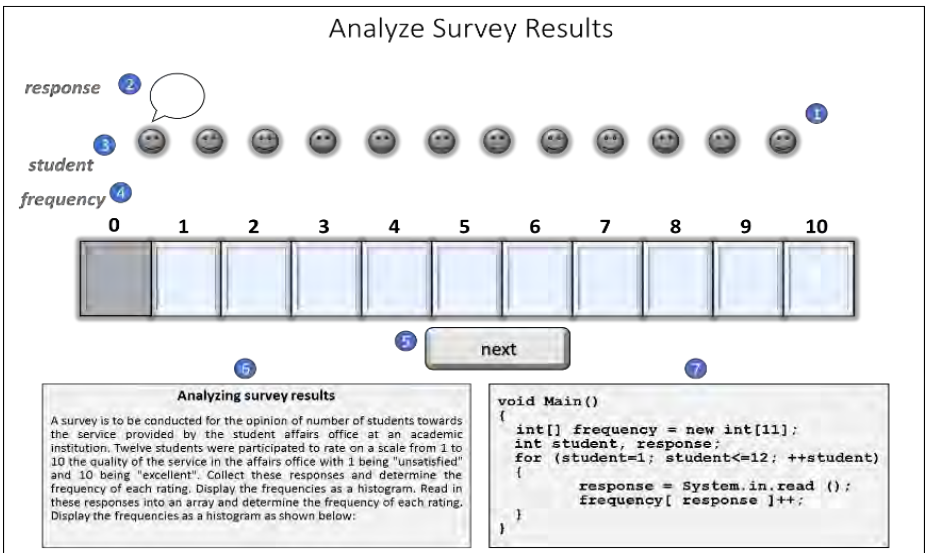

Figure 3. Exhibit for the Learning Object of the Survey Problem-solving

Next, based on the nature of the problem under consideration and the analysis of the programming code, the actions that will be applied to the learning object to show the mechanism of the programming-based problem-solving are generated. For the survey problem, the generated actions are tabulated in Table 3. A complete learning object for the problem-solving of the survey problem was implemented using integrated animation tools. The implemented learning object can be used in the learning of several programming structures and concepts of the problem-solving.

Table 3. Sequence of Actions for the Students' Survey Learning Object Problem-solving

Actions Generation
1. display the LO initial state
2. wait until *next* control button is pressed by the LO user
3. disable *next* control button
4. display 0s in all *frequency* counters
5. disable (dim) *frequency* counter number 0 (indicate "never appearing response" state)
6. enable *next* control button
7. wait until *next* control button is pressed
8. set *student* to 1
9. repeat
10. disable *next* control button
11. input *response* value in the callout
12. blink and color *frequency* counter matching *response* value
13. increment value of matching *frequency* counter by 1
14. reset color in 12 back to original (indicate update completion)
15. enable *next* control button
16. wait until *next* control button is pressed
17. increment *student* by 1
18. until *student* value exceeds *n*
19. Display "*Done*" message (indicate termination)

The learner can start using this learning object by pressing the *next* control button. This action will enable the execution of the problem-solving learning object (PSLO). Figure 4 displays the learning object initial state. It demonstrates a frame that shows the concept of initialization of the *frequency* counters which corresponds to the array used in the programming code. As can be noticed, all of the counters are initialized with the value 0. Furthermore, the frame illustrates the values of the possible range of the responses of the survey experiments, as the 0 value can never appear from any student response. This is indicated by the darkening of the counter with entry 0 among the set of the *frequency* counters.

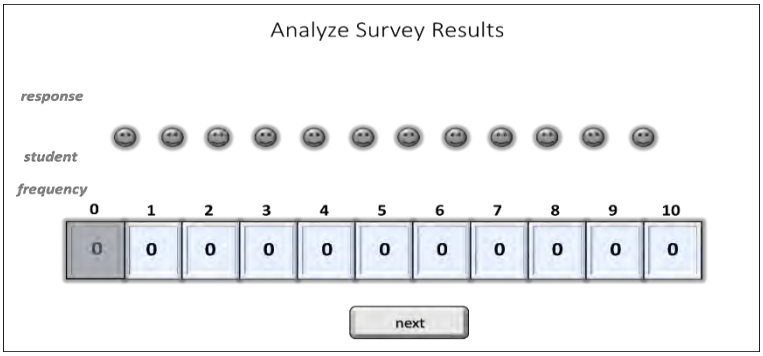

Figure 4. Initial State of the Survey PSLO

Figure 5 demonstrate a frame of running the learning object after clicking the command button *next* for one time. The frame illustrates the concept of inputting response, where the student response is displayed in the callout.

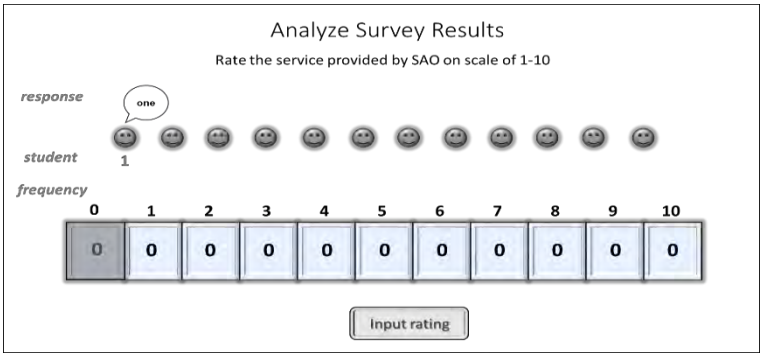

Figure 5. Inputting Response to the Survey PSLO

Figure 6 shows the state after the completion of this loop iteration in the programming code. As can be observed, the counter inside box 1 of *frequency* is incremented by a value of 1 as a result to the observed student response to the survey.

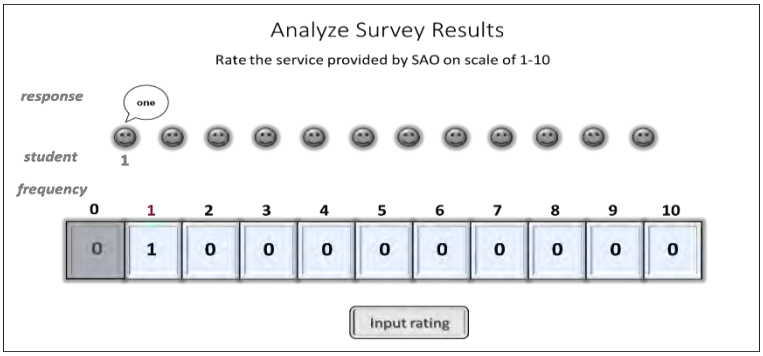

Figure 6. Manipulation of the Survey PSLO

For more illustration, the second iteration of the survey experiment is shown in Figure 7 which is performed by clicking the *next* control button for the second time. The *student* is corresponding to the current loop iteration and the *response* value is used as an entry index to increment the counter inside *frequency* box number1.

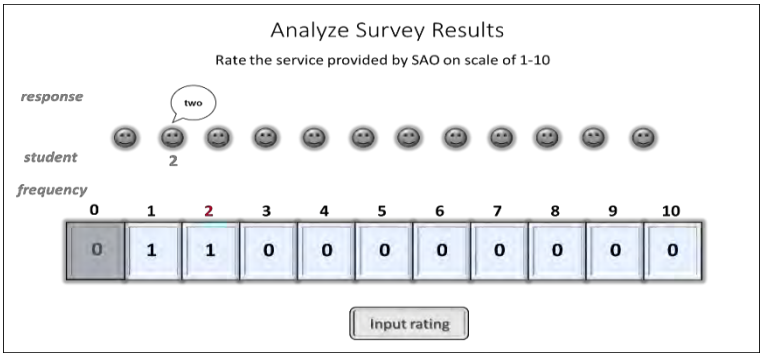

Figure 7. Iteration 2 of Execution of the Survey PSLO

Based on the performed experiments from 12 runs of PSLO, the responses of all the iterations were as follows: 1, 2, 6, 2, 8, 5, 6, 6, 3, 10, 7, 5. The frequency of each rating based on the input responses is listed in Table 4.

Table 4. Survey Responses Outcomes of Running PSLO for 12 Times

Rating	1	2	3	4	5	6	7	8	9	10	Total
Frequency	1	2	1	0	2	3	1	1	0	1	12

Figure 8 illustrates the last frame after the completion of all the iterations performed by running the learning object. As can be observed, the summation of the values of the counters reflects the total number of experiments performed; consequently, it corresponds to the number of iterations that has been completed by the loop code before the loop condition became false.

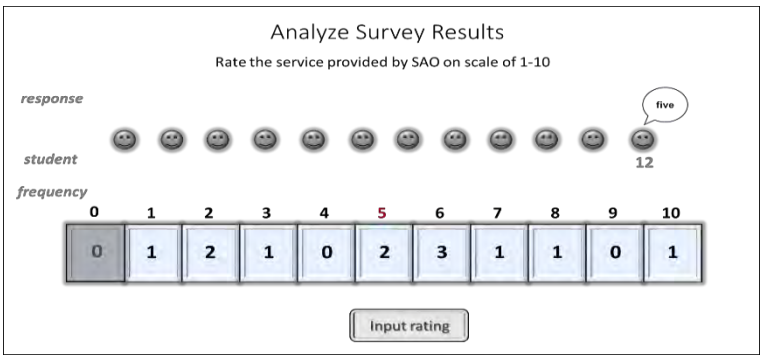

Figure 8. State at Iteration 12 of Execution of Survey PSLO

After the completion of the 12th experiment, *student* variable, which tracks loop control variable, is incremented by 1 according to the loop header code. This value makes the loop condition to be false, which indicates the completion of the experiments. Thus, it will lead to loop termination. This situation is shown in Figure 9. The control button will indicate the learning object termination by disabling the button and displaying the message *Done*.

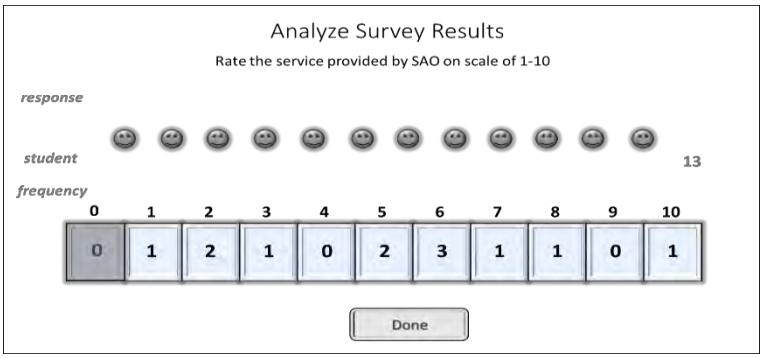

Figure 9. PSLO after Execution Completion

The use of the PSLO by a learner for repeated times is possible. This illustrates the feature of the learning object reusability; thus, it enhances the learning process by highlighting the same information as much as the need of the learner.

Discussion and Conclusions

This paper proposes a technique that utilizes electronic learning objects supported with computer animations and graphics to teach and learn programming concepts and structures for problem-solving. It shows a method for tracking programs' execution mechanism in correspondence with the problem-solving. The technique can be, as well, moderated to be fitted and adopted in various educational disciplines. Based on the literature review that we have conducted, the method that we suggested here can be considered as a distinct methodology of applying the learning objects in teaching and learning programming-based problem-solving techniques. The method of exploiting the learning objects for programming-based problem-solving has been used as supporting learning supplements in college-level related courses. We have used the developed learning objects and shared them among students and instructors for several semesters of teaching programming-based problem-solving techniques. The students stated that they benefited from the use of the learning objects by the improved understanding of the programming techniques used in the problem-solving. Usually, after the students experienced with the learning objects that have been implemented for some problems, the students asked for more learning objects that they could use for other problems which have no implemented learning objects. This indicates the importance of the supplemental learning objects to the learning styles of the students. As students mentioned, "the learning objects helped them to understand the notions of the problem-solving by representing the topic matters in a visualized manner." It demonstrated the process and facilitated the association of the scattered pieces of the subject components. Some students commented that it is much easy to visualize the concept, to understand it, and to remember using such trends.

Another advantage of using the learning objects is that students have the chance to use the learning objects to work with and manipulate on their own, and to replay as the need arises. From our observations and comments of other instructors who have been involved in the use of the implemented teaching objects, the students showed faster understanding of the related topic matters. They showed increased involvement and motivated to be engaged during class sessions. The instructors who used the learning objects during class sessions were very enthusiastic and expressed their appreciation with this way of teaching. Some commented that it gave them an opportunity to gain insight into the teaching process and saved times during the teaching of the related topic matters compared to the traditional techniques of teaching. Some instructors expressed their desire to use this model in teaching other topics as well. The approach demonstrated in this paper was presented in some institutions to the academic staff members in quite a few workshops. Several related workshops were also presented. Most of the attendants were highly interacting during the sessions and they expressed their appreciation and welling to develop educational material using the proposed technique.

This paper demonstrated an approach for the design stages of learning objects for programming-based problem-solving using animations and computer graphics. Practices and experiences during the teaching and delivering of related courses were adopted in the design approach. The main feature of this approach is its effectiveness and support in teaching and learning the programming concepts and structures used in the problem-solving techniques. The produced outcomes can function as practical experience for the learners of the subject matter. The developed learning objects are characterized by the reusability, and operability in an e-learning environment. The technique we applied for implementing the learning objects for computer programming-based problems could be, as well, extended to other fields and disciplines of science and technology. Appropriate introduce of the problem-solving learning objects in the teaching-learning process is essential to encourage frequent usage. However, it is expected that the usage to be based on the learning styles of the learners. Learning objects are not the witchcraft that we might rely on in teaching programming concepts and structures, but it could be useful, specifically, if they are integrated into teaching and learning as a natural part of the learners programming practices.

There are some limitations of the technique illustrated in this paper. As an example, the experiments that can be conducted using the learning object that we showed here are limited to the implementation. There is no flexibility to expand the experiment by the user; for example, in the illustrated application, changing the number of survey participants. This modification requires reprogramming, and in turn it involves dedicating more resources. Another limitation arises in terms of the problem that is handled by the learning object. It is a specific problem that is chosen by the learning object developer. This means the considered problem should be carefully selected to be a good representative to the topic matter. Otherwise, the learning object might not serve the intended purpose. These limitations and some others can be resolved by the careful analysis of the design process of the learning objects. As future work, it is intended to use this approach to implement more learning objects for problem-solving of different nature in order to revisit the proposed model that can better fit in other related educational paradigms, such as flipped-learning, active-learning, and m-learning. It is also suggested to make the use of the learning objects to be more flexible by accepting inputs by other means than keyboard or mouse inputs, such as accepting voice inputs that can make the learner to interact with the learning object through speech.

The author of this paper recommends using the methodology illustrated in this article to develop and utilize the learning objects in teaching and learning as it can facilitate grasping the techniques related to the problem-solving by the learners. It is also recommended to encourage the instructors to get benefit of the approach illustrated here to implement learning objects due to their effectiveness in terms of use and resources utilization. Another recommendation is to raise the awareness among instructors for the importance of the learning objects development. This matter might requires holding short-courses and workshops to familiarize the instructors with the production of the learning objects using relatively straightforward and attainable tools based upon their current capabilities and experiences. Finally, it is recommended that instructors share their developed learning objects with other colleagues as it is will open the doors for exchanging ideas and enhancing and enriching education process using technology.

References

Adamchik, V., & Gunawardena, A. (2003). A learning objects approach to teaching programming. Proceedings of the International Conference on Information Technology: Computers and Communications (ITCC 03), (pp. 96-99). Retrieved from https://ieeexplore.ieee.org/document/1197507

Bastos, A., & Francisco, J. (2019). Inclusive Model Application Using Accessible Learning Objects to Support the Teaching of Mathematics. Informatics in Education, 18, 213-226.

Begosso, L., Begosso, L., & Begosso, R. (2016). An approach for the use of Learning Objects in teaching Computer Programming concepts. IEEE Frontiers in Education Conference (FIE), (pp. 1–8). USA. Retrieved from https://ieeexplore.ieee.org/document/7757619

Begosso, L., Begosso, L., Begosso, R., Ribeiro, A., & Martins, R. (2015). The use of learning objects for teaching computer programming. IEEE Frontiers in Education Conference, (pp. 786–791). USA. Retrieved from https://ieeexplore.ieee.org/document/7344148

Cavus, N., & Ibrahim, D. (2004). Using learning objects to teach programming languages. Creating the Future 3rd FAE International Symposium, (pp. 303-308). Retrieved from https://files.eric.ed.gov/fulltext/ED503158.pdf

Falvo, D. (2008). Animations and simulations for teaching and learning molecular chemistry. International Journal of Technology in Teaching and Learning, 4(1), 68–77.

Halverson, R., Wolfenstein, M., Williams, C., & Rockman, C. (2009). Remembering Math: The Design of Digital Learning Objects to Spark Professional Learning. E-Learning and Digital Media, 6(1), 97–118.

Henderson, W. (1994). Animated models for teaching aspects of computer systems organization. IEEE Transactions on Education, 37(3). Retrieved from https://ieeexplore.ieee.org/document/312133

Hoffler, T., & Leutner, D. (2007). Instructional animation versus static pictures: A meta-analysis. Journal of Learning and Instruction, 17, 722–738.

Hwang, I., Tam, M., Lam, P., & Lam, S. (2012). Review of use of animation as a supplementary learning material of physiology content in four academic years. The Electronic Journal of e-Learning, 10(4), 368–377. Retrieved from https://files.eric.ed.gov/fulltext/EJ986645.pdf

Jaimez-González, C., García-Mendoza, B., Luna-Ramírez, W., Nápoles-Duarte, M., & Vargas-Vargas, A. (2018). Learning Objects to Support the Teaching-Learning Process of a Web Fundamentals Undergraduate Course, American Journal of Educational Research, 6, 1573-1580. Retrieved from http://pubs.sciepub.com/education/6/11/17/index.html

Lowe, R. (2004). Animation and learning: value for money? Proceedings of the 21st ASCILITE Conference, (pp. 558–561). Retrieved from https://www.ascilite.org/conferences/perth04/procs/pdf/lowe-r.pdf

Luna-Ramírez, W., & Jaimez-González, C. (2014). Supporting structured programming courses through a set of learning objects. International Conference on Information Society (i-Society 2014). Retrieved from https://ieeexplore.ieee.org/document/7009024

Matthiasdottir, A. (2006). Usefulness of Learning Objects in Computer Science Learning. The Codewitz project. Proceedings of the Codewitz Open Conference Methods, Materials and Tools for Programming Education. Tampere, Finland. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.559.1156&rep=rep1&type=pdf

Narasimhamurthy, U., & Al Shawkani, K. (2009). Teaching of programming languages: an introduction to dynamic learning objects. International Workshop on Technology for Education (T4E), (pp. 114-115). Retrieved from https://ieeexplore.ieee.org/document/5314123

Osman, W., & Elmusharaf, M. (2014). Effectiveness of combining algorithm and program Animation: A case study with data structure course. Issues in Informing Science and Information Technology, 11, 155-168. Retrieved from https://www.semanticscholar.org/paper/Effectiveness-of-Combining-Algorithm-and-Program-A-Osman-Elmusharaf/026e947b16148e19e995435fe5e3abec48f0e1c1

Rotbain, Y., Marbach, G., & Stavy, R. (2008). Using a computer animation to teach high school molecular biology. Journal of Science Education and Technology, 17(1), 49-58.

Shindo, Y. (2000). Programming Education Based on Computer Graphics Animation. Proceedings International Workshop on Advanced Learning Technologies (pp. 164-165). Palmerston North: New Zealand. Retrieved from https://ieeexplore.ieee.org/document/890639

Smith, G., & Escott, E. (2004). Using animations to support teaching of general computing concepts. Sixth Australian Computing Education Conference (ACE2004). Retrieved from https://pdfs.semanticscholar.org/75e4/d3f46b6fd0a0cfc231a927d532842337a6a8.pdf

Standard for learning object metadata. (2002, July). IEEE Computer Society. Retrieved from https://standards.ieee.org/project/1484_12_1.html

Tapoli, P., & Mikropoulos, T. (2019). Digital Learning Objects for Teaching Computer Programming in Primary Students. In M. T. Mikropoulos, Technology and Innovation in Learning, Teaching and Education (Vol. 993). Springer. Retrieved from https://link.springer.com/chapter/10.1007/978-3-030-20954-4_19

Villalobos, J. C., & Jiménez, C. (2009). Developing programming skills by using interactive learning objects. ITiCSE '09 Proceedings of the 14th annual ACM SIGCSE conference on Innovation and technology in computer science, 41, pp. 151-155. New York, USA: ACM. Retrieved from https://www.researchgate.net/publication/220807957_Developing_programming_skills_by_using_interactive_learning_objects

Author Information

Abdullah Basuhail
King Abdulaziz University
Faculty of Computing and Information Technology
Jeddah, KSA
Contact e-mail: *abasuhail@kau.edu.sa*