

Developing A 3D Action-Adventure Game Called “Code - E” for Learning C++ Function Codes

Muhammad Kamil Zainol Abidin¹, Azniah Ismail^{2*}

¹*Department of Computing, FSKIK, Universiti Pendidikan Sultan Idris; d078449@siswa.upsi.edu.my*

²*Department of Computing, FSKIK, Universiti Pendidikan Sultan Idris; azniah@fskik.upsi.edu.my*

* *correspondence author*

To cite this article (APA): Abidin, M.K.Z. & Ismail, A. (2021). Developing a 3d action-adventure game called “Code - E” for learning C++ function codes. *Journal of ICT in Education*, 8(3), 13-26. <https://doi.org/10.37134/jictie.vol8.sp.1.2.2021>

To link to this article: <https://doi.org/10.37134/jictie.vol8.sp.1.2.2021>

Abstract

This paper describes a game prototype development project named “Code-E” for players learning and practicing their knowledge about coding while playing. Our research is based on design and development research (DDR) methodology. The main purpose of our project was actually to create a 3D action-adventure game with interesting gameplay for educational purposes that can be used by students (i.e., players) as part of their coding practice. Our chosen topic was C++ function codes. Rapid Application Development Model was used as the development method to allow us to focus on minimizing the planning stage and maximizing prototype development. First, the gameplay idea was designed. A quick prototype was developed to assist the potential users to understand the gameplay easily when we were asking for their feedback on requirements. To collect them from potential users, we used interviews. Very fortunately, our first gameplay idea was well accepted which required minimal improvement. Then, we designed and developed a full game prototype using Roblox. A questionnaire session was then conducted to get further feedback from our potential users. The 5-Likert scale questions were categorized into three categories: the game’s technical features, its gameplay concepts, and its usability as a learning tool. We used median values to interpret the Likert data. Our findings showed that the “Code-E” game prototype was well-functioned and well-accepted by our potential users. Some improvements were still required to enhance its ability in terms of difficulty level and motivation.

Keywords: game development, 3D action-adventure game, learning programming language, C++ function codes, prototype.

INTRODUCTION

Nowadays, in line with technological advancements, games have become part of many students' lives. Games are not only for entertainment purposes as they are now also widely accepted as learning and training tools. Using games for learning has been largely discussed among researchers such as Papastergiou (2009), Cahyana et al. (2017), Hieftje et al. (2017), Braghirolli et al. (2016), Teeds (2019), and Gao et al (2020). It has been shown to be effective, motivating, and enjoyable (Papastergiou, 2009; Braghirolli et al., 2016). According to Hieftje et al. (2017), educational video games have also shown effectiveness in promoting academic achievement.

Learning programming languages have posed several difficulties for students at all educational levels for so long (Malliarakis, Satratzemi, & Xinogalos, 2016). More than 30 percent of students failed in their computer science courses (Watson & Li, 2014, as cited in Tan Zalilah, 2019). Tan Zalilah (2019) had also mentioned that many researchers relate the problem with coding. Coding, including writing and reading function codes, might be a bit hard for many people, but practicing will always make any learning easier. According to Priyaadharshini et al. (2020), using games for learning programming can stimulate the curiosity to learn and understand the programming concepts in a better way. The learners can be encouraged to play mobile gaming app for programming courses which were designed to attend various tasks like debugging, drag and drop activity, find the odd object out, game quiz, and multiple-choice questions. There are several games like that available for learning coding, however, they might be less suitable for adult learners. Children's games for learning programming such as Code Adventure and Code Combat are more complex and interesting as users can learn programming by creating codes to control spawn movement in the game given. Code Combat is more challenging as users will be asked to code with real programming languages and users do not only create codes to control spawns but also methods and event handlers in the game. Users play the character as the game master that decides what will happen in the game given, not role-playing the main character as most action-adventure games do.

Watch Dogs and *Do I Have a Right?* were among a limited number of games in the action-adventure game genre that were created for learning purposes. However, potential gameplay ideas under the 3D action-adventure genre that are created especially for learning coding are still lacking. This research area is ready to be explored further. In our research, we were very keen to create one 3D action-adventure game with interesting gameplay that is possible to help students learn and practice their C++ function codes. To have a 3D action-adventure game as an educational tool to practice coding would be a great addition to current educational games.

LITERATURE REVIEW

Games for Learning

According to Teed (2019), using games for learning has gained considerable traction since Gee (2003) described the impact of gameplay on cognitive development. Some important advantages from using games for learning as lined by Teed include the followings: (i) a game usually utilizes an interesting narrative, (ii) it has competitive exercises to motivate students learning according to a specific design of learning objectives, (iii) can engage students with the material, (iv) make significant improvement over those participating in learning with other educational software due to game's feature of inductive reasoning and frequent interactions with content. An extensive overview of the games for learning examples can be found in the review by Gao et al. (2020).

Children's games for learning programming such as Code Adventure is one of the interesting types of interactive games available today as users can learn programming by creating codes to control spawn movement in the game given. For example, users create codes that allow a monkey character to move to get to bananas (see Figure 1). An example of the game can be found here: <https://www.codemonkey.com/hour-of-code/coding-adventure/>.

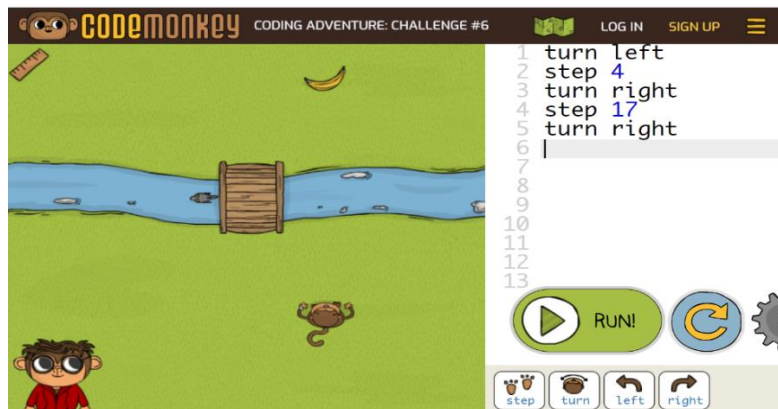


Figure 1: Code adventure allows users to write code to control the spawn (monkey) to get to the banana

Code Combat is another type of creating code game that is more challenging with real programming languages and users do not only create codes to control spawns but also methods and event handlers in the game. However, the game does not involve the role-playing game element as most action-adventure games do. In Code Combat, users play the character as the game master that decides what will happen in the game given (see Figure 2). An example of Code Combat game can be found here <https://codecombat.com/play/hoc-2018>.



Figure 2: Code Combat interface allows users to write code to control the spawn, the methods, and the event handlers to create interesting gameplay. Users become the game master in this game


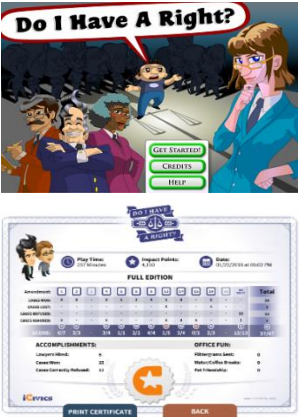
Action-adventure Games

The 3D action-adventure genre was the result of (i) the 'happy marriage' of action games and adventure games, marked by Super Mario Bros. in 1985, and (ii) the 3D revolution in the mid-1990s (Klevjer, 2011). Examples of games in this genre were Doom, Tomb Raider, Quake, Super Mario 64, and Silent Hill. GTAIII was an example of genre expansion that notably hybridizing with role-playing games in 2001. According to Arjoranta (2011), digital games, especially with role-playing game elements, effectively convey visual cues with unlimited imaginative virtual spaces, and the only limitation is lacking the possibility to convey emotions through facial expressions since the player communicates through his or her avatar. Arjoranta also emphasized important terms to be aware of by game developers which include game world (that supported by scenes in the game), participants (game master and players), characters, interaction, and narrative.

Action-adventure Game Play for Learning Purpose Examples

To get some ideas about gameplays, we took a look at two different games: Watch Dogs and Do I Have a Right? Both are from the action-adventure game genre that has a role-playing game element, and both were used for learning purposes, but in different fields. We also explored their features and identify their strengths and weaknesses.

Table 1: Differences between two action-adventure games: characteristics, strengths, and weaknesses

Game Name	Characteristics	Strength	Weakness
<p>Watch Dogs</p> 	<p>How to use it?</p> <ol style="list-style-type: none"> 1. A 3D video game. 2. For learning networking. 3. Contains puzzle/ riddle to be solved. 4. Gameplay: player become the third person who can control a hacker named Aiden Pearce that can control all system network in the big city. 	<ol style="list-style-type: none"> 1. The graphics used are great. 2. User friendly. 3. Fun to play while learning. 	<ol style="list-style-type: none"> 1. Not free. 2. No coding learns in this game just the basic concept of networking.
<p>Do I Have A Right?</p> 	<ol style="list-style-type: none"> 1. An online 2D game. Also available as a mobile app. 2. For learning civil rights. 3. Classic simulation game type. 4. Gameplay: the player controls an avatar of a lawyer, who specializes in constitutional law, managing cases in a law. 	<ol style="list-style-type: none"> 1. The gameplay is easy to play. 2. Free to play. 3. The interesting avatar that can be changed/ chosen accordingly. 	<ol style="list-style-type: none"> 1. The content (law) used in the game is only applicable for Americans only. 2. Lots of text to read while playing.

Watch Dogs

The gameplay allows the player controls a hacker named Aiden Pearce, who uses his smartphone to control trains and traffic lights, infiltrate security systems, jam cell phones, access pedestrians' private information, and empty their bank accounts. This game shows the player how the network is connected by showing the player to hack. For example, one of the gameplays involves a mission to hack a CCTV. By doing this mission, the player will learn how the networking of the CCTV works (for security reasons, the player will only be given basic knowledge like how the CCTV works and not actually hacking it using real-life hacking skills). The difficulty of each game ranges from simple to hard with riddles to solve makes this Watch Dogs game interesting. The strength of this game is its

gameplay and graphic. It is also user-friendly. However, this game is not free to play. Nonetheless, it is worth buying.

Do I Have Rights?

A game developed by iCivic.org. This game is a classic simulation game that gives users some experience to practice constitutional laws. The player controls an avatar who manages a law firm. This game is easy and free to play. The only drawback is that this game only focuses on Americans' civil rights. This game is accessible at <https://www.icivics.org/games/do-i-have-right>.

Both games offer interesting and unique gameplay, respectively. Their features were both different from each other. More interestingly, both games give controls to the players and make learning more enjoyable with their amazing graphics.

C++ Function Code

Programming languages are very important in computer science or related field. C++ is one of the popular programming languages that is being taught in schools, colleges, and universities, other than Java and Python. In C++, a function represents a group of statements that can be called from some point in a C++ program. It allows us to organize our programs in a well-structured segment of codes. Each function can be designed to perform certain actions or individual tasks.

Functions are part of reusable codes. Therefore, a function has to be named uniquely so that the right one can be called many times when needed. Functions must be declared before they can be used in the C++ program. We can also pass data (or parameters) into a function to be processed and the output can be sent back to the main() function or any other function that calls it. In general, a C++ function code consist of two parts: (i) declaration: the function's name, return type, and parameters (if any), and (ii) definition: the body of the function (code to be executed). However, the main() function is a pre-defined function name that must be used to execute code in C++. Our self-defined functions are usually will be called from this main() function. Another way to make the code better organized and easier to be read is by separating the declaration and definition. Function codes require some practice before one can get familiar with them and be able to write and read them properly and correctly.

METHODOLOGY

This study was based on design and development research (DDR) methodology that involved designing and developing a game as an educational tool for practicing C++ function codes. The game prototype was built based on the Rapid Application Development Model to minimize the planning stage and maximize prototype development.

Research design

In general, our research design involved the following: one gameplay was designed and a quick prototype was built before getting feedback from potential users to see whether the gameplay idea was well accepted. Data were collected using interviews. Based on the findings, we developed a full prototype. Questionnaires were then administered to get feedback on the game functions and prototype acceptance.

Participants

Students who were pursuing information technology education bachelor program at Faculty of Arts, Computing and Creative Industry, and who were also familiar with the action-adventure type of games. Only five students had given their full consent to participate in our requirements planning and evaluation phase.

Rapid Application Development (RAD) Model.

a) Requirement Planning

A basic breakdown of this stage includes researching the current problem, defining the requirements for the project, and finalizing the requirements with each stakeholder's approval. As we employed the RAD model, we tried to minimize planning duration by creating a quick prototype to accelerate the decision-making process at this stage. By doing that it was much easier for us to deliver our gameplay idea. We collected sufficient requirements for the game such as improvement on scenes and characters based on the prototype as well as other current and potential issues that need to be addressed.

b) Game Design

Design artifacts including gameplay concept and layout, storyboard, character design, and scene design were developed before going through the actual development. During this phase, we double-checked our work with other experts in the field to ensure the needs were met at every step in the design process. We also considered the multiplayer feature as requested.

- **Gameplay Concept Design:** an online 3D action-adventure genre where the player is trapped and needs to find a button to open the gate.
 - The button is hidden somewhere in a room where the room is locked with a passcode. To find the passcode, the player has to search the entire place and solve several C++ functions codes on the wall.
 - The player is free to roam the place searching for an exit way but there are hints available through riddles. If the player is not familiar with the C++ function codes, the player can find related hints on other walls too.

- However, to make the game more challenging, the place is loaded with zombies. The player has to escape these zombies while finding the way out.
- Content Design: Every C++ function code provided in the games shall provide an answer that has to be collected by the player to reveal the passcode that can open the locked gate. Therefore, each player must know how to read the C++ function codes and be able to solve the answer. Some examples of C++ function codes used in the game are as follows:

```
//function example#include <iostream>using namespace
std;
int subtraction (int a, int b)
{
    int r;
    r=a-b;
    return r;
}
int main ()
{
    int z;
    z = subtraction (5,1);
    cout << "The result is " << z;
}

//function example#include <iostream>using
namespace std;
int subtraction (int a, int b)
{
    int r;
    r=a-b;
    return r;
}
int main ()
{
    // The third result is for the third code.
    int x=5, y=3, z;
    z = subtraction (7,2);
    cout << "The first result is " << z << "\n";
    cout << "The second result is " << subtraction
(7,2) << "\n";
    cout << "The third result is " << subtraction (x,y) <<
"\n";
    z= 4 + subtraction (x,y);
    cout << "The fourth result is " << z << "\n";
}
```

- Character and Scene Design: There were four main characters including the player that can be found in the game (see Figure 3). One was the zombie character that each player has to avoid. There were many of them and they will chase the player around all the time in the game. If a player got defeated by any of these zombies, it means the game has ended. The other character in the game was Mr. Dol who was responsible to give a task to the player and awaits at the end when the player won. Last but not least, the knight who guards the entrance to the locked door can give a hint on the passcode correct number arrangement. Figure 4 shows some examples of scenes that we developed in the game.



Figure 3: Characters specifically design for the Code-E game

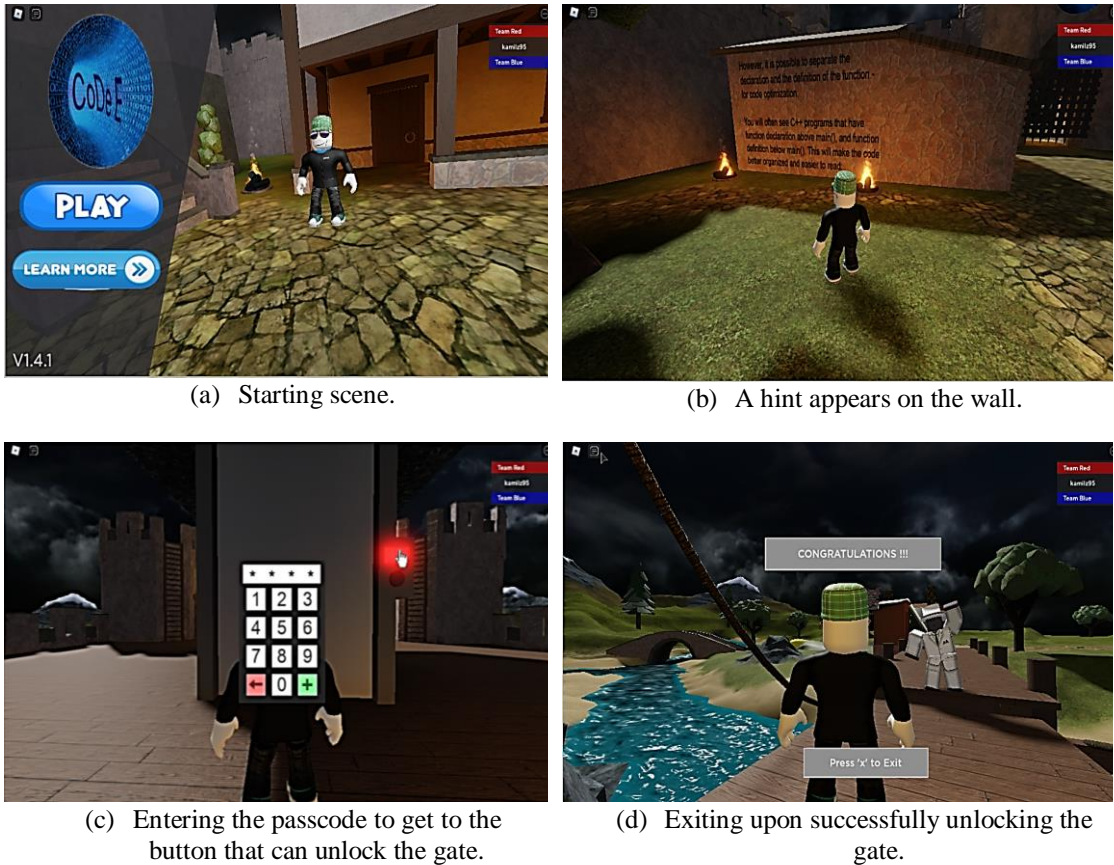


Figure 4: Some examples of scene design in the Code-E game.

c) *Development*

The quick prototypes from previous phases were converted into a working prototype using Roblox. Coding development still running even there were changes in any of the designs from time to time due to the iterative design phase. We spent more time here to develop the full prototype. The phase breaks down into several smaller steps including preparation for rapid construction, program and application development and configuration, and coding.

d) *Testing and Evaluation*

To ensure that the development and design of the existing game were implemented according to the requirements and planning set. All five participants had the chance to play the Code-E game before a questionnaire was administered to them right away.

Table 2: The 5-Likert scale was used in the questionnaire

Agree	Description
1	Strongly disagree
2	Disagree
3	Not sure
4	Agree
5	Strongly agree

Table 3: Question items and codes were used based on category

Category	Question
[A] INTERFACE, CONTROL, AND AUDIO FEATURES	A1 - The quality of the user interface is great.
	A2 - The control is smooth while playing the game
	A3 - The quality of sound is great.
	A4 - The interaction between the Player and other characters is great.
	A5 - The interaction with the surrounding in the game is great.
[B] GAMEPLAY CONCEPT	B1- The gameplay is unique.
	B2 - The game difficulty level is hard.
	B3 - The game is interesting.
[C] USABILITY	C1 - The game helps me to learn C++ function code.
	C2 - The game helps me to practice C++ function code.
	C3 - The game motivates me to learn C++ function code.
	C4 - The game is beneficial for people who learning C++ programming.
	C5 - I would recommend the game to other people.

RESULTS AND DISCUSSION

Table 4 shows the demographic analysis results on the participants' gender and background. We have mentioned previously that only five students willing to participate in the requirements planning and testing the game. It is considered sufficient to get brief feedback at the moment before another evaluation can be conducted later. Three of the participants (60%), consist of two males (40%) and one female (20%) have learned programming through a course named Principle of Programming, whereas two other participants (40%), all males, have yet taken the course.

Table 4: Demographic analysis results

Gender	Num.	Percentage	Have taken the MTS3063 Principle of Programming course?	
			Yes	No
Male	4	80%	2 (40%)	2 (40%)
Female	1	20%	1 (20%)	0 (0%)
		100%	3 (60%)	2 (40%)

In terms of technical features in the game such as the control and the audio, 60% to 80% of the participants have agreed that all the features in Code-E game under this category are great and smooth (see Figure 5). However, the results also show that there might be room for improvement. Further investigations using open-ended questions might allow us to get more feedback for this category.

For more constructive ways to approach the Likert data, we used a median rather than a mean (see Table 5). We arranged all the responses in sequence. As there were only five responses, the third response would fall at the numerical midpoint. For this category A, the median values for all items were 4 and 5, which means most participants agreed. Thus, we concluded that all technical features in the game were considered well-accepted by the participants.

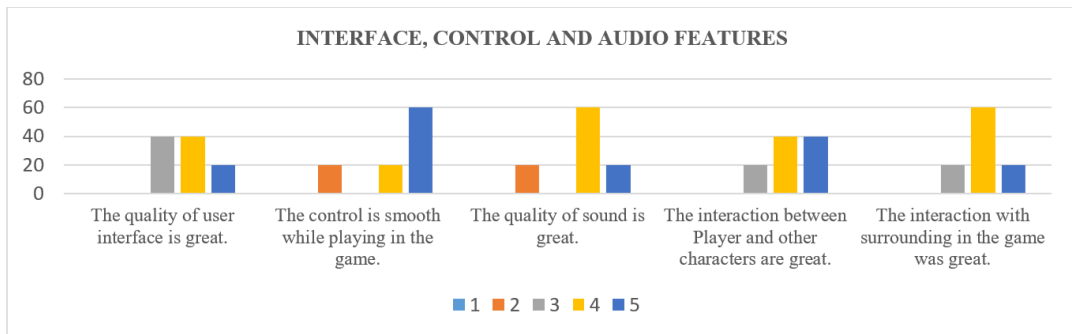


Figure 5: Interface, control, and audio features category results

Table 5: Median analysis results for interface, control, and audio features category

Category	Features	Median value
[A] Interface, control, and audio features	The quality of the user interface is great.	4
	The control is smooth while playing the game.	5
	The quality of sound is great.	4
	The interaction between the Player and other characters is great.	4
	The interaction with the surrounding in the game was great.	4

In terms of the gameplay concept of Code-E, 60% of the participants have agreed that the gameplay is unique (see Figure 6). Only 20% of the participants have strongly agreed that the game is hard. 80% have agreed that the game is interesting. The median value for the game difficulty is only 3

which means most respondents disagreed that the game was hard (see Table 6). However, most participants agreed that the game is unique (median = 4) and interesting (median = 5).

It might either means that less difficulty level has not affected the fun they had when they were playing the game, or maybe that less difficulty level makes them more attached to the game. The game's uniqueness might also make them interested. Further investigations would be preferable for this category. We conclude the gameplay works well but it is better to provide a different range of difficulty levels in the game

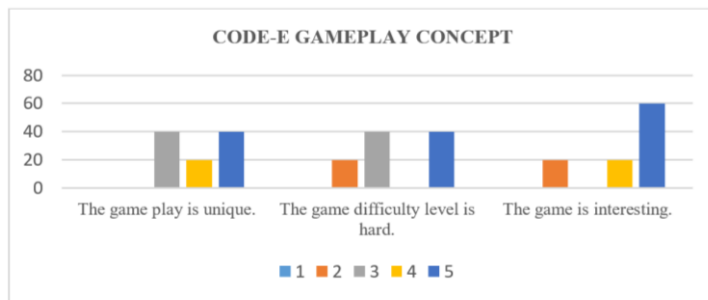


Figure 6: Gameplay concept category results

Table 6: Mean analysis results for gameplay concepts category

Category	Features	Median value
[B] Gameplay concepts	The gameplay is unique.	4
	The game difficulty level is hard.	3
	The game is interesting.	3

Under the usability category, 60% to 80% of the participants have agreed that the Code-E game is beneficial for people learning C++ programming, and the game can help them to learn and practice C++ function codes (see Figure 7). 80% have agreed to recommend the game to other people. However, only 40% of the participants have agreed that the game can motivate them to learn code functions whereas the rests (60%) were unsure. As we looked at the median values in Table 7, all items were more than 3 except that most participants disagreed (median = 3) that the game can motivate them to learn. Perhaps, creating a more interesting scoring system, and providing a reward or special weapons would increase their motivations. Notably, we concluded that the Code-E game has successfully passed as an educational tool for learning C++ function codes.

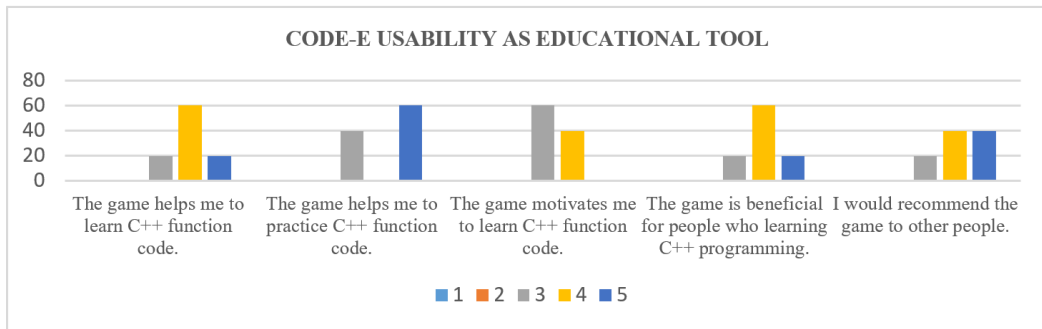


Figure 7: Usability category results

Table 7: Mean analysis results for usability as educational tool category

Category	Features	Median value
[C] Usability as an educational tool	The game helps me to learn C++ function code.	4
	The game helps me to practice C++ function code.	5
	The game motivates me to learn C++ function code.	3
	The game is beneficial for people who learning C++ programming.	4
	I would recommend the game to other people.	4

CONCLUSION

We have described our Code-E game project development in this paper. We successfully developed a full 3D action-adventure game prototype using the RAD model by involving five potential users to participate in our requirements planning and evaluation phases. Based on their feedback, we conclude that our Code-E game is suitable for learning and practicing C++ function codes although some improvements were still required to enhance its ability in terms of difficulty level and motivation. Hence, our aim to create a 3D action-adventure game prototype with interesting gameplay for learning purposes has been successfully achieved. We are looking forward to another testing session with more participants to get more insights into the potential of our Code-E game. We hope that our efforts would encourage more action-adventure gameplay ideas to be explored and developed, especially for learning programming for adult learners such as college or university students.

REFERENCES

- Arjoranta, J. (2011). Defining Role-Playing Games as Language Games.” *The International Journal of Role-Playing*, 1(2), 3-17. Retrieved from <https://jyx.jyu.fi/handle/123456789/37331>
- Braghirolli, L. F., Ribeiro, J. L. D., Weise, A. D., & Pizzolato, M. (2016). Benefits of educational games as an introductory activity in industrial engineering education. *Computers in Human Behavior*, 58, 315–324. <https://doi.org/10.1016/j.chb.2015.12.063>.

- Cahyana, U., Paristiowati, M., Savitri, D. A., & Hasyrin, S. N. (2017). Developing and application of mobile game based learning (M-GBL) for high school students' performance in Chemistry. *Eurasia Journal of Mathematics, Science and Technology Education*, 13(10), 7037-7047. <https://doi.org/10.12973/ejmste/78728>
- Gao, F., Li, L. & Sun, Y. A systematic review of mobile game-based learning in STEM education. *Education Tech Research and Development*, 68, 1791–1827 (2020). <https://doi.org/10.1007/s11423-020-09787-0>
- Gee, J. P. (2003). What video games have to teach us about learning and literacy? *Computers in Entertainment (CIE)*, 1(1), 20-20. <https://doi.org/10.1145/950566.950595>
- Hieftje, K., Pendergrass, T., Kyriakides, T., Gilliam, W., & Fiellin, L. (2017). An evaluation of an educational video game on Mathematics achievement in first-grade students. *Technologies*, 5(2). <https://doi.org/10.3390/technologies5020030>
- Klevjer, R. (2011). *Telepresence, cinema, role-playing. The structure of player identity in 3D action-adventure games*. The Philosophy of Computer Games 2011. Retrieved from <https://www.academia.edu/download/30883417/runeklevjerathenstalk.pdf>
- Malliarakis, C., Satratzemi, M., & Xinogalos, S. (2016). CMX: The effects of an educational MMORPG on learning and teaching computer programming. *IEEE Transactions on Learning Technologies*, 10(2), 219-235. <https://doi.org/10.1109/TLT.2016.2556666>
- Papastergiou, M. (2009). Exploring the potential of computer and video games for health and physical education: A literature review. *Computers & Education*, 53(3), 603-622. <https://doi.org/10.1016/j.compedu.2009.04.001>
- Priyaadharshini, M., Natha Mayil N., Dakshina, R., Sandhya S., & Bettina Shirley R. (2020). Learning analytics: game-based learning for programming course in higher education, *Procedia Computer Science*, 172, 468-472. <https://doi.org/10.1016/j.procs.2020.05.143>.
- Tan Zalilah, M. A. (2019). Penjelasan Identiti Asas Pengaturcaraan dan Perubahan Konseptual Melalui Adegan Permainan. *Journal of ICT in Education*, 5, 48-57. <https://doi.org/10.37134/jictie.vol5.6.2018>
- Teed, R. (2019). *Game-based learning*. Science Education Resource Center Carleton College. Retrieved from <https://serc.carleton.edu/introgeo/games/index.html>
- Watson, C. & Li, F. W. B. (2014). Failure rates in introductory programming revisited. In *Conference On Innovation Technology in Computer Science Education (ITiCSE '14)*. New York: Association for Computing Machinery (ACM), pp. 39-44. <https://dro.dur.ac.uk/19223/1/19223.pdf>