JADCM
Journal of Art, Design and Creative Media
Universiti Pendidikan Sultan Idris
Sultan Idris Education University

Development and Evaluation of RoboCodin: Insights into Learners' Perceptions of Programming Learning

Muhammad Abdul Malik Saedon*, Muhammad Shameer Sawar, Tan Wei Jie

Department of Creative Multimedia, Faculty of Art, Sustainability and Creative Industry, Universiti Pendidikan Sultan Idris, 35900, Tanjong Malim, Perak, Malaysia

***Corresponding author email:** abdulmalik@fskik.upsi.edu.my

ARTICLE HISTORY
Received: 10th August 2025
Revised: 25th February 2026
Accepted: 3rd March 2026
Published: 18th March 2026

KEYWORDS
Serious game
Game-based learning
Turn-based game
Programming
JavaScript

ABSTRACT - The aim of this study is to propose and develop a learning tool that can enhance programming learning. The researchers' experience suggests the need for various approaches to programming learning in addition to the conventional approach. Accordingly, the first research objective is formulated, which is to propose a suitable game for programming learning that applies game-based learning approaches. It is proposed based on issues and challenges occurring among tertiary students. As found by the researchers, conventional learning is not appropriate for the learners to understand syntax, fundamental concepts, and debugging. They need a different approach to face the challenges. To address these challenges, the researchers then formulated the second research objective, which is to develop a serious game named RoboCodin that applies turn-based mechanics and also embeds programming learning content. The game was developed through a comprehensive process that included stages from concept design to deployment. Once the game's development was complete, the researchers conducted alpha and beta testing to fix any design flaws. Then, the third research objective is formulated, which is to evaluate the RoboCodin turn-based serious game. To evaluate the game, a playtest is conducted among tertiary students using purposive sampling. During the playtesting, they had to play the game thoroughly, and then they were given a questionnaire to complete. Thereafter, data from the survey questionnaire are distributed in SPSS, and descriptive statistics are conducted. The results indicated that the game has a favorable impact on its efficiency, helpfulness, and learnability, with high mean values ($\bar{x} > 4.1$). The learners consider the RoboCodin turn-based serious game to have an impact on learning programming. Playing RoboCodin has improved the learners' cognitive skills in understanding the programming concepts in a fun way.

INTRODUCTION

Preparing tertiary students as well as learners in programming in this digital age is important because many sectors of employment are required to do programming in many workforces such as website design, purchasing systems, video games, and digital educational tools. In education, preparing learners with programming skills is increasingly important due to the demand of the industry that wants to hire employees who have expertise in programming. To meet the growing demands, the education system often adapts its curricula with digital learning aids, as many governments support the Industry 4.0 transformation. Additionally, the governments lay out plans to introduce students to more advanced technologies, promote creativity, and put 21st-century learning initiatives into action. This trend is in line with a significant increase in the inclusion of programming in the core curriculum (Bachiller-Burgos et al., 2020). This effort prepares the younger generation for an increasingly digital future. To prepare them, education should be in line with the industry development. One of the preparations to meet the growing industry demands is mastering programming in tertiary-level education. Otherwise,

https://doi.org/10.37134/jadcm.vol1.1.5.2026

unemployment issues may arise from an imbalance between the human supply and industry demand due to a mismatch in skill requirements (Salmiah, 2023).

Issues and Challenges

Learning programming is challenging in several aspects including learners, educators, syllabus, and teaching approaches (Ahmad & Ghazali, 2020), and it remains a significant challenge among young learners at the beginning stage (Abu Bakar et al., 2025). As beginners, they often find errors in programming compared to intermediate and advanced learners who have been learning it for a long time. The intermediate and advanced learners can rectify the error because they have sufficient knowledge and a comprehensive understanding of writing codes, while beginners often approach programming line by line instead of using meaningful structured programs (Abdul et al., 2018). It is not an easy task because learning programming takes years for the beginners to become expert programmers (Kadar et al., 2021) and also requires them to master and know multi-layered skills. Besides that, teaching and learning programming can be difficult due to pedagogical obstacles, as many learners struggle to comprehend programming languages. Instructors are facing challenges in making learners understand the programming concept in order to solve the real problem. As stressed by Kadar, Wahab, Othman, Shamsuddin, and Mahlan (2021), lack of logical, creative, and critical thinking with limited surface knowledge drives weaknesses in the implementation of problem-based learning programming. Additionally, ineffective textbooks also contribute to challenges learners face in developing programming competency. Although there are many learning tools, the difficulties of teaching and learning programming remain unsolved (Ahmad & Ghazali, 2020).

The researchers of this study found that students' background knowledge and teaching approaches are several factors that cause difficulties in learning programming. The programming questions assigned by instructors appear to exceed the students' cognitive abilities because they lack comprehension, problem-solving, and mathematical skills. This is because mastering programming requires comprehending the structure of the programming language, interpreting requirements into an algorithm, writing code in the correct syntax, debugging, applying correct logic, and using the program development environment (Abdul et al., 2018). According to Ahmad and Ghazali (2020), most of the learners had negative perceptions towards learning programming, concluding that programming is one of the killer subjects. This negative perception becomes even worse when their seniors claim the same thought. This condition causes juniors as beginners to lose interest and have less motivation towards the programming subject even before learning it. Furthermore, the scenario made them less diligent in the class and eventually may lead to low grades in the subject. As a result, programming subjects often have high dropout rates because beginners suffer from difficulties and deficits (Massoudi, 2019), despite the many teaching and learning methods and tools that are used (Medeiros et al., 2019). Muhamad Yusof et al. (2021) assert that the difficulty of understanding programming language led to a decrease in student motivation to learn it.

Solution Methods

To face the issues and challenges in learning programming, sophisticated changes must be implemented regarding study methods, attitudes, and teaching approaches. As addressed by Xinogalos (2016), hands-on practice on learning programming and contemporary innovative teaching and learning as well as instructional methods must be applied to increase learners' interest in computer education. The instructors need to apply appropriate subject content and pedagogical approaches that can increase the level of learners' interest and motivation, not just traditional approaches. In response to these unsolved challenges, game-based learning can be applied to make learning programming more enjoyable among the learners. Although there are many digital games on the market, few have integrated programming languages like JavaScript. In addition, there is a dearth of research evaluating the precise ways in which gamification can support teaching, learning, and assessment of specific programming concepts (Rojas-Lopez, 2019). By combining engaging gameplay and learning content, the game-based learning approach can help players enhance their interest and motivation in programming and, at the same time, sharpen their skills. It is due to the approach which provides an innovative and interactive way to learn programming by combining education and entertainment elements.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Research Objectives

According to Mehmood et al. (2020), many studies have helped students acquire programming skills and overcome learning difficulties using specific models, notional machines, or games. To achieve one of the learning difficulties (games), this study is conducted to propose and develop a serious game as a programming learning tool. Accordingly, three objectives are addressed in this study as follows:

1. To propose a game that is suitable for learning programming.
2. To develop RoboCodin, a turn-based serious game.
3. To evaluate the learners' perceptions of the RoboCodin turn-based serious game.

Based on the research objectives, the researchers of the study, as well as game developers, focus on designing, developing, implementing, testing, and evaluating a turn-based serious game as a programming learning tool. The researchers make use of the game mechanics and game-based learning to draw learners' attention towards the learning process.

LITERATURE REVIEW

Game-Based Learning

Educational strategies like game-based learning that apply game-like elements are used to improve the learnability of programming. Game-based learning combines interesting gameplay with educational and instructional content to inspire learners, foster problem-solving skills, and encourage involvement. In contrast to conventional teaching techniques, game-based learning as in this study makes use of game mechanics and interactive features to enhance the learning process and make it more fun and engaging. As stated by Monteiro et al. (2021), engaging younger children, fostering spatial awareness, and managing the children's anxiousness are among the tactics to be aware of in teaching programming. To counter these tactics, this study proposed and developed a turn-based serious game as a game-based learning tool to promote engaging learning. Learning programming in fun and intriguing ways by applying a game-based learning approach with gamification techniques can increase learners' confidence and motivation (Ahmad & Ghazali, 2020). As stressed by Rojas-Lopez et al. (2019), the application of games can improve students' engagements in learning programming.

Turn-Based Game

Turn-based games have a long history of evolution from ancient board games and tabletop games to sophisticated digital games. One of the examples during the evolution is text-based adventure and strategy games like Colossal Cave Adventure (1976), Zork (1977), Rogue (1980), and Ultima (1981). Nowadays, turn-based games have become one of the unique game genres because they employ cunning strategy and careful planning with the demand of role-playing games and tactical games (Agate, 2025). It is a type of strategy game where players as combatants take turns performing actions (Makela & Schmidt, 2020), unlike real-time gameplay that plays continuously and simultaneously (Agate, 2025; Fiveable, 2025; Mahtgician Game, 2025). The playing field is typically defined by a grid of hexagons or rectangles called grid-based movement, and the objective is usually to destroy all enemy troops or buildings. The player has a designated time or opportunity to perform certain tasks, such as moving characters, attacking opponents, or completing puzzles, before passing control to the next player or the game's artificial intelligence (AI).

Players who like to prepare strategies, analyse possibilities, and have time to think through the consequences of their actions will love this type of game. It is because the turn-based game enables players to think critically and consider their options. These playing patterns are commonly found in several game genres like strategy games (Civilization, XCOM, Fire Emblem), role-playing games (Final Fantasy, Pokemon), and board games (chess, Monopoly). Turn-based games also incorporate gamification mechanics and principles, including score, life scale, timer, cutscene, and reward systems. The application of gamification is a beneficial active strategy for learning programming (Garcia & Lemos, 2023). These features indicate the turn-based game has high difficulty, a complex system, and wise mechanics (Agate, 2025) compared to games that are not turn-based. To compare these games, Table 1 shows a comparison of existing game-based programming learning tools and the proposed turn-based serious game.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Table 1. Comparison of Game-Based Programming Learning Tools

No	Game	Genre	Description	Differences	Turn-based Mechanic
1	CodinGame	2D puzzle	An online game to strengthen programmers' skills through interactive puzzles and algorithmic challenges, and it is also a multiplayer game.	• Solve coding that control visual games like racing and battles. • Not provide a full-stack training.	No
2	Lightbot	2D puzzle	An introductory coding game for kids and beginners that introduces several principles of programming.	• Guide a robot to light up tiles using commands. • The game gets progressively harder.	No
3	Human Resource Machine	3D puzzle	A puzzle game where the boss of the player gives a job, then solve problem using assembly-like code. Automate it by programming as little office worker.	• Office-themed environment. • Problem-solving using assembly-like code.	No
4	CodeCombat	2D puzzle-based RPG	A video game designed to teach programming languages through an interactive approach. The player writes real code to move characters and defeat enemies.	• Advance to higher levels and more complex challenges. • Type the actual code to direct the hero.	No
5	Code World	2D puzzle	A platform game where a player controls the robot, walks around the world, and sends messages to devices. Every device reacts to those messages according to its code.	• It does not help user in memorizing JavaScript syntax. • Cannot be considered as an adventure game.	No
6	Proposed turn-based game (RoboCodin)	3D action-adventure	A turn-based game to get started to learn programming. The player controls a robot in a fictitious world where he needs to bring back all the historical rock.	• Provide tutorials in memorizing JavaScript syntax before combat. • A gamified approach with robotic features.	Yes

Robocodin as a Turn-Based Serious Game

RoboCodin is proposed as an educational game that serves as a learning tool for programming. It is not only a learning tool, the RoboCodin turn-based serious game focuses on character and surroundings design which aims to make it more engaging while playing it. Playing the game requires a player as a combatant to act in turns like one move or decision at a time, while the other combatant (enemy AI) is unable to act outside of his turn (Makela & Schmidt, 2020). This approach will help the game appeal to a wider audience and keep them interested. RoboCodin is an example of a 3D turn-based serious game that offers an immersive experience, and its graphics are stunning with detailed character designs and immersive environments. Its isometric camera view with a 30-degree perspective gives players a unique view of the action.

The storyline of the RoboCodin turn-based serious game revolves around a young human child named Xorex who accidentally finds himself transported to a fictitious world dominated by the JavaScript programming language. Because Xorex is only familiar with other programming languages, he must

https://doi.org/10.37134/jadcm.vol1.5.4.2026

learn JavaScript in order to manipulate robots, defeat enemies, and collect a series of JavaScript history stones. In the gameplay, Xorex needs to take turns to make his move to attack the enemy before passing control to the enemy AI. In this way, he is forced to think strategically and learn how to apply programming knowledge by answering questions in a limited time. As the player progresses through the game, he will be encouraged to grow quicker in solving coding issues.

The application of turn-based mechanics means that a player will learn to write JavaScript code to control the robot to defeat enemy AI. It requires problem-solving skills and fundamental concept understanding in terms of variables, functions, and conditional statements. Each level will pose new challenges, requiring the player to use programming knowledge to progress. The game will introduce different JavaScript concepts and techniques with hints and tutorials to help them overcome obstacles. It is in line with the gamified approach to learning where the player will do a self-study, immerse themselves in an interactive game design, achieve scores, and progress to the final combat. Figure 1 illustrates a flow in applying the gamified model throughout the RoboCodin turn-based serious game.

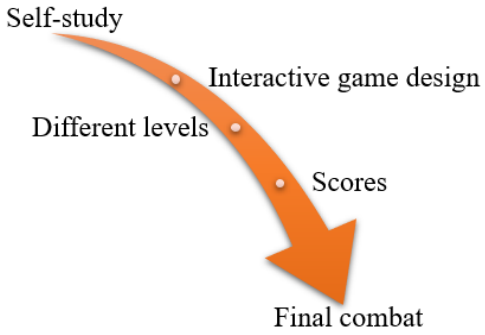

Figure 1. Gamified Approach of Learning in the RoboCodin Turn-based Serious Game

Based on Figure 1, the player learns programming (self-study) prior to progressing to different interactive game levels known as battlefields. To progress, the player needs to read tutorials and then enter the battlefield to answer programming questions in JavaScript. Next, he will achieve scores and play the RoboCodin turn-based serious game until the final combat.

JavaScript

JavaScript is widely used in web development and interactive front-end design (Patil et al., 2025). It is not limited to web development but is also used in non-browser applications across the entire technology stack. Now, JavaScript is a powerful language that has its unique characteristics and challenges that developers should be aware of when writing applications. It has grown significantly to become the backbone of modern web applications, supporting everything from basic user interfaces to complex server-side systems (Kappagantula, 2024). Hence, learning this programming language is becoming increasingly important for future job demand as students frequently prioritize programming languages that improve employability and align with current market trends. The popularity of online education, for instance, could increase through the use of JavaScript. Research indicates that web-based applications like Facebook, Netflix, Uber, and LinkedIn utilize JavaScript due to its versatility and stability.

With rapid demand for skilled developers in programming as well as computer science, choosing the right programming language can impact learning outcomes and ultimately future job opportunities. Despite the availability of more than 200 programming languages, the real-world industry demands only a select few (Logan, 2025). The demand is generally based on two reasons, which are efficiency and functionality (Rachmawati et al., 2024). Based on the researcher's experience, some of the criteria for choosing a programming language are based on ease of learning basic concepts, demand in the digital creative industry, and the availability of code resources and references. Furthermore, the selection is also based on simple syntax, the availability of resources, community support, and practical uses (Patil et al., 2025). Figure 2 shows the rank of the programming languages among developers.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

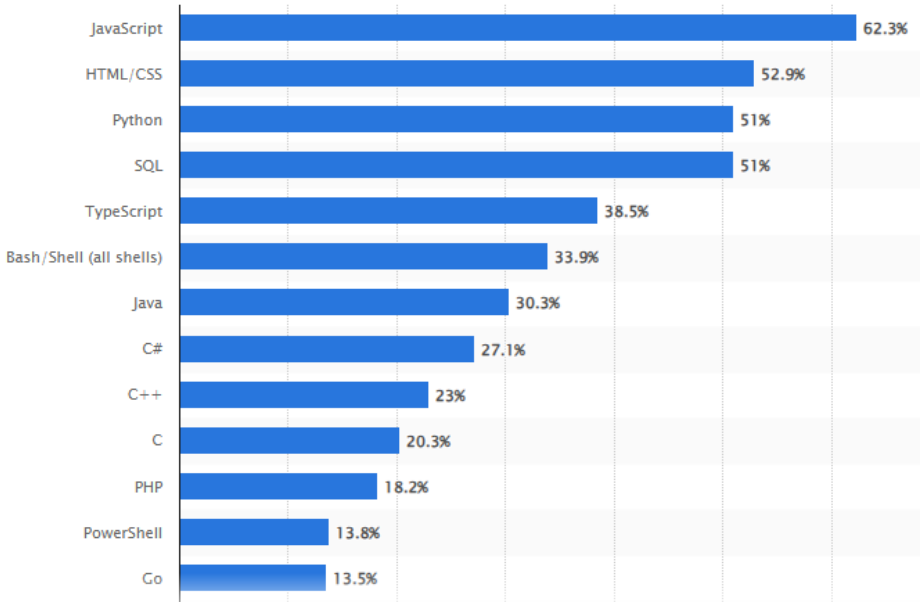

Figure 2. The Most Popular Programming Language among Developer. Source: Adapted from Vailshery (2025)

As shown in Figure 2, JavaScript is indicated as the most popular programming language among software and web developers, with 62.3 percent of them using it. HTML/CSS comes in second with 52.9 percent, and Python follows with 51 percent. The rest, like SQL and TypeScript, rounded out the top five most popular programming languages around the world (Vailshery, 2025). Accordingly, beginners benefit from programming languages that provide comprehensive documentation, robust community support, and useful real-world applications (Patil et al., 2025). Despite the many programming languages available, there is still an unclear consensus on which one offers the most accessible learning path for beginners. Only a few of them are in demanded in the labour market, with Python and JavaScript being popular programming languages due to their uncomplicated syntax, ease of use, versatility, and flexible applications (Patil et al., 2025) (Islam et al., 2024).

DEVELOPMENT OF THE ROBOCODIN TURN-BASED SERIOUS GAME

Development of the RoboCodin turn-based serious game is by using the Game Development Life Cycle (GDLC). The cycle of game development consists of five stages which are (i) concept design, (ii) design, (iii) implementation, (iv) testing, and (v) deployment. The GDLC is divided into three phases which are pre-production, production, and post-production. To initiate the development of the RoboCodin turn-based serious game, the game developers put great effort into designing unique and engaging game mechanics. Figure 3 illustrates the game development life cycle that is applied throughout the development of the game.

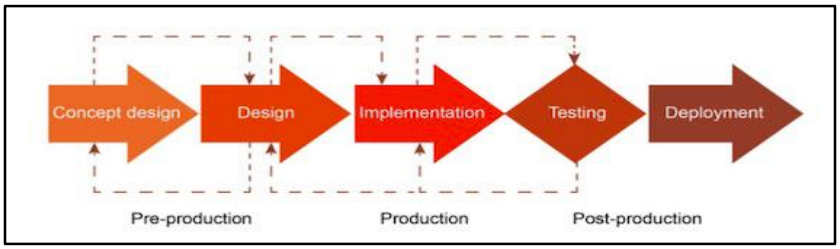

Figure 3. Game development life cycle. Source: Adapted from Conteras-Espinaso & Eguina-Gomez (2017)

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Concept Design

The RoboCodin turn-based serious game is proposed through a game design document (GDD) that consist of (i) a storyboard, (ii) a flowchart, (iii) a mood board, and (iv) a character mood board. It is proposed based on the game ideas and the relevancy gained from the literature review as in the section of Issues and Challenges and Table 1. To conduct the concept design, Figure 4 illustrates an example of the process that occurs in the stage of concept design.

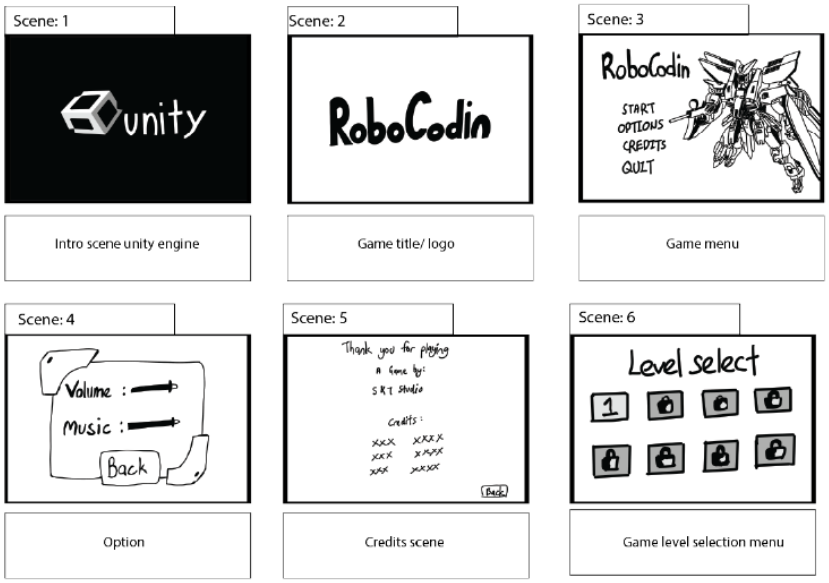

Figure 4. Storyboard of the RoboCodin Turn-Based Serious Game

Design

Gamified modules are applied in this design stage where game-design elements are integrated into the educational content of JavaScript programming. This stage includes features like level design, platforms, experience points (XP), badges, rewards, and obstacles. The game mood board and storyboard illustrate these features through colourful and inspirational sketches. Figure 5 showcases several inspirational sketches of the main characters in the RoboCodin turn-based serious game.

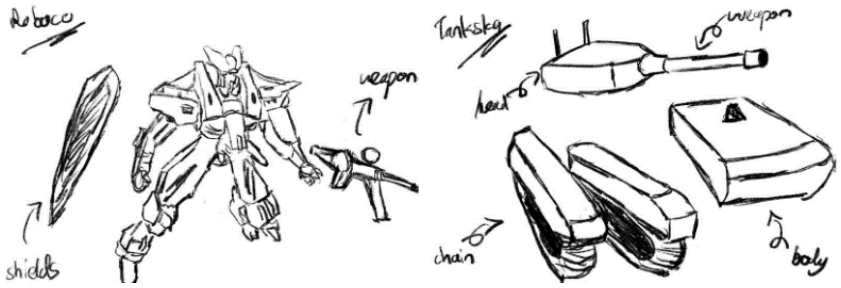

Figure 5. Inspirational Sketches of the RoboCodin Turn-Based Serious Game.

In the design stage, all characters and other game assets are started to be designed. It includes sketches of small and large objects such as weapons, shields, non-player characters, buildings, a basement, and others. Maps are also created to visualize the level structure.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Implementation

After completing the design stage, core development of the RoboCodin turn-based serious game has begun. It includes asset creation and integration of systems physics and AI. For the asset creation, it involves art, audio, and animation components that aim to add depth, visual appeal, and immersion to the game. Every level has a different design, atmosphere, objects, and difficulties. The implementation process also involves modelling, programming, and game mechanics.

Modeling

3D character models are developed using the Blender 3D computer graphics software tool. The animation character is also created using the software. Figure 6 depicts one of the modeling processes of the main player character.

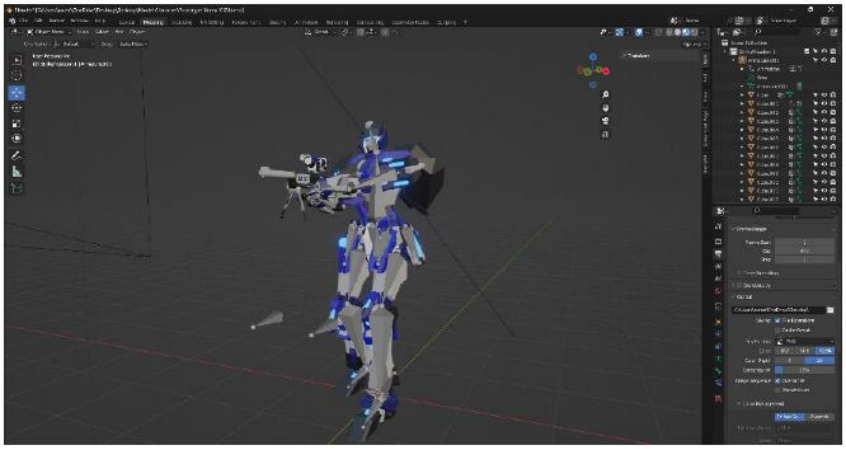

Figure 6. The Modeling Process of the Main Player Character

After character modeling is complete, the rigging process is conducted to animate character movement. It involves complex and comprehensive programming processes and techniques.

Programming

Unity3D is utilized as the main game engine, while Visual Studio is used as a code editor to write and debug the game code using the C# programming language. One of the examples is the animation code which is used to animate the player and non-player characters. The utilization of both Unity 3D and Visual Studio is crucial to bring the RoboCodin turn-based serious game to life through coding and scripting. The researchers of this study as well as game developers, write code that governs the game mechanics, AI, physics, and sound. A sound system application is utilized to add audio effects and music to the game, aiming to enhance the atmosphere and game immersion.

Game Mechanics

Game mechanics for the RoboCodin turn-based serious game are related to rules, systems, and interactions that govern the way players play the game to achieve the objective of learning JavaScript programming. To visualize the game mechanic work throughout the gameplay, Figure 7 depicts a flowchart of the gameplay of the RoboCodin turn-based serious game.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

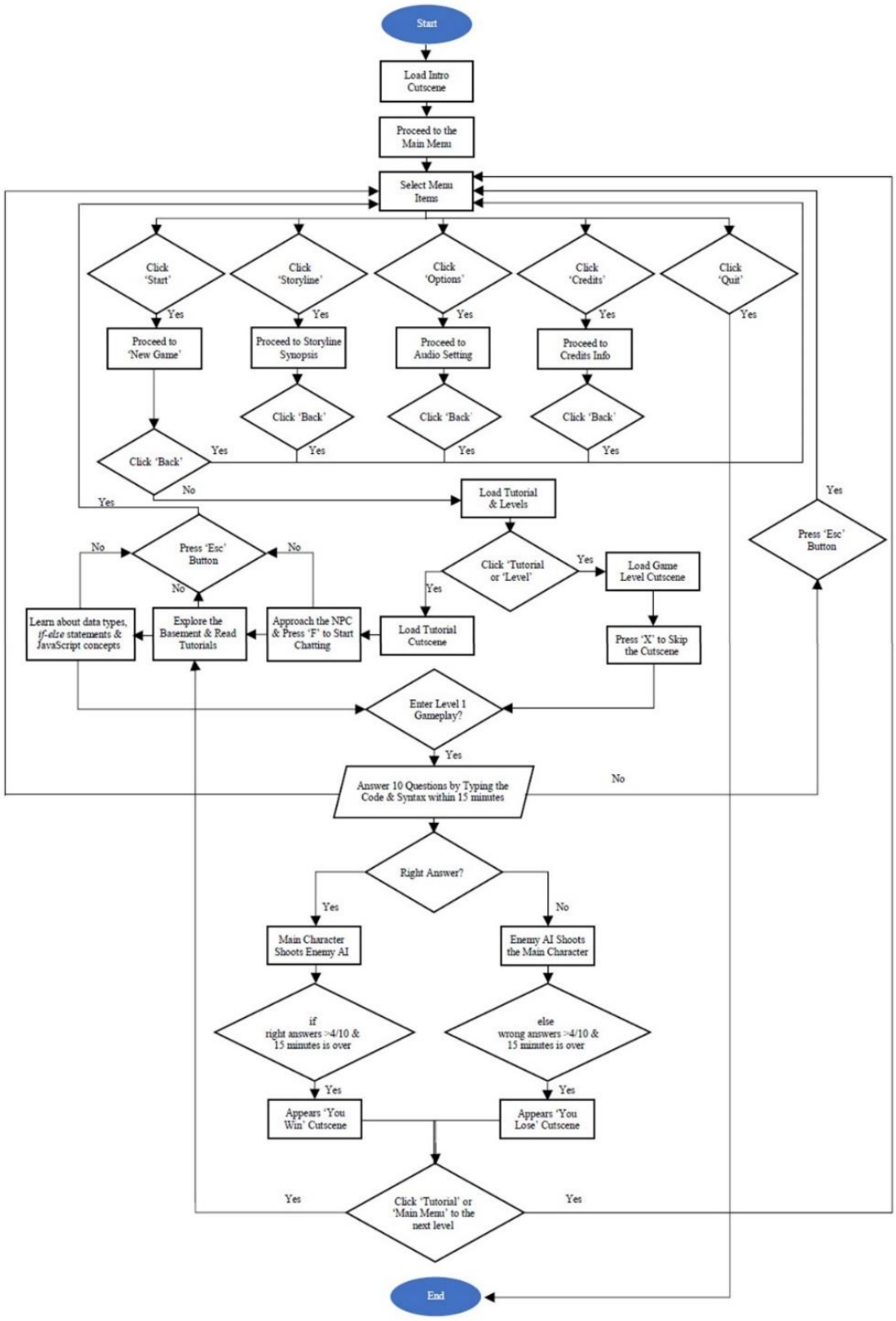

Figure 7. Flowchart of the RoboCodin Turn-Based Serious Game's Gameplay

In the game, six mechanics are applied which are (i) aesthetic mechanic, (ii) narrative mechanic, (iii) exploration mechanic, (iv) action mechanic, (v) strategy mechanic, and (vi) progression mechanic. These mechanics make the game work like the skeleton that holds the game together (Wayline, 2025). They are applied throughout the RoboCodin turn-based serious game where the player as a robot takes turns by answering JavaScript questions before shooting an enemy.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Aesthetic Mechanic

An aesthetic mechanic is employed to contribute to the visual and auditory style of the game. It refers to the sensory phenomena like visual, aural, haptic, and embodied that make players attracted to and feel the game (Goethe, 2019). To visualize the game aesthetic, Figure 8 depicts two examples of the main character pose in the RoboCodin turn-based serious game.

Figure 8. Robot Aesthetic in the RoboCodin Turn-Based Serious Game

Aesthetic mechanics in the game also include the transition and manipulation of graphics, lighting, animation, audio and sound, and realism. Sound effects from gunshots, smoke as bullets hit characters, and thundering sounds are set to enhance the auditory sense of the RoboCodin turn-based serious game.

Narrative Mechanic

The narrative mechanic used in the RoboCodin turn-based serious game enhances player engagement during gameplay. Based on the storyline as shown in Figure 9 on the left, the player character called Xorex is only proficient in C++ and C#, so he struggles to complete his programming tasks in JavaScript. However, he hears a whisper that offers to assist him, and that's when the adventure of the game begins.

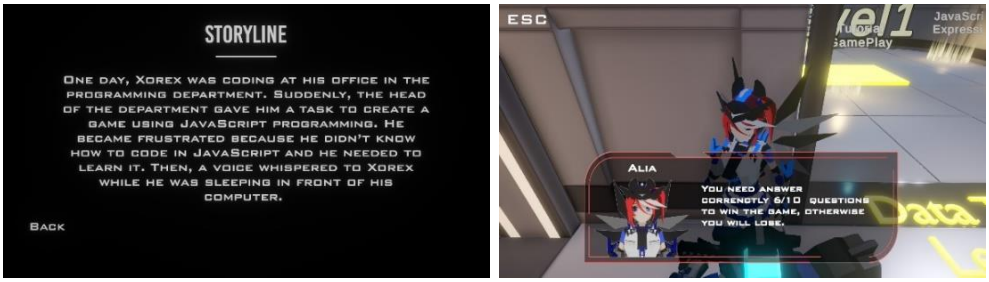

Figure 9. Narration of the RoboCodin Game's Story

Figure 9 on the right depicts narration in the game and conversation between the player and a non-player character. It shows a dialogue between the player character and the non-player character named Alia that provides instruction on the gameplay. This narrative mechanic allows the player to engage in conversations with the non-player character to obtain instructions as well as clues and enables the player character to explore the game.

Exploration Mechanic

The exploration mechanic allows the player to walk to discover new areas of three different game levels. The player explores in a parallel robotic base to uncover secrets from the beginning of the level to the last area. Each area has different game content including hidden objects, items, information, and programming tutorials. Figure 10 depicts exploration in the base of the RoboCodin turn-based serious game.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

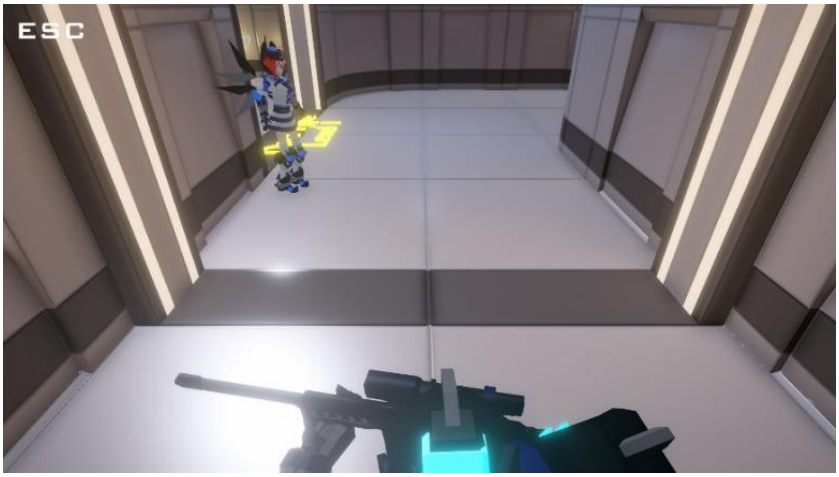

Figure 10. Base Exploration in the RoboCodin Turn-based Serious Game

With the application of the exploration mechanic, the player becomes increasingly interested and curious about the base. It motivates the player to explore the game environment and further to act in all levels of the RoboCodin turn-based serious game.

Action Mechanic

The player interacts directly with the game world in real-time where the player has a limited number of actions to execute during his turn to answer programming questions. To progress the action, a sub-mechanic called movement is applied where the player is able to control the character's movement across the game world. The programming questions in the RoboCodin turn-based serious game are simple, and it is straightforward for the player to make a choice. The mechanic also enables the player to learn more about the game's lore and interact with enemy AI in a meaningful way as in Figure 11.

Figure 11. Answering a Syntax Question in the Gameplay.

Based on Figure 11, a sub-mechanic called shooting allows the player character to engage in combat. The player needs to answer a programming question before being enabled to fire a projectile at the enemy AI. The RoboCodin turn-based serious game is designed with a typing system that can only read the correct answer provided by the AI mechanic. To answer the question, the player needs to key in the JavaScript code and syntax. This execution enables the player to control the robot's movements, and attacks. A sub-mechanic called Quick Time Events (QTEs) follows accordingly where the player quickly presses the 'Enter' button to perform an action (shooting).

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Strategy Mechanic

The RoboCodin turn-based gameplay allows the player to strategize his plan to attack and defeat the enemy. He needs to plan strategies before answering questions because all questions have different levels of difficulty. To strategize, the player needs to remember the syntax provided in the early tutorials. Next, the player should make decisions to enter JavaScript code into the answer box within a limited time. Then, a sub-mechanic turn is applied to enable the player to take a turn to fire at an enemy. Figure 12 illustrates the application of strategy mechanics.

Figure 12. Shooting Mechanics in the Gameplay.

The player needs to think strategically to remember the programming tutorial in a limited time before engaging in the shooting. This turn-based serious game will also present several challenges for the player. The player needs to manage his time wisely and plan strategies appropriately to defeat the enemy.

Progression Mechanic

Every level in the RoboCodin turn-based serious game features a tutorial mode that guides players to learn JavaScript programming. The tutorials are designed to be accessible and easy to understand, even for players who have no prior programming experience. Table 2 shows the three different levels for progress in the RoboCodin turn-based serious game.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Table 2. The Three Levels in the RoboCodin Turn-Based Serious Game

Level	Figure	Description
1		Novice level: Data Types is the name of the first game level. From the beginning to the end of this level, the player must write a JavaScript output to respond to programming questions.
2		Intermediate level: The *if-else* statement is the name of the second level. From the beginning to the end of this level, the player must write JavaScript syntax. He is given four choices to select the correct answer and then write it in an answer box.
3		Advanced level: At the third level, the player needs to write code by himself. The player faces JavaScript questions from the beginning to the end of this level. Then he needs to write the correct coding.

To progress each level, a timer script is applied in the development of the RoboCodin turn-based serious game. It keeps track of the elapsed time and can be used to trigger events such as spawning enemies, creating power-ups, or ending the game after a certain amount of time has passed. One common use of a timer script in the game is to create time-limited challenges, such as completing a level or defeating an enemy within a limited timeframe. The timer script is used to display a countdown timer on the screen which creates a sense of urgency and adds an extra level of challenge to the game to gain achievements. Figure 13 depicts the sub-mechanic for the timer.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Figure 13. Timer System while Answering Question.

As a player progresses in the game, he will learn different coding concepts including conditional statements, loops, and functions that will guide him to complete the complex programming mission. When he finishes a level, he will be motivated to progress to the next levels. This sense of progression serves as an engaging factor to learn more about programming in the RoboCodin turn-based serious game.

Testing

After the implementation stage of the RoboCodin turn-based serious game is complete, it undergoes an initial playtest called alpha testing. It is nearly complete in the post-production phase of the game development life cycle where the alpha testing is conducted internally by the researchers of this study to identify design flaws including bugs, glitches, and also to gather feedback. This test aims to optimize performance and polish the playing experience to ensure its playability and control. After the alpha testing is complete, beta testing is conducted to identify usability issues, performance problems and also to gain general feedback from the target audience which is tertiary students.

Evaluation

The evaluation is then conducted to ensure the usability of the RoboCodin turn-based serious game for the purpose of JavaScript programming learning. It involves evaluating the player interaction in the game, focusing on game mechanics, navigation, instruction, control, user interface, tutorials, and overall flow. To evaluate the game, a research instrument is developed.

Instruments

The instrument of this study is developed based on the third research objective which is to evaluate the learners' perceptions of the RoboCodin turn-based serious game. To ascertain their perceptions of the game, a quantitative technique is adopted in this study where a questionnaire is created. It is divided into (i) demographic profile, (ii) construct of efficiency, (iii) construct of control, (iv) construct of helpfulness, and (v) construct of learnability. For the demographic profile, it consists of three questions of nominal and ordinal data in measuring participants' profiles. The other four constructs are created based on existing research and the researchers' ideas where each construct has 15, 5, 5, and 10 questions/items, respectively. All the items use the psychometric technique which is the Likert scale that consists of fixed-response or multiple-choice questions.

Furthermore, the questionnaire is tested to gain its reliability through the Cronbach's Alpha test. This test is conducted to determine data-collecting inter-item reliability which is a measure of the degree of internal consistency between measurements of a construct in the questionnaire of this study. To calculate the reliability of the questionnaire, the Cronbach's Alpha test is then conducted, and the values are shown in Table 3.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Table 3. Cronbach's Alpha Values of the Questionnaire

No	Constructs	Number of Items	Cronbach's Alpha
1	The efficiency of the RoboCodin	15	0.853
2	The control of the RoboCodin	5	0.652*
3	The helpfulness of the RoboCodin	5	0.701
4	The learnability of the RoboCodin	10	0.944

Based on Table 3, the Cronbach's Alpha values are more than 0.700 ($\alpha > 0.7$) and range between 0.701 and 0.944. Therefore, the instrument demonstrates satisfactory reliability (Santos, 1999). However, the control construct shows a mean value of 0.652 ($\alpha < 0.7$), thus, it is removed from the instrument. After the instrument is completely developed, participants are determined to playtest the RoboCodin turn-based serious game.

Participants

Participants involved in the playtesting for this study are determined using purposive sampling. In the purposive sampling, tertiary students who play games frequently are selected to playtest the RoboCodin turn-based serious game. They are chosen due to their homogeneity, as they have experience playing games and are familiar with the purpose of game development. Thus, they can compare games that apply turn-based mechanics with games that don't.

Procedure

After selecting the participants, the RoboCodin turn-based serious game undergoes a playtest. All of the participants are invited periodically by the researchers of this study to participate in the face-to-face playtesting. When the participants agree and are ready for it, the researchers present a briefing on the objective of the study, followed by instructions on how to play the game. During the playtesting, they are free to play until they have thorough experience playing the game. After finishing playing the game, each of them is given a survey questionnaire to be completed.

Data Collection and Analysis

Data collection of the RoboCodin turn-based serious game is gathered within two weeks where all questionnaires are returned to the researchers of this study. Now, the data is ready to be distributed in the SPSS statistical software package and analysed empirically through descriptive statistics. This descriptive statistic is conducted to analyse the participants' profiles and the four constructs. Based on the data, the participants' age is between 18 and 40 years. Most of them are aged 24-29 years. Figure 14 depicts the percentage of participants' demographics.

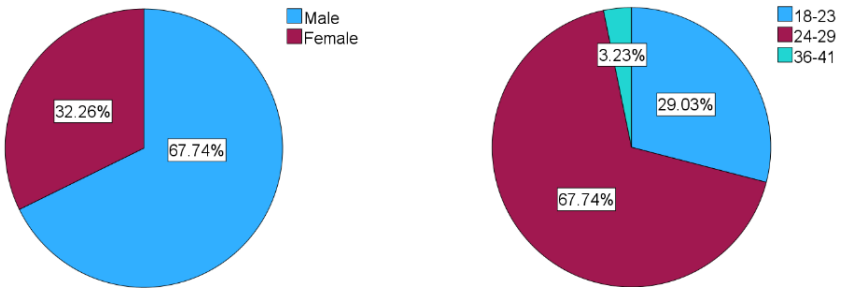

Figure 14. Percentage of Participants Based on Gender and Age

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Furthermore, descriptive statistics for all items in the three constructs are conducted. It is conducted to measure the participants' attitudes toward the favourable and unfavourable statements of all the items. Table 4 shows the descriptive statistics for questionnaire-based metrics of 30 items.

Table 4. Descriptive Statistic of the Items of the Three Constructs

	Construct Items	Mean	Standard Deviation
The efficiency of the RoboCodin			
1	The organization of the menus lists seems logical.	4.7097	.46141
2	It is easy to see what the options are in each stage.	4.2581	.77321
3	I am familiar with the gameplay in RoboCodin.	4.3548	.79785
4	The movement of RoboCodin is fast enough.	4.3548	.95038
5	Tasks can be performed in a straightforward manner.	4.6774	.54081
6	The information is clearly presented.	4.6452	.55066
7	The RoboCodin tutorial explains all the information.	4.6129	.49514
8	There are no issues or bugs with the control system in RoboCodin.	4.0323	1.19677
9	I am confused with the instructions given while playing RoboCodin.	3.1613	1.50769
10	RoboCodin is using many keystrokes.	3.8710	1.08756
11	The steps required to type and submit the answer are complicated.	3.0968	1.53252
12	The input answer is visible.	4.5806	.67202
13	Is the game consistent?	4.5161	.67680
14	The game makes me dizzy while playing.	3.1935	1.72084
15	I think RoboCodin fit my expectation.	4.4194	.84751
The helpfulness of the RoboCodin			
1	Is the tutorial helpful?	4.6774	.54081
2	RoboCodin provides explanations on how to solve syntax errors in JavaScript.	4.6129	.49514
3	It is easy for me to play the game after reading "how to play" in the tutorial.	4.4839	.72438
4	I think that player's needs have been fully considered.	4.1613	.89803
5	I need external assistance most of the time while playing RoboCodin.	3.8387	1.26746
The learnability of the RoboCodin			
1	Playing RoboCodin helps me to learn more about JavaScript programming.	4.6452	.48637
2	The questions are easy to read while playing RoboCodin.	4.5484	.67521
3	I learn more about "data type" in level 1.	4.5161	.81121
4	I learn more about "*if-else* statements" in level 2.	4.5161	.85131
5	I learn more about "syntax error" in level 3.	4.5161	.81121
6	Is it simple to learn JavaScript?	4.2903	.93785
7	I learn the basic JavaScript in the tutorial map.	4.5484	.88840
8	I can learn the concepts of JavaScript through the game.	4.6129	.55842
9	Playing RoboCodin helps me gain a better understanding of JavaScript.	4.5806	.62044
10	Playing RoboCodin makes me more interested in learning JavaScript.	4.5484	.62390

https://doi.org/10.37134/jadcm.vol1.5.4.2026

RESULTS

The RoboCodin turn-based serious game significantly impacted tertiary students' learning of JavaScript programming. Based on Table 4, all the construct items showed a mean value greater than three ($\bar{x} > 3$). This descriptive interpretation of the findings suggests that participants in the playtesting tended to agree with the statements related to the items. To interpret the findings, Table 5 presents the interpretation of mean values on the 5-point Likert scale.

Table 5. Interpretation for Mean Values of 5-point Likert Scale

Mean Value Range	Interpretation	Explanation
1.00 – 1.99	Very Low	Strong disagreement; participants perceive the construct very negatively.
2.00 – 2.99	Low	Disagreement; participants perceive the construct negatively.
3.00 - 3.49	Neutral/ Moderate	Neither agree nor disagree; mixed or neutral perception.
3.50 – 3.99	Fairly High	Leaning towards agreement; generally positive perception but not strong.
4.00 – 4.49	High	Agreement; construct is positively perceived.
4.50 – 5.00	Very High	Strong agreement; construct is perceived very positively close to optimal.

To interpret the relationship between the mean values and the descriptive statistical results of the constructs, Table 6 presents the mean values for the three constructs. The mean values indicate the tendency of participants to agree with the RoboCodin turn-based serious game.

Table 6. Interpretation for Descriptive Statistical Results of the Constructs

No	Constructs	N	Mean	Standard Deviation	Interpretation
1	The efficiency of the RoboCodin	31	4.1656	.57320	High
2	The helpfulness of the RoboCodin	31	4.3742	.55074	High
3	The learnability of the RoboCodin	31	4.5323	.60464	Very High

Each mean value of the three constructs is greater than four of the five points of the Likert scale ($\bar{x} > 4.1$). It indicates the participants agreed and strongly agreed with the statements about the items of each construct. This agreement reflects the level of optimism perceived by the participants (Alkharusi, 2022). Their perceptions of (1) efficiency, (2) helpfulness, and (3) learnability of the RoboCodin turn-based serious game are significantly favourable. The RoboCodin turn-based serious game is perceived as efficient where gameplay and tasks in the game can be completed without unnecessary effort and complexity ($\bar{x} > 4.1656$). For the helpfulness construct, the players who play the RoboCodin turn-based serious game found the game to be useful and supportive in achieving the goal of learning JavaScript programming ($\bar{x} > 4.3742$). The RoboCodin turn-based serious game is regarded as beneficial for learning JavaScript programming, with a mean score of 4.5323.

DISCUSSION

The RoboCodin turn-based serious game provides fundamental concepts of JavaScript programming such as data types and if-else statements, thus, it might be suitable for beginners. Learners who are

https://doi.org/10.37134/jadcm.vol1.5.4.2026

proficient with JavaScript may find the game easy to play. However, the initiative in developing this type of game has become a crucial effort in education. This endeavour is important because studying potential learning tools can enhance pedagogical practices. According to Bachillar-Burgos et al., learning programming is important at the early stages of education because it can shape problem-solving, logical thinking, and application development. Furthermore, the researchers of this study found that learning programming through practical application with a game-based learning tool is an efficient mechanism for improving programming comprehension. It is because using a game-based learning tool such as the RoboCodin turn-based serious game conveys learning content in an enjoyable manner and helps learners solve coding errors more quickly. As stated in the result section, the RoboCodin turn-based serious game indicates a positive perception in terms of efficiency, helpfulness, and learnability. It is user-friendly and well-accepted. Overall, the participants significantly agreed that the game is useful and advantageous as a learning tool for programming. These findings indicate that learners consider the RoboCodin turn-based serious game to be efficient, helpful, and highly learnable.

Based on the research objectives, the first research objective was achieved where the suitable game mechanic for learning programming was identified. The game mechanics applied in the RoboCodin serious game were turn-based mechanics. The developers of the game consider techniques and ways to adjust the mood in a subtle way in its game mechanics. Various methods exist to engage players' expectations, including altering the sense of place and developing unexpected relationships between the player and the protagonist. In the RoboCodin turn-based serious game, players can be more interested in exploring the game world and experiencing the game mechanics. Experiencing the game mechanics can enhance player engagement and further motivate players to explore more in JavaScript programming. Furthermore, the game was started to be designed and developed using the application of the Game Development Life Cycle. The process of developing the RoboCodin turn-based serious game took about 10 months to complete from the game's pre-production to post-production. All the processes were conducted to achieve the second research objective which is to develop an isometric turn-based serious game. Then, the game went through iterative processes including alpha and beta testing, eventually the RoboCodin turn-based serious game was free from critical errors, bugs, and glitches. Finally, it was evaluated through a playtesting session. It is to achieve the third research objective which is to evaluate the learners' perceptions of the RoboCodin turn-based serious game.

CONCLUSION

In the final stage of the game development life cycle as depicted in Figure 3, the RoboCodin turn-based serious game is ready to be released to the public. The tertiary students as well as learners, are able to install the game on personal computers and play it to experience the gameplay of the RoboCodin turn-based serious game. This game serves as an initial step for them to engage in interactive learning and acquire JavaScript programming skills. When they enjoy and engage with this turn-based serious game, they will build their comprehension of JavaScript programming and ultimately master it.

Overall, the RoboCodin turn-based serious game is an innovative way for learners in tertiary education to learn JavaScript programming in an advanced way. This innovative approach is made possible by a unique turn-based serious game that incorporates game-based learning and interactive features, including 3D game objects, turn-based mechanics, and a robotic theme. It leverages the benefits of gaming to make learning programming a more exciting and interactive experience. By using turn-based gameplay, the learners will be encouraged to develop their problem-solving skills and enjoy the programming learning process. Therefore, the RoboCodin turn-based serious game could be a valuable tool for learning JavaScript programming, as mastering programming skills is important for job prospects in various industries.

ACKNOWLEDGEMENT

Muhammad Abdul Malik Saedon is an instructor in Bachelor Degree of Design (Digital Games) at Universiti Pendidikan Sultan Idris. As an instructor as well as a researcher, teaching, supervising and consulting other two researchers of this study are his roles. The two researchers are Muhammad Shameer Sawar and Tan Wei Jie who play roles in designing, modelling, animating, developing, coding, and playtesting the RoboCodin turn-based serious game.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

No funding

This study was not supported by any grants from funding bodies in the public, private, or not-for-profit sectors.

CONFLICT OF INTEREST

The authors declare no conflicts of interest

AUTHORS CONTRIBUTION

CRediT author statement: **Author 1. Muhammad Abdul Malik Saedon**: Conceptualization, Methodology, Writing-Original draft preparation, Visualization, Investigation, Software, Validation, Writing-Reviewing and Editing, Data Analysis, Data Curation, Supervision. **Author 2. Muhammad Shameer bin Sawar**: Conceptualization, Writing-Original draft preparation, Software, Visualization, Investigation, Game Modeling, Data Collection. **Author 3. Tan Wei Jie**: Conceptualization, Writing-Original draft preparation, Software, Visualization, Investigation, Game Modeling, Data Collection.

AVAILABILITY OF DATA AND MATERIALS

Data is openly available in a public repository, with a permanent identifier (such as a DOI).

DECLARATION OF GENERATIVE AI

The authors declare that no generative AI was used in the writing of the manuscript.

ETHIC STATEMENTS

Not applicable

REFERENCES

Abu Bakar, E. E., Abd Halim, M. D., Abdul Hanid, M. F., & Inderawati, R. (2025). The Challenge of Teaching and Learning Programming in School: Insights from a Systematic Literature Review, *Karya Journal of Emerging Technologies in Human Services*, 1(1), 48-63. doi.org/10.37934/kjeths.1.1.4863.

Abdul, T. F. B., Anuar, N., & Said, M. R. F. (2018). How the nature of programming and learning materials affects novice learner's motivation and programming ability, *ACM International Conference Proceeding Series,* 124–128. https://doi.org/10.1145/3178158.3178184.

Ahmad, S. N., & Ghazali, J. (2020). *Programming Teaching and Learning: Issues and Challenges*, In Proceeding of ICITS 2020, 6th International Conference on Information Technology & Society.

Agate (2025, August 7). *The Art of Strategic Gameplay: Turn-based Game*. https://agate.id/the-art-of-strategic-gameplay-turn-based-games/.

Alkharusi, H. (2022). A Descriptive Analysis and Interpretation of Data from Likert Scales in Educational and Psychological Research*, Indian Journal of Psychology and Education*, 12(2), 13-16.

Bachiller-Burgos, P., Barbecho, B., Calderita, L. V., Bustos, P., & Manso, L. J. (2020). LearnBlock: A Robot-Agnostic Educational Programming Tool, *IEEE Access*, 8, 30012-30026. doi: 10.1109/ACCESS.2020.2972410.

Contreras-Espinosa, R. S., & Eguia-Gomez, J. L. (2017). Usability and user experience methodologies used by games companies, Observatory of Communication, Games and Entertainment, Autonomous University of Barcerlona and University of Vic-Central of Catalonia, Barcelona, Spain, *Catalonia Research Project Rep*.

Fiveable. (2025, August 5). *Turn-based Strategy - Intro to Humanities*. https://library.fiveable.me/key-terms/introduction-humanities/turn-based-strategy.

https://doi.org/10.37134/jadcm.vol1.5.4.2026

Garcia, C., & Lemos, N. (2023). The Gamification of E-learning Environment for Learning Programming, *International Journal on Informatics Visualization*, 7(2), 455-462.

Goethe, O. (2019). Visual Aesthetics in Games and Gamification, in Gamification Mindset, Ed. Cham: *Springer International Publishing*, 85-92.

Islam, M. T., Islam, M. R., Jhilik, R. A., Islam, M. A., Raihan, P. M. S., Faruque, M. S., & Shahjahan, A. M. (2024). A relative Analysis of Programming Language Preferences Among Computer Science and Noncomputer Science Students, *European Journal of Theoretical and Applied Sciences*, 2(3), 900- 912. doi.org/10.59324/ejtas.2024.2(3).70.

Kadar, R., Wahab, N. A., Othman, J., Shamsuddin, M., & Mahlan, S. B. (2021). A Study of Difficulties in Teaching and Learning Programming: A Systematic Literature Review, *International Journal of Academic Research in Progressive Education and Development*, 10(3), pp. 591–605. doi.org/10.6007/IJARPED/v10-i3/11100.

Kappagantula, A. (2024). The Evolution of JavaScript: From Client-side Scripting to Universal Computing Platform, *International Journal of Computer Engineering and Technology*, 15(6), 1509-1517.

Logan, D. (2025, June 9). *Top 8 Most Demanded Programming Languages in 2024*. https://www.devjobsscanner.com/blog/top-8-most-demanded-programming-languages/.

Makela, V., & Schmidt, A. (2020). *I Don't Care as Long as It's Good: Player Preferences for Real-time and Turn-based Combat Systems in Computer RPGs.* In Proceedings of the CHI PLAY '20, Virtual Event, Canada, Nov. 2-4. doi.org/10.1145/3410404.3414248.

Mahtgician Game (2025, June 26). *Designing Turn-based vs Real-time Game Mechanics*. https://mahtgiciangames.com/blogs/the-creative-workshop-game-design-blueprints/designing-turn-based-vs-real-time-game-mechanics?srsltid=AfmBOorLy_4MxeuIo6F6E2eifniqksGMs7ApiVwG3ZcGvOoQug7GRkKk

Massoudi, M. (2019). A Review on Challenges and Solutions in Learning Programming Courses at Undergraduate Level, *International Journal of Applied Research*, 5(8), 146-149.

Medeiros, R. P., Ramalho, G. L., & Falcao, T. P. (2019). A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education, *IEEE Trans. Educ*., 62(2), 77–90.

Mehmood, E., Abid, A., Farooq, M. S., & Nawaz, N. A. (2020). Curriculum, Teaching and Learning, and Assessment for Introductory Programming Course, *IEEE Assess*, 8, 125961-125981.

Monteiro, A. F., Miranda, P. M., & Osório, A. J. (2021). Coding as Literacy in Preschool: A Case Study. *Educ. Sci.,* 11(198). doi.org/10.3390/educsci11050198.

Muhamad Yusof, M., Ab Jalil, H., & Perumal, T. (2021). Exploring Teachers' Practices in Teaching Robotics Programming in Primary School, *Asian Social Science*, 17(11). doi.org/10.5539/ass.v17n11p122.

Patil, R. S., Gawande, H. P., Rai, P. R., Padwal, R. M., Ram, N. D., Kushwaha, B. O., Kene, K. D., & Patil, R. R. (2025). Choosing the Right Programming Language for Beginners: A Study of Learning Preferences and Career Impact*, International Research Journal of Modernization in Engineering Technology and Science*, 7(3), 588-593. doi.org/10.56726/IRJMETS68520.

Rachmawati, O. C. R.. Barakbah, A., & Karlita, T. (2024). Programming Language Selection for the Development of Deep Learning Library*, International Journal of Informatics Visualization*, 8(1), 434-441.

Rojas-López, A., Rincón-Flores, E. G., Mena, J., García-Peñalvo, F. J., & Ramírez-Montoya, M. S. (2019). Engagement in the Course of Programming in Higher Education Through the Use of Gamification, *Universal Access in the Information Society*, 18(3), 583–597. doi: 10.1007/s10209-019-00680-z.

Salmiah, S. H. (2023). A Systematic Review of the Challenges in Teaching Programming for Primary Schools' Students, *Online Journal for TVET Practitioners*, 8(1), 75-88. doi.org/10.30880/ojtp.2023.08.01.008.

Santos, J. A. (1999). Cronbach's Alpha: A Tool for Assessing the Reliability of Scales, *The Journal of Extension*, 37(2).

Vailshery, L. S. (2025, June 12). *Most Used Programming Languages Among Developer Worldwide 2024*. https://www.statista.com/statistics/793628/worldwide-developer-survey-most-used-languages/.

Wayline (2025, June 25). *A Beginner's Guide to Game Mechanics*. https://www.wayline.io/blog/intro-to-game-mechanics.

Xinogalos, S. (2016). Designing and Deploying Programming Courses: Strategies, Tools, Difficulties and Pedagogy, *Education and Information Technologies*, 21(3), 559–588. doi.org/10.1007/s10639-014-9341-9

https://doi.org/10.37134/jadcm.vol1.5.4.2026