1. Project Background

STEM education is a teaching method that is a combination of Science, Technology, Engineering, and Math (Hom & Dobrijevic, 2022). There are lots of STEM lessons provided by different organizations. For example, Scratch (Scratch, n.d.) and Micro:bit (OXXO.STUDIO, n.d.) Moreover, some big-scale STEM competitions, like the International Youth Tech Olympics (Techbob Academy, 2022), are organized for students to participate in. Those lessons and competitions provide an opportunity for students to learn about the current technology in the world or even gain experience using those technologies.

In recent years, the popularity of STEM has increased in several education stages (Irwanto et al., 2022). The success of this teaching approach is related to the contribution it makes to society. There are several advantages for students studying STEM. First, STEM education involves lots of current technology. For example, drones (Drone Legends, 2022) and artificial intelligence (STEM, n.d.). Students have an opportunity to try out those technologies and learn related knowledge, which can broaden their horizons. Moreover, the use of technology in deeper learning can provide a chance for students to use different skills. For example, critical thinking, problem solving, teamwork, and self-directed learning (Ismail, 2018) can enhance students’ skills in those categories. These kinds of skills are useful in various areas. For example, in their studies and workplace, These show that STEM education brings benefits to students, which is a big contribution to the education sector.

Because of the popularity of STEM education and its advantages to students, it is worth choosing this teaching approach as the final year topic to explore more. Moreover, I am interested in developing such educational software due to my internship experience.

Through my internship at a STEM educational company, I saw a lot of teaching materials and had a chance to participate in the teaching of programming. I saw that most of the teachings are telling students to follow the steps in order to create a function or even an app. However, students have difficulty extracting the programming concept from the STEM lesson because of their lack of programming knowledge and experience. Through the lesson, they do not know why this function or code is needed in order to create that functionality and pick up some important concepts from the tasks. As a result, I think the final year project is a good opportunity to give me a try at creating a STEM education platform that teaches some programming concepts from games.
After searching and experiencing some STEM education platforms, those platforms can be divided into three different types.

First, the platform requires some hardware components to support STEM teaching. For example, Matatalab (Matatalab, n.d.) and Micro: bit (OXXO.STUDIO, n.d.) Those pieces of hardware can provide a hands-on experience for the students, which can increase their attractiveness. On the other hand, these kinds of STEM teaching methods have a number of restrictions on program creation due to the hardware used, and some commands are only used for a specific purpose, like the Music and Led command (Microsoft MakeCode, n.d.), which may not be good for programming concept teaching as the underlying principles of these commands are hidden. Moreover, students are required to buy the hardware needed so that they can learn those topics.

Second, the platform only requires a compiler, like a mobile phone or computer, to support STEM teaching. For example, Scratch (Scratch, n.d.) and App Inventor (MIT App Inventor, n.d.). The major advantage of these platforms is that students have the freedom to create whatever they want on the frontend and backend. However, their main limitation is that they do not have a goal for users to complete, which is a disadvantage for self-study in programming as users do not have a guide or target to follow when they are studying.

Third, the platform has guidelines to follow, and the user can complete the tasks step-by-step. For example, Code.org (Code.org, 2023). These kinds of platforms focus on the backend and provide guidelines on every topic for students to study. One of the improvements for this kind of platform is that although they are using block programming for the program part, which will be explained in the project objective part, the length of the whole program code will still be long when the tasks become more difficult and require lots of code blocks to complete the whole program. These limitations may reduce interest in using those platforms, as students need to spend a long time to finish the whole task due to the length of the program without seeing any significant progress.

After the observation of different kinds of programming learning platforms, several points are summarized based on their advantages and limitations. First, those platforms have clear grouping on coding blocks, which can reduce the difficulty for students finding the corresponding code block. Second, guidelines or tips are needed for students to follow so that when they are facing problems, the platform can provide support to them in order to finish the tasks or understand the concept. Third, the balance between gameplay and educational elements is important to make the game interesting enough that students can learn from it.
Fourth, the use of programming concepts should be obvious to reduce difficulty and be easy
to understand. Fifth, divide a task into several subtasks to shorten the length of the program
code. Moreover, using some visuals like a message box or animation between different
subtasks will let the student understand their progress on a long task and teach them step-by-
step instead of only showing the final result when they finish the long task. In the project
objective part, the implementation of the points mentioned will be mentioned.

2. Project Objective & Description

The goal of this project is to create a STEM education game that focuses on
programming and teaches different kinds of programming concepts. This game project will
learn from the advantages and limitations of the existing programming teaching platform to
build a more competitive product. The measure for the finalized project will consist of two
parts. First, personal testing. This part focuses on both programming and interfaces. For
example, are all the objects, like the graphics and buttons, correctly shown on the screen and
work properly? Are any unexpected behaviors and the logic flow of the game tasks correct?
Second, user testing, which will be mentioned in the Project Schedule and Milestones part,
This part focuses on the user experience, like user friendliness and suitable difficulty. These
two tests are to evaluate the programming implementation, gaming experience, and
effectiveness of programming teaching. The final presentation, final report, and project
exhibition will be held in April 2024. As a result, the internal deadline for the finalized
product should be March 2024, which reserves time for user testing and product modification
if anything goes wrong. Below is detailed information about the expected product.

2.1 Product Name

The name of the product at this stage is “Life Operator”. This name combines two
main elements in the game: daily life and using programming concepts to control, which
will be explained in the Major Features part.

2.2 Workable Operating System

The operating system of the game project will be Windows at this stage to avoid
any extra modifications to the game when there is some conflict or unexpected behavior
when installing on another operating system. Moreover, it will be easier to test the game
on my own machine.
2.3 Gameplay Overall Description

The user needs to control the character in order to finish the tasks, which are the whole process of some daily activities, by completing the block programming part of different tasks. The block programming part will act as a skill, which is a normal game element. The user is required to combine several code blocks into a complete action for the character to follow. If the action is correct, the game will move on to the next procedure. If the action is not correct, tips will be given, like highlighting the key point in the subtask description or mentioning the missing component with the help of the message box.

2.4 Gameplay Example

This part will have a brief description of a game task example. It is just a simplified task to demonstrate what a task looks like and how to connect programming and daily life. The topic of this simplified task is buying things in a supermarket.

1. When a sale sees a customer arrive, the sale needs to greet the customer. This part requires a **print concept** to form the action.

2. The customer wants to buy an apple from the supermarket. This part requires a **print concept** to tell the customer what he wants. Also, **input concept** are required for the sale to remember that he needs an apple.

3. An alternative situation is if a customer wants to buy other things instead of an apple. **Variable concept** is required. Instead of saving the word “apple” in the sale's mind, variable should be used so that whenever customers change their choice, the customer can save the product name in their mind.

4. Sale needs to calculate the price, so **arithmetic concept** is required. However, we need to know the price of apples, so we need the **database concept** to extract the price.

5. The customer needs to take out enough money from his wallet, which requires **arithmetic concept**.

6. If the money taken out is enough, finish the buying process. If not, customers need to take out the money from their wallets again. This part shows the **if-else concept**.

7. Extra tasks can also be set up, like if a customer wants to buy several things. This is a **looping concept**, and the user can understand some related concepts like the **range**.
concept on counting the loop and the break and continue concept.

By dividing the daily activities into multiple subtasks and extending the scale of the tasks, many basic concepts or even some advanced concepts like databases can be included in one task. This can test the user's understanding of each concept and the combination of them.

2.5 UI Design

The screen will be divided into two halves horizontally. The left part will show the programming result, while the right part is for the user to work on the block programming.

The left part will show the result of the block programming. To simplify the animation part, simple stop motion is preferred. Moreover, guidelines, like hints, a message box, and immediate reflection on whether the programming part is correct or not, will also be shown to show the progress of the game tasks or guide the user to fix some mistakes. As a result, users can understand how the programming concept was implemented before finishing the whole task or whether they have some misunderstandings about the concept.

The right part will be the block programming part. The user can grab the code block from different organized topics of concept, like logic and math. After pressing the submit button, those picked concepts will form an action for the character in the left part, and the user can check if they are correct or not.

2.6 Major Features

There are five features related to the game. They are Block Programming, Daily Life Activities, Programming Concept Control, Project Extension, and Learning Board.

1: Block Programming

Block programming is mainly used in lots of programming development tools or learning platforms. For example, Scratch, App Inventor, and Code.org are mentioned in the project background part. It makes use of the concept of “Drag and Drop”, which is by putting different code blocks together in order to form a complete instruction for the compiler to understand what needs to be done (Ryan, 2020 March 25). Compared to writing a complete program, block programming will be easier as the user can directly apply different programming concepts by simply dragging and dropping the code block.
needed to their program. This will be useful for this game project to reduce the difficulties of learning programming.

2: Daily Life Activities

In this game project, the topic of the tasks will be some daily activities. For example, buying things in a supermarket and ordering food in a restaurant. The reason for choosing daily life as the topic of the game is because these kinds of activities are very common in our daily lives. Therefore, students normally have those experience, which makes them easier to understand the target they need to finish through the tasks compared to complicated topics.

3: Programming Concept Control

Compared to other games, this project will not use WASD buttons on the keyboard or mouse clicks on the ground to control the movement or action of the character. Instead, users are required to combine different kinds of programming concept blocks to form some action in order to let the character move and finish the tasks.

4: Project Extension

As this project is designed into multiple tasks instead of a main task with a long storyline, extensions on the project scale, like adding new code blocks and tasks, can be easily implemented. The independence of tasks and code blocks is an advantage for the continuous development of the game, as the need for modification of existing material is minimized.

5: Learning Board

The learning board is a place for recording the user's learning progress. All concepts used in the finished tasks will be unlocked on the board. The user can track their learning progress in the game. Moreover, users can make use of the learning board for revisions on concepts.

2.7 Improvement on the game idea based on the existing platform

As mentioned in the Project Background part, there are some key points that can be learned from the existing learning platform, and this section will describe how those ideas can apply to the game implementation.

1: Clear grouping on coding block

The included concept list will be prepared by learning from some programming teaching sites for beginners to see how those concepts are grouped for later
implementation on the learning board and game interface.

2: Guideline or Tips through the tasks

As mentioned in the Gameplay Overall Description section in the Game Objective part, tips will be given when the user answers wrongly for them to adjust their answer. Moreover, a guideline button will be implemented in the game interface so the user can revise the concept immediately by explaining the concept to the user.

3: Balance on gameplay and educative element

Task design should be well organized to avoid problems like topic overlapping, which will lower the level of interest in playing some similar tasks, and a lack of certain concepts, which will affect the educational aspect.

4: Understandability on the tasks and programming concept

A more detailed task description should be provided to the user, and the difficulty of the tasks should be tested to ensure the difficulty is suitable for secondary or even primary school students. Stop motion animation and message box will be used to support the task description and help students understand what the tasks are about.

5: Suitable length of tasks

The task will be divided into several subtasks to shorten the concept needed in each subtask. Moreover, immediate guidelines and results are shown between subtasks for users to catch up on progress and learn the programming concept step by step.

2.8 Expected Learning Outcome

There are three expected learning outcomes for the user. First, the user can learn different kinds of programming concepts that will be used when creating a program. Second, the effect of different combinations of concepts. Third, problem solving skills for finishing different tasks.

3. Project Methodology

This game project will make use of Unity (https://unity.com/), which is a free game engine to develop due to my experience using this engine in game creation. For the programming language, this project will use C#, which is a preferred language for Unity, to avoid any conflict with Unity. The creation of animations such as message box interaction and stop motion will make use of the animation feature provided by Unity, which is enough for this project as the main focus will be the concept and also their interaction instead of the
animation part. The resources needed, like animation, effects, and audio, will make use of the internet resources instead of being created to reserve time for the programming part. Extra customization or creation may be needed if no required materials are found.

4. Budget Spending Prediction

As this game project makes use of Unity, there is no extra cost to the game engine. The possible areas to use the budget will be the resource parts like animation, objects (e.g., characters, background environments), and audio (e.g., button click sounds, background music) if no suitable resources are found on the internet.

5. Project Schedule and Milestones

Below is the different organized work needed to be done in this game project. Every work will have a brief description and the expected number of learning and working hours. Moreover, an expected project schedule will be shown in this section.

1. Programming concept list

A list of programming concepts is required for later task design and implementation. Also, those concepts should be well organized into different groups, which is more user-friendly for searching. The expected learning hour will be around 24 to 36 hours.

2. Required resources or customization tools

Different resources like stop-motion animation, characters, and audio will first be found on the internet. Customization or creation of the resource may be required based on the resources found. The expected learning and searching hours will be at least 40 hours, which will be affected by the design of the game tasks and the quality of the resources found.

3. Game task design and implementation

This section included designing each task’s topic and the content involved, task implementation, testing, and modification. The reason for putting design and implementation together is that content may change according to the actual implementation to fix some problems like overlap of topics or a lack of certain concept tasks. The expected learning hour will be at least 200 hours. The ratio of design and implementation should be around 3:7.

4. Graphical User Interface (GUI) design and implementation
This section included different parts of UI elements like the Start Game Interface, Menu, Learning Board, and Game Interface. The expected learning and working hours will be at least 50 hours.

### 5. User testing

This section is related to the evaluation of the gaming experience and concept implementation. There are some possible user groups. For example, primary and secondary school students, teachers, and university students whose major is related to teaching. Some important considerations, like whether the game interface is user-friendly, the game difficulty is suitable, the game content is interesting, and the game is educational, will be measured.

### 6. Other deliverables and preparation

- **First Presentation Preparation:** The expected working hour will be at least 40 hours
- **Final Presentation Preparation:** The expected working hour will be at least 60 hours
- **Interim Report:** The expected working hour will be at least 60 hours
- **Final Report:** The expected working hour will be at least 80 hours

**Expected time Schedule:**

September, 2023
- Programming Concept list
- Programming Concept List
- Required resources
- Game Task Design and Implementation

October, 2023
- Game Task & GUI Design and Implementation
- Game Task & GUI Design, Implementation and Testing
- First Presentation Preparation

November, 2023
- Game Task & GUI Design, Implementation and Testing
- First Presentation Preparation

December, 2023
It was noted that the game task & GUI design, implementation, and testing in December will mainly focus on the tasks that need to be shown in the first presentation and interim report. Therefore, the overall GUI structure and number of tasks should be finished for demonstration purposes.

6. Reference List


