Computing Project Documentation

Name: <Name>

Candidate Number: 3328

Centre Number: 10816

## Table of Contents

Table of Contents

[Table of Contents 2](#_Toc92707818)

[Description of Project 4](#_Toc92707819)

[Overview 4](#_Toc92707820)

[Nature of the problem 4](#_Toc92707821)

[End User 4](#_Toc92707822)

[Stakeholders 4](#_Toc92707823)

[Research on Existing Systems 4](#_Toc92707824)

[My Approach 4](#_Toc92707825)

[Alternative Methods 4](#_Toc92707826)

[Solution 4](#_Toc92707827)

[Limitations 4](#_Toc92707828)

[Investigation and Analysis 5](#_Toc92707829)

[Interview 1 5](#_Toc92707830)

[Summary of Interview: 5](#_Toc92707831)

[Hardware requirements 5](#_Toc92707832)

[Software Requirements 5](#_Toc92707833)

[Requirements Specification (MoSCoW) 5](#_Toc92707834)

[Approach to development 5](#_Toc92707835)

[Design 6](#_Toc92707836)

[User Interface Drawings 6](#_Toc92707837)

[Data structures and Variables 6](#_Toc92707838)

[Algorithms 6](#_Toc92707839)

[Test Plan 6](#_Toc92707840)

[Iteration 1 7](#_Toc92707841)

[Implementation 7](#_Toc92707842)

[My Evaluation 7](#_Toc92707843)

[Feedback from end user: 7](#_Toc92707844)

[Conclusions 7](#_Toc92707845)

[Iteration 2 8](#_Toc92707846)

[Implementation 8](#_Toc92707847)

[My Evaluation 8](#_Toc92707848)

[Feedback from end user: 8](#_Toc92707849)

[Conclusions 8](#_Toc92707850)

[Testing 9](#_Toc92707851)

[Evaluation 10](#_Toc92707852)

[Success Criteria Evaluation 10](#_Toc92707853)

[End User Evaluation 10](#_Toc92707854)

[Adaptive Maintenance 10](#_Toc92707855)

[Corrective Maintenance 10](#_Toc92707856)

[Extensions 10](#_Toc92707857)

[Bibliography 11](#_Toc92707858)

[Appendix 1 (code listing) 12](#_Toc92707859)

# Description of Project

## Overview

The current games in the market have a lot of elements and detailed graphics, allowing the user to immerse in the virtual environment. However, most of these games contain a lot of different mechanics, like different skill combinations, tactics and characters. These games often take hundreds of hours of playing and reading to learn and master these games. Hence, I have used the classic game Super Mario Bros as an inspiration and created a platform game which only has the necessary mechanics to minimise the learning curve of this game.

It is a platform game with a similar layout as the Super Mario Bros. Super Mario Bros is a platform game developed by Nintendo and first released in 1985. The player control Mario as they travel the Mushroom Kingdom to rescue Princess Toadstool.

Similarly, in this game, the player will control a character. The objective is to pass through a series of obstacles and enemies to reach the final destination and pull down the flag to complete the level.

It will have two levels with different maps and layouts. The map will be predetermined and imported locally. There will be a death-count counting the number of death the player has in that level.

## Nature of the problem

The well-known classic Super Mario Bros was created more than 30 years ago. It is no longer compatible with modern machines, and the game's visual quality is poor in the current standard. It will inherit a similar idea of a horizontal platform game.

The number of enemies in this game is reduced, but a large variety of traps are introduced to increase the dynamic of the game.

## End User

The users of the game are a group of teenage students ranging from the age of 16 to 25. These users have a shorter attention span. This game is straightforward, and the times needed for each level are relatively short, which suits the short attention span and busy life of the current adolescents. Although the basic idea and concept of the platform game are similar to the original Super Mario Bros, it has a different play style as the player needs to focus on different traps instead of different enemies. It can give a new taste of a similar game to users that have tried the Super Mario series.

## Stakeholders

## Research on Existing Systems

#### Super Mario Bros

It is a platform game. The character Mario is controlled by the player, who has to go through the Mushroom Kingdom. At each level, there are coins throughout the map that can be collected. There is also special gear that Mario can be collected, which will then give him special powers like invincibility, and shooting fireballs, et al. There are different enemies, including Goombas, who can be stepped on to defeat; Koopas, who are turtle-like creatures that can be bounced around; Browser, the boss of the game.

![Graphical user interface

Description automatically generated]()

#### Syobon Action (Cat Mario)

It is a parody of Super Mario Bros, created in 2007. The player controlling a cat must travel through a series of traps. The player has to memorise the traps and dodge them accordingly.

Diagram

Description automatically generated

#### Game Analysis

Both of the games have minimal graphics with little to no animations. It shows that the quality of animation or graphics is not necessary as long as the game mechanics are fun to play. Both games move the camera when the character leaves a specific boundary on the screen. In both games, the player can only increase or decrease the character's speed instead of directly changing the character's speed. Although it is more realistic to do it this way, the player will have a more challenging time picking it up. Both games have a life system in which the player can see how many lives they have left. Both games have a single jump system where double jumps are not allowed.

While Super Mario Bros has shorter levels, Syobon Action has longer levels with a checkpoint system where the player can respawn in the checkpoint after they die.

While Super Mario Bros is proven to be enjoyable to play, it has been out for so long that everyone has already gotten used to the play style. On the other hand, while Syobon Action has much more mechanics involved and much more features, the game is too complicated that it becomes frustrating to play.

## My Approach

I will use Python Module Pygame as the main framework for my game. It is a Python Module set designed for making video games in Python. I will use it to design the game and user interface. I will mix up existing tile maps for the game. I aimed to design a game that is easy to pick up but fun and challenging to play. The game mechanics will be as simple as possible, only moving horizontally, jumping and shooting horizontally. I will introduce some new enemies and traps. The game will contain long levels with checkpoints for the player to respawn.

The game will not have a life system where the player must start again after they die a certain amount of times. Instead, it will have a death system where the game will only record the number of deaths the players have but will not force a restart after a certain amount of death.

The game has a user interface where the player can access the control menu and all the levels.

## Alternative Methods

Other software and programming language can also be used to create this game. We can use C++ or C# for development through Unreal Engine and Unity, Java on jMonkeyEngine, JavaScript on PixiJS or Phaser.

However, after comparing many of the options, Python and Pygame are the most suitable. Although it is significantly slower than the other options, it is more straightforward and simple for a student to start with. Moreover, the game is uncomplicated, so the slowness of Python and Pygame will have limited effect. On the other hand, Python and Pygame are much easier to pick up and require less development experience.

Hence, Python and Pygame are chosen as the tools for this project.

## Solution

I will have a game loop containing the interaction between the user and the menu, instructions and buttons. I will check the coordinate of the mouse pointer whenever a click is detected. The coordinate will then be checked to see if it collides with any of the buttons

## Limitations

# Investigation and Analysis

## Interview 1

## Summary of Interview:

## Hardware requirements

## Software Requirements

## Requirements Specification (MoSCoW)

## Approach to development

# Design

## User Interface Drawings

## Data structures and Variables

## Algorithms

## Test Plan

# Iteration 1

## Implementation

## My Evaluation

## Feedback from end user:

## Conclusions

# Iteration 2

## Implementation

## My Evaluation

## Feedback from end user:

## Conclusions

# Testing

# Evaluation

## Success Criteria Evaluation

## End User Evaluation

## Adaptive Maintenance

## Corrective Maintenance

## Extensions

# Bibliography

# Appendix 1 (code listing)