**Super Lario – Coursework 4**

## Contributors’ Names

*‘The names and student numbers of all students who worked on the submission’*

* Person 1 – Fadi Mohamed Mostefai; K23160103
* Person 2 – Aldrich Antonio Fernandes; K23171918
* Person 3 – Mehdi Belhadj; K24054794
* Person 4 – Kishan Prakash; K23153494

## Description

### Overview

We created a platformer game called Super Lario, which took inspiration from Super Mario, that is primarily made up of vector graphics. The player has to navigate through 6 varying maps and collect coins and a key to pass through the locked door; avoiding all the enemies thwarting the player’s attempts throughout.

### Showcasing the game

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.Before entering the game, the user first sees the Title Screen, as shown in figures 1 and 2 below:

Where the user has the option to read about how the game works and its basic controls, from the menu bar, before and during play. The code implementation of this can be seen in the TitleScreen() and BaseScreen() classes.

*A screenshot of a video game

Description automatically generated*Following this we go onto the actual game:

*A screenshot of a video game

Description automatically generated*

*Figure 3 Figure 4*

The first and final maps always stays the same, with the maps in the middle varying from game to game. The following table best represent the features of our game:

|  |  |  |
| --- | --- | --- |
| **Object Picture** | **Object Name** | **Behaviour** |
|  | Player | The player has a set health which will deteriorate every time it hits a spike  It can move in any NESW direction through basic controls: ‘A,D + space bar’ or the arrow keys (←,→) |
|  | Static Spike | It is static – it deals damage to the player |
|  | Dynamic Spike | It is dynamic, it moves in a set space from left to right repeatedly – it deals damage to the player |
|  | Coin | They are static and multiple of them spawn in each map.  The coins act as the point system for the player – the more coins the higher their score |
|  | Key | The key is static and there is only one of them throughout all 5 maps. The player needs 10 coins to collect it to unlock the door at the end |
|  | Door | The door in the end room the user needs to pass through to finish the game |

### Additional Features

We implemented an information bar at the top of the screen (figure 5) that shows to the player their remaining health and time, as well as displaying their current FPS. The FPS display is a trait of many modern games and is often the most common excuse for bad gamers hence we tried implementing it as well.



*Figure 5*

We also added a translucent pause screen that overlays the game and pauses the timer when the user presses Esc on their keyboard:

A screenshot of a video game

Description automatically generated

*FIGURE 6*

A screenshot of a game

AI-generated content may be incorrect.We also an ending screen with varying quotes depending on the way you died, such as in figure 7

## Issues and Future Implementations

### Issues

One issue we encountered during development was the use of the Animation Timer class to run the game. Animation Timer runs the game at the monitor’s refresh rate, so the game would run drastically differently depending on what computer is being used. To fix this issue of inconsistent gaming, we used a delta time implementation, that instead looks at the difference in time between generated frames and this information is used to help with calculations to do with player movement. This allows for more consistent gameplay, across different computers.

Another issue we faced is collisions, specifically phasing through objects and getting out of bounds. Since the game is running at specific FPS, the game only updates, for example, 60 times a second. If a player were to hold down a key, such as spacebar, the player would move quicker than the game can check for collisions so the player can jump through terrain. We solved this by changing the collision method to check for what direction the collision is taking place and move the player away from the terrain; as well as making the method more efficient by only looping through the terrain tiles instead of all the tiles in the room.

### Future Implementations

In the future we were potentially thinking about implementing a save function feature which records all the movements the player made in the turn they save. We would then allow the player to play whilst having a replay of their saved movements appearing translucently in the background so it appears as if the player is playing against themselves.

Another implementation could be to make the game more roguelike by introducing an enemy character which chases after you, and some extra power-ups to escape from it. However, we would add this as an additional game mode with this game being the ‘Classic’ mode.