**Obstacle Game (Mario edition): Multithreading in Game Development**

**Introduction:**

A major problem that is faced by game developers is performance and user experience. This project is centered around a multithreaded obstacle game that addresses the traditional issue faced by these developers. This project uses multithreading as a programming technique that enhances multiple threads. This report will cover the design, development, test phases, and challenges and concludes by discussing future improvements.

**Solution approach:**

The game uses Pygame, a game development framework for Python, to handle graphics and user input. This project also incorporates other programming techniques such as object-oriented programming (OOP) as well as other libraries such as DateTime, Numpy, Image, and random.

Player:

The player class represents the main character of the game. This class is implemented using OOP. It includes the speed and the score which are initialized to zero. The update method handles the movement based on the player's input and is called each frame. These are specific keys on the keyboard that have been stated using the key.get\_pressed() function. The player can move up, down left, and right. The update method also handles cases where the player exceeds the boundaries of the game area.

**Below is a snippet of the player class :**

Text

Description automatically generated

Obstacles:

The obstacle class represents the objects that the player must avoid in the game. It contains a list of images to represent different obstacles. The images are randomly chosen by the init method. The position and the speed of the obstacles on the screen are randomized with the aim of increasing the difficulty level. The update method moves the obstacle(s) to the left by the randomly generated speed. Upon reaching the left end of the screen the obstacle is killed

**Below is a snippet of the obstacle class:**

Text

Description automatically generated

Main Game:

The obstacle game has the famous video game, Mario, theme. This game uses multithreading to run separate processes simultaneously- one is used to continuously change the background. And the other is to spawn the obstacles to be avoided by the player.

The background thread is started at the beginning of the program. This thread runs the “update\_background” function. The function loops between a selected amount of background images. The obstacle spawning thread is also started at the beginning of the program and runs the “spawn\_obstacle” function, this function continuously spawns obstacles at random intervals. Within the function, the number of obstacles to spawn is randomly generated.

By running the two processes in separate threads, the game can continue to update the background and spawn obstacles without being slowed down.

The main game loop listens to the user's input, updates the player's position, and scores them accordingly. It also checks for collisions with the spawning obstacles and coins. Additionally, it updates and displays the background image and score text.

**Experiment**

In the first experiment, I tested the effect of changing the chance of getting coins on the game’s difficulty level. This was done by increasing the probability of getting a coin within the 10% to 50% range. I found that 20% yielded an adequate difficulty level while providing the user with an enjoyable experience.

**Below are images with various probabilities from 10% to 50% (left to right)** 



To investigate the impact of the obstacle frequency on player engagement and difficulty level, an experiment was conducted with varying frequencies from 1 to 5. The result showed that an increased frequency beyond 3 was too difficult given other factors such as the size of the window. A frequency maximum of 3 increased the payer engagement and yielded an adequate difficulty level.

**Below are images of various frequencies from 1 to 5:**

A picture containing background pattern

Description automatically generatedA screenshot of a video game

Description automatically generated with medium confidenceA screenshot of a video game

Description automatically generated

**Challenges:**

During the implementation phase of my project, one challenge I faced was using the pygame framework as I had never used it before. It was challenging to figure out how to use it properly. I main issue I had with this framework was implementing the game loop. The game would enter an infinite loop instead of exiting when the player lost the game.

To overcome this challenge, I spent a lot of time watching tutorials and experimenting with different solutions. After several trials, I was able to identify the root of the problem. I declared a variable in the wrong part of my code. while it was exiting one loop if didn’t exit the other. I modified the loop to incorporate proper exit conditions which helped in resolving the issue.

Although I was able to solve the major issue I encountered, there were other challenges that I could not address due to time constraints such as implementing a start and a play again button. I was trying to incorporate this while still having issues with exiting the while loop which made it more difficult, so I discarded this and focused on the more pressing issue which was exiting the while loop.

One of the unique features I would say that my project has is the incorporation of numerous obstacle images and the changes in the background. This keeps the player engaged. These features add variety and challenge to the game. Which makes it more interesting and enjoyable for the player.

**Below are images of the obstacles and background :**Graphical user interface, application

Description automatically generatedA picture containing text

Description automatically generated

**Conclusion:**

This project has helped me to understand the importance of threading in game development to improve user experience by handling multiple tasks simultaneously. I was also able to learn about game development.

Moving forward, I plan to incorporate game levels that increase in difficulty, providing a greater challenge and increasing the player's engagement.

**References:**

* "Python Documentation", Python Software Foundation, <https://www.python.org/doc/>
* "Pygame Documentation", Pygame Community, <https://www.pygame.org/docs/>