Daniel Martin

S2025604

Introduction to Games Programming

I confirm that the code contained in this file (other than that provided or authorised) is all my own work and has not been submitted elsewhere in fulfilment of this or any other award.

Signature. *Daniel Martin*

Contents

[Description of Code](#_hyobctuq96wm)

[Challenging Piece Of Code](#_9vruzdtayhtq)

# Description of Code

Ball

The Ball script contained everything from the balls movement and collision to calculating lives, score and the win/lose conditions.

“case "Brick":

directionY = -directionY; // Invert the direction horizontally

Destroy(other.gameObject);

IncreaseScore(score);”

Bat

The bat script contained the code for the movement and collision of the bat, allowing it to be used by the user using the arrow keys.

“bool isLeftPressed = Input.GetKey(moveLeftKey);

bool isRightPressed = Input.GetKey(moveRightKey);

if (isLeftPressed && canMoveLeft)

{

direction = -1.0f;

}"

# Challenging Piece Of Code

Implementing the ball resetting after hitting the bottom was the most difficult piece of code I found to try and implement. I struggled to find a method of despawning the ball and spawning in a new one after it hits the bottom so switched up my approach to it, instead of despawning and spawning a new ball I would instead just change the position of the ball to the centre of the level. Though this worked in theory you had no time to react as the ball would hit the bottom, reposition to the centre and fly to the left or right before you could catch up. Instead opted for the ball to set its position to the same as the bat.

After finding the code I needed to set a game objects position to the same as another game object using transform.position. Using this I made the ball teleport to the bat after hitting the bottom, this created an issue as the ball would spawn inside the bat and could sometimes fall through, it also gave you 0 time to react. My final solution to this problem was to create a new child game object under the bat, positioning it a couple units above the bat. The game object would follow the bat as it moves back and forth, just a couple of units above it, using this as the game object my ball would teleport to, the problem was solved. Since the game object was just slightly above the bat it would give the ball travel time before hitting the bat, giving the player time to react and allow them to predict where it will hit.





