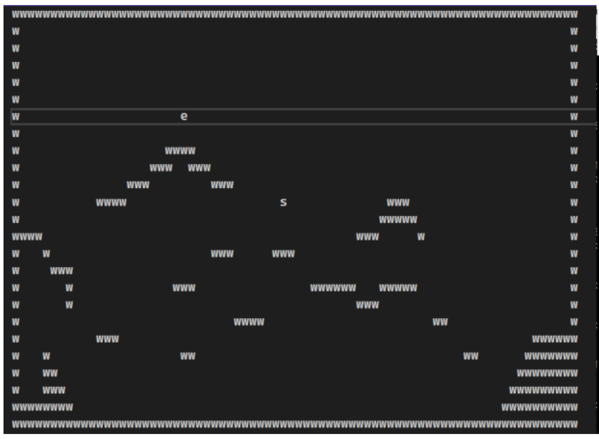
Games Engineering Report

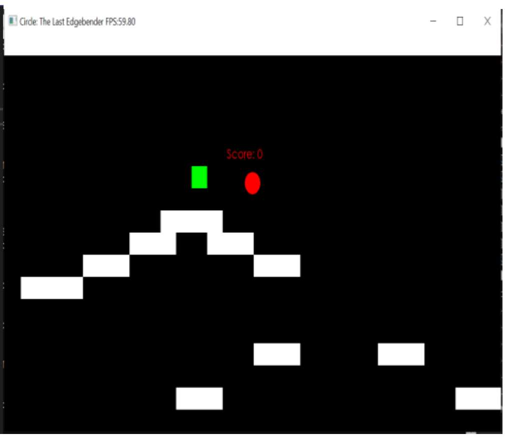
Daniel Seavers, Jack Millar and Maja Heltowska

1. The assignment we were given was to create a game, this game needing to be of a playable state by a stranger coming into it with no experience of how it worked or what it entailed, from this we decided to create a game where the user would control a physics based ball that would move around the stage through controlled bounces, and combat enemies generated in the stage. The inspirations for this came from horde based games like vampire survivors and blockhead and the idea of using simplistic graphics came from games such as diep.io and agar.io.
2. In our GDD we had planned to have more levels available to the player than we did in the end, however due to the complexity of creating the level code repeatedly for new levels this did not end up feeling worth the time and thus was left out. The task of implementing a boss did not feel worth the required effort after making the rest of the game as we felt it would not add much and would cause far more complications. While the current implementation of the game is endless, the player cycles through a repeating set of three rooms. We were not able to add co-operative play to the game and felt that it would take away from the enjoyment the main player would have while playing. In the end we decided to not have the player able to shoot the enemies as between time constraints and difficulty implementing such systems, we in the end opted for bounces to be the sole source of damage. Reducing the number of edges the circle had and eventually turning it into a square, while a cool idea was impractical for the project scope. The speed of the player is inconsistent as is to be expected with a physics object thus at times it moves far slower than a triangle and at times it moves far faster. Squares were not implemented as the enemies ended up taking far too long to implement for it to be worth adding variety. The wave effect is invisible but triggers every time the player lands in the current implementation. Pentagons have also not been implemented at this time. Health and damage values have been changed since the original plan and are still being tuned as I write this report. There is no squangle or other shooting enemy currently. Due to a bug causing enemies to spawn in walls we currently open the exit on its own after 2 seconds, but if this bug is fixed it will be changed to when all enemies die. The player uses arrow keys and the space bar for ball control. Currently enemies do not pass through walls. There is no in game story currently. Currently there are no ui buttons implemented in the game and the title is displayed in a standard font. There is currently only a plain black background, and the platforms are plain white rectangles, the doors are green squares. None of the planned animations have been implemented. The game does not currently use any sound effects. The game runs at over 2000 fps when uncapped so should run on most devices. The game may not be complete to a state where it can be uploaded to itch.
3. The Enemy State Diagram at Appendix 1 shows how the enemy AI works and the states it uses. The enemy starts in its stationary state, then if the player is in range of the enemy, the enemy will enter the seek state and move towards the player and when it gets to closer distance it will enter the flee state which makes the enemy stop at a close distance to the player, it was planned to stop at a close distance and shoot at the player, with more time that would have been implemented. The states are managed by the state machine component implemented in the game.

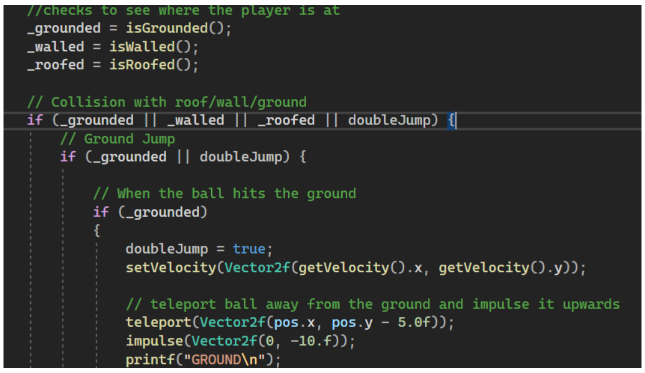
4. The Game was implemented using the C++ programming language on the Visual Studio 2022 IDE. Initially we started by creating a basic level design, using a txt file to hold certain characters that would represent sprites within the game. E represented Exit, S represented Player Spawn, W represented Wall.



```
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
w                                                                         w
w                                                                         w
w                                                                         w
w                                                                         w
w                                                                         w
w                     e                                                   w
w                                                                         w
w                   wwww                                                  w
w                 www  www                                                w
w              www        www                                             w
w          wwww                    s             www                      w
w                                               wwwww                     w
wwww                                         www     w                    w
w   w                     www     www                                     w
w    www                                                                  w
w      w             www               wwwwww   wwwww                     w
w      w                                     www                          w
w                            wwww                      ww                 w
w          www                                                       wwwwww
w   w                 ww                                   ww       wwwwwww
w   ww                                                             wwwwwwww
w   www                                                           wwwwwwwww
wwwwwwww                                                         wwwwwwwwww
wwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwwww
```

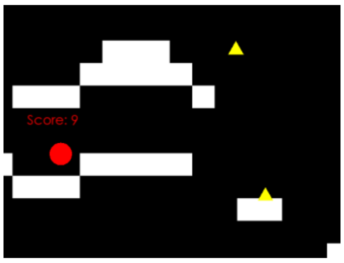


For the Player, we gave it the functionality to bounce against other sprites, mimicking the movement of a red bouncy ball.

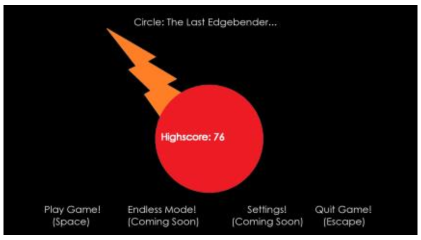


```
//checks to see where the player is at
_grounded = isGrounded();
_walled = isWalled();
_roofed = isRoofed();

// Collision with roof/wall/ground
if (_grounded || _walled || _roofed || doubleJump) {
    // Ground Jump
    if (_grounded || doubleJump) {

        // When the ball hits the ground
        if (_grounded)
        {
            doubleJump = true;
            setVelocity(Vector2f(getVelocity().x, getVelocity().y));

            // teleport ball away from the ground and impulse it upwards
            teleport(Vector2f(pos.x, pos.y - 5.0f));
            impulse(Vector2f(0, -10.f));
            printf("GROUND\n");
```

For the enemies, we implemented a yellow triangle that would fly towards the player, upon hitting the player it would die and cause some damage.



We implemented a main menu screen that would hold the players high score as well as provide as a way to play the game or exit the game.



Finally, we have an exit door that leads to the next levels and its implemented so that when the last level is completed it, the game loops through to the first level and scales enemies until the player dies, upon death, it will take the player back to the main menu screen with their score (if it beat their current high score).

5. The game is on the right track to match the original plan. The player bounces around the room as was described in the plan and the enemies follow the player to attack. Not all the characters were implemented but with more time they would have appeared. The player gained a double jump ability through the process which makes moving around the level more interesting. We did not have time to include the sound effects or animations from the original plan. The aim of the game is still the same, to survive and escape. The level design is also more slightly more complex and maze like than originally planned to make the game more engaging for the player. We also changed the player controls to arrow keys instead of WASD but kept the spacebar for jump movement.
   It is not finished so it would be unfair to truly compare it to fully finished games however we were inspired by games such as diep.io for its minimalistic design and our game does resemble that style with simple colourful shapes for the player character and the enemies. It doesn't have the attack and shooting like diep.io but with more time it could have the potential to be more similar. Our game does have some resemblance to the horde like fighting and survival we were inspired by from games such as Vampire Survivors, since there are many triangles attacking the player who needs to survive the level, this does need much more development with the enemy attacks and spawning.

   The game is unfinished but it's a good start for more serious development. It's a good start since the player and enemies are established as well as the level design and the game engine which can be worked from. The player movement is quite good for this early part of the game development. The enemy movement could be more refined as well as the attacks and combat in the game. The game does still have some bugs, for example some of the enemies spawning outside the walls of the level which requires fixing so the game definitely requires further development, but the main idea is present. The game is quite short since it only has two levels however due to the slight complexity of the levels it keeps the player focused on completing the two levels.

   There are many improvements that could be done with more time which would lead to more varied characters being added such as square enemies, squangles and pentagons that we had in our design, to develop the horde fighting survival physics game we had in mind. We also could include instructions for the player controls so that new players can be able to play the game without getting too confused. More defined attacks for both the player and enemies which would then lead to better scoring, some of the sound effects and possibly animations that were planned could be added too. We had ideas for some player power ups that could be implemented in a further stage of development as well as more levels with more complex designs to keep the player more engaged with the game.

6. We used the SFML library (SFML, n.d.) for the graphics of the game, and it would also be used for implementing the sound effects into the game. We also used the Box2D physics engine library (Box2D, n.d.) for the physics in the game.

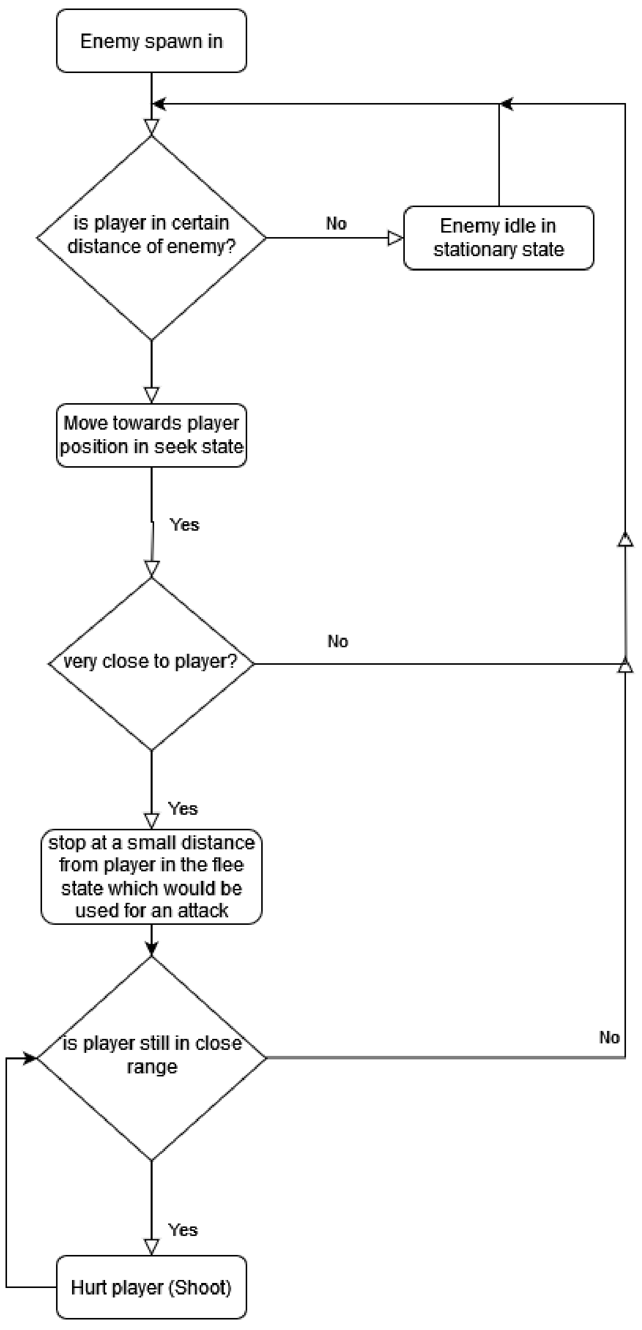
References:

SFML. (n.d.). https://www.sfml-dev.org/

Box2D. (n.d.). https://box2d.org/

Appendix

Appendix1:


Enemy spawn in
is player in certain distance of enemy?
No
Enemy idle in stationary state
Move towards player position in seek state
Yes
very close to player?
No
Yes
stop at a small distance from player in the flee state which would be used for an attack
is player still in close range
No
Yes
Hurt player (Shoot)