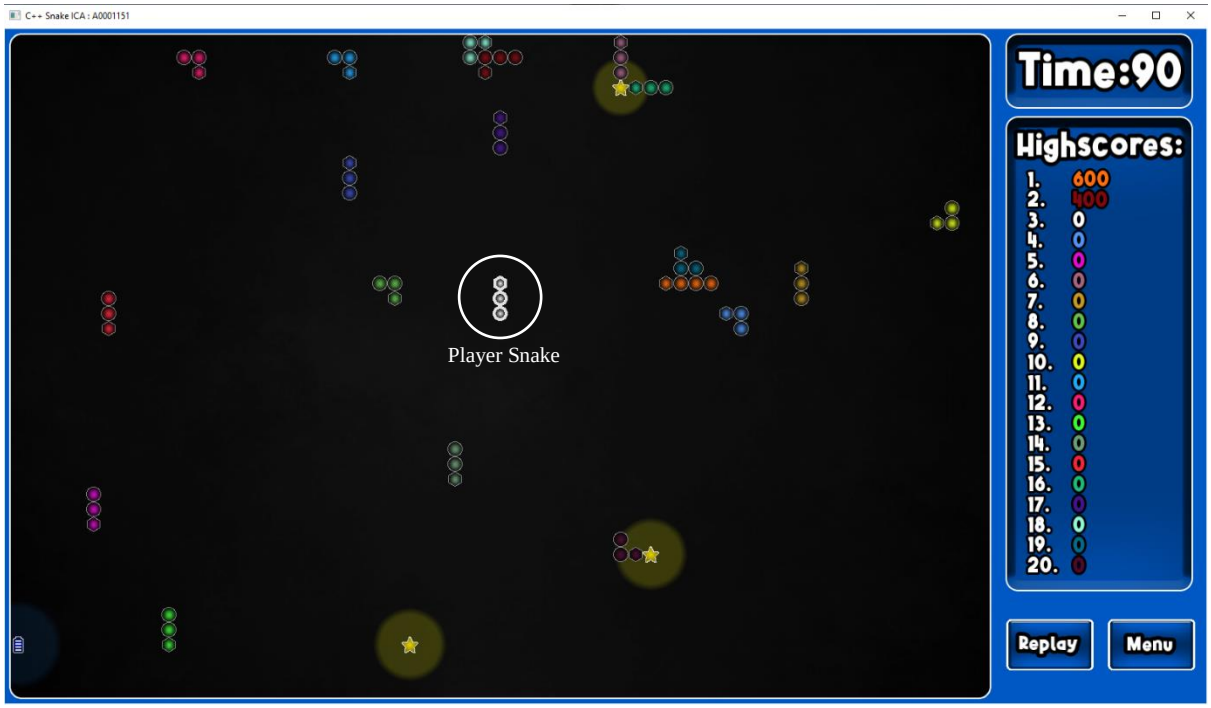
Games Development with C++

Electric Snakes ICA

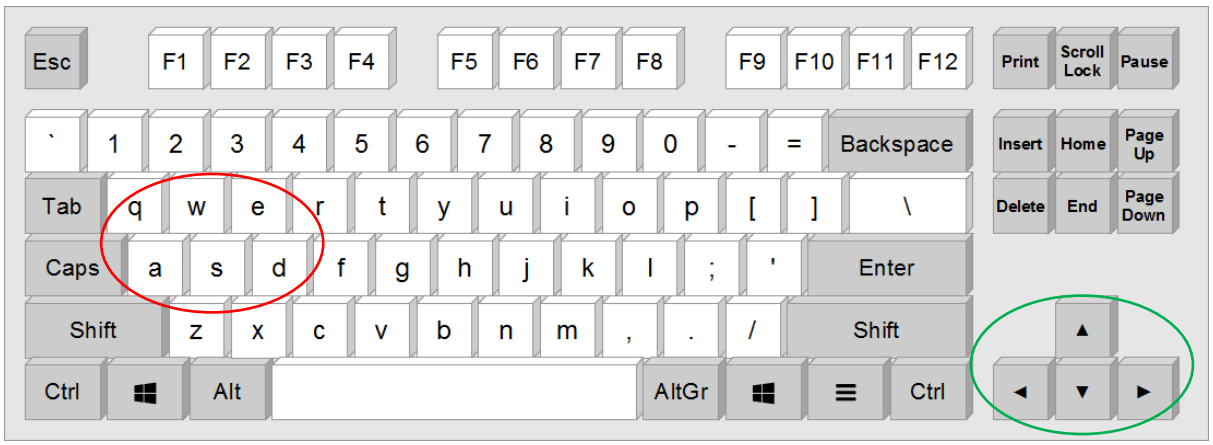
Ivan Razvan – A0001151

1. User Guide

The player controls a white snake that always spawns in the middle of the game field. The player will have to face 19 AI snakes in a race to collect as many stars, in order to hold the highest score until the end of the countdown.



The player can control the snake's direction by using either the WASD keys or the arrow keys.

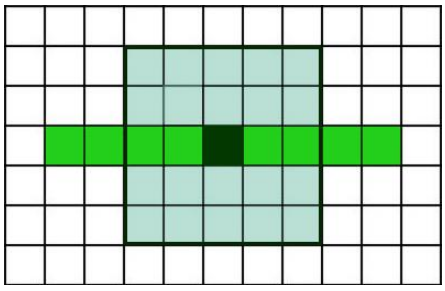


2. Problems Encountered and Solutions

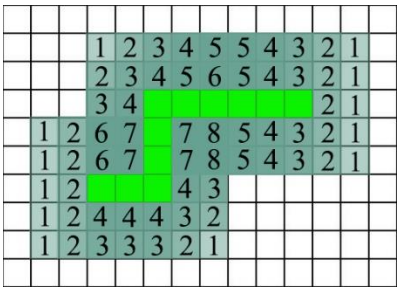
Throughout the entire coding process, I encountered many problems that I was unfamiliar with. Although I managed to solve a portion of them through research alone, there were a few more intricate ones that stood out from the rest:

2.1 Handling the collision with an electrified snake

My first implementation was brute force, and consisted of looping through every single segment of an electrified snake, then checking if any other snake was in the 5x5 square centered in said segment. This implementation used 5 nested for loops, raising the complexity to $O(n^5)$. Another flaw was that it checked the same positions numerous times while looping through the entire snake.

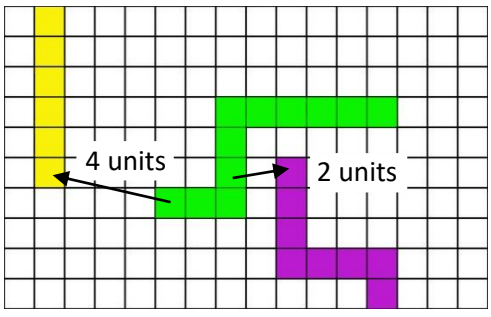


(1 iteration)



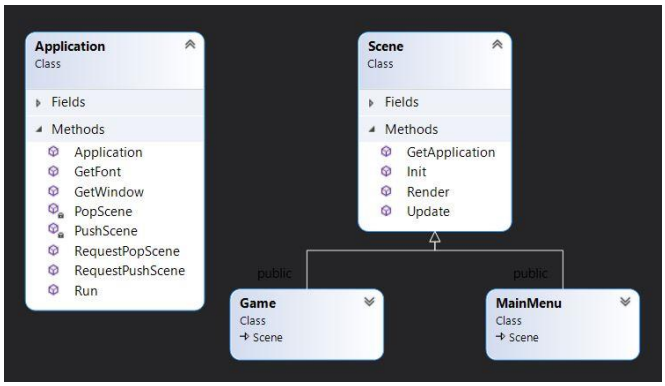
(all iterations)

My final implementation uses the Manhattan Distance to find which snakes are at 2 or less units away. The complexity is now reduced to $O(n^2)$ and the code is much easier to read and understand.



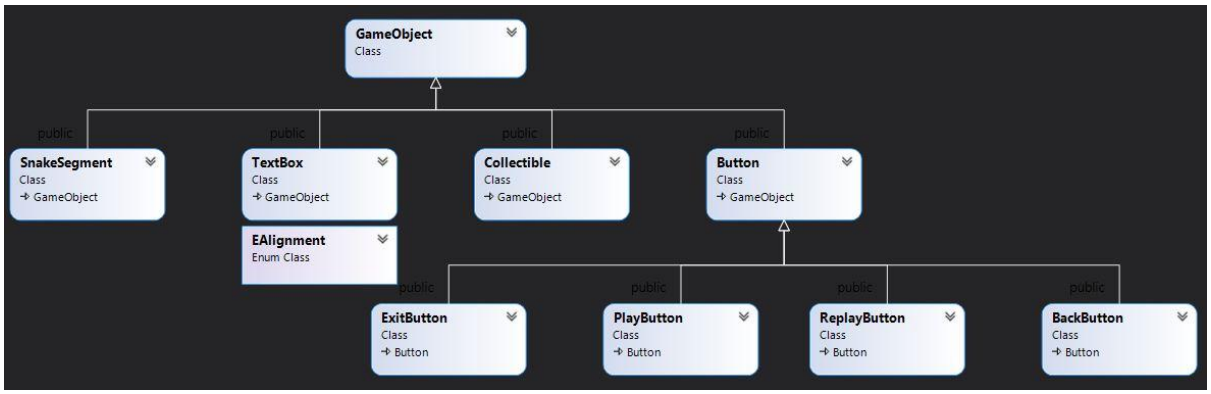
2.2 Handling the scenes

After finishing the logic behind the snake game, I wanted to implement a menu screen. In turn, I needed to design a class that would handle all of my scenes. In the end, I came up with the idea of having the class use a stack in order to update and render the scene found at the top of it. This way, when jumping to a new screen, the program would push the scene onto the stack. By the same logic, when returning to a previous screen, the program would pop the scene on top of the stack.



3. Reflections

While working on this project, I gained a better understanding on how to develop a larger program and how classes should be implemented to achieve this scope. Inheritance played a vital role when it came to creating and storing objects with similar behaviour.



Besides the usage of classes, I have also practiced using several data structures and algorithms. Along the previously mentioned scene stack, I used a linked list to achieve the snake's movement, instead of storing each segment in a vector and changing their positions on every step.

In conclusion, when it comes to future projects, I will approach the entire process in a different manner. After seeing the amount of problems I have encountered along the way, I finally realised the importance of preparing a good model before any coding. Having this in mind, I will start my next projects by creating more diagrams that would help me avoid unnecessary refactoring.

4. References

Font used: https://www.dafont.com/alphakind.font

Music used: https://youtu.be/4Yn174EAEwk