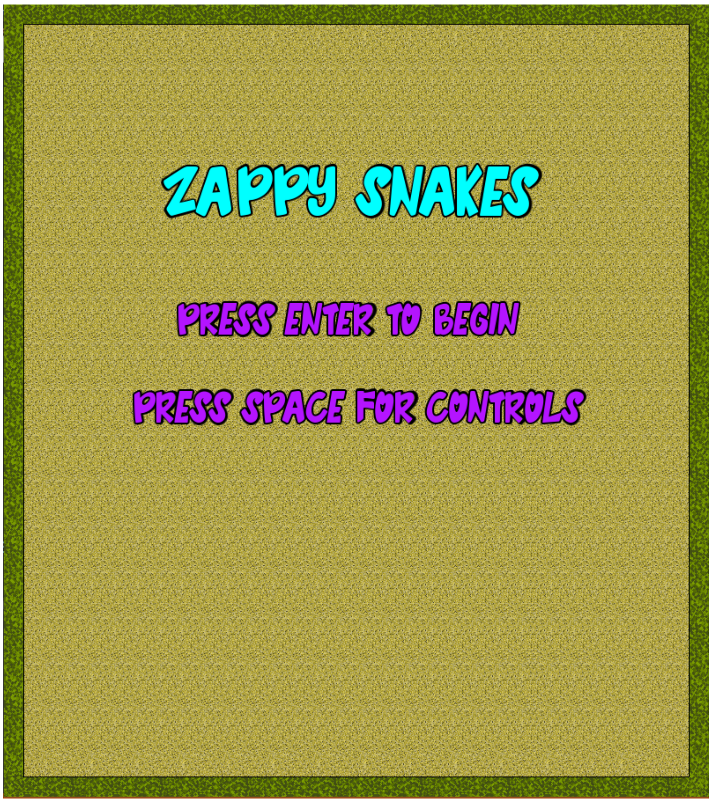


ZAPPY SNAKES

PRESS ENTER TO BEGIN

PRESS SPACE FOR CONTROLS

ELECTRIC SNAKE

GAME REPORT

Games Development with C++ ICA

ZAPPY SNAKES

A short report covering Issues I encountered, how they were solved, and what I would do next time.

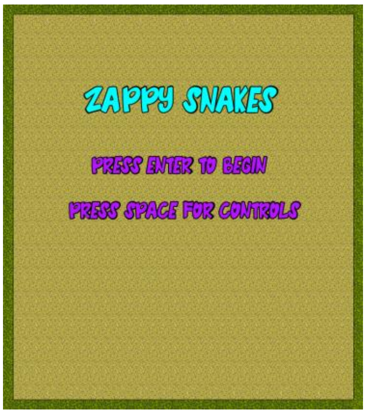
SENIOR, COLETTE (Student)
A0208468

Please note: All Textures within the game were made by myself using GIMP. The font that I used was free to download and use for personal use was found here https://www.dafont.com/waffle-crisp.font

Game Controls:

My game, titled “Zappy Snakes”, has very simple controls that are outlined to the player. Thanks to my Menu, I have a screen showing players the controls, which are just simple arrow key movements. Below, I have created a table to quickly demonstrate the key presses required to navigate the menu and game, most of which are displayed to the player.

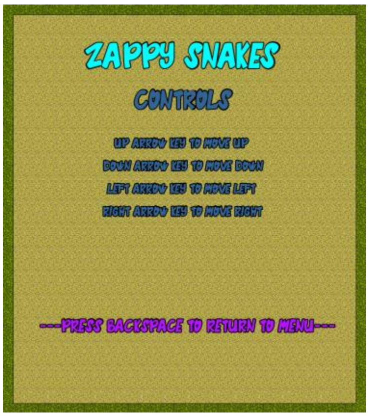
Where	What Key	What it does	Do players know?
Main Menu	Enter	Begins the Game	Yes
Main Menu	Space	Shows player Controls	Yes
Controls Menu	Backspace	Returns player to Main Menu	Yes
Game	Up Arrow, Down Arrow, Left Arrow, Right Arrow.	Changes player snake’s direction	Yes
Game Over Screen	Tab	Returns player to Main Menu	Yes
Main Menu	Esc (Escape)	Closes the game entirely	No



ZAPPY SNAKES

PRESS ENTER TO BEGIN

PRESS SPACE FOR CONTROLS



ZAPPY SNAKES

CONTROLS

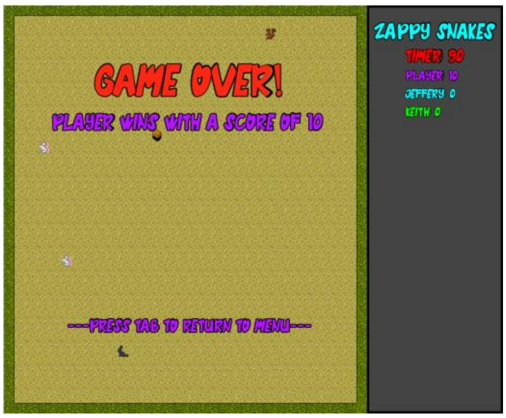
UP ARROW KEY TO MOVE UP

DOWN ARROW KEY TO MOVE DOWN

LEFT ARROW KEY TO MOVE LEFT

RIGHT ARROW KEY TO MOVE RIGHT

---PRESS BACKSPACE TO RETURN TO MENU---



GAME OVER!

PLAYER WINS WITH A SCORE OF 10

---PRESS TAB TO RETURN TO MENU---

ZAPPY SNAKES

TIMER 90

PLAYER 10

JEFFERY 0

KEITH 0

Bugs & other issues…

Sadly, I did not give myself enough time to work on this game as much as I would have liked. However, despite coming across many compiler issues during my time building this game, I am happy to say that I have no compiler errors at the end of my build.

Most of my issues encountered were the silly little mistakes; with spelling errors, not enough brackets, missing semi colons, and the occasional missing header file. I also found that when building in C++, and probably in any coding language, ordering is key to a functioning game. For example, the importance of drawing to the SFML window AFTER you have cleared the window in the loop very quickly became clear to me when coding this game. I had a lot of issues with getting my collisions between snakes to work correctly. I often found that my player snake would die to my Ai snake, but my Ai snake would not die to my player snake. After a lot of time thinking, and discussing with others, the key here was to check and return if both snakes were alive before initiating the death sequence, and not during.

Unfortunately, there were several bugs that I found that I simply did not have enough time to iron out. The remaining issues that I have found whilst testing my game are shown in the table below.

Bug	How does it affect the game	Potential way to fix
Battery effect doesn't always kill other snakes straight away.	It means that when a snake should die, it doesn't so does lead to inaccurate gameplay.	I would need time to really focus on my battery effect collision. It may be more beneficial to find a new way to manage this collision entirely.
Ai Snakes collide with the walls instead of avoiding them.	Often leads to shorter and easier gameplay as the Ai snakes often die rather quickly due to the randomised movement.	Again, looking closer at the collisions. A potential but rather long-winded fix would be to hard code the collision to move a specific way when it's about to collide with the walls.
Ai Snakes rarely collide on themselves without reason to, but it still occurs.	Can lead to shorter gameplay. Also, it demonstrates how unintelligent the Ai Snakes are.	This was something I had a big issue with and took a while to fix. My code for this now is much stronger than what it was, but it does still occur.

Reflection Time…

From start to where I have gotten to with building Zappy Snakes, I have really enjoyed my journey. I've learnt how to construct a simple 2D game using SFML. My knowledge and skills with classes especially have gotten stronger, and I feel very comfortable when working with classes and inheritance. I have learnt to take more breaks when struggling to fix errors, as every time I come back with a fresh mind, I have a new idea to try, or have mentally solved my issue. I believe I overshot myself with adding some extra features. Some of these features remain incomplete purely because I did not allow myself the right amount of time. Below Is a short list of what I desired to finish but ran out of time for.

- Snake Textures: Sadly, I ran out of time, and was unable to solve the puzzle of how to change the snake texture, one segment at a time. Instead, the entire snake's body textures will change direction based on the direction the snake head is moving. I did begin looking at using a struct to help with this, but I needed to polish off other elements before the deadline.
- Ai Snakes: Despite having Ai snakes, they're not very clever. They will move at random and not care if there is food nearby or not. I did want to smooth out my Ai movement to allow them to hunt for food and batteries, but again I had little time left and chose to focus on the more important elements to the game.
- Different Snakes: I did wish to change the colours of the snakes, but with my added textures, I found it rather difficult to determine the colour of each one individually.