**Assignment 1 A.I.**

**Content**

Introduction

Technique used

Screenshots

What went well.

What did not go well.

What I can do to improve.

Why a star algorithm was the best choice.

Why Finite state machine was the best choice.

References

**Introduction**

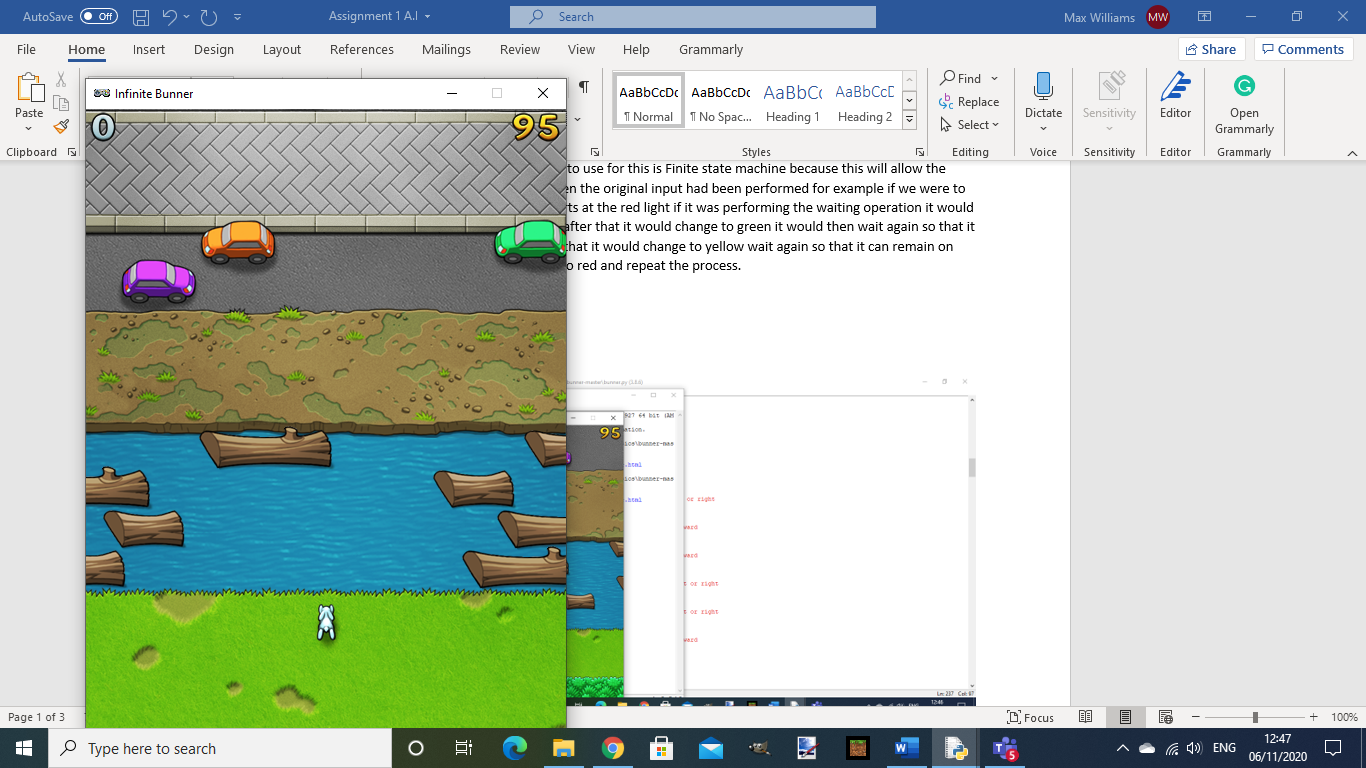
For this assignment I was tasked to download any one of the five choices of games that were on github and we were tasked to try and create an A.I for one of them that would allow them to successfully complete a task that’s required for the game for example in the Bunner game the task is to create an A.I that will allow the player to go the furthest distance in the game. With the Mayripod game we are required to make the AI try and get the highest score in the game and with the soccer game we are required to make the AI try and beat the CPU players.

**Technique used**

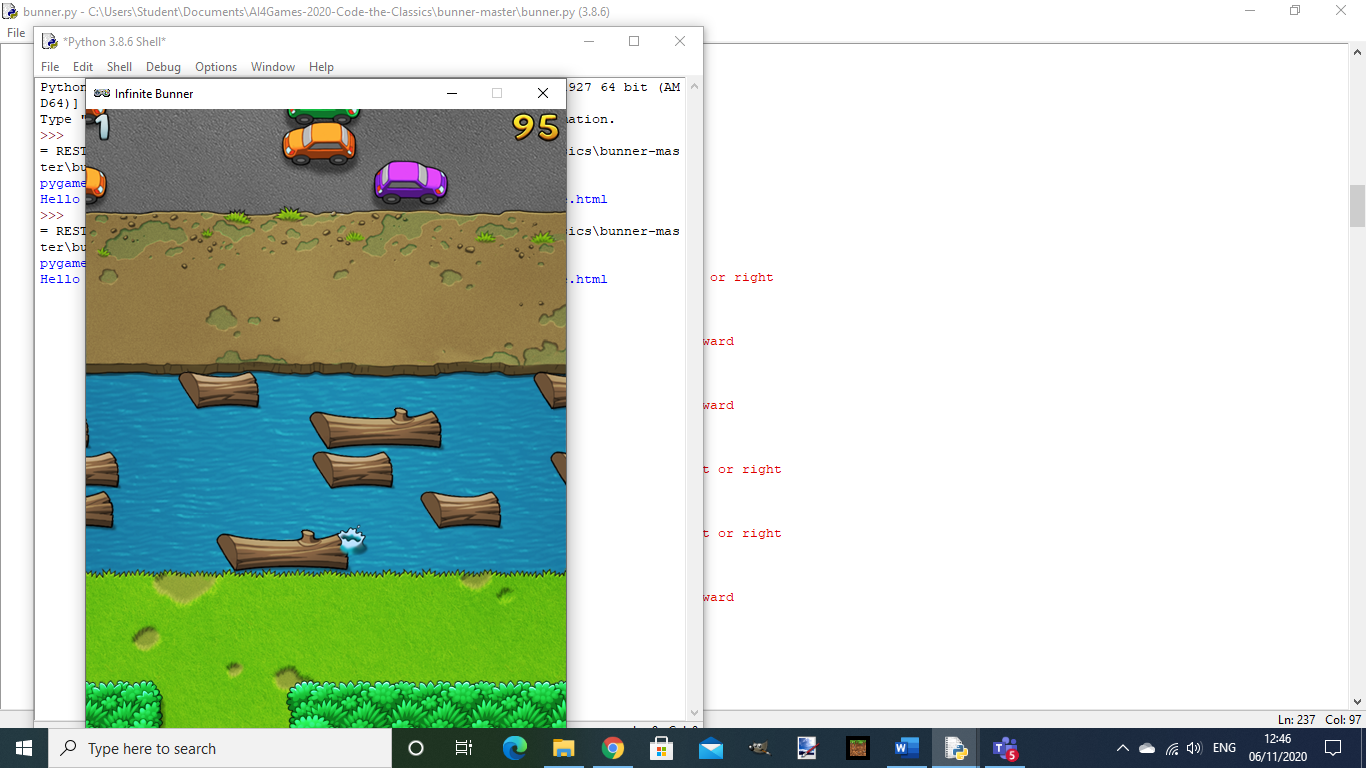
The technique I wanted to use was the A star algorithm technique because since it is used for a program to navigate through a maze, I thought it would be the best solution for the program. The way how A star algorithm works is that the program navigates through a maze and detects any obstacles that are in its path, when it does find an obstacle in the path it will try to avoid it by changing its direction. One example you would use a\* algorithm is on a sat nav because what it would do it would find the shortest route on the journey you are taking to get to your destination. If the program follows a path that it had already gone through however it will recognise this and instead go a completely different route that it has not already gone through. Also, if it was navigating through a maze and it finds a dead end through the maze it will go back to its last direction that it went through and change its direction to a new pathway. Finally, if the maze has more than one route you can use the A star algorithm to find the shortest route there is.

The other technique that I need to use for this is Finite state machine because this will allow the program to change an input when the original input had been performed for example if we were to look at a traffic light when it starts at the red light if it was performing the waiting operation it would continue on being red but then after that it would change to green it would then wait again so that it can remain on green then after that it would change to yellow wait again so that it can remain on yellow and then finally go back to red and repeat the process. Another example of a Finite state machine would be with a light switch being turned on and off the way how that would work is that the switch it might start off as just being switched off and would continue to switch off performing the waiting operation however when a person turns the switch on it would then change to the next operation which is of course switching the light on after that it would perform the waiting operation again until the light is switched off again.

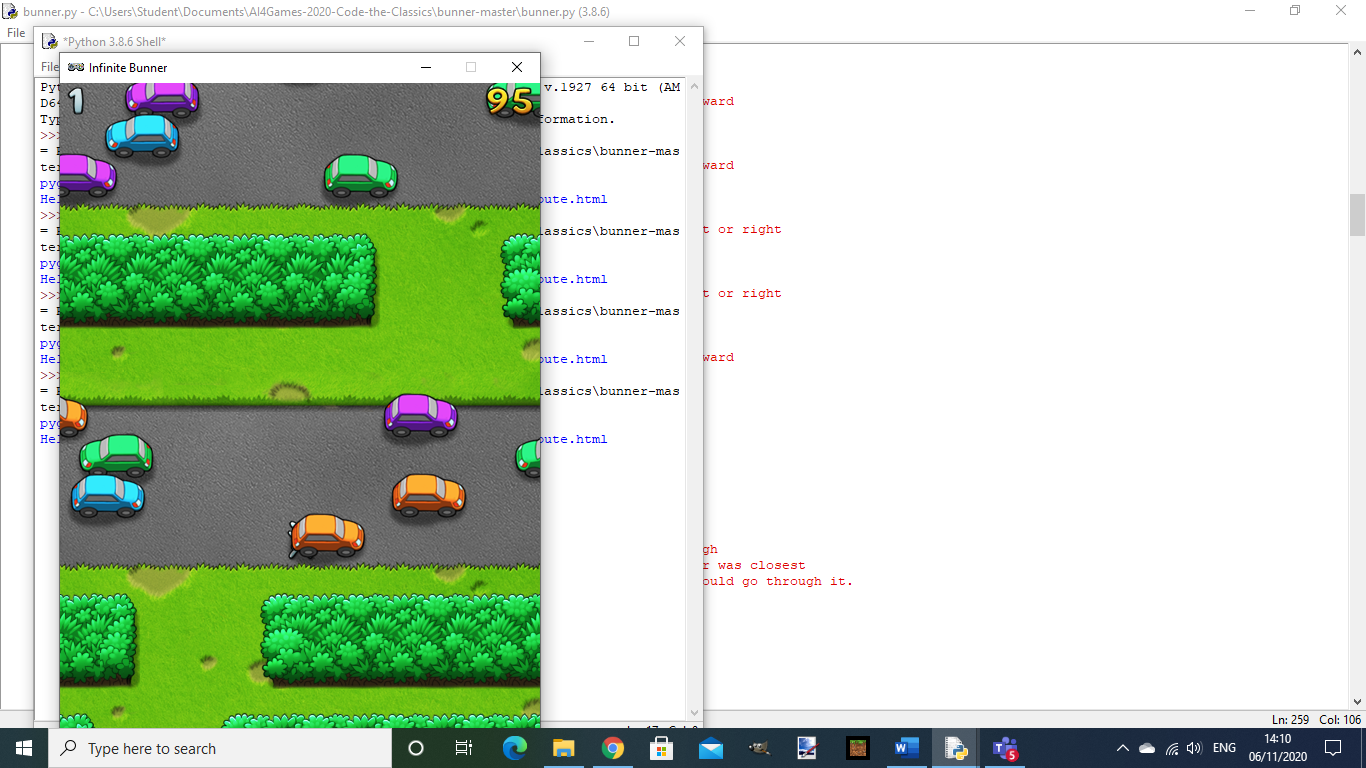
**Screenshots**



The screen shot here is of the actual game itself. You might not be able to tell but I am not pressing any of the keys on the keyboard and yet the rabbit is moving at complete random.



This screen shot shows of the rabbit falling into the water because I unfortunately was not able to make a function which would have allowed the rabbit to have crossed the river using the logs.



This screen shot shoes of the rabbit getting hit by a car because like the river issue I unfortunately was not able to make a function which would have allowed the rabbit to avoid the cars when they passed.

**What went well.**

What I feel went well was I was able to get my rabbit to move all by itself in random directions without the need of using the controls as well as the rabbit moving forward I was able to also allow it to move randomly. Also, I was able to allow it to move forward all by itself without the need of any input from the keyboard.

**What did not go well**

What I feel did not go well was I unfortunately was not able to input any detection for the bunny rabbit so I could not have him avoid any of the hazards that were littered across the stage as well as having the ability to navigate through the maze.

With regards to avoiding the hazards that are littered around the stage I planned to put them in a finite state machine function so that it could tell the program that if a hazard was right in front of them they would automatically stop in position or just move either left or right trying to avoid the hazard. I would have also used this technique for allowing the rabbit to cross the river using logs that are right in front of it.

With the navigating through the maze section I planned to make a function that would have been used to make the a star algorithm since that technique is used to navigate through mazes trying to find the shortest route so I would have used this so the AI would have tried to find the shortest route towards the entrance to the maze.

**What I can do to improve**

What I feel I could do to improve for next time is I could have tried to do some more practice with python because before hand I thought I had a good knowledge of Python so I was confident I’d be able to do this easily but unfortunately I did not have the knowledge I thought I had for python so because of this I struggled a lot trying to actually do the assignment itself since every time I tried to do something new I would always cause my program to crash.

**Why A star algorithm was the best choice**

The reason why a star algorithm was the best choice for this program was because since it is normally used for navigating through mazes from the start to the finish so by using this technique in the bunner code you can use it to detect if there is a hazard in front of you or on the side of you. Furthermore, when you get to the hedge section you can use the a star algorithm to find the shortest way through the hedge or just the entrance through the hedge.

**Why finite state machine was the best choice**

The reason why finite state machine was the best choice for this program was because it can allow a program to wait for an action before it can actually perform that action for example in the bunner program you would use finite state machine to make the bunny wait for the hazard to pass in front of it an would then continue on once that hazard has passed by. So since the game has cars and a train that can pass every now and then finite state machine will check to see if a car or train is in front of the rabbit the program will tell the playable character to stay in place until the hazard has passed by. When it comes to the logs however you will need to tell the program to actually stop at the bank of the river and wait until a log in in front of the bunny so that it can jump onto the log otherwise it will fall into the river and the player will lose the game.

**References**

Pecurka (2019) *Frogger-AI-bot* Available at: <https://github.com/pecurka/Frogger-AI-bot> (Accessed: 15th October 2020)

Tianjason6 (2018) *Frogger-AI* Available at: <https://github.com/tianjason6/Frogger-AI> (Accessed: 15th October 2020)

Red Blob Games (2020) *Introduction to A\** Available at: <http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html> (Accessed: 3rd November 2020)

Lancelot Bors, M (2018) *What is a Finite State Machine?* Available at: <https://medium.com/@mlbors/what-is-a-finite-state-machine-6d8dec727e2c> (Accessed: 3rd November 2020)

DoctorMikeReddy (2020) *mike-bunner-demo* Available at: <https://github.com/PlaceholderGames/AI4Games-2020-Code-the-Classics/tree/mike-bunner-demo> (Accessed: 3rd November 2020)

ryanlambie (2020) *Code-the-classics* Available at: <https://github.com/PlaceholderGames/AI4Games-2020-Code-the-Classics/tree/master> (Accessed: 9th October 2020)

Swift, N (2017) *Easy A\* (star) Pathfinding* Available at:<https://medium.com/@nicholas.w.swift/easy-a-star-pathfinding-7e6689c7f7b2> (Accessed: 10th October 2020)

Jones, A. (2018) *Applying the A\* Path Finding Algorithm in Python* Available at: <https://www.analytics-link.com/post/2018/09/14/applying-the-a-path-finding-algorithm-in-python-part-1-2d-square-grid> (Accessed: 10th October 2020)

Trevor Payne (2013) *Let’s Learn Python #20 – A\* Algorithm* Available at: <https://www.youtube.com/watch?v=ob4faIum4kQ&feature=emb_title> (Accessed: 11th October 2020)

Trevor Payne (2013) *Let’s Learn Python #19 – Finite-State Machines (FSM)* Available at: <https://www.youtube.com/watch?v=E45v2dD3IQU&feature=emb_title> (Accessed: 04th October 2020)