**CSY1018: Web Development**

**Assignment 2: Programming (JavaScript)**

Ana Rosa Neto Gaspar

18400473

BSc Computing



University of Northampton

Submission: **Sunday, 22nd April 2018, by 23h59.**

**TABLE OF CONTENTS**

University of Northampton 1

1. INTRODUCTION & PROBLEM SPECIFICATION 3

1.1. Aim 3

1.2. Objectives 3

2. Game design 5

3. Program design 6

4. Testing 7

5. Evaluation 11

**TABLE OF FIGURE**

Figure 1 - Move functions loop 6

Figure 2 - Horse turn up if statement (function moveRight) 6

Figure 3 - Finish race code 6

Figure 4 - Horses moving right (Google Chrome and Internet Explore) 7

Figure 5 - Horses moving up (Internet Explore) 7

Figure 6 - Horses move up (Google Chrome) 8

Figure 7 - Horses move left (Google Chrome) 8

Figure 8 - Horses move down (Internet Explore) 9

Figure 9 - Horses move down bug (Google Chrome) 9

Figure 10 - End of race 10

# INTRODUCTION & PROBLEM SPECIFICATION

## 1.1. Aim

The purpose of this assignment is to assess our ability to create an interactive web page using JavaScript, CSS and HTML.

Using the zip file as2.zip which contained some HTML and CSS for a horse racing game. Our task is to build a web based horse racing/betting game by adding the relevant JavaScript code.

The problem should be solved using the existing HTML/CSS as a basis and build onto it within the rules below.

## 1.2. Objectives

**Rules (Basic)** Make the horses perform one lap of the track, where:

* Each horse should move to the bottom right of the screen, up to the top, across the top of the page, back down and across the line.
* The horses should continue following the track until they reach the start/finish line.
* As horses change direction and start/stop their animation should change. The animations are provided for you as CSS classes. The horses should always face the direction they are travelling. (Classes runDown, runUp, runLeft, runRight, standRight).
* As each horse reaches the start/finish line it completes the lap and should stop racing (the horses don’t have to stop exactly on the line, but must go over it).
* Pressing the start button again should reset the horses and start another lap The speed the horses run at is up to you, but a lap should not take more than one minute.

**Rules (Intermediate)** Make the race randomised and implement betting, where:

Rules (Intermediate)

* Each horse should run at a different speed around the track.
* To make it so you can’t see who will win by identifying who’s fastest at the start of the race, the speed should vary as the horse goes around the track (e.g. one horse starts slow but later overtakes another horse by speeding up).
* Don’t have each horse turn at exactly the same point on the track.
* Detect the winner (the first horse to complete a lap) and display the results by listing the position and the name of the horse.
* The start button should be disabled while the race is in progress.

Rules (Intermediate advanced)

* + The user starts with £100 and can bet on a horse by entering the amount they wish to bet and selecting a horse.
  + If they win, they get double their money back.
  + They can then bet on another horse with their new funds and run the race again.

**Rules (Advanced)** Implement one of the following:

* Randomly generate the odds for each horse at the beginning of the game. E.g. one horse might be 2 to 1 (you get twice your money back if you win). Another might be 100 to 1 (you get 100 times your bet back if you win). Once a horse wins, it’s odds should decrease. E.g. if it wins, on 100 to 1 odds, its odds should be reduced for the next race, if a horse comes second its odds should decrease. This should stack, for example, if a horse has lost 5 times in a row it should have high odds.
* Allow the user to specify a number of laps around the track that the horses will run over the course of the race before a winner is declared.
* Adjust the shape of the track so it’s more interesting than just a square, e.g. a figure of eight. You will need to amend the HTML/CSS and paths that the horses run.
* You may also implement other features you feel would be interesting to add, the level of difficultly will need to be in line with the other options in this section.

# Game design

To come up with a solution, I analysed the rules and choose from the exercises done in the class the ones I thought could help.

The first step was to make the horses move and turn at the relevant points on the track. To make that happen I created four functions to make the horses move (moveRight, moveUp, moveLeft, moveDown) and a variable timer, to indicate when each function should be executed or stop. So, in each function I put an if statement to indicate when that function had to stop (clear the timer variable) and turn (change the time variable, to a new function).

After that all the horses where moving together, so to make them race I had to create a new variable called speed, in all move functions, to define how much each horse should go forward. So, to make a different horse win each time this variable was a random number between 0 and 5.

# Program design

To facilitate the process of writhing the code I divided the problem in the follow way:

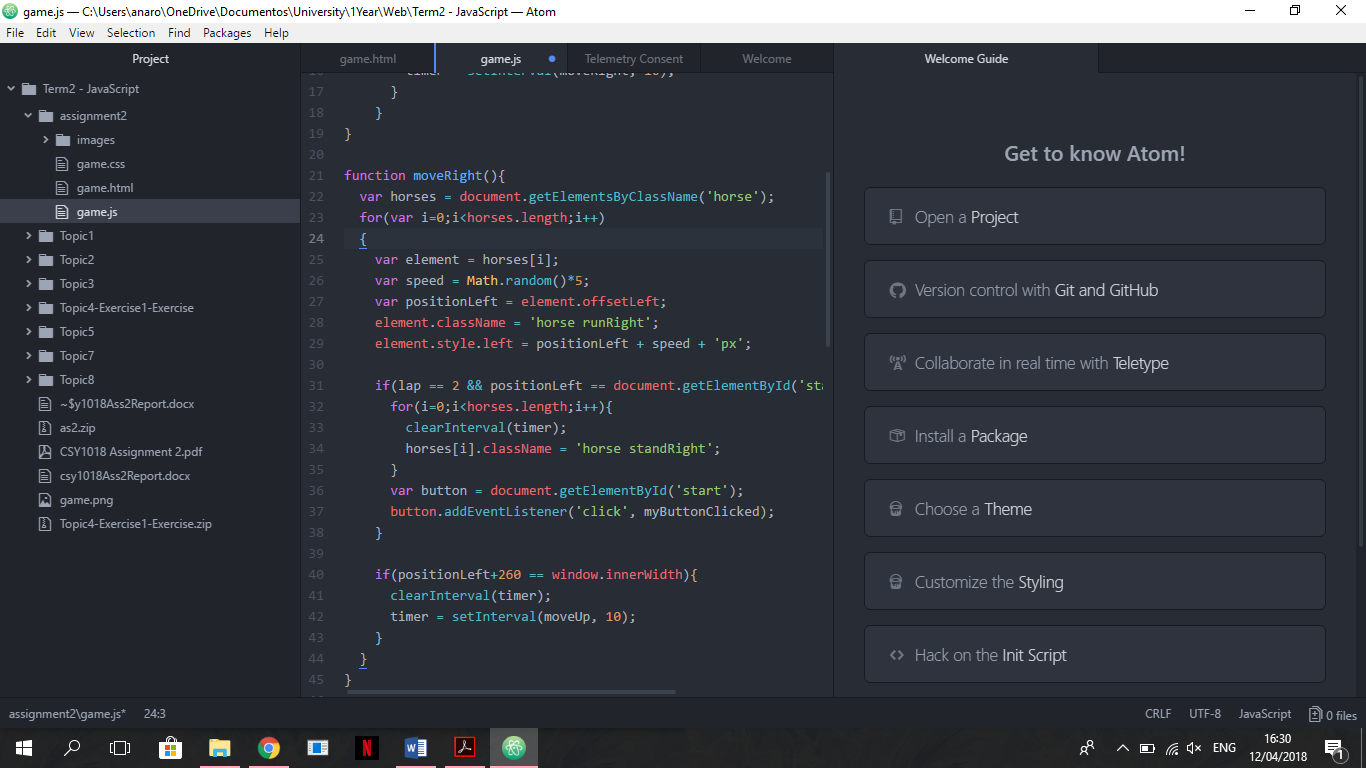
* **Horse move**  
  As I said before, to make the horses move I created four functions.

Figure 1 - Move functions loop

Because all the horses had to do these four functions at same time, I used a loop in the beginning of each function. That way I wouldn’t have to repeat the four functions for each horse.

* + **Horse turns**

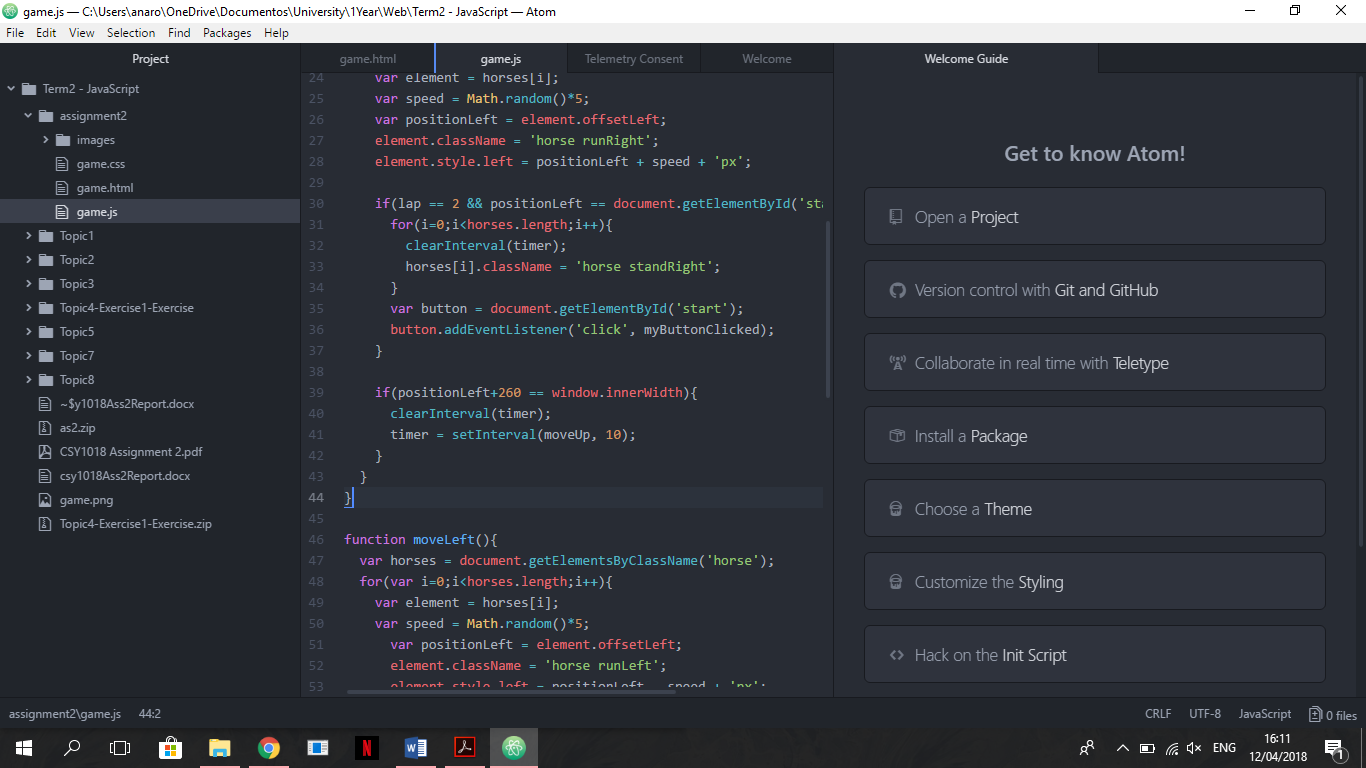
To make the horses turn at the relevant points on the track in different screen sizes I had to use the offsetLeft and offsetTop proprieties to get the actual position of the horses, and innerWidth and innerHeight properties in an if statement to check the end of the window.

Figure 2 - Horse turn up if statement (function moveRight)

* **Finish race**

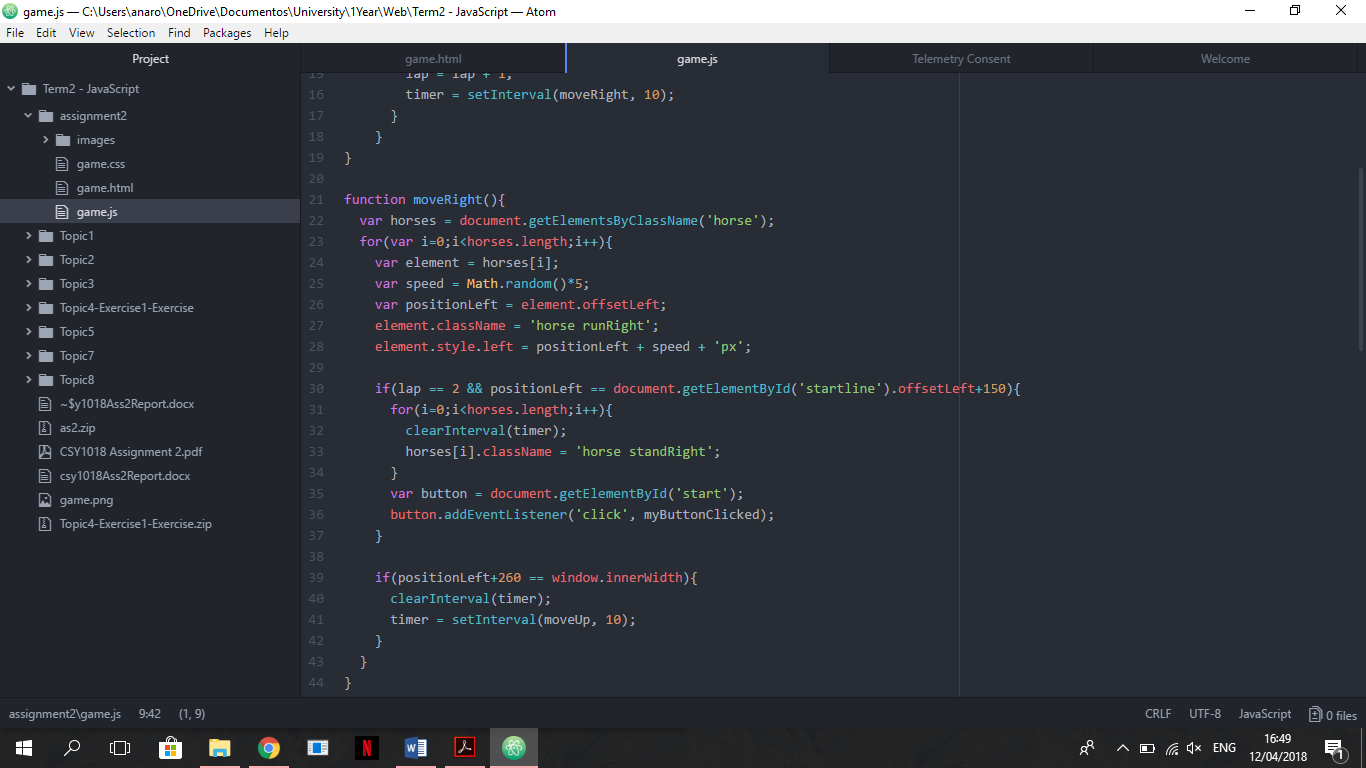
The race should finish after the horses pass the start line again. To make that happening I created a new if statement, inside the moveRight function. In the if statement, I used a new variable (lap) to count the laps of the race and the offsetLeft propriety from the element ‘startline’ to declare where they should stop.

Figure 3 - Finish race code

* **Race randomized**
* **Implement betting**

# Testing

To test if the code worked I run it in different web browsers, always checking the console and resizing the browser windows.

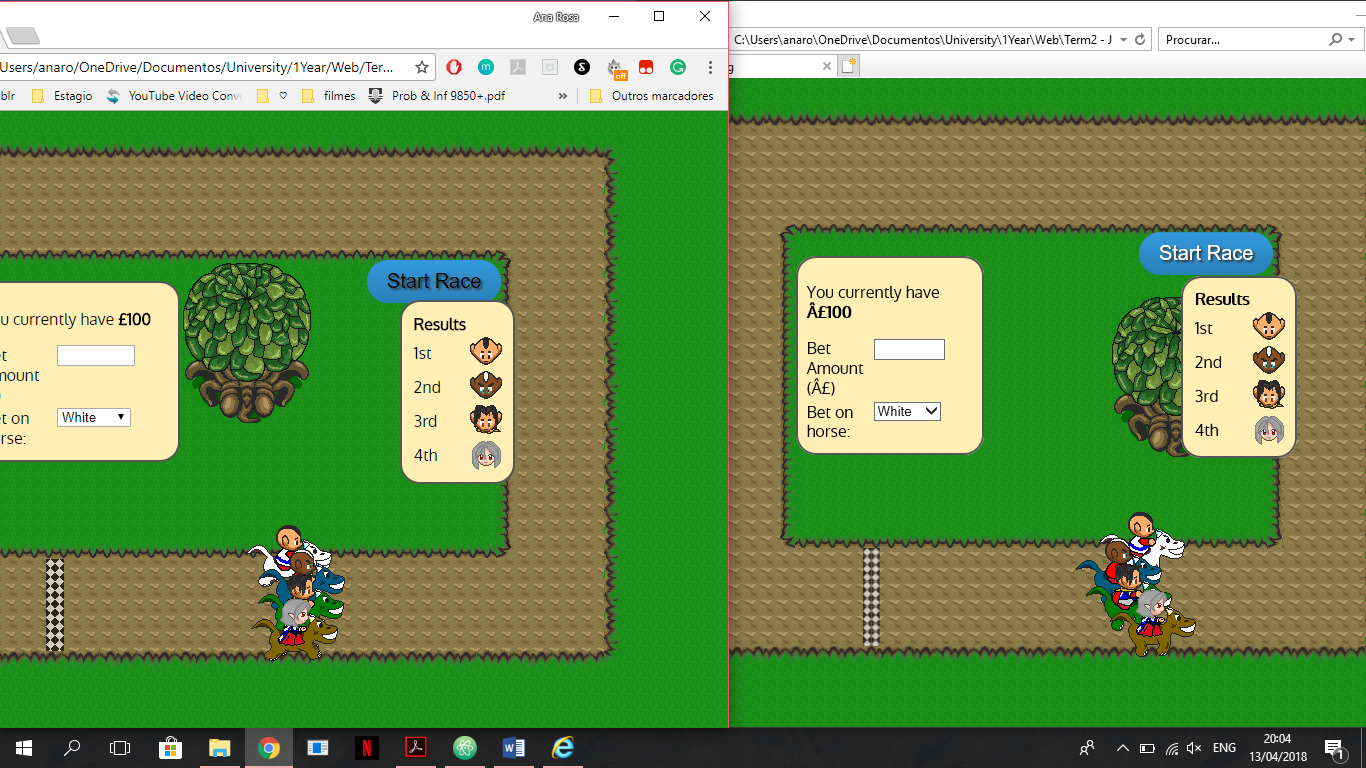


Figure 4 - Horses moving right (Google Chrome and Internet Explore)

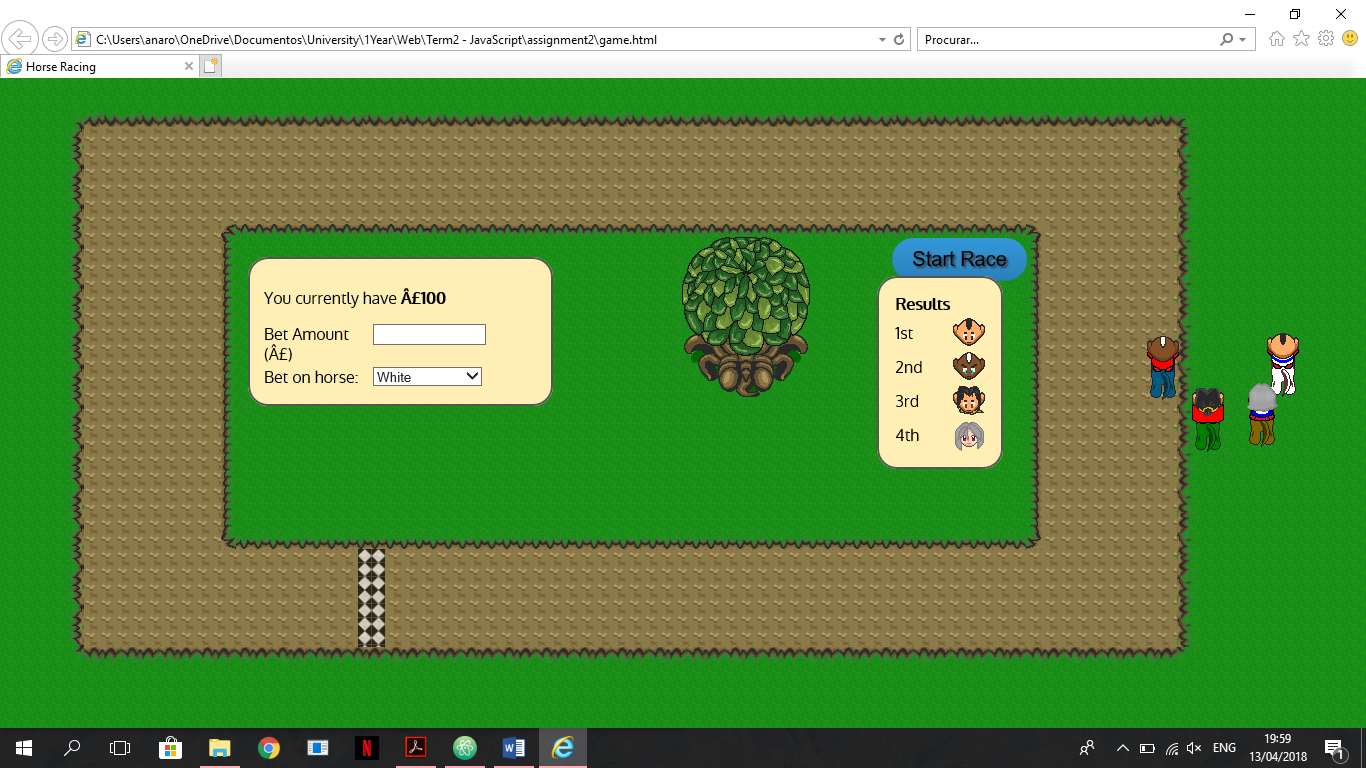


Figure 5 - Horses moving up (Internet Explore)

Figure 6 - Horses move up (Google Chrome)

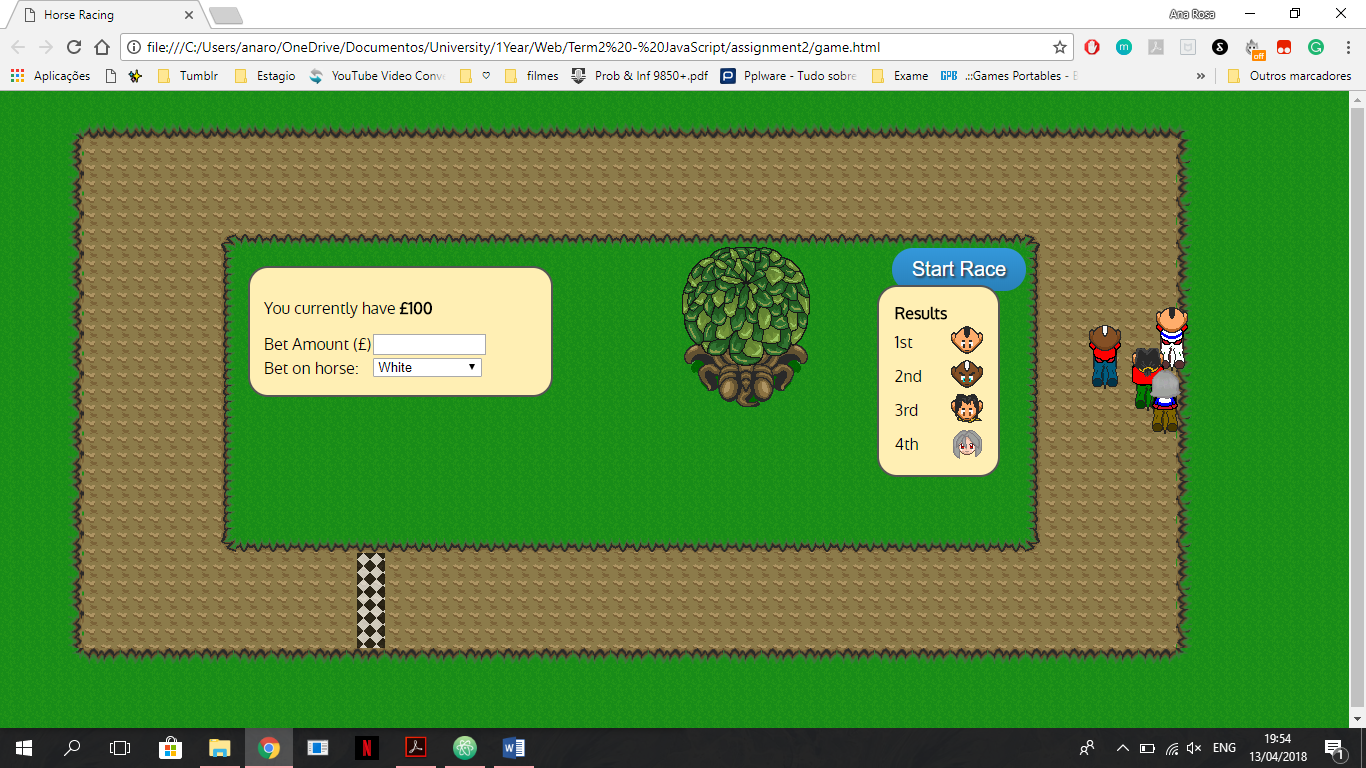
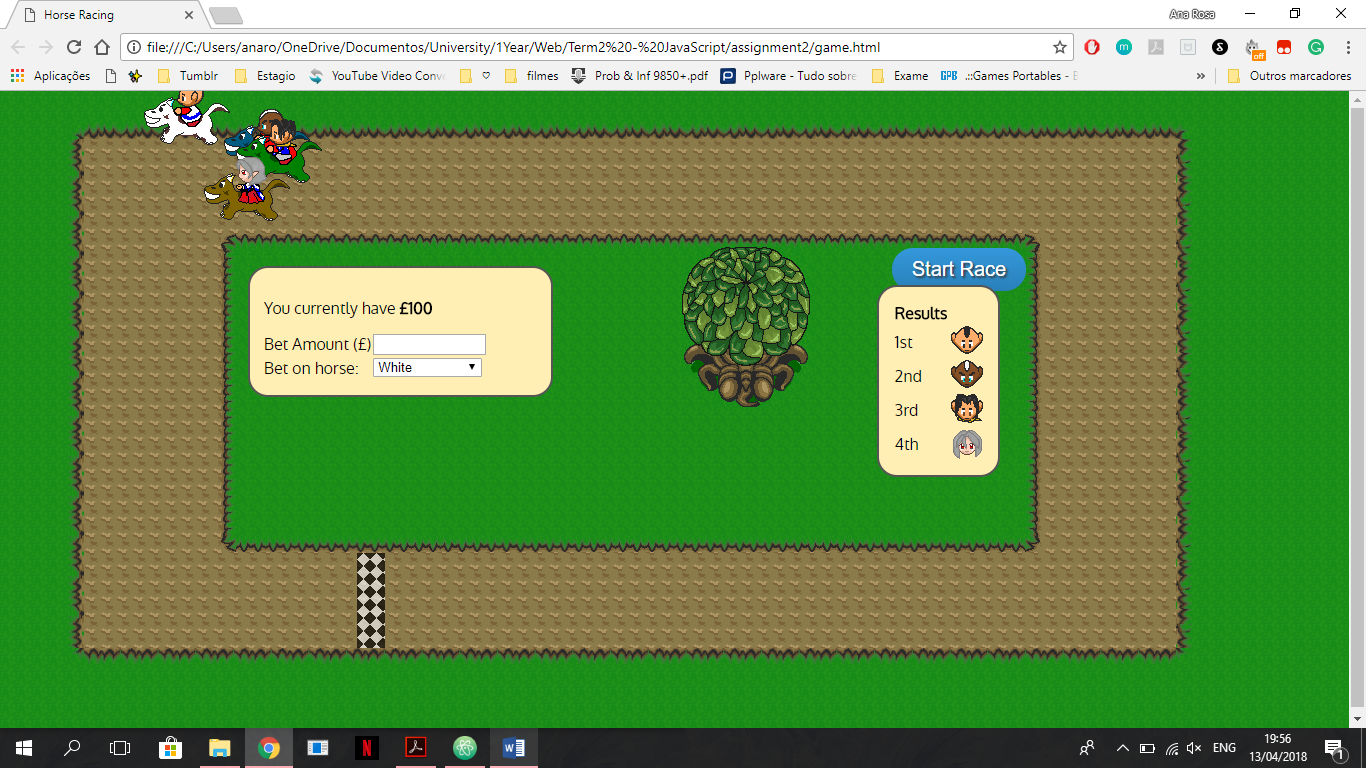


Figure 7 - Horses move left (Google Chrome)

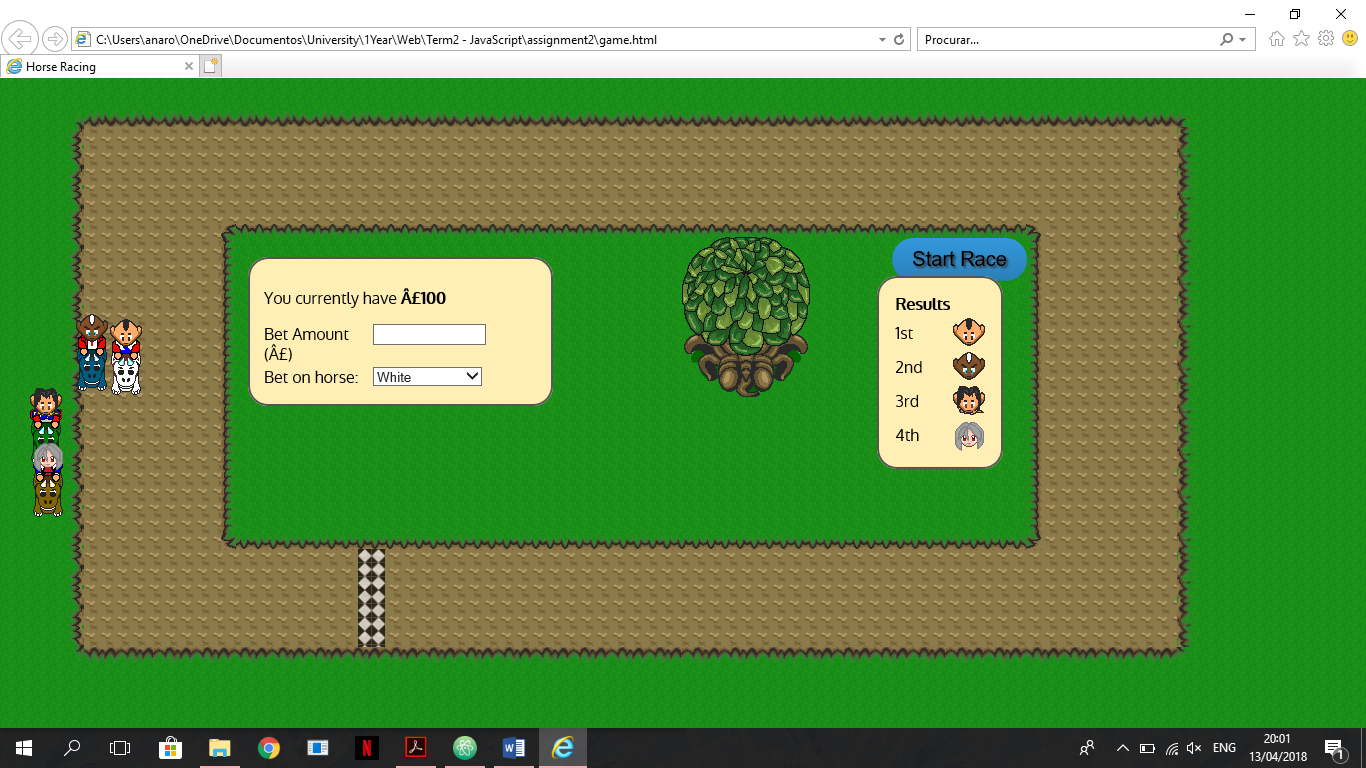


Figure 8 - Horses move down (Internet Explore)

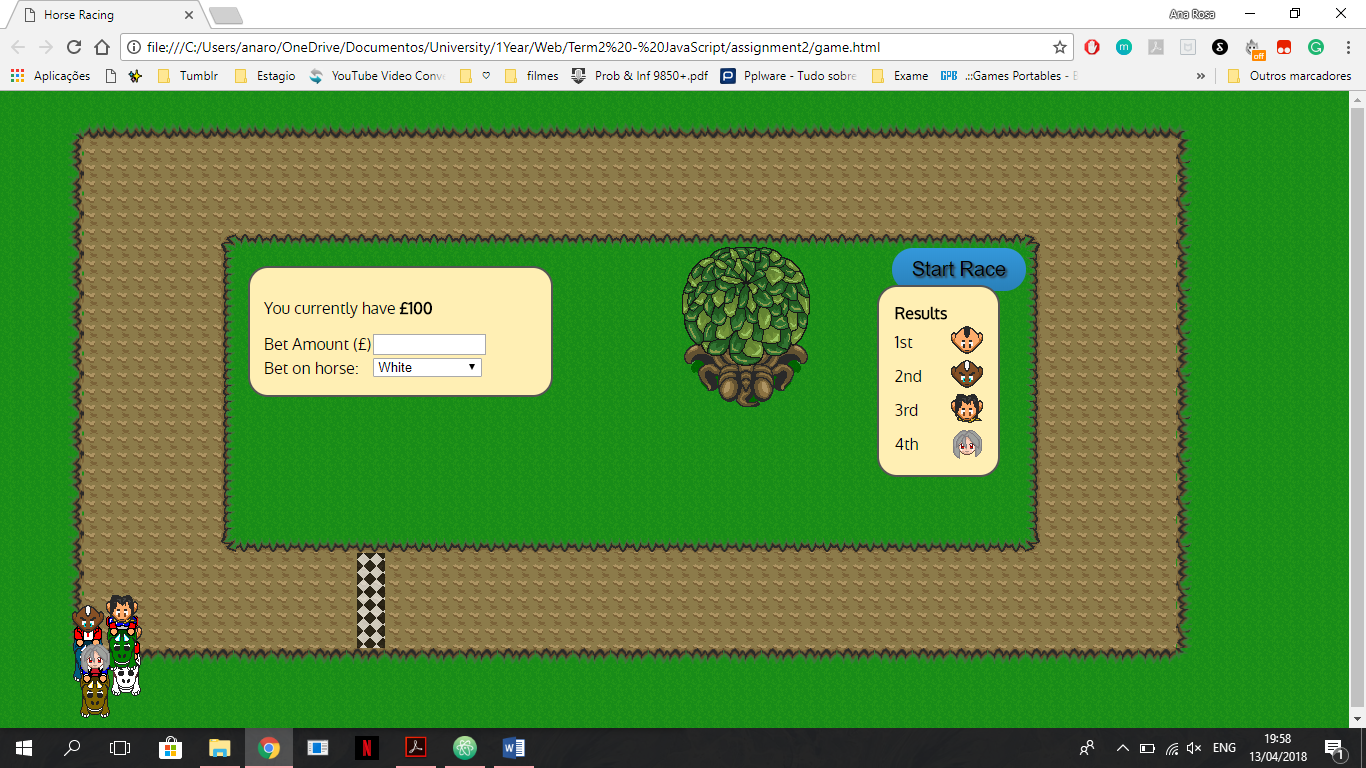


Figure 9 - Horses move down bug (Google Chrome)

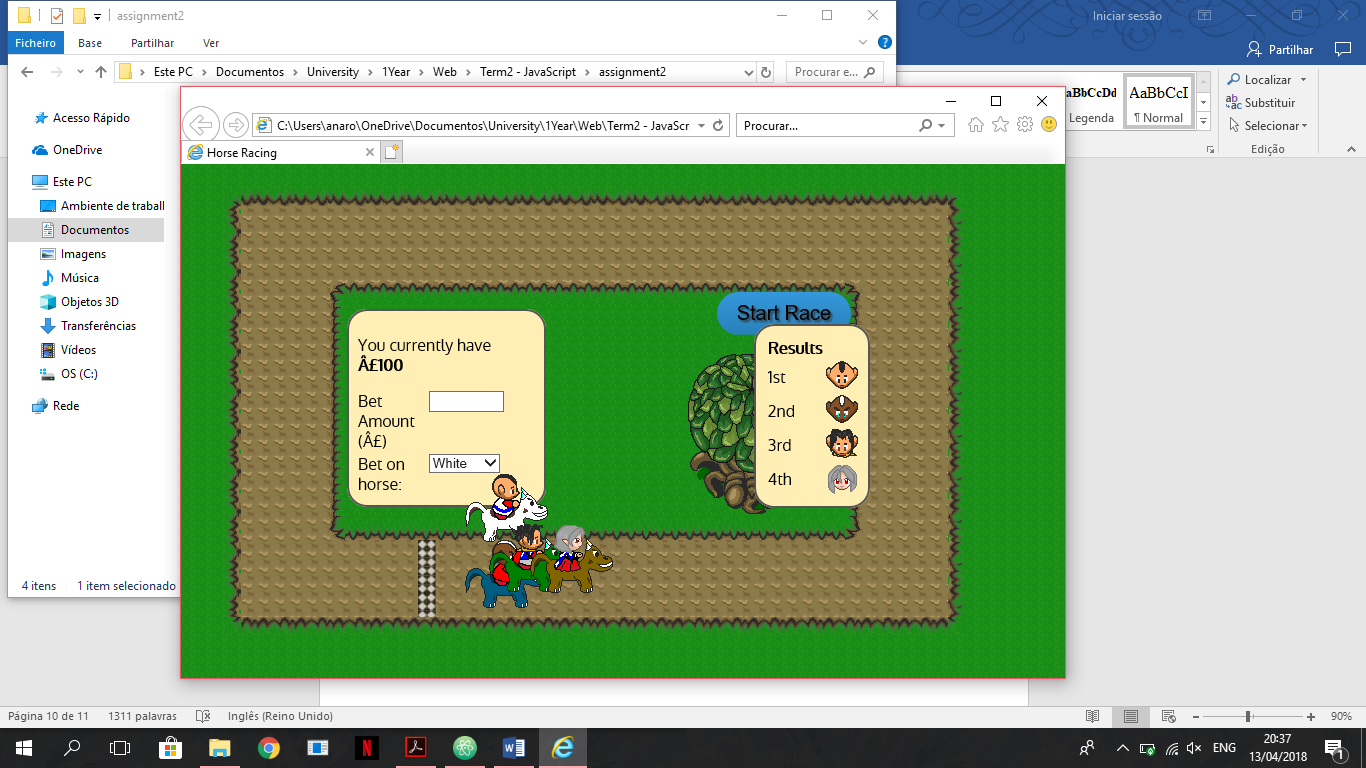


Figure 10 - End of race

# Evaluation

Although the game works and the objectives were fulfilled, there is some bugs/weakness on it.

The main bug of the game happens when the horses must turn. One of the bugs is that the horses never turn in the same place as they should, which causes them to be out of the track sometimes. The other is when I click on the button to start a new race, the page refresh and the new race begins but sometimes the horses don’t turn.

Apart from that the game work in all browser and page sizes, so I’m satisfied with my solution results that can be seen on <https://anarosang.github.io/Assignment1/HorseRacing_GoogleChrome11_04_2018.mp4>. If I had more time I would work on the betting part of the game and after that done try to find the best way to resolve the actual bugs.

If I were to add a 5th horse I think it would be quite easy as the changes are simple and should be made in the code was on the loops and add a new horse on the CSS and HTML file.