**FINAL PROJECT**

**ALGORITHM AND PRGORAMMING**

**Project name: 2D Racer**

**Name: Pandya Limawan**

**Student ID: 2502022433**

**Class: L1CC**

# **Table of Contents**

[**Table of Contents** 2](#_Toc93083623)

[**Description** 3](#_Toc93083624)

[**Introduction** 3](#_Toc93083625)

[**Project Specification** 3](#_Toc93083626)

[**Solution Design Plan** 4](#_Toc93083627)

[**Implementation** 5](#_Toc93083628)

[**Explanation of the code** 5](#_Toc93083629)

[**Flowchart** 15](#_Toc93083630)

[**Evaluation** 16](#_Toc93083631)

[**What can be Improved?** 16](#_Toc93083632)

[**Reflection and Experience** 16](#_Toc93083633)

[**References** 17](#_Toc93083634)

# **Description**

## **Introduction**

When the final project was first introduced, I had an idea to make a game, but I wasn’t sure of what to make. I first got an idea to create a Tetris game, but I wasn’t sure with my coding ability that I’ll be able to make Tetris. After further research, I came across a 2 players website that has a racing game. I decided to make a racing game after seeing that game.

For this project, I chose Visual Studio Code as my IDE and I commit all the file for this project to my GitHub account. The link for this specific project: <https://github.com/pan-dya/FinalExamAlgo>

## **Project Specification**

The purpose of this project is to create a simple 2D two player racing game. This game is able to be played by 2 players on the same computer/device. The goal is simple, to cross the finish line before the other player. First to win 5 races, will win the game. There is no advantage for anyone, because both cars have the exact same maximum speed and acceleration. This game is fairly simple to play, the control is very user-friendly only by using WASD and arrow keys to move the cars. Even though the game is simple, I think that simple 2 player games can be fun to play with friends and relatives not only that, it will create fun memories.

# **Solution Design Plan**

**Input**

1. Player movements

**Output**

1. Car moves
2. Winner of the race
3. Winner of the entire game

**Main menu**

The main menu is not really a menu, it is just a simple “press to start” button that allows the game to start. Pretty straightforward, any key in the keyboard can be used to start the game. This window shows the user what keys to press to move the cars.

**Main Game**

This window is what the user sees after pressing the button, the user can see how many wins does each player have on the top left of the screen. The game is playable afterwards.

**Results**

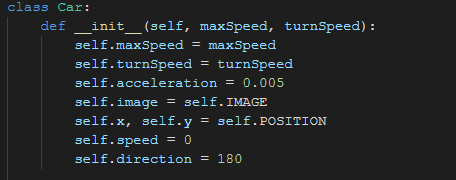
This window shows the winner of the game

# **Implementation**

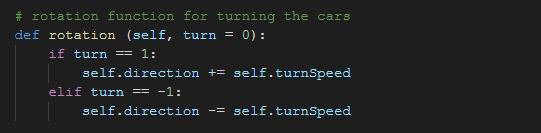
For this project, I’m using two modules which are pygame and math module. There are 4 classes that I made for this project, Car class, Player1 class, Player3 class, and game class. The car class is where I store functions used for the car itself, whether it’s player1’s or player2’s. For player1 and player2 class, I inherited the car class with 2 parameters, max speed and turn speed these 2 parameters are basically making it easy to change the velocity of specific car.

## **Explanation of the code**

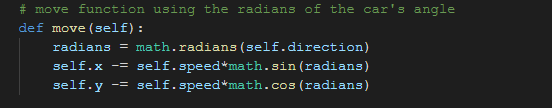
**Car Class**



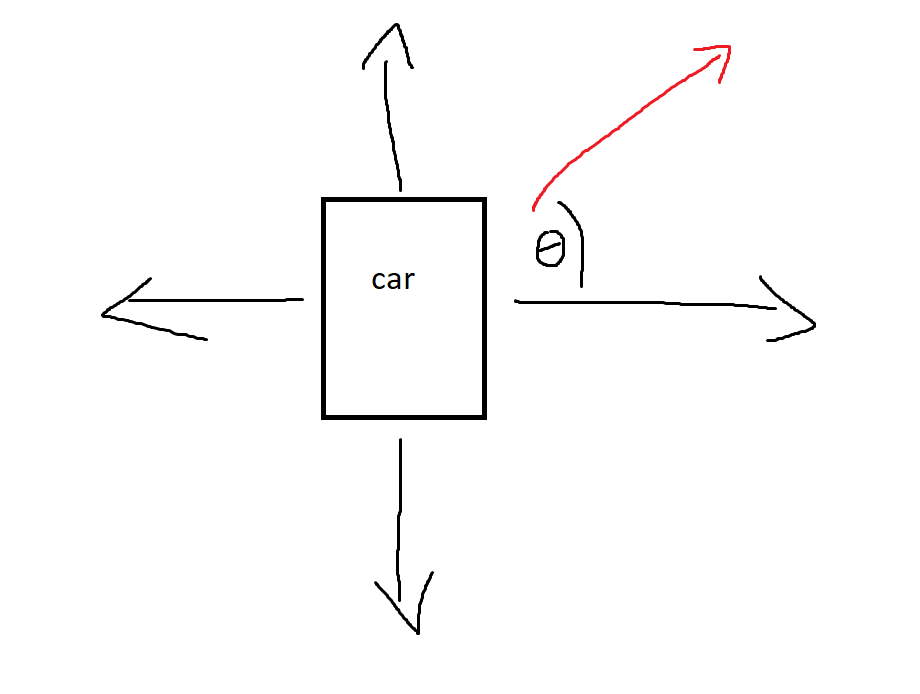
In the init method, I initialize the attributes for the car movement and position. maxSpeed and turnSpeed are inside the parameters because I want to make the car speed specific for different cars. For IMAGE and POSITION, I used constant variables so I am able to call self.image, self.x, and self.y from other class.



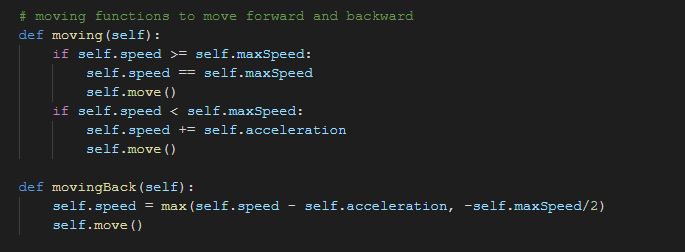
The rotation method here it’s just to decide where the car will turn. The default turn value is 0 and I make 2 conditions here if the value of the turn is 1, it will add the angle where the car is facing towards the turn speed. If the value of turn is -1, it will be the other way round.



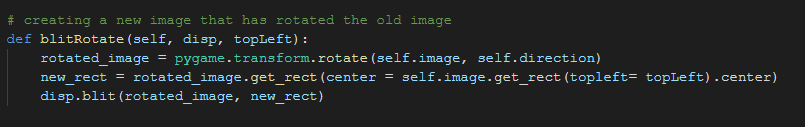
Move method is used to change the car position, from one place to another. I used sin and cos from math module for specific reason.



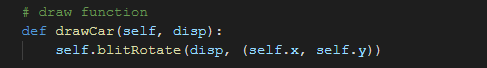
When the car is moving on an angle. The car will not move on the same amount of speed as the velocity, hence I used sin and cos of the radians (θ) to have the car move at the correct speed.



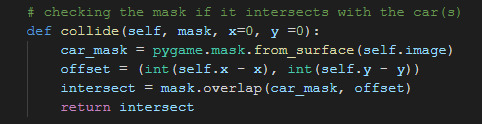
Moving functions here are used to determine what’s the current speed of the car. When a car is moving, it will not have the same speed all the time. So, for the move forward function if the speed exceeds the max speed, it will use the max speed as the speed. For the move backwards, the max method is used. Then self.move() is called after each method to move the car.



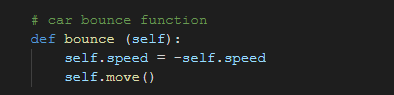
blitRotate is a function to draw the car in a rotated way. So basically, every image in pygame is a rectangle. And if I try to just use the rotate() function, the image will rotate on the top left hand corner of the rectangle. This function created a new rectangle that change the top left of the original rectangle to the center of the new rectangle. Then it will blit (Draw) the image. I didn’t come up with this function, but the one who created it is user “Rabbid76” on stackoverflow.



drawCar function called blitRotate() taking the position of the car. This function will be called later on to draw the image.

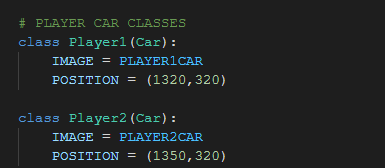


Collide function is used to check collisions with the border of the track by creating a mask for the car. Offset is a calculator for the car’s x and y values and reduce it with the x and y of the track border mask. Offset uses int function because overlap() must use an integer value and can’t use float.



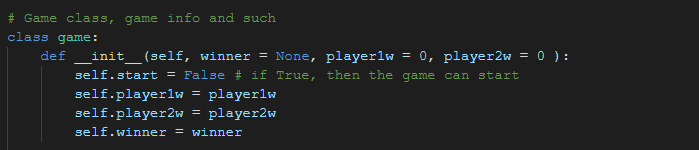
Bounce function is a function to bounce the car if it collides with the border. Simply by changing the velocity of the car to negative will do just the trick.

**Player car classes**

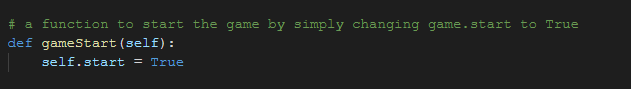
****

Car class is inherited to these 2 classes. This class is only used to define the specific x and y of the cars and which image to use for the car.

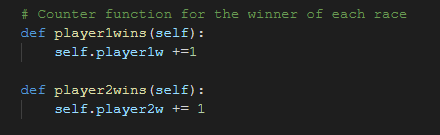
**Game class**



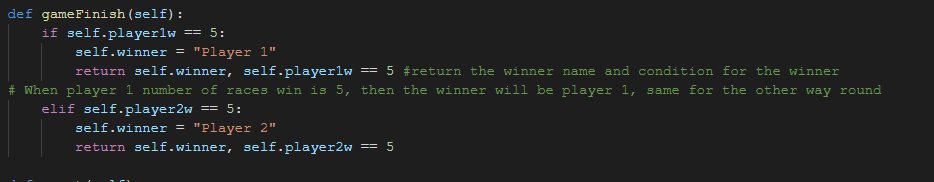
The init function initialize start to check whether the game is started or not, player1w and player2w to see the amount of wins each player has during the game and winner which the default value is None.



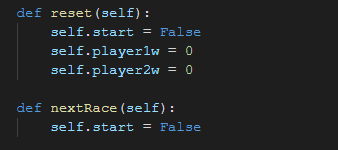
gameStart() is a function to start the game. This is the only function which has self.start = True. Not any other function have this so gameStart() must be called to start the game.



These 2 functions are basically a counter for the winner.



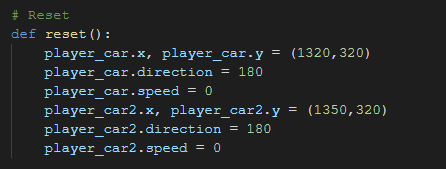
This function checks who’s the winner of the game. If any player has 5 wins, then the winner will be set to Player 1 or Player 2.



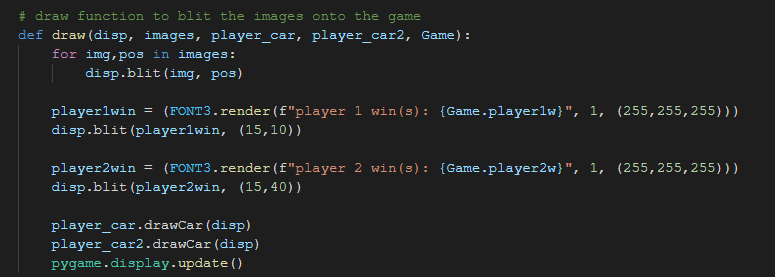
Reset() is used to reset everything, including the wins. This function is to prepare the next game.

nextRace() is a function to start the next game.

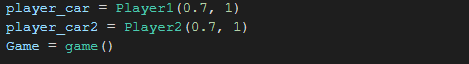
**Outside of class**



To reset everything, including car positions. Car speed

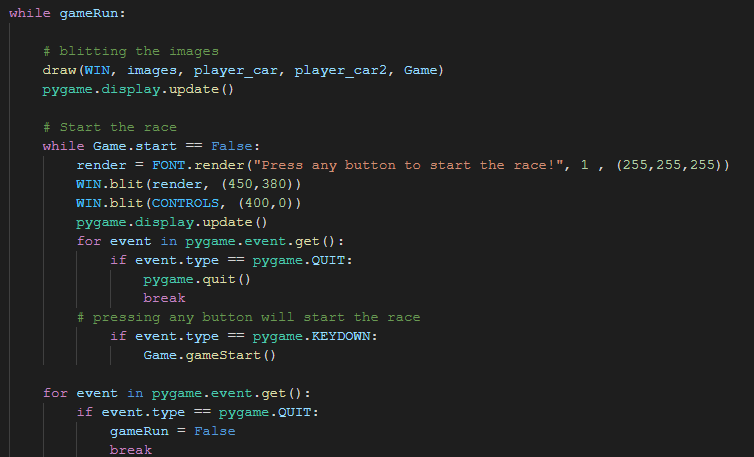


To draw everything on to the screen. The first segment of this function is to draw everything on to the screen except the texts and cars. Player1win and player2win is used to draw the text on the top left corner of the screen as a counter. The last segment of this function is the drawCar function from car classes to draw the car. Last on the function is pygame.display.update() which is needed to update everything.



Classes being placed in variables

**Main Loop**

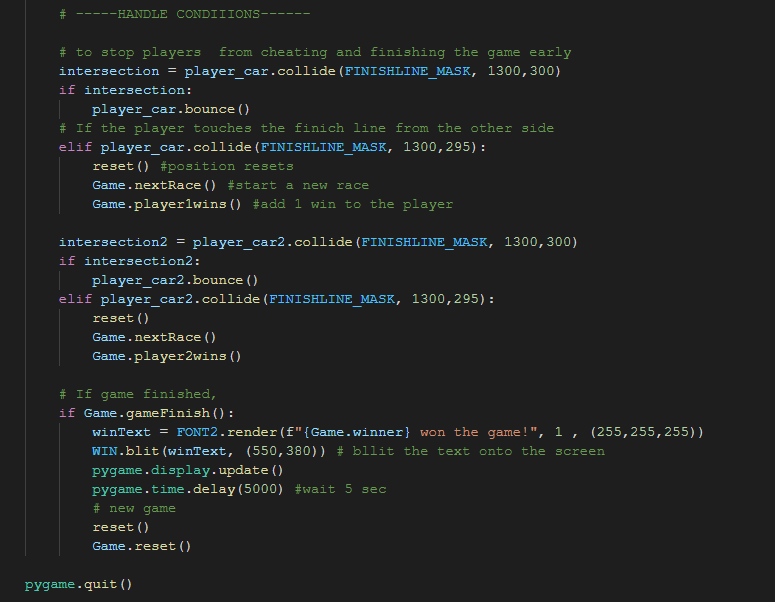


draw() is called first thing first to draw the elements onto the screen. Next segment is another loop inside the main loop. This loop here is just before the game starts, so it will show render (the text before the race), controls (the controls shown to show which button to press to move the car). A loop to exit pygame is next on this loop then a condition if a key is pressed to start the game.

Next segment of the main loop is a condition if pygame is closed.



This here is the controls handling and collision handling.

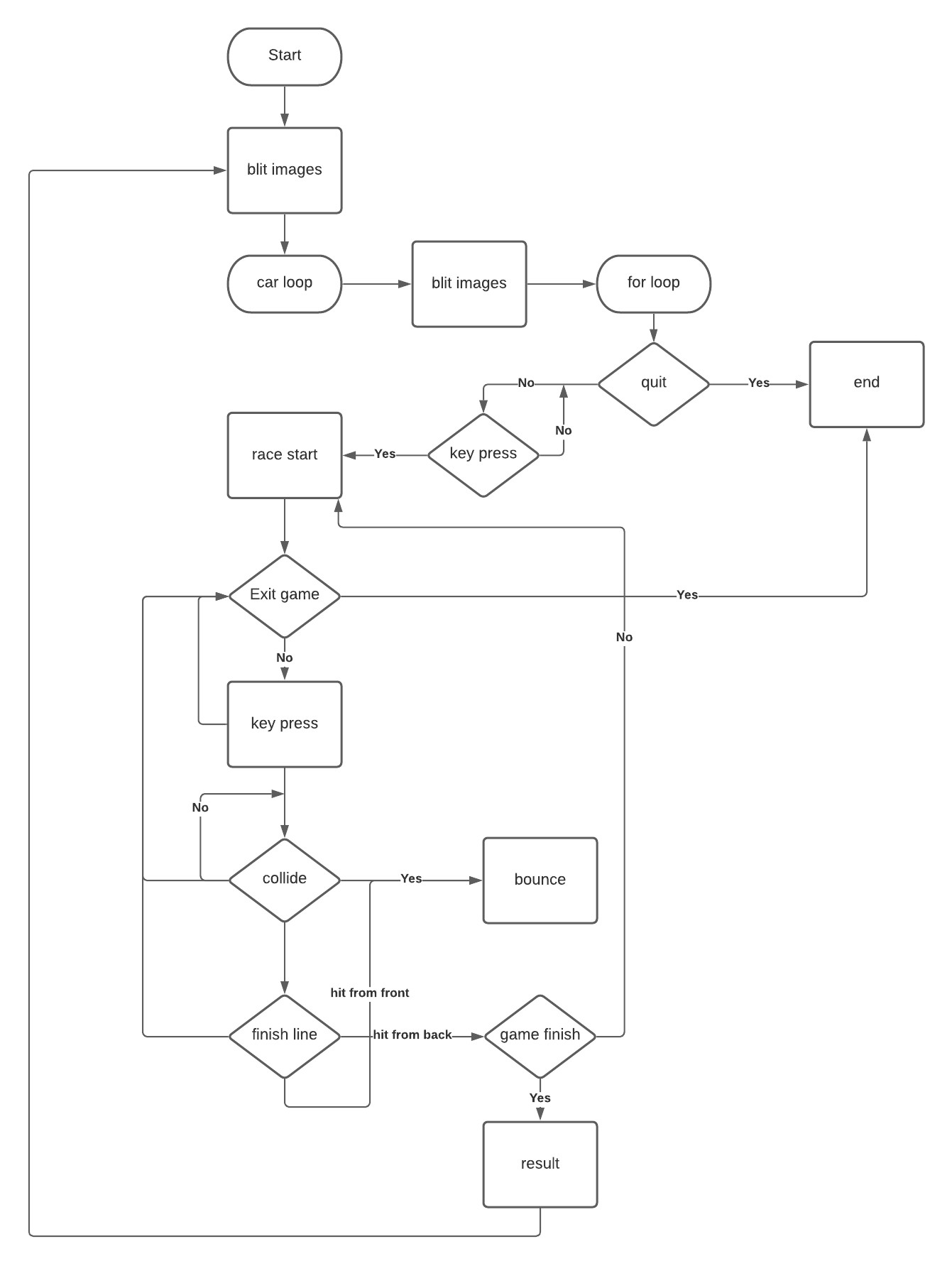


On here, if the car hit the finish line from the start side, the car will bounce so it is not possible to cheat. If the car hit the finish line from the correct side, reset function will be called.

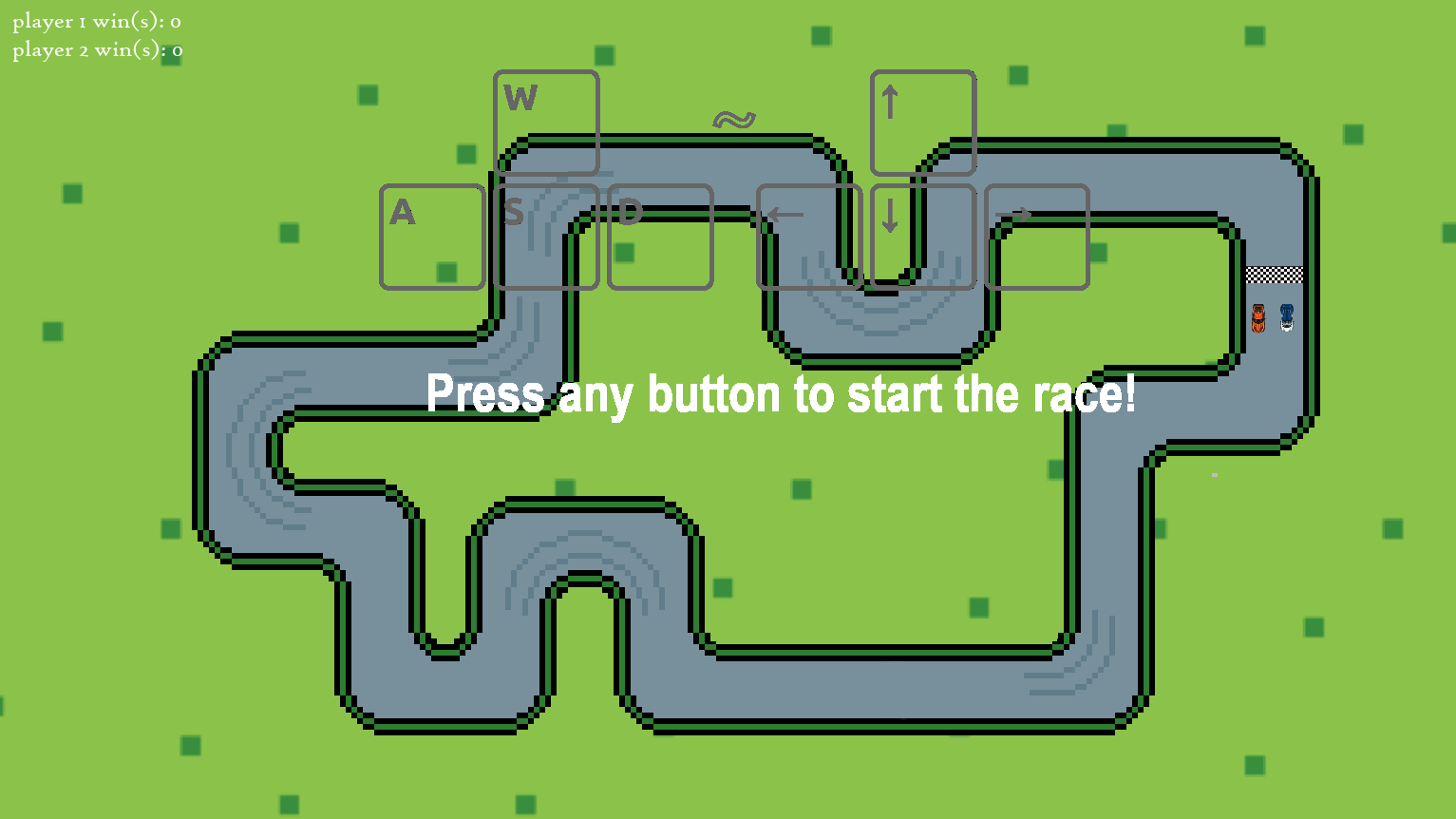
If the game is finished, a text will pop up onto the screen saying which player won the game. There will be a 5 second delay after that before a new game will start.

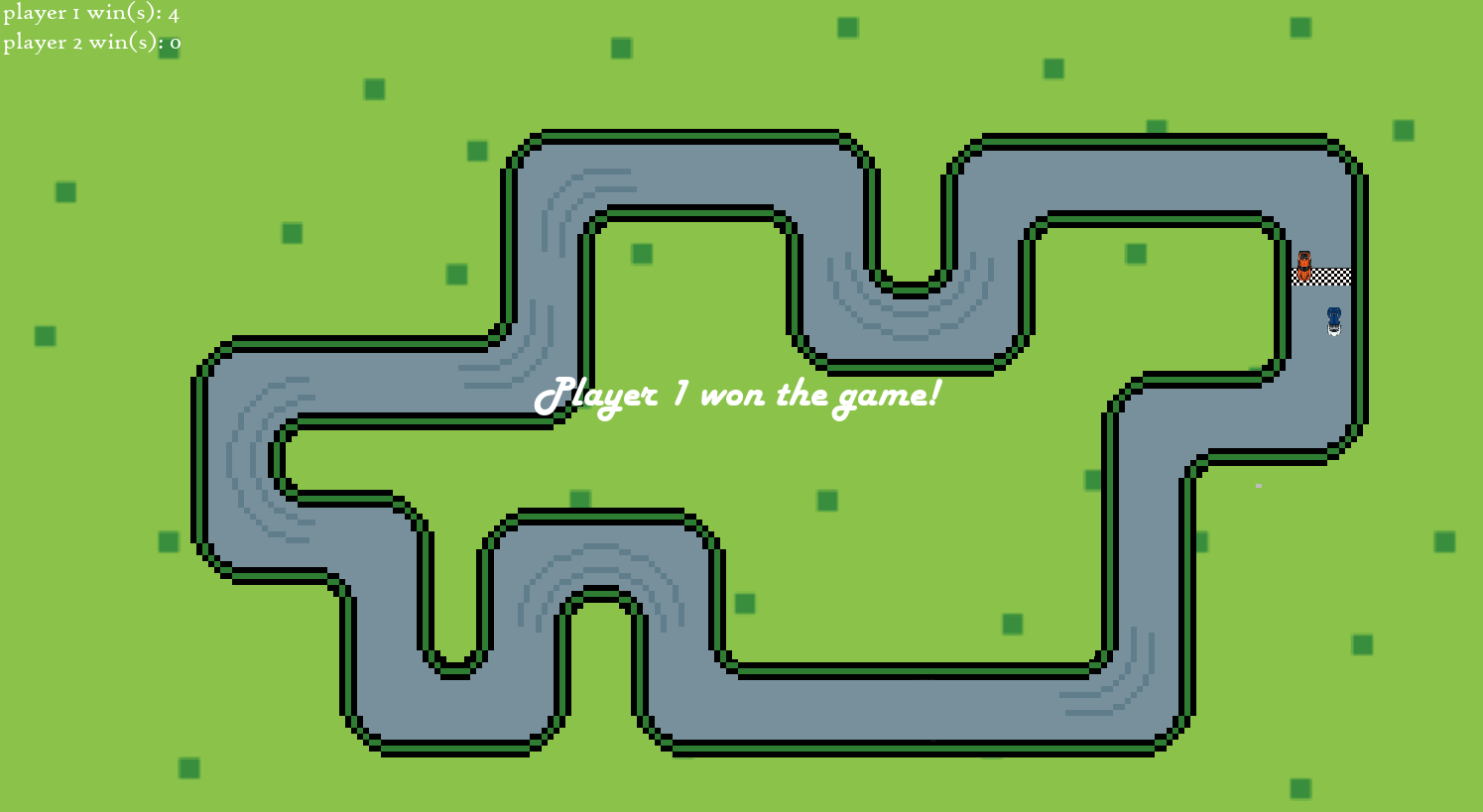
## **Flowchart**

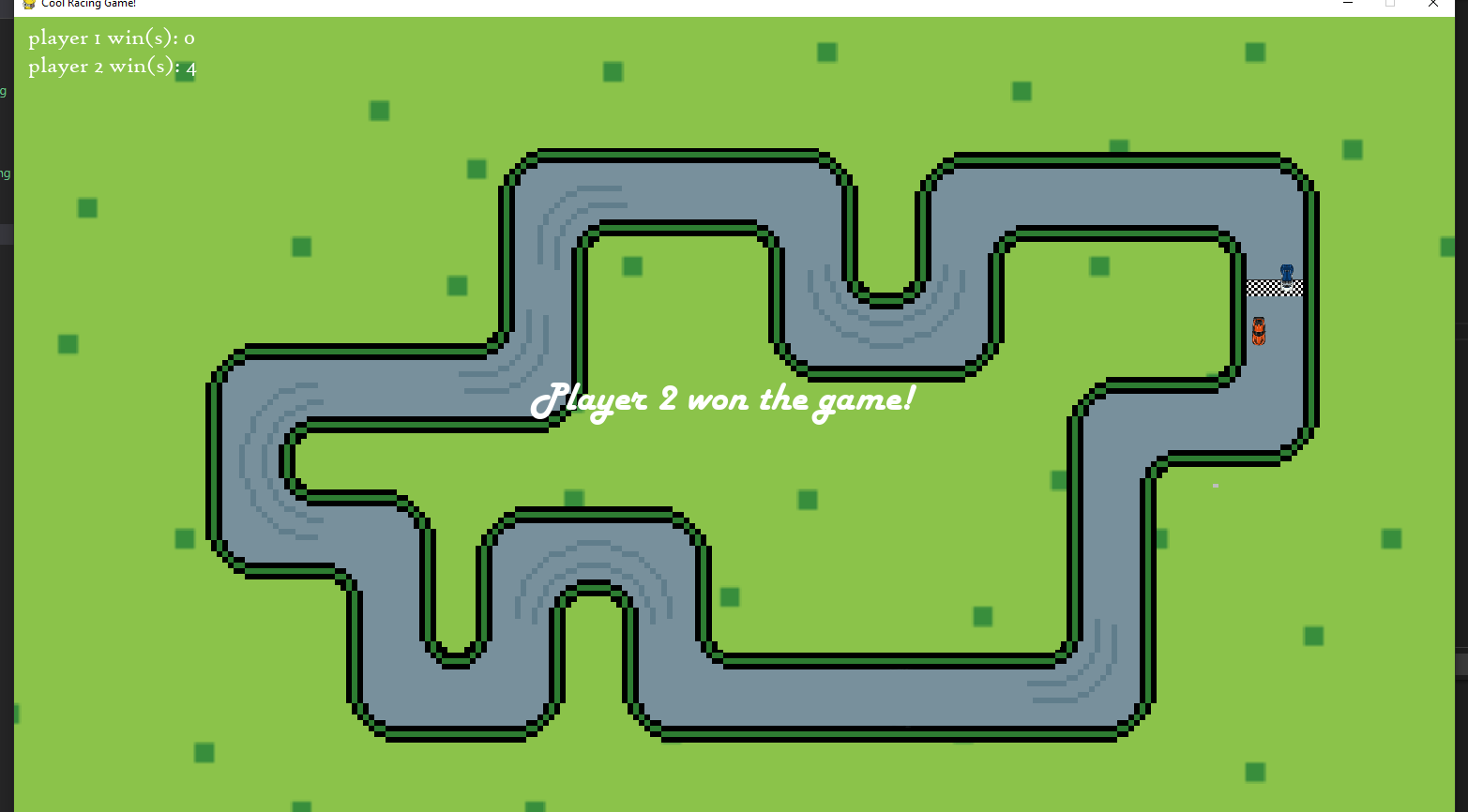
**Main Loop**



# **Proof of Working Program**







# **Evaluation**

## **What can be Improved?**

The program works fine right now, every input works just fine, the collision and finish line work fine. In terms of efficiency and consistency, there are rooms to be improved on. First, the border of the track in some part is a little bit glitchy, the car can bounce of the track even though it doesn’t touch the border at all. This problem is due to the masking not being clean. Next, for different devices, this program hasn’t been tested yet and who knows if this program can run properly in lower-end devices. The efficiency of the code is not the best. Some part of the code can be put in a better place for better looks and to run the game smoother. Although there are some flaws in this project, it mostly works well to the initial purpose of the project.

## **Reflection and Experience**

At first, this project was something new for me and I knew it was going to be hard. First, it was not easy to find an idea that suits my current sill level. Some ideas came to my mind, but I didn’t know how to executed properly, other than that, I am not that confident with my skill set to do a pretty hard challenge at the time. I thought of some games that I used to play in school with my friends, a simple web game that can be fun and challenging at the same time. Next, I remember a game that I play with a few friends of mine a few years ago. It was a racing game from a different angle. So, a 2d racign game doesn’t seem to hard to make so I decided to make a 2d racing game that can be played by 2 players offline. Learning pygame at first doesn’t seem like a big deal for me then because it is just as simple as drawing an image onto the screen. But, I found it difficult for me to move the image, especially turning the car. I did some research after that to find a solution to my problem. Eventually, I found a pretty good solution on stackoverflow by this user “Rabbid76”. When I thought everyting was going as planned, I realise I have to do some calculations regarding the car movement. Math wasn’t really my subject back in school so it was a bit of a challenge for me to find a good formula. So, I did more research and eventually I found a solution towards my problem.

# **References**

<https://appoftheday.downloadastro.com/app/race-it-2d-racing-game/>

<https://stackoverflow.com/questions/4183208/how-do-i-rotate-an-image-around-its-center-using-pygame>

<https://www.youtube.com/watch?v=Idu8XfwKUao>

Images:

<https://gameartpartners.com/downloads/2d-super-cars/>

<https://appoftheday.downloadastro.com/app/race-it-2d-racing-game/>

<https://pngimage.net/starting-line-png-1/>