Project Summary:

PyRace Group 09

Group Members:

Tommy Kaplan	University High School, Zionsville, IN
Dylan Klasing	Springboro High School, Springboro, OH
Tristen Shields	Brown County High School, Nashville, IN

Introduction:

Our project assignment was to learn how to code in a programming language called Python, and then to create a game using Python code. Python is a programming language that is growing in popularity, since it allows for easy access to others' code by importing packages of code called "libraries", which can make coding substantially easier by providing built-in functions. The game was made using the Python library "Pygame", which is designed to allow programmers to easily make games in Python code by allowing built-in functions such as setting up screens, drawing shapes, and importing images.

Problem Statement:

For our project, we set out to build a two-dimensional racing game in Python code, hence the name "PyRace". It will work by allowing the user to accelerate and decelerate a race car by pushing the 'W' and 'S' keys, which won't make the car itself move, but will make the track around the car move. It will also allow the player to turn by using the 'A' and 'D' keys, which will rotate the image of the car and change its direction of motion relative to the track.

The track has coded in borders ("colliders"), meaning if the car tries to move across the track's walls, it will bounce off. This will keep the user on the track.

The game keeps track of laps, the time of individual laps, and the rankings of those times. It displays this information on the screen.

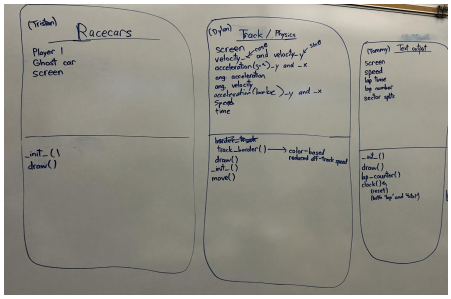
Background Information:

Projects written in Python coding are often organized in "classes", which often represent certain elements of the game. These classes are then broken down into "functions", which allow the code to perform certain organized tasks. Different classes will have different variables used to perform different functions within the class. All of this is executed by one "main" function at the end of the code, which goes and executes all of the functions in those respective classes in the intended order.

The Design Process:

The first step we took in designing code for PyRace was organizing the game into three main classes: 'Racecar', 'Track/Physics', and 'Text output'. As the names suggest, the 'Racecar' class puts an image of a custom race car on the screen and located on the track, the

'Track/Physics' class contains the track and the systems of motion that allow the track to move along with the image of the car, and the 'Text output' class simply displays the speed, lap time, and lap number on the board. We then took those classes, and determined both the variables and functions that each class should contain. We mapped it out on a whiteboard, as shown on the next page. (These weren't our final classes, but they allowed us to organize the project early on).



Once the layouts of these classes were written in code, we worked on drawing a track, and implementing our physics (motion) system on that track. At first we had a curvy track with a desert background, but after running into issues with pixels being out of range due to its complex shape, we drew another track that ran more like a maze in order to make coding colliders easier.

Once we could move the car along the track, we added code that would keep track of lap times, start/finish lines, and code that would display that information on the screen. After that, we set out to add sound to our game. We found racing music that was fan-made, and then implemented it to play as background music. Finally, we coded "colliders" for all of the track's walls, making it so the player cannot cut across the walls.

Results:

While the game looks different than we originally intended, due to issues regarding setting up track borders for complex shapes and limiting pixel ranges, we succeeded in making a two-dimensional racing game. The physics, controls, and racing all perform how we originally expected. The track design and colliders were the only two elements of the game we had to modify throughout the design process.

Reflection:

Over the course of our design process, we had to try six different methods of instituting colliders. Everything about the game was complete except for the track's colliders, and we spent days figuring out how to set up colliders without running into issues with FPS (frames per second) or compatibility with Pygame code and a moving track. From color-based colliders, to complex-shaped colliders, to specific Pygame libraries for colliders, it seemed like nothing would fit into our game and work.

Each member had to think out and try different methods over and over until we finally found a way to code the colliders into the track walls ourselves. We had to change the track so the borders weren't in a complex shape, which then changed the entire map design before we could implement the correct method. In conclusion, this project required us to be very adaptable and very persistent.