ID1110 Course Project
Car Racing!!!

INTRODUCTION

Background and context:

"Car Racing!!" is a basic racing game where you drive a car on a twisty track. You can enjoy the thrill of overtaking other cars. There are two cars: one you control and the other is controlled by the computer. The computer-controlled car follows a set path along the track. You control your car using commands to steer and move around the track.

Problem Statement:

We're making an exciting racing game that players will love! It'll keep you hooked with its unique and enjoyable racing challenges. We want you to have a blast while racing your vehicle around the track, staying fully engaged in the game.

Objectives:

Create a good gameplay experience that captures the thrill of car racing. To provide a diverse range of levels with increasing difficulty

Project Overview

Project goals and scope:

Make a video game that accurately depicts the thrill and excitement of racing, giving players a sensation of speed. To offer a variety of game levels. To create a computer car as an opponent which will follow a well-defined path. The race track is set on visually appealing green grass. Implementing physics such as bounce back and collision. Providing well-defined command controls to player car

Project Timeline:

1) 15-17 May: Selection of Topic
2) 18-21 May: Selection of track, cars, background, finish line
3) 22 May: Defining utils
4) 23-25 May: Learning libraries and modules required
5) 26-30 May: Defining “Gamedata” and “Car” class
6) 31 May: Defining “Gamercar” class
7) 1 June: Defining “Computercar” class
8) 2-3 June: Defining “move_player” and “manage_collision” functions and main game loop

Project Repository:

Tejaswini-0607/Python-Car-Racing (github.com)

Team Members and Contribution:

(i) Mullaguri Tejaswini (122301027):

Defined the “GameCar” class and all the functions involved. Defined the “move_player” function which helps in moving the car based on the commands provided. Formed the main game loop for the game.

(ii) Gubbala Sai Deepa (112301008):

Defined the “ComputerCar” class and all the functions involved. Defined the “manage_collision” function for handling the collision of cars. Defined path for the computer car.

(iii) Beere Harshitha (102301006):

Created the “utils” module. Selected cars, tracks, backgrounds, and finish line. Defined “Car” and “Gamedata” classes.

Methodology

Approach and methodology employed:

1) After doing research, we selected the required libraries (pygame, math, and time) and used their functions. We also created the “utils” module.

2) Selection of cars, tracks, background, finish line.
3) Defining the path for the computer car.
4) Defining the “Gamedata” class which contains functions related to game levels, resetting of the game, and game completion.
5) We defined the “Car” class which contains attributes common to both the cars and also the functions common to both.
6) We defined the “Gamercar” class with the functions “bounce”, “decrease_speed”.
7) Then, we defined the “Computercar” class with the required functions in it
8) We then defined the “draw” function which displays the current level, time passed, and the velocity of the player car.
9) Furthermore, we defined the “move_player” function which regulates how the player car will move based on the given commands.
10) Then we defined the “manage_collision” function which handles the collision of both the cars with the finish line and the collision of the player car with the track border.
11) At last, we defined the main game loop which controls the game.

Tools, technologies, and frameworks used:

Python: Programming language used for developing the game
Pygame Library: We extracted major functions from the library required for the game

VS Code: The text editor used for writing, running, and debugging the code.

Conclusion and Future Work

Summary of outcomes and contributions:

The first major successful outcome was to define a proper smooth path for the computer car. The second outcome was to develop commands to control the movement of the player car. The third outcome was to define collision and its effects on the player car(like bouncing back). The fourth outcome was to play the right music at the right time. Finally, the last outcome was to successfully run the game.

Assessment of project success:

This game runs smoothly without any runtime errors. If we closely observe, in certain instances, the player car gets stuck for some time. We tried our best to debug this by adjusting the size of the car and using bounce. With this, the game provides a good racing experience to the user.

Lessons learned and recommendations for future improvements:

Lessons learned:

1) Debugging is an essential part of programming.

2) Python libraries such as pygame, math, and time.

3) Application of physics concepts like velocity, acceleration, collision, etc.

4) Efficient use of object-oriented programming.

Recommendations:

1) Creating obstacles for the player car.
2) Reward system and scoreboard
3) Game Menu which opens initially where it provides options for Starting a new game or continuing the game.
4) Providing scope for buying and using new cars.

Team Members' GitHub Accounts:

1) Mullaguri Tejaswini:https://github.com/Tejaswini-0607

2) Gubbala Sai Deepa:https://github.com/Deepa-1309

3) Beere Harshitha:https://github.com/Harshi-2706

References:

1) https://www.pygame.org/docs/

2) https://docs.python.org/3/library/math.html

3) https://docs.python.org/3/library/time.html

4)https://pythonprogramming.net/pygame-python-3-part-1-intro/

Appendices:

1) https://www.pygame.org/docs/ref/mixer.html