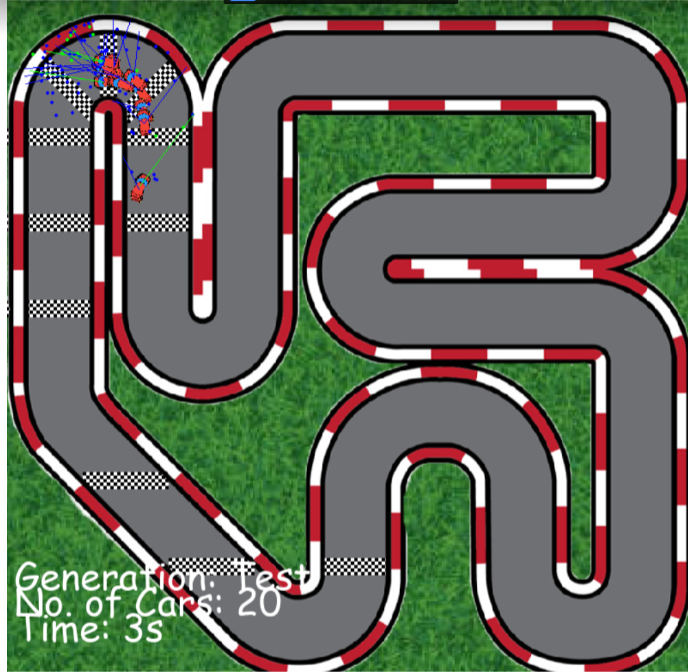
**Genetic Algorithms on Cars**

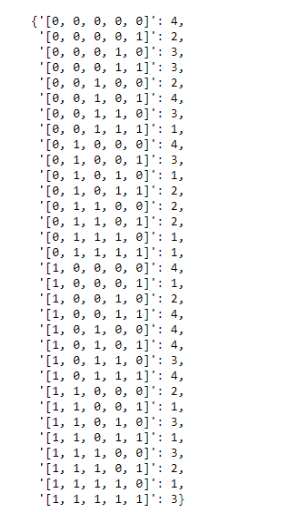
**Overview**

This project implements a genetic algorithm to optimize cars' navigation through a track with checkpoints. Each car is equipped with sensors that detect obstacles and determine movement directions based on predefined rules. The goal is to maximize the number of checkpoints crossed by evolving car behaviors across generations.



**How it Works**

* **Sensors**: Cars are equipped with 5 sensors (left, top-left, top, top-right, right), detecting obstacles.
* **Chromosome**: Each car has a chromosome dictating sensor combinations and corresponding movements.
* **Fitness**: Cars earn scores based on checkpoints crossed. Higher scores indicate better performance.
* **Algorithm**: Genetic algorithm iteratively evolves car behaviors, selecting, crossing over, and mutating chromosomes to optimize fitness.



**Features**

* **Selection**: Choose fittest cars for reproduction.
* **Crossover**: Combine genetic material of selected cars.
* **Mutation**: Introduce variations in chromosome values.
* **Goal Finding**: Identify cars completing all checkpoints.
* **Output Saving**: Save chromosome outputs.
* **Display**: Visualize algorithm output and test with provided code.

**Instructions**

1. Run **main.py** to start the genetic algorithm.
2. Check outputs in **outputs** directory.
3. The cars list must contain individuals belonging to the AgentCar class.
4. Children created must also be instances of the AgentCar class before being inserted back into the cars list.
5. When mutating the chromosome of each car, do NOT change the keys, only change the values and keep the values between 1 and 4 inclusive (since only 4 directional movements are implemented).
6. The time limit per generation is a self-adjusting one, calculated by the formula: (4 \* Score of Best car + 10). This is already implemented so no need to worry about it.
7. Code for saving your chromosomes and testing the goal population is provided.
8. You can create an instance of AgentCar by simply doing xyz = AgentCar().
9. There aren't many error checks in place, so your terminals might hang or crash if your code is going in the wrong direction. It's recommended to use print statements to debug your algorithm.

**Requirements**

* Python 3.x
* Dependencies: N/A

**Author**

Raffay Arshad