Interview questions

Create a folder of your name and submit files of your answers here: https://tinyurl.com/cdasimquestion

Question 1:

Suppose we developed a traffic simulation software where we can simulate vehicle movements at a signalized intersection. Now suppose we want to introduce a new feature to add **pedestrians** crossing simulation at traffic lights and interacting with vehicles realistically.

- Suppose we will develop a new module to model pedestrian behavior. What functions should be included in this module?
- How will you update the existing vehicle behavior to handle interactions with pedestrain?
- How will you update the traffic light control logic?

Question 2:

After adding the new pedestrain crossing feature, we observed that some pedestrains ignore red light and some vehicles do not stop and wait for pedestrains.

No errors were reported in the simulation.

How would you troubleshooting these abnormal behaviors?

Question 3:

This problem requires you to write an **actual Python program**.

Suppose we want to model different types of vehicles, design a code **structure** using an object-oriented approach to handle the three types of vehicles with two required methods below.

The test script for this problem is `Question3_TestScript.py`. We have the assumption that all your codes are inside one python script named `MyVehicleClass.py`. And we assume you use class names `Car`, `Minivan`, `Truck` to define the three types of vehicles.

Three types of vehicles with following attributes:

	Car	Minivan	Truck
Weight (kg)	1500	3000	8000
Maximum Acceleration (m/s^2)	8	5	3
Maximum speed (mph)	120	100	80
Fuel efficiency (MPG)	40	15	10

Two methods (If short of time, you can select to complete only one of the two methods)

1. `get_next_speed(current_speed, current_headway)` - Car-Following Model to determine its speed

Your task is to implement this `get_next_speed` function which updates the vehicle's speed. The $current\_acceleration$ is updated based on equation below:

$$current\_acceleration = a_{\max}\left(1 - \left(\frac{v}{v_{\max}}\right)^4 - \left(\frac{2 + 2 \cdot v}{s}\right)^2\right)$$

Where:

- v = Current speed (m/s)
- $v_{\max}$ = Maximum speed (m/s)
- $a_{\max}$ = Maximum acceleration (m/s^2)
- s = Headway/gap to the preceding vehicle (m)

Then the $next\_speed$ is calculated as

$$next\_speed = current\_speed + current\_acceleration$$

`Parameters:`

current_speed: float or **numpy float array**, m/s
the current speed of the vehicle

current_headway: float or **numpy float array**, meter
the current headway/gap of the vehicle with respect to the preceding vehicle

`Returns:`

next_speed: float, m/s
next speed of the vehicle

---

2. `get_safe_score(headway, relative_speed)` - Safety Metric (Time-to-Collision - TTC)to assess risk

Your task is to implement `get_safe_score()` using the formula below:

$$TTC = \frac{\text{headway}}{\text{relative speed}}$$

Assign the safety score as follows:

- If `relative_speed <= 0`, return **100** (no risk).
- Otherwise:
  - $TTC > 3$ → **Safe (Score = 100)**
  - $1 \leq TTC \leq 3$ → **Moderate Risk (Score = 50)**
  - $TTC < 1$ → **High Risk (Score = 0)**

`Parameters:`

headway: float or **numpy float array**, meter
the current headway/gap of the vehicle with respect to the preceding vehicle

relative_speed: float or **numpy float array**, m/s
the current relative_speed of the vehicle with respect to the preceding vehicle

`Returns:`

TTC: int, 0, 50, 100
time to collision score