# **Design Document for DSAD Assignment 1 – PS9**

## **Problem Statement:**

A car race is organized among N cars of the world. Every car has a speed (in km/hr) that is denoted by an array speed[ ] where speed[i] represents the speed of the ith car. The indexing of the array is 0-based.

All the cars are standing at different starting positions. The starting position of every car is represented by an array pos[ ] where pos[i] denotes the starting position of the ith car. The indexing of the array is 0-based. The pos[ ] array is a permutation of integers 0 to N-1. A car who is standing at i position is considered to be ahead of the car which is standing at position j if and only if i > j.

The finish line of the race is 100000 kilo meters ahead. An overtake occurs when car A, whose starting position is less than car B, finishes the race earlier than car B. Your task is to determine the number of overtakes that has occurred during the race.

## **Proposed Solution:**

## **Optimized Algorithm using Divide and Conquer Algo (Proposed Solution)**

* ***Idea:***
  + ***step 1:*** *(Divide) For each car, identify the cars behind it using position of the car*
  + **step 2:** (conquer) find how many of these cars have speed more than that of the current car because all these cars would overtake the current car.
  + **step 3:** (combine) find the total overtake by adding all these overtake.
* ***Code:***

Please refer to class ***get\_overtake\_count*** in DSAD\_assignment2\_main.py

It has following function:

1. ***get\_overtake\_counts()***: That Calculate the number of overtakes by the cars on finishing the race based on initial position and speed of cars

*Pseudocode:*

1. *get input no\_of\_cars, speed, position*
2. *For each car:*
   1. *Get the current\_speed and current\_position of car*
   2. *Iterate through the rest of car list and find the car behind the current car*
   3. *Check how many of these cars have speed more than current car*
   4. *This count would be number of cars overtaking current car*
   5. *Add this count to total\_overtake\_count*
3. *Return the total\_overtake\_count*

* ***Data Structure used:***

We have used list to store the speed and position of the cars

* ***Time Complexity of solution:***

*Total complexity for code is* ***O(N^2)*** *rough calculation can be seen below for same*

## **Time Complexity Calculation**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr.no** | **function name** | **Code line** | **Complexity** |
| 3 | ***get\_overtake\_counts*** |  |  |
| **overtake\_count = 0** | **O(1)** |
| **for i in range(self.no\_of\_cars):** |  |
| ***init\_position = self.pos[i]*** | **O(1) \* N => O(N)** |
| **speed\_of\_car = self.speed[i]** | **O(1) \* N => O(N)** |
| **overtaking\_cars = [j for j in range(self.no\_of\_cars) if self.pos[j] < init\_position and self.speed[j] > speed\_of\_car]** | ***O(N) \* N => O(N^2)*** |
| ***overtake\_count += len(overtaking\_cars)*** | **O(3) \* N = O(3N)** |
| **return overtake\_count** | ***O(1)*** |