Project (v1.0)

General Instructions:

(1) Timeline:
- Release of project requirements: 21 Oct 2019 (Mon, Week 10)
- Deadline for Q1: **30 Oct 2019**, 23:00hrs (Wed, Week 11) + release for Q2
- Deadline for Q2: **8 Nov 2019**, 23:00hrs (Fri, Week 12)

(2) Q1 is to be done in a team of 2 or 3 members. Your team allocation is found in a separate document in eLearn (Contents→Project). Each team submits one solution.

(3) Q2 is to be done individually. Each student submits one solution.

(4) Teams or individuals with the best algorithms for each section may be invited to present their solutions to the class in week 13/14.

(5) Refer to "Announcements" at red.smu.edu.sg for bug discoveries/fixes, dates and new data sets that will be provided to test your solutions.

(6) **WARNING**: Plagiarism is strictly not tolerated and plagiarism cases will be referred to the university's disciplinary committee. You MUST acknowledge all third-party contributions (including assistance acquired) in your write-up and list all sources in a reference list there.

Deliverables:

Please submit the following by the stipulated deadlines:

(1) For Q1, each team must submit:
- (i) a working **q1.py** to red.smu.edu.sg (the latest submitted version by any member of the team before the deadline will be taken to the final submission)
- (ii) an identical copy of **q1.py** to eLearn (Assignments)
- (iii) a write-up (PDF or Word doc) to eLearn (Assignments)
  - Include your team ID and names of all members on the write-up
  - Explain how your algorithm works and, if possible, derive the time complexity of your algorithm (with clear steps on how it is derived). Your explanation **may NOT exceed 1 A4-sized page**
  - Include your reference list and acknowledgement list (if any)
  - You will be graded on the clarity of your explanation and steps to derive the time complexity. Use "point form" and diagrams if relevant.

(2) For Q2, each student must submit:
- (i) a working **q2.py** to red.smu.edu.sg,
- (ii) an identical copy of **q2.py** to eLearn (Assignments)
- (iii) a write-up (maximum 1 page + acknowledgements/references) to eLearn (Assignments). Similar requirements apply.

Grading:

This project is worth ~~15%~~ 20% of your final course grade. Each question is worth ~~7.5%~~10%. This is the breakdown for each question:

(i) ~~1%~~ 1.5% for write-up

(ii) ~~2%~~ 3% for performance (time taken)

(iii) ~~4%~~ 5.5% for quality of algorithm

~~(iv) 0.5% to be awarded as bonus points to the top scorers in performance and quality~~

Scoring is competitive, and determined by your relative rank in terms of quality and performance on red. Your relative rank on the scoreboard at red is only indicative and not final. After the deadline, new data sets will be loaded to red in order to determine your final score. The time limit for each question, as well as errata announcements, will be shown as an "Announcement" at the home page of red.

The context

A delivery company receives multiple orders which need to be scheduled. Each order includes the following information: order ID and target location (in 2D coordinates). The delivery trucks must start and end at the company (which is at (0, 0)). All the items to be delivered are stored at the company in the beginning. The company has a limited number of trucks. All trucks are identical, and the speed of each truck is 1 length unit per time unit. Each truck can deliver more than one order at a time.

Here is the description of given CSV files:

- **orders.csv**: The list of orders is shown in this file with the following fields: order ID, x-coordinate, y-coordinate. Each line in this CSV file represents one order. For example, the following three lines represent three orders that need to be fulfilled:

  O00000, -39.0888931396, 14.9550391799
  O00001, -10.3413339776, -14.6545825259
  O00002, 25.7389470222, 32.5374354386

  The first order's ID is O00000 and a truck needs to deliver the item to (-39.0888931396, 14.9550391799).
- **parameters.csv:** The parameters are shown in this file with the following fields: number of trucks, truck speed and plane speed. Truck speed and plane speed are irrelevant for Q1 and will only be used in Q2. For example, the following in **parameters.csv** mean that up to 25 trucks are available for delivery purposes.

  25, 1, 1.1

Your solution should work with different **orders.csv** and **parameters.csv** files.

Q1

For this question, your task is to come up with an algorithm that outputs the delivery schedule of each truck. The schedules must ensure that all orders are delivered. Your algorithm should strive to minimize the time required to deliver all the orders. Since all trucks start off from the same location (0, 0) and travel at the same speed, the objective here then is to minimize the distance of the truck which travels the furthest distance based on its schedule. All distances are calculated as the Euclidean distance between two locations (i.e. the geometric between two points on the coordinate system).

You are required to fill up the body of this function in **q1.py**:

schedule_q1(orders, number_trucks)

where:

- **orders** is a list of items describing the order ID and delivery location of each order. Each item represents the order ID, its x-coordinate, its y-coordinate. This list may look like this:

  [[O001, 5.5, 3.4], [O002, -10.0, -8.9], …]
- **number_trucks** is a positive integer that denotes the number of available trucks.

The function should return a list of schedules in a list of items in which each item describes the schedule of each truck. The list contains the same number of items as the number of trucks used in your proposed schedule. This number may be equal to or less than the total number of available trucks (you need not use all the available trucks).

For example, assuming that you have 4 delivery trucks, and there are 10 orders to be fulfilled, your function may return this:

[[O002, O005, O003, O001], [O008, O004, O006, O010], [O007, O009]]

This particular solution deploys only 3 of the 4 available trucks. The first truck's schedule is **[O002, O005, O003, O001]**. This is to be interpreted as:

1) it starts from the company (0, 0) and goes to the delivery location of the order '**O002**' to fulfil the delivery.
2) it goes to the delivery location of the order '**O005**' to fulfil the delivery
3) it goes to the delivery location of the order '**O003**' to fulfil the delivery
4) it goes to the delivery location of the order '**O001**' to fulfil the delivery
5) it then returns to the company (0, 0).

In this case, the total mileage of the first truck would be the sum of Euclidian distances between:

1) The locations of the company and '**O002**'
2) The locations of '**O002**' and '**O005**'
3) The locations of '**O005**' and '**O003**'
4) The locations of '**O003**' and '**O001**'
5) The locations of '**O001**' and the company.

The total mileage of each of the other two trucks can be calculated in the same way. The mileage of the truck which travels the furthest will determine the time taken for all orders to be fulfilled. Remember that your algorithm aims to minimize this time.

The number of items in the returned list should be fewer or equal to **number_trucks.**

Your algorithm will be scored in 2 ways:

1) Performance: the time taken for your algorithm to complete (the faster the better)
2) Quality: the quality score will be the mileage of the deployed truck which travels the furthest as determined by your proposed schedule (the lower the better). Quality is more important than performance.

Q2 will be released near the deadline for Q1.

Notes:

- Q2 is to be done individually, and is an extension of Q1.
- Hence, it is very important that every student needs to be familiar with what the team has submitted for Q1 because (s)he is likely to build upon the team's solution in order to come up with a solution for Q2.