Design/Coding Exercise

- You may not use any external libraries to solve the problem itself, but you may use external libraries or tools for building or testing purposes.

How we evaluate your code

We will be looking at a number of things including the design aspect of your solution and your object oriented programming skills. Whilst these are small problems, we expect you to submit what you believe is “production-quality” code that you would be able to run, maintain and evolve. You don’t need to “gold plate” your solution, but we are looking for something more than a bare-bones algorithm. You should submit code that you would be happy to produce in a real project, or that you would be happy to receive from a colleague. We recommend you to provide adequate tests to indicate full code coverage. We also recommend you follow good code hygiene and clean code practices to reduce cyclomatic complexity of your classes.

How to submit your code

Please upload the code to github, and submit a link.

Problem: Airport Baggage

Denver International Airport has decided to give an automated baggage system another shot. The hardware and tracking systems from the previous attempt are still in place, they just need a system to route the baggage. The system will route baggage checked, connecting, and terminating in Denver.

You have been asked to implement a system that will route bags to their flights or the proper baggage claim. The input describes the airport conveyor system, the departing flights, and the bags to be routed. The output is the optimal routing to get bags to their destinations. Bags with a flight id of "ARRIVAL" are terminating in Denver are routed to Baggage Claim.

Input: The input consists of several sections. The beginning of each section is marked by a line starting: "# Section:"

```
Section 1: A weighted bi-directional graph describing the conveyor system.
Format: <Node 1> <Node 2> <travel_time>
Section 2: Departure list Format:
<flight_id> <flight_gate> <destination> <flight_time>
Section 3: Bag list Format:
<bag_number> <entry_point> <flight_id>
```

Output: The optimized route for each bag

```
<Bag_Number> <point_1> <point_2> [<point_3>, …] : <total_travel_time>
```

The output should be in the same order as the Bag list section of the input.

Example Input:

```
# Section: Conveyor System
Concourse_A_Ticketing A5 5
A5 BaggageClaim 5
A5 A10 4
A5 A1 6
A1 A2 1
A2 A3 1
A3 A4 1
A10 A9 1
A9 A8 1
A8 A7 1
A7 A6 1
# Section: Departures
UA10 A1 MIA 08:00
UA11 A1 LAX 09:00
UA12 A1 JFK 09:45
UA13 A2 JFK 08:30
UA14 A2 JFK 09:45
UA15 A2 JFK 10:00
UA16 A3 JFK 09:00
UA17 A4 MHT 09:15
UA18 A5 LAX 10:15
# Section: Bags
0001 Concourse_A_Ticketing UA12
0002 A5 UA17
0003 A2 UA10
0004 A8 UA18
0005 A7 ARRIVAL
```

Example Output:

```
0001 Concourse_A_Ticketing A5 A1 : 11
0002 A5 A1 A2 A3 A4 : 9
0003 A2 A1 : 1
0004 A8 A9 A10 A5 : 6
0005 A7 A8 A9 A10 A5 BaggageClaim : 12
```