COL106: Data Structures and Algorithms

Assignment 4

Instructor: Prof. Saroj Kaushik *Deadline: 23/04/18 11:55pm*

1 Statement

Amravathi, is a newly established township on the banks of river Krishna in Andhra Pradesh. The government is looking forward to strengthening the road network of the city in order to boost the development. At an initial stage, the government has decided a few prime junctions/points in the city (each junction has a value associated - lets call it **traffic_light_time**). The chief engineer has presented a list of roads to be built in the future and the time required for building each road (**build_time**: an integer representing number of months required) as well as the time required to move on the road from one end to another once the road is complete (**traverse_time**: an integer representing minutes required to go across the road).

In order to speed up things, we wish to connect all the identified prime junctions/points in the least possible time (assume work on roads cannot be simultaneous i.e. only work on one road can progress at any given time). This reduces problem to identification of the minimum spanning tree in the network of roads (assume traffic signals to be not functional yet). Note that this operation (of finding the MST) might be needed even after initial construction of the roads while repairing the roads.

Soon, all the roads in the original plan (irrespective of whether they were in the MST or not) are built. The government now wishes to develop a travel app for the citizens to ease their movement within the city. Any person wanting to travel by road, enters a source and a destination and hopes that the app returns the quickest way to reach the destination. This eventually boils down to the following:

You want to go from one node to another in the shortest amount of time. Time required to traverse for edge (road) e is $e.traverse\_time$ time units. At any junction x having the traffic signal value $x.traffic\_light\_time$, going through that node will force you to wait until the time spent is divisible by $x.traffic\_light\_time$.

NOTE: You are allowed (& encouraged) to use the priority_queue from the Standard Template Library (STL) (Hint: STL priority queue does not support updation outright. However, it is possible to still use it for implementation of Dijkstra's algorithm with a very slight change in time and space complexity)

You thus need to support the following operations:

- **add-junction** <x><y>: Add a junction x with $traffic\_light\_time = y$.
- **add-road** <i> <j> <m> <n>: Add a road between junction i and j with $build\_time = m$ and $traverse\_time = n$.
- **demolish-road** <i> <j>: Demolish the road between the prime point i and j.
- **print-mst**: Print the roads (represented as $(x\ y)$ where $x < y$ and the road connects junctions x and y) in the minimum spanning tree of the current junction-road graph ordered such that if a road $(x_1\ y_1)$ is placed before $(x_2\ y_2)$, then either $x_1 < x_2$ or if $x_1 = x_2$ then $y_1 < y_2$. If multiple

MSTs exist, then output any one of them. **You must use STL for sorting**. If there is tree spanning all vertices, just output -1.

- **quick-travel** <i> <j>: Print the minimum time needed to reach junction i starting at j at time 0. Also, output the junction ids in order of your suggested visit. If multipltiple such paths with the same travel time exist, then you are free to print any one of them. If it is not possible to reach i from j, then simply print -1.

2 Input Format

- All IO operations are to be done using standard input/output.
- The first line of the input contains two space separated integers V - the no. of prime junctions/points and E - the no. of proposed roads.
- Following V lines contains 2 space separated integers - $junction{-}id$ and corresponding $traffic\_light\_time$ - representing the V prime junctions.
- Following E lines contains 4 space separated integers - $junction - id - 1$, $junction - id - 2$, $build\_time$ and $traverse\_time$ corresponding to the E proposed roads.
- Next line has a single integer Q - the no. of operations to be performed.
- Q lines follow, each consisting of 1 operation in the following format: <operation-code> [params].

Code	**Operation**
1	add-junction
2	add-road
3	demolish-road
4	print-mst
5	quick-travel

3 Output Format

- Operations 1, 2 and 3 have no outputs.
- Operation 4 outputs the roads (one road per line) to keep in the MST in sorted order as described in the operation description, where each edge is represented by space separated junction-ids in ascending order.
- Operation 5 must output, in a single line, the minimum time, along with the junction-ids in expected order of visit (including the source and the destination junction ids), separated by spaces.

4 Limits

- $0 \leq V, E \leq 10^6$
- $1 \leq traverse\_time, build\_time, traffic\_light\_time \leq 10^9$
- For every junction, the $0 \leq junction - id \leq 10^6$
- At any time there is atmost 1 road between any 2 junctions

5 Sample IO

5.1 Input

```
4 4
2 7
3 5
1 3
4 4
2 3 10 2
4 2 1 1
4 3 3 2
1 3 3 2
7
4
1 5 10
2 5 3 3 10
1 10 4
2 10 1 3 2
2 4 10 2 2
5 10 2
```

5.2 Output

```
1 3
2 4
3 4
3 4 10 1 2
7 10 1 3 2
```

6 Submission Instructions

- You are required strictly follow the instructions given below.
- You must submit a single program file which must be named with your *KerberosID.cpp* in small case. For example, if your kerberos id is ch1150999 then the file name should be ch1150999.cpp. WARNING: Do not submit any other file formats just renaming them to cpp.
- **You will be penalized if your submission does not conform to this requirement**.

7 Other Instructions

- The assignment has to be done individually.
- All submissions must use C++ for the programming assignment.
- You are **encouraged** to use the standard template library (STL) to ease out your implementation of the assignment. It will be useful in almost all your future projects that require you to write C++ code.
- You are not allowed to discuss or take help from anybody outside the class. You may only discuss but are not allowed to share code with anyone inside the class.

- We will run plagiarism detection on your submissions. People found guilty will be penalised as per the instructions mentioned in the beginning of the course. Copying code or parts of code from sources available on the Internet is as much unacceptable as copying from one another.