ECM5605 (5076) S'18: Homework 02

Graph Problem on Real Topology

DUE: 2018/06/12 23:59:59

In this homework, you need to implement some graph algorithms. By analyzing the number of nodes and edges, you have to find a proper data structure and algorithm to outperform your classmates. **Please upload a zip file (<student_id>.zip) with two files on E3: (1) report that include your design concept and (2) source code for the problem (pb.cpp).**

[Problem 1] (100%) Single Source Shortest Path

The shortest path problem is the problem of finding a path between two vertices (or nodes) in a graph such that the sum of the weights of its constituent edges is minimized. Single source shortest path (SSSP) problem is to find the shortest path between a vertex to all the other vertices. In this homework, you are going to solve the SSSP problem efficiently with your implementation.

There will be several different test cases for you to evaluate the performance of your implementation. Some of these test cases are from real network topology. For example, there is Flickr image relationship network with relating rate as "weight", or some Gnutella peer-to-peer file sharing networks with connecting costs as "weight".

These test cases are all in the same format in <.in> files and the networks are all indirected graphs. During the performance evaluation, a <.in> file and a source node will be fed into the program. Your program should calculate the shortest (minimal weight) path from the given source to EVERY other node. **(Note that the weight might be negative.)** The following is the format of the files:

Input file <.in>

Format:

Nodes: <Node Number> Edges: <Edge Number>
< Node ID 1> <Node ID 2> <Weight> <= this is an edge

(Note that node ID is not always started by 1)

Example:

```
Nodes: 105938      Edges: 2316948
512        321        4.2
545        351        -2.4
127        5013       5.2
45         8145       -3.1
```

Output file <.out>

Format:

```
Nodes: <Node Number in the shortest path>
Runtime: <CPU runtime in mini-second> ms
Memory Usage: <memory usage in kByte> kB
<Node1 ID>      <Cost to the Node1>
<Node2 ID>      <Cost to the Node2>
<Node3 ID>      <Cost to the Node3>          <= Sorted by Node ID
```

Example:

```
Nodes: 105938
Runtime: 350 ms
Memory Usage: 125 kB
254     4.2
324     2.4
625     5.2
952     -9999                               <= If the vertex is isolated
```

Execution

Format:

```
./ans   <input.in>   <source node>   <out.out>
```

Example:

```
./ans   test1.in   50   test1.out
```

Performance Measurement

Please measure the run time **(from the very beginning of read file to the very end after write file.)**, and memory usage. You can refer to the measurement functions in previous homework.

Your program will be ranked by both runtime and memory, separately and follows the score table below for each test case.

Performance	Runtime	Memory
$x \leq 10\%$	70	30
$10\% < x \leq 20\%$	65	28
$20\% < x \leq 30\%$	60	26
$30\% < x \leq 50\%$	55	24
$50\% < x \leq 75\%$	50	22
$75\% < x$	45	20

Note

1. The weight might be negative!
2. Remember to upload the report as well.
3. Node ID are unsigned long integers.
4. In the output file, the cost should be sorted by the Node ID.
5. Please put the runtime measurement function in the right place.
6. Please follow the formats! Otherwise, some points will be taken off.