CS 211 Data Structures and Algorithms Lab
July -- December, 2018
Assignment 10
Total Marks: 10
Due on 3rd November

The objective of this assignment is to implement Dijkstra's algorithm to find shortest path distances from a source vertex to every vertex in the input graph, which is directed and has non-negative weights on edges.

Inputs

Your program should accept two command-line arguments: an input file and the label of a source vertex. A typical execution of your program will be ./a.out sample.graph 34.

The input file represents a directed graph with non-negative integer weights on edges. Every node in the graph is uniquely labelled with a non-negative integer. Every line in the input file is of the form *x y w*, which represents a directed edge from node *x* to node *y*, where the weight of the edge is *w*. No edge is repeated in the input file. The second command-line argument is the label of a source vertex, which is guaranteed to be a vertex in the given graph.

Task

Implement Dijkstra's algorithm to find shortest path distances from the given source vertex to all vertices in the given graph. It is recommended that you use min-priority queue with the binary min-heap implementation, but a simpler implementation is also accepted with full credits.

Output

Your program should create a file named 'dijkstra.txt'. Every line in the output file should corresponds to a shortest path distance from source to a vertex and should be of the form:

<target> <shortest-path-distance-from-source>

For example, if there is a line "21 10023" in the output file and 34 is the source vertex, then it implies that the shortest path distance from 34 to 21 is 10023. If there is no path from the source to a vertex, say 59, then the corresponding output line must be "59 -1".

Shortest path distances of vertices from the source can be printed in the output file in any order.

Submission and Evaluation

- The program you submit should output a file named 'dijkstra.txt' when run.

- There should be only one main file and it should be named as main.<extension>, where the extension depends on the language you choose (You must use either C or C++).
- Test well before submission. We have some hidden inputs with us to test your program. The mark you obtain is purely based on whether your program correctly gives outputs for the hidden inputs.
- Submit your code as a .zip file (even if there is only one file) where the name of the zip file is your roll number**.** It is important that you follow the input/output conventions exactly (including the naming scheme) as we may be doing an automated evaluation.
- **Penalty for not following the naming conventions is 5% of the marks obtained.**
- This assignment is due on 3rd November. Penalty for late submission is 5% per week; i.e., if you submit on 5th November, you will get only 95% of the mark you deserve otherwise.
- Follow some coding style uniformly. Provide proper comments in your code.
- Submit only through Moodle. Submit well in advance. Any hiccups in the Moodle/internet at the last minute is never acceptable as an excuse for late submission.
- Acknowledge the people (other than the instructor and TAs) who helped you to solve this assignment. The details of the help you received and the names of the people who helped you (including internet sources, if applicable) should come in the main file or in a separate file (acknowledge.txt). Copying others' programs is a serious offence.
- Honesty policy of the institute will be strictly followed. Note that we have access to a very good software to check plagiarism.