Assignment 8:

I. Overview

In this assignment, you will write a program to use Dijkstra's Algorithm to solve the single source shortest path problem:

Read a directed weighted graph G from an input file:

small\_graph.txt

The first line of the file contains n, the number of vertices of G. The names of the vertices are the integers from 1 to n.

Each remaining line of the graph is a list of integers separated by spaces, and represents the list of outneighbors of one vertex.

The first number in the line is the name of the vertex.

After that, there is a sequence of pairs, consisting of a vertex name and a weight.

The source will be vertex 1.

You are to print a shortest path from the source to each other vertex.

Here is a small output if the input is a small graph:

sample\_output.txt

There will be no more than 1000 vertices in the graph.

A large graph file is here:

large\_graph.txt

II. Requirements:

Your program should be written in Java programming language. In addtion, you are not allowed to import any package (except java.util.Scanner) that performs heap operations, such as decrease(), insert() and delete(). The only variable types allowed in your program are int and int[]. Instead, you should use your implemented HeapPQueue class you have completed in HW7. Therefore, you are required to use PQueue interface (PQueue.java) for this assignment. Again, if your submitted code is not compliant with this requirement, your HW8 will not be graded and you will receive 0 credit.

Hint1: A slight modification of your Heap code in HW7 is required. As I indicated, you should use vertex ID when you store the vertices in your heap, but the key values (of those vertices) are stored separately, e.g., an array (indexed by vertex IDs). Since both vertex ID and the key values are integers, int, int[] are enough for this assignment. No other data type is required.

Hint2: To store the adjacency list of an input graph, you can use an array (using vertex ID as the index) to store the lists. One list for each vertex. You will have an array of lists. You can represent a list by using an array (exactly as the input file).

For example, for vertex 1, you can have:

adjacency\_list[1] is an int array with elements 2, 5, 6, 3, 8, 4 (vertex 1 is connected two vertices 2, 6, 8 with their corresponding weights of edges 5, 3, 4, respectively).

Write all your own code by yourself. You are not allowed to use "off the shelf" code from any source, or from your classmates.

Your program (by default) will read the input from keyboard (the format of the input file has already been provided). Your program does not read a file, because the file input can be done by using input redirection in Linux. For example, if your test program is Test.java, the execusion command should be "java Test < small\_graph.txt".

The processed vertices should be stored in a priority queue implemented as a heap. The heap should contain the names of the vertices, but the key is the value.

In your program, each vertex record should have fields:

whether the vertex is visited;

the current back pointer of that vertex;

value: the current minium weight of any path from 1 to that vertex;

the current index of that vertex in the heap.