CS 3345 Programming Project 3

Write a program to:
Read an edge list for an **undirected, weighted graph** from a file named **Graphs.txt** and
1. use Dijkstra's Single Source Shortest Path Algorithm to construct the shortest path from a given vertex S to all other vertices.
2. use Kruskal's algorithm to construct a minimal spanning tree for the graph.

Input:
The input will begin with a line containing two space-separated integers, N giving the number of vertices in the graph (vertices are numbered 1,2,3,. . . ,N), and S giving the source vertex for Dijkstra's algorithm. Then the edge list will follow. For each graph edge there will be a line containing three space-separated integers, vertex, vertex, and weight. The edges are not listed in any particular order.

Here is an example input file:

```
7 1        // number of vertices and source vertex
1 2 2      // undirected edge from v1 to v2 of weight 2
1 4 1
2 5 10     // undirected edge from v2 to v5 with weight 10
2 4 3
5 7 6      // there will not be comments in the input
3 1 4
3 6 5
4 3 2
7 6 1
4 5 2
4 7 4
4 6 8
0 0 0
```

There will not be more than 1024 vertices. Edge weights will be in [1, 200]. The edge list will be terminated by a line containing three zeros.

Expected Output:
Use Dijkstra's algorithm to form shortest paths and output the lists of vertices on the shortest paths from the given vertex to all the others as follows:

```
1 1 0      // shortest path from 1 to 1 has length 0
1 2 2      // this list must be in increasing order of
1 4 3 3    // terminal vertex
1 4 1
1 4 5 3    // shortest path from 1 to 5 is via 4 and has length 3
1 4 7 6 6  // shortest path from 1 to 6 is via 4, 7 and has length 6
1 4 7 5    // do not print these comments
```

Then print a blank line followed by the minimal spanning tree as follows:

```
1 2   // each line must begin with the smaller vertex number
1 4   // lines with equal first numbers should be in order of the
3 4   // second number
4 5
4 7
6 7
```

Finally print a line giving the total length of the edges in the minimal spanning tree as follows:
Minimal spanning tree length = 12

Sample Input	Expected Output
7 1	1 1 0
1 2 2	1 2 2
1 4 1	1 4 3 3
2 5 10	1 4 1
2 4 3	1 4 5 3
5 7 6	1 4 7 6 6
3 1 4	1 4 7 5
3 6 5	
4 3 2	1 2
7 6 1	1 4
4 5 2	3 4
4 7 4	4 5
4 6 8	4 7
0 0 0	6 7
	Minimal spanning tree length = 12

RULES FOR PROGRAMMING AND SUBMISSION:

1. Your submitted code must be entirely your own. If you do copy code from a text or the web, cite the copied section with a comment. Points could be lost, depending on what is copied.
2. Write your program as one source file and remove the package construct from your Java source before submitting.
3. Name your source file as N1N2F1F2P3.java where your given name begins with the characters N1N2 and your family name begins with the characters F1F2. For example my name is Ivor Page, so my source file will be called IVPAP3.java. Note that in all but the java extension, all characters are upper case.
4. Your program's output must exactly match the format of the Expected Output above.
5. Do not use any Java Collection Classes except the String, arrays, and PriorityQueue. You may use Java's sort() function.
6. You program must read from **Graphs.txt** and output to **System.out**.
7. Use good style and layout and comment your code well.
8. Use the test files provided on the eLearning webpage for this class to test your program.
9. Submit your ONE source code file to the eLearning Assignment Dropbox for this project.
10. Don't submit a compressed file. Don't submit a .class file.

There will be a 1% penalty for each minute of lateness, up to 60 minutes. After 60 minutes of lateness a grade of zero will be recorded.

Send any questions/corrections to ivor@utdallas.edu.