Algorithms and Data Structures

MST Algorithm Assignment

You are required to implement both Prim's and Kruskal's algorithm for finding the minimum spanning tree for a weighted connected graph. If the implementation is too difficult and/or you can't debug it, a paper based simulation of both Prim and Kruskal will suffice to pass the assignment.

The program when run will request the name of a text file with a sample graph and a staring vertex. The user will then enter this name and vertex (as a number). The graph will then be read from the text file and a graph data structure will be constructed. A method should then be called to compute the graph's MST after which the MST should be outputted to the console. While the algorithm is running, it should output some of its workings to the console.

Implement Prim for a sparse graph using an array of adjacency lists. You are provided with some skeleton code to get you started. Instead, if you wish you can do a 2-D array version of Prim which is suitable for dense graphs.

Kruskal has only one version which uses an array of edges. Prim with adjacency lists requires a priority queue or heap. You may use a heap for Kruskal too. The heap code is quite different for each. With Kruskal you can use QuickSort or HeapSort to order the edges instead of using a heap. Also two improvements on Kruskal are*:* ***union by rank*** and ***path compression***, you are not required to implement these but path compression is quite simple to implement.

I have provided a sample graph for you and some of the code for the Graph and Heap classes. You should construct a text file myGraph.txt to store the second weighted that you created and drew yourself.

Make sure that

- your code is well commented and well structured
- messy code will lose marks, even if it works

Report

This is to be submitted thru Webcourses. It should include:

1. brief introduction
2. diagram of a sample input graph other than those in the notes, also save this graph to a file called myGraph.txt and paste its contents into your report near the graph diagram.
3. an adjacency lists diagram showing the graph representation of your sample graph.
4. a step by step construction of the MST for your sample graph which should include the contents of parent[] and dist[] arrays for Prim and the union-find partition and set representations for Kruskal.
5. **screen captures showing the output of all MST implementation**s for your sample graph.
6. a diagram showing the MST superimposed on the graph (use a highlighter if you want).
7. all documented program code for the algorithm (single line spacing size 10 font ok here)

For submission via Webcourses

1. Report in a file named **MST.pdf**. Make sure your name and student number is on the report cover page.
2. Two code files (Prim & Kruskal) and sample graph **myGraph.txt**. I only want one code file, **no RAR files**.