Algorithms and Data Structure

Exam Replacement / Assignment (70% of module)

DF Traversal & Prim's MST Algorithms

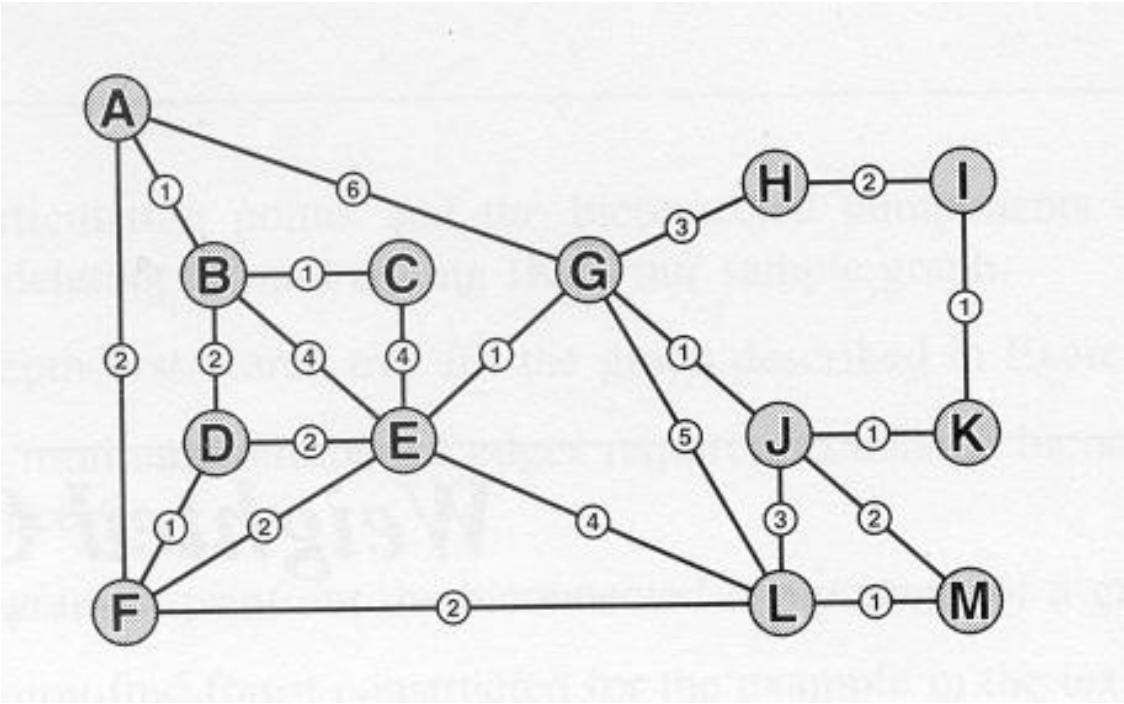
You are required to implement in Java:

1. Depth First (DF) Traversal of connected graph using both recursion and iteration.
2. A modification of DF to count the number of connected components in a disconnected graph.
3. Prim'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 will suffice to pass the assignment.

Two graphs that you construct yourself will be required to test the code. One for tasks 1 & 3, the other for task 2. In all three tasks, the graph is to be represented using an array of adjacency lists, as was discussed in class. Tasks 1 & 3 when working, could be in a single Java program, with different methods for each task, 2 methods for DF and 1 for MST. Task 2 will have to be a separate Java program because it require a disconnected graph as input.

The program DF and MST when run will request the name of a text file which it will load the graph from a text file into a data structure in memory and it will also request a staring vertex as input (for both DF and MST). The user will then enter this name and vertex (as a number or letter). You are to make up two of your own graphs, one connected and one disconnected, save them to a text file and test the code on them.

Also test DF and MST code on the following graph from a textbook by Sedgewick:



When DF traversal or Prim's algorithm are running, they should output some of their workings to the console. Also, after the MST has been computed, it should be displayed to the console.

You should construct 3 text files: sGraph.txt , myGraph.txt and myDiscGraph.txt to store the above graph from Sedgewick, your own one for DF and MST and thirdly your own disconnected graph.

If your code fails to compile or run correctly or you simply can't implement it, you can pass based on the quality of **your** written report.

Make sure that:

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

Report

This should include:

1. introduction/explanation of what you are going to do and a bit about the history of prim's algorithm.
2. diagrams of graphs you constructed yourself. Try to use a drawing tool.
3. adjacency lists diagrams showing the graph representation of both your sample graphs.
4. a section with detailed explanations showing how all three algorithms that you have been asked to implement will work on the graphs you constructed yourself. For MST, it should include the contents of parent[] and dist[] arrays as well as the heap. Diagrams are important here, even thought they will be time consuming to draw.
5. a section on implementation. No need to include whole programs in report, just some relevant part with explanatory text.
6. diagrams showing the output of DF and MST superimposed on the graph diagrams, and discuss if it corresponds with what you expected from step 4.
7. screen captures showing the output of for all implementations for the above graph and your sample graphs.
8. a discussion/analysis/reflection on what you learned or found useful in the assignment.

Save the report in a PDF file named **GraphReport.pdf**. Make sure your name and student number is on the report cover page as well as a declaration saying the report is your own work. If you cooperate with another student(s) in the coding, mention this on cover page and state their name(s).

For submission to Brightspace

The report, Java code files (two of them) and your sample graphs files **sGraph.txt, myGraph.txt** and **myDiscGraph.txt.**