COMP 221 Homework 7

Due: Friday, November 1, by 11:59 pm

1 Guidelines

Please **type up** and submit a PDF of your solutions. I recommend LaTex, but you are certainly welcome to use Google Docs or some other word processing program. Just make sure to save it as a PDF before submission. If you have to draw any pictures for a question (such as trees, graphs), you can do so **neatly** and then add a picture of it to your PDF file.

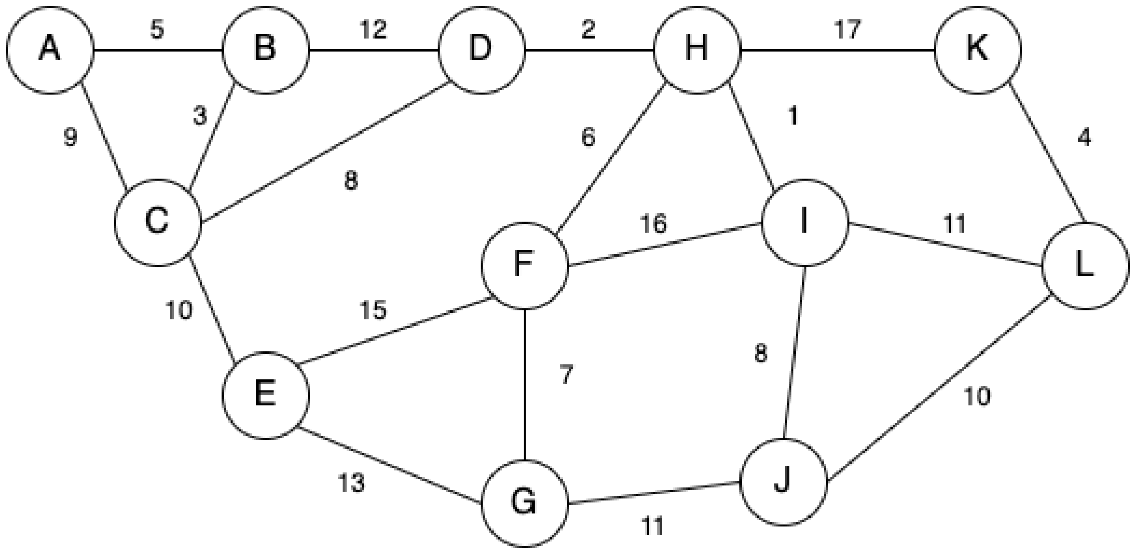
If a problem is explicitly marked as a **Programming** question, you will probably need to submit the source code for it (unless otherwise indicated). You should always be writing and submitting a README.md file that answers the requested questions.

Programming language: You are welcome to use Python or Java.

Do your own work: Homework should be individual work. You may discuss problems with other people, but you must write up your solutions yourself. I will not say that the web is off limits, but handing in solutions you find online as if they are your own is also not acceptable. Speaking of, most of these questions are shamelessly stolen from Susan :)

Make sure you have your name on your homework!

Use the graph below for the next three questions



- **Question 1**

  Find a minimum spanning tree (MST) of the graph above using Prim's algorithm.

- **Question 2**

  Find a MST of the graph using Kruskal's algorithm.

- **Question 3**

  Assuming node **F** is the source, use Dijkstra's Algorithm to find the shortest path to all other vertices.

- **Question 4 - Programming**

  Your task is to write a program that charts a path from a **source** three letter word to a **destination** three letter word using only "valid" words as intermediaries. Each intermediary must differ by only one letter.

  For example, to go from DOG to CAT, a possible sequence could be:

  DOG → COG → COT → CAT

  (note that this is just one I made up and may not be an actual solution)

  Your program will take three arguments:

  1. An input file containing a list of all three letter words, one per line.
  2. A source word (e.g. DOG).
  3. A destination word (e.g. CAT).

  Your program will find and print a 'track' from the source word to the target word, consisting of other words from the list, where each word varies from the previous one by one character.

Try to convert the words plus the neighbors (i.e., words that differ by only one letter) to a graph/network, and create edges between adjoining words. Then, use a BFS search starting at the source word and ending at the destination word. The `reconstructPath` algorithm that was provided after the BFS pseudocode may be useful after a successful BFS search.

The three letter word list is on Moodle with this homework.