**CPSC 6109:** [**Advanced**](https://colstate.view.usg.edu/d2l/lp/ouHome/home.d2l?ou=1218642) **Algorithms**

**Spring 2018**

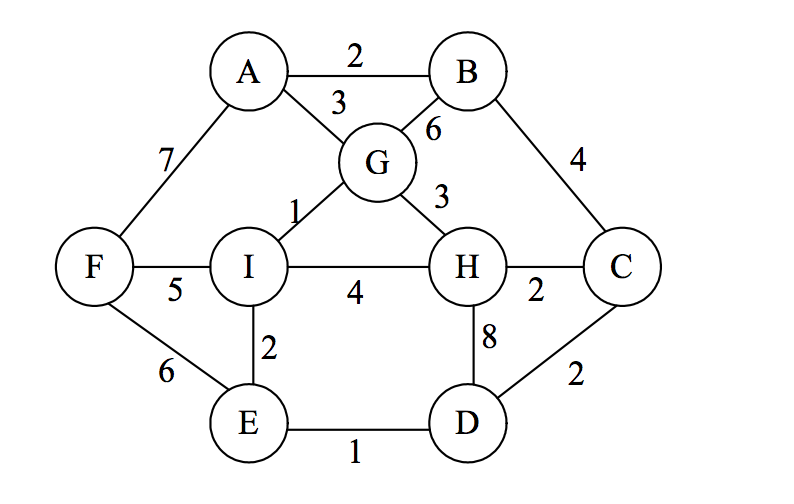
**Assignment #9**

**Student: Lu Lin**

**Due: 11:59 PM Tuesday, May 1**

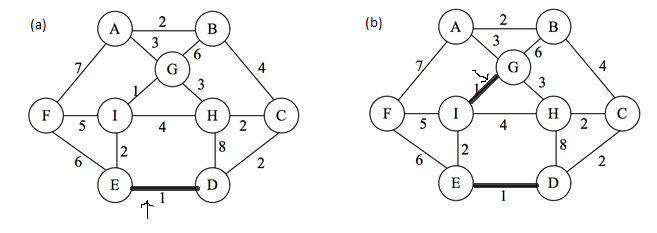
Do the following exercises/problems. Each problem is worth 50 points with a total of 100 points. **Problem 23-1 on page 638**, you need to complete parts (a) and (b). For parts (c) and (d), you ONLY need to do one of them, not both.

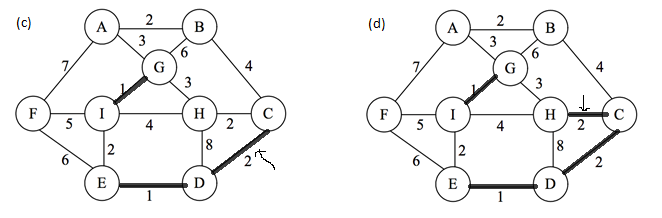
1. Run the Prim’s algorithm and the Kruskal’s algorithm to compute an Minimum Spanning Tree (MST) for the weighted graph below. For each algorithm, you must list all the edges in the MST **in the order they are added into the tree**.

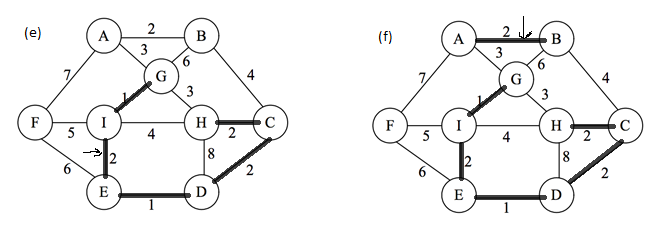


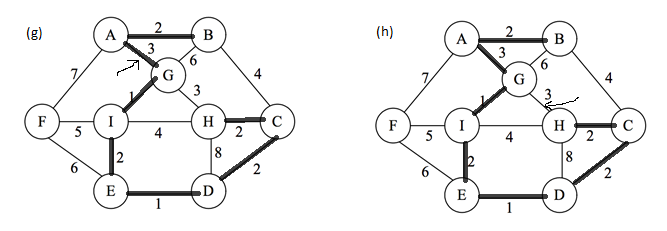
Solution:

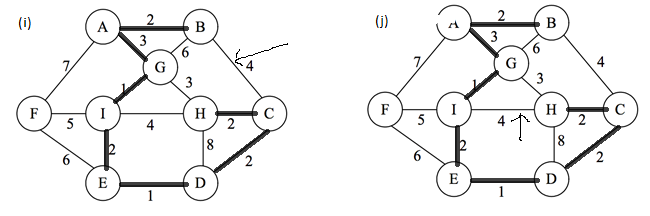
As textbook, further steps in the execution of Kruskal’s algorithm are as follows:

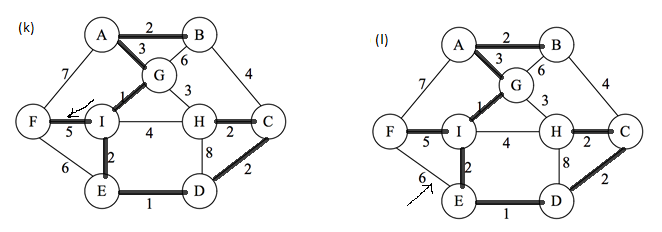


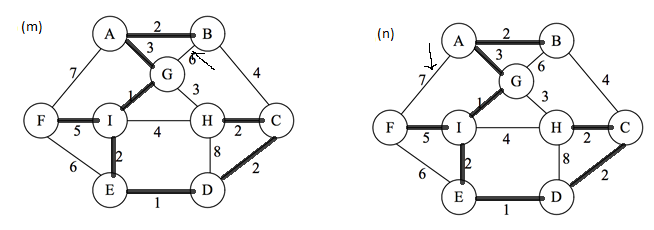


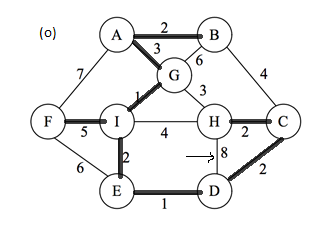








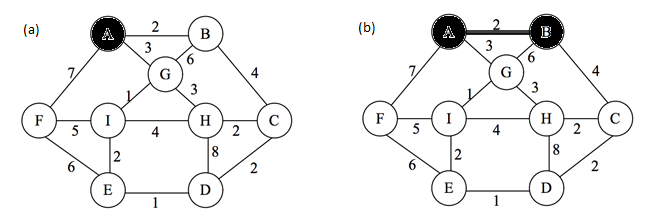


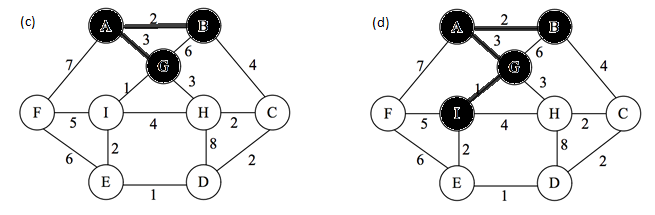


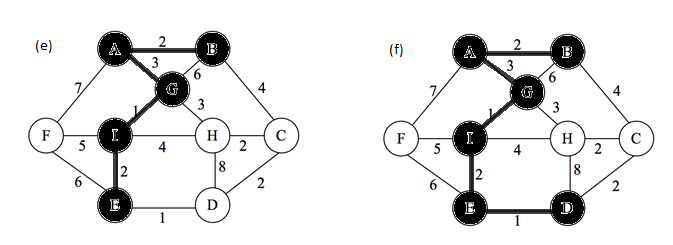
Using Kruskal’s algorithm edges in the MST in the order they are added into the tree are:

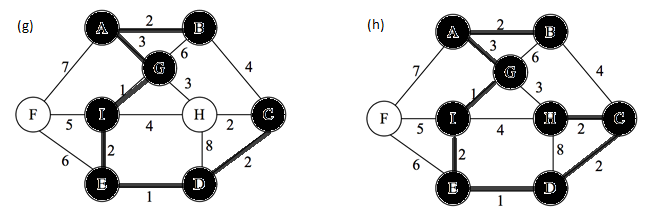
(E, D), (G, I), (D, C), (C, H), (I, E), (A, B), (A, G), (F, I)

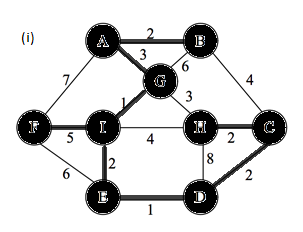
Take A as root vertex, further steps in the execution of Prim’s algorithm are as follows:











Using Prim’s algorithm edges in the MST in the order they are added into the tree are:

(A, B), (A, G), (G, I), (I, E), (E, D), (D, C), (C, H), (I, F)

1. Problem **23-1** on page 638. Second-best minimum spanning tree

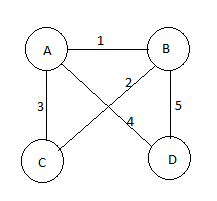
Solution:

1. Show that the minimum spanning tree is unique:

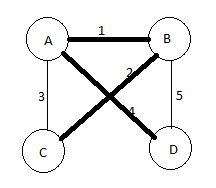
Assuming there are two minimum spanning trees called T1 and T2. For any edge e in T1, if we remove e from T1, then T1 becomes unconnected and we have a cut (S, V - S). Edge e is a light edge crossing cut (S, V - S) of the graph. If edge t is in T2 and through cut (S, V - S), then t is also a light weight one. Because the light edge is unique, we can view e and t is the same edge, e is also in T2. Because we choose any edge at random, of all edges in T1, it is also in T2. As a result, T1 is T2, the minimum spanning tree is unique.

Show that the second-best minimum spanning tree need not be unique:

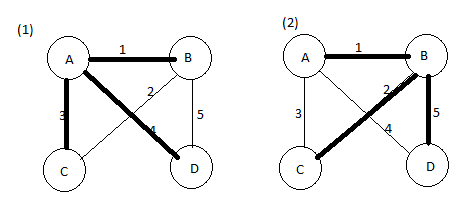
Here we consider the following example graph on four vertices. Suppose the vertices are {A, B, C, D} and the edge weights are as follows:



The minimum spanning tree is as follows, it has weight of 7, tree edges are (A, B), (B, C), (A, D)



There are two second best minimum spanning trees, they have weight of 8 as follows:



The tree edges for (1) are (A, B), (A, C), (A, D) and for (2) are (A, B), (B, C), (B, D)

1. My strategy is to prove that if we substitute two or more edges in the minimum spanning tree, we can not get a second-best minimum spanning tree. Let T1 be a MST and T2 be a second best MST, let (u, v) belongs to T1- T2. Then T2 (u, v) contains a cycle where one of the edges in the cycle is not in T1. Let this edge by (x, y). Then we must have w(x, y) > w(u, v), for otherwise we could replace (x, y) in T2 by (u, v) to get a MST better than T2. Now we note that S = T2 – (x, y) (u, v) is also a spanning tree since (u, v) and (x, y) are in the same cycle. In addition, w(S) < s(T2), so S is a best MST. By the uniqueness of MST proven in step a, S must be equal to T1. Therefore T1 and T2 differ with only one edge.
2. The path between two vertices u and v in the MST is unique. Here is the algorithm:

For each u ∊ V, perform either BFS or DFS to find the maximum weight between u and every other v ∊ V. Since T is a spanning tree of G, the BFS Tree or DFS Tree has only tree edges. When we visit edge (x, y) (from vertices x to y), the max-weight edge in the path from root u to y is found as follows:

max[x, y] =

For each u, the time for BFS or DFS is O(|V|) because a tree has only |V| - 1 edges. So the total time is O(|V|2)