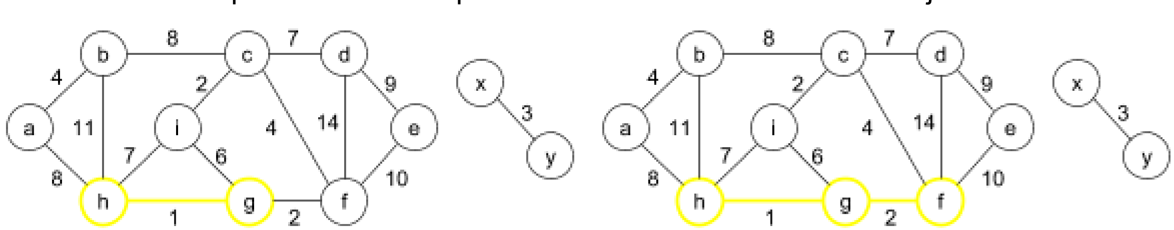
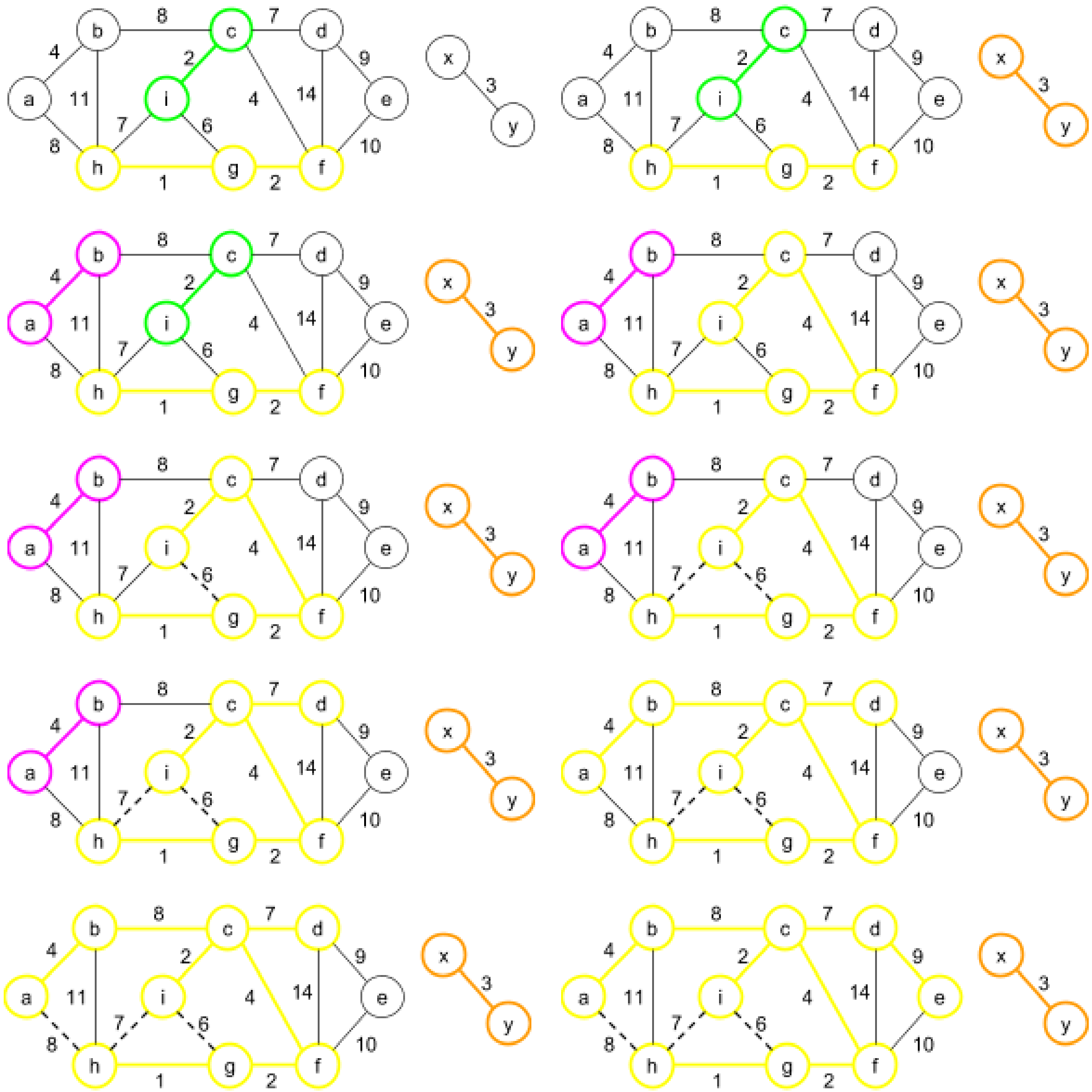
CS 1332 - Week 14

Monday

<u>Topics</u>

I. Kruskal's MST Algorithm
   A. Minimum Spanning Tree (MST)
      1. Spanning trees are acyclic, undirected, and connected
         a) Minimum spanning trees also minimize the total weight of the edges in the spanning tree
      2. MSTs of graphs are not necessarily unique, especially when some edges have the same weight
      3. For a graph with v vertices, the MST will be composed of (v-1) undirected edges, or 2(v-1) directed edges (where each undirected edge is represented by two directed edges)
   B. Requirements
      1. Priority Queue - holds all the edges in the edge list
      2. Disjoint Set - hold "clusters" of vertices to keep track of which vertices have already been connected in the MST
         a) find(v) - finds the "root" of the subset vertex v is in
         b) union(v, u) - combines the clusters containing v and u
      3. MST Set - holds the edges in the MST we're creating
   C. Algorithm
      1. Initialize Priority Queue (PQ) with all the edges from the graph
      2. Initialize Disjoint Set (DS) with all the vertices from the graph
      3. Initialize empty MST Edge Set (MST)
      4. While PQ is not empty and MST size is not valid yet
         a) e ← PQ.remove()
         b) if find(e.u) does not equal find(e.v)
            (1) union(e.u, e.v)
            (2) add e to the MST (if the graph is undirected, add the opposite edge as well: edge(e.v, e.u))
      5. Check if the MST is valid
         a) If the size is valid → return the list
         b) If the size is less than what it should be → throw it out
   D. Example: each color represents a different cluster in the disjoint set





E. Valid MST? No, there are only 9 edges in the MST when there should be 10.

No Content - Wednesday

No Content - Friday