MA/CSSE 473 Day 36 **minimal spanning tree for a connected graph G.**

1. The following lemma can be used to prove that Krushkal's algorithm produces a MST.
   Lemma: Let G be a weighted connected graph with a MST T; let G′ be any subgraph of T, and let C be any connected component of G′.
   If we add to C an edge *e=(v,w)* that has minimum-weight among all of the edges that have one vertex in C and the other vertex not in C, then G has an MST that contains the union of G′ and *e*. [Let v be our name for the vertex of e that is in C, and w our name for the vertex of e that is not in C].

2. Use the above lemma to prove that Kruskal's algorithm is correct:
   Claim: After every step of Kruskal's algorithm, we have a set of edges that is part of an MST of G
   Proof of claim: Base case …

   Induction Assumption: before adding an edge we have a subgraph of an MST
   We must show that after adding the next edge we have a subgraph of an MST
   Details:

3. Data Structures for Prim's algorithm.

4. What are the operations for the Disjoint Set datatype?

   a.

   b.

   c.

5. Outline Kruskal's algorithm in terms of the disjoint set ADT.

6. Based on the high-level code, what can we say about efficiency of Kruskal algorithm (in terms of n= |V| and m= |E|)?

7. What is the simple representation we can use for a DisjointSet datatype?

8. (4) Using the above representation, write

   makeset(i):

   findset(i):

   mergetrees(i,j):

   union(i, j):

9. Suppose we always make shorter trees subtrees of taller trees.

10. Write new versions of `makeset` and `mergetrees`