15-351 / 15-650 / 02-613: Homework #1

Due: Jan. 30 by the start of class.
(note: we may go over the solutions in class, so no late homeworks will be accepted.)

You may discuss these problems with your current classmates, but you must write up your solutions independently, without using common notes or worksheets. You must indicate at the top of your homework who you worked with. Your write up should be clear, and concise. You are trying to convince a skeptical reader that your answers are correct. Your homework should be submitted via Canvas as a typeset PDF. A LaTeX tutorial and template are available on Piazza (linked under Resources) if you choose to use that system to typeset.

For problems asking for an algorithm: describe the algorithm, give an argument why it is correct, and an estimation of how fast it will run. For this assignment, your argument about run times need not be sophisticated: just argue that it will be practical.

1. Let G be a graph where every edge has a different weight (as we assumed in class). Show that the edge with the smallest edge weight overall is in the minimum spanning tree.

2. Let G be a graph where every edge has a different weight (as we assumed in class). Show that there is only one, unique minimum spanning tree in this case.

3. Let G be a graph of n nodes where every edge weight is either 1 or 2 (now edges can have the same weight). Let G' be G with edges of weight 2 removed. Show that the weight of the minimum spanning tree of G is $n - 2 + \text{cc}(G')$, where $\text{cc}(G')$ is the number of connected components of G'.

4. Let G be a graph with positive, integer edge weights where several edges may have the same weights. Show that, for all integers i, the number of edges of weight i is the same in every minimum spanning tree of G. *Hint: Start your answer with: "Suppose not. Let T_1 and T_2 be two minimum spanning trees that do not have the same number of edges of some weights."*

5. You are given an undirected graph $G = (V, E)$ with edge weights $d(u, v)$, which you can assume are all distinct, and you are given a minimum spanning tree T on G. You expect one of the edges of G to disappear at some time in the future, but you don't know which edge it will be, and when the edge does disappear, you'll need to find another minimum spanning tree very quickly. For example, the MST represents a communication network for stock traders, and if a link fails, you have to fix it as fast as possible.

   Give an efficient algorithm to preprocess T and G to label each edge e in T with another edge $r(e)$ of G so that if e disappears, adding $r(e)$ to the tree creates a new minimum spanning tree in the modified graph.