CS2223
HW#4

DUE: Tuesday, November 27

1 (4 points) Suppose the following algorithm is given a complete graph $G=(N,A)$ with distinct edge lengths, that is $(\forall a_1, a_2 \in A)\, length(a_1) \neq length(a_2)$. Prove that it either always produces a minimum spanning tree *T* or provide a counterexample to show that it fails.

$T \leftarrow \varnothing$
$N^* \leftarrow$ an arbitrary vertex of *N*
for *i* ← 1 **to** *n*-1 **do**
    let $a=\{v_i, v_j\}$ be a shortest edge from v_i, the last vertex to be added to N^*, and some vertex $v_j \in N \setminus N^*$
    $T \leftarrow T \cup \{a\}$
    $N^* \leftarrow N^* \cup \{v_j\}$

That is, the algorithm always adds a shortest edge between the vertex most recently added to N^* and a vertex not yet in N^*.

2. (5 points) Prove or give a counterexample to the following.

CONJECTURE: For any connected graph $G=(N,A)$ with distinct edge lengths, if there is an edge $a \in A$ and a cycle in *G* such that *a* is the shortest edge of the cycle, then *a* belongs to every minimum spanning tree.

3. (6 points) Suppose that in a *vertex-weighted graph* $G=(N,A)$ we extend the definition of *length* to be $length: N \cup A \rightarrow \mathbb{R}^+$, that is, we associate a length with each vertex **and** each edge. We define the length of path $v_i, a_i, v_{i+1}, a_{i+1}, ..., a_{j-1}, v_j$ to be $\sum_{i \le k \le j} v_k + \sum_{i \le k < j} a_k$. That is, the length of a path is the sum of the lengths of the vertices on the path **plus** the sum of the lengths of the edges on the path. Describe an algorithm to solve the single-source shortest path problem in a vertex-weighted graph. The worst-case execution time of your algorithm should be in $\Theta(n^2)$.