1. **Suppose we are given a directed graph G=(V,E) in which every edge has a distinct positive edge weight. A directed graph is acyclic if it has no directed cycle. Suppose that we want to compute the maximum-weight acyclic subgraph of G (where the weight of a subgraph is the sum of its edges’ weights). Assume that G is weakly connected, meaning that there is no cut with no edges crossing it in either direction.**

**Here is an analog of Prim’s algorithm for directed graphs. Start from an arbitrary vertex s, initialize S={s} and F=∅. While S≠V, find the maximum-weight edge (u,v) with one endpoint in S and one endpoint in V−S. Add this edge to F, and add the appropriate endpoint to S.**

**Here is an analog of Kruskal’s algorithm. Sort the edges from highest to lowest weight. Initialize F=∅. Scan through the edges; at each iteration, add the current edge i to F if and only if it does not create a directed cycle.**

1. **Consider a connected undirected graph G with edge costs that are not necessarily distinct. Suppose we replace each edge cost ce by −ce; call this new graph G′. Consider running either Kruskal’s or Prim’s minimum spanning tree algorithm on G′, with ties between edge costs broken arbitrarily, and possibly differently, in each algorithm. Which of the following is true?**
2. **Consider the following algorithm that attempts to compute a minimum spanning tree of a connected undirected graph G with distinct edge costs. First, sort the edges in decreasing cost order (i.e., the opposite of Kruskal’s algorithm). Initialize T to be all edges of G. Scan through the edges (in the sorted order), and remove the current edge from T if and only if it lies on a cycle of T.**

**Which of the following statements is true?**

1. **Consider an undirected graph G=(V,E) where edge e∈E has cost ce. A minimum bottleneck spanning tree T is a spanning tree that minimizes the maximum edge cost maxe∈Tce. Which of the following statements is true? Assume that the edge costs are distinct.**
2. **You are given a connected undirected graph G with distinct edge costs, in adjacency list representation. You are also given the edges of a minimum spanning tree T of G. This question asks how quickly you can recompute the MST if we change the cost of a single edge. Which of the following are true? [RECALL: It is not known how to deterministically compute an MST from scratch in O(m) time, where m is the number of edges of G.] [Check all that apply.]**

**Solution :** Suppose e∈T and we increase the cost of e. Then, the new MST can be recomputed in O(m) deterministic time(Let A,B be the two connected components of T−{e}. Edge e no longer belongs to the new MST if and only if it is no longer the cheapest edge crossing the cut (A,B) (this can be checked in O(m) time). If f is the new cheapest edge crossing (A,B), then the new MST is T−{e}∪{f})

Suppose e∈T and we decrease the cost of e. Then, the new MST can be recomputed in O(m) deterministic time(The MST does not change (by the Cut Property), so no re-computation is needed)

Suppose e∉T and we decrease the cost of e. Then, the new MST can be recomputed in O(m) deterministic time(Let C be the cycle of T∪{e}. Edge e belongs to the new MST if and only if it is no longer the most expensive edge of C (this can be checked in O(n) time). If f is the new most expensive edge of C, then the new MST is T∪{e}−{f})

Suppose e∉T and we increase the cost of e. Then, the new MST can be recomputed in O(m) deterministic time(The MST does not change (by the Cycle Property of the previous problem), so no re-computation is needed)