**7.17) a,b,c)** In all the 3 cases the run time will be O(NlogN).

**7.20) a)** O(NlogN)

**b)** O(NlogN)

**c)** O(N2 )

**7.22) a)** The run-time is O(NlogN)

**b)**

**9.1)** s, G, D, H, A, B, E, I, F, C, t.

**9.2)** Using a queue, the topological order that results is:

s, G, D, H, A, B, E, I, F, C, t.

Using a stack, the topological order that results is:

s, G, H, D, A, E, I, F, B, C, t.

A topological sort that uses queues produces a more natural ordering as it processes vertices in the same manner as a breadth-first search.

**9.16)** Yes, Prim’s and Krushkal’s algorithms do work with negative edge weights.

**9.20)** The code would be written like this using Krushal’s method:

1. Sort the edges of G into decreasing order by weight. Let T be the set of edges comprising the maximum weight spanning tree. And we’d set T = (empty set)

2. Then we add the first edge to T

3. Then we add the next edge to T making sure it does not form a cycle in T. If there aren’t any remaining edges we will exit and report G to be disconnected.

4. If T has (n-1) edges, we stop and output “T”. Other we go back to step 3.