CS302 - Problem Set 11

1. **[6 points]** What is the worst case runtime of the following algorithm for depth-first search? Explain.

```
Algorithm 1: DepthFirstSearch(A, X, s, f)

Input : Adjacency list A for a graph G = (V, E), an array X
        of length |V| such that X[v] = 1 if v has been
        explored and 0 otherwise, a starting vertex s, a goal
        vertex f

Output: String “f found!” or “f not found” depending on
        whether f can be found from s.

1 if s == f then
2     Return “f found!”;
3 else
4     X[s] = 1;
5     d = A[s].length;
6     for k = 1 to d do
7         if X[A[s,k]] == 0 then
8             DepthFirstSearch(A, X[A[s,k]], f);
7     end
10 end
11 Return “f not found”
```

2. **[9 points]** Describe (using pseudocode) a modification of Dijkstra’s algorithm that solves this problem. (Hint - the algorithm stays the same, just the criterion changes.) I’ve started it for you:

```
Algorithm 2: BottleNeck(A, s)

Input : Graph G(V, E) with positive edge weights l(u, v), and a vertex
        s ∈ V.

Output: Array B such that B[v] is the path from s to v that has the
        minimum bottleneck of all paths from s to v, and an array A
        such that A[v] is the value of the bottleneck of the path B[v].
```

(b) **[11 points]** Prove your algorithm finds the path with smallest bottleneck from s to every other vertex in G.
3. Detecting Negative Cycles

(a) [6 points] In the version of Bellman-Ford we looked at in class, we assume that there are no negative cycles in the graph. Describe using words how you would change the Bellman Ford algorithm either detect a negative cycle, or return the shortest path if there are no negative cycles. Also explain why your modification is correct.

(b) [6 points] What is the runtime of your modified algorithm. Explain.