Homework #5
CMSC351 - Spring 2013

PRINT Name

- Grades depend on neatness and clarity.
- Write your answers with enough detail about your approach and concepts used, so that the grader will be able to understand it easily. You should ALWAYS prove the correctness of your algorithms either directly or by referring to a proof in the book.
- Write your answers in the spaces provided. If needed, attach other pages.
- The grades would be out of 16. Four problems would be selected and everyone’s grade would be based only on those problems. You will also get 4 bonus points for trying to solve all problems.

1. [Prob 7.2, pg. 248] Let $G = (V, E)$ be a connected, undirected graph, and let $T$ be a DFS tree of $G$ rooted at $v$.
   a. Let $H$ be an arbitrary induced subgraph of $G$. Show that the intersection of $H$ and $T$ is not necessarily a spanning tree of $H$.

   b. Let $R$ be a subtree of $T$, and let $S$ be the subgraph of $G$ induced by the vertices in $R$. Prove that $R$ could be a DFS tree of $S$. 
2. Let $G$ be a directed acyclic graph. Does $G$ have a unique topological ordering? If so, prove it or else give an example of a directed acyclic graph having at least two topological orderings.
3. Consider the undirected graph below which is represented by its adjacency matrix.
   a. Run the DFS algorithm starting from vertex 1, and draw the DFS tree.
   b. Run the BFS algorithm starting from vertex 1, and draw the BFS tree.

\[
\begin{bmatrix}
1 & 1 & 0 & 0 & 0 & 1 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 1 & 1 & 1 & 0 \\
1 & 0 & 1 & 1 & 1 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1
\end{bmatrix}
\]
4. Give an example that the Dijkstra algorithm is not working properly with negative edge weights. The graph should not contain a negative cycle (a negative cycle is a cycle which the sum of the weights of its edges is negative). Briefly explain why Dijkstra does not work in your example.
5. Consider the weighted graph below which is represented by its adjacency matrix.
   
a. Run the Dijkstra algorithm starting from vertex 1. Write the vertices in the order which they are marked.

b. Run the Prim’s algorithm starting from vertex 1. Again write the vertices in the order which they are marked.

\[
\begin{bmatrix}
0 & 5 & 0 & 0 & 8 & 0 & 6 \\
5 & 0 & 0 & 7 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 3 \\
0 & 7 & 0 & 0 & 2 & 0 & 0 \\
8 & 0 & 0 & 2 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 4 \\
6 & 0 & 3 & 0 & 0 & 4 & 0
\end{bmatrix}
\]
6. Recall that given a graph $G$, the Bellman-Ford algorithm runs for $n - 1$ iterations, where $n$ is the number of vertices in $G$. Prove that $G$ contains a negative cycle, if and only if at least one edge gets relaxed if we run the algorithm for one extra iteration.

```plaintext
procedure BellmanFord(list vertices, list edges, vertex source)
// Step 1: initialize graph
for each vertex v in vertices:
    if v is source then v.distance := 0
    else v.distance := infinity

// Step 2: relax edges repeatedly
for i from 1 to size(vertices)-1:
    for each edge uv in edges: // Try relaxing the edge from u to v
        u := uv.source
        v := uv.destination
        if u.distance + uv.weight < v.distance: // uv gets relaxed
            v.distance := u.distance + uv.weight
```
7. Suppose that \( G = (V, E) \) is a connected graph with weights on the edges. Show that the following algorithm gives a minimum cost spanning tree of \( G \):

- Arrange the edges in \( E \) in decreasing order of weights
- Traverse the edges according this order
- For each edge, keep the edge if its deletion disconnects the graph. Otherwise delete the edge.

**Hint:** The proof of correctness is similar to the one given in Section 7.6 of the book.
8. A Hamiltonian path in graph $G=(V,E)$ is a simple path that includes every vertex in $V$. Design an algorithm that runs in $O(n+m)$ time, to determine if a Hamiltonian path exists in a given directed acyclic graph.
9. [Prob. 7.61, pg. 256]. Let \( G=(V,E) \) be a connected, undirected graph, and let \( T \) be a minimum cost spanning tree of \( G \). Suppose the cost of one edge \( e \) in \( G \) is changed. Discuss the conditions in which \( T \) is no longer a MCST of \( G \). Design an efficient algorithm to either find a new MCST or to determine that \( T \) is still an MCST. (\( e \) may or may not belong to \( T \)).