**HW3**

**Question 1 (30 Points):** Consider the above mentioned graph. Now,

I. Apply BFS considering 1 as the source node (level-0 node). Draw the BFS tree and mark each level. If there are multiple options at any step, pick the node with smaller index.

Schematic

Description automatically generated

II. Repeat part II considering 6 as the source node (level-0 node).

A picture containing diagram

Description automatically generated

**Question 2 (20 Points):** Consider the above mentioned graph with many job dependencies. Each node denotes a job and x -> y means job y is dependent on job x. Determine a feasible schedule by applying topological sort. If there are multiple options at any step, pick the job with smaller index.  
A picture containing text, orange, vector graphics

Description automatically generated

**A feasible schedule would be: 0-1-2-7-3-6-4-5**

**Question 3 (20 points):** Consider the following tree. All the nodes under the same parent are siblings. For example, 70 and 100 are siblings (25 is their parent). Is it possible to determine the siblings for any node using a queue? Why or why not? Describe you logic and provide a pseudocode with adequate comments.

**I believe that it is possible to determine the siblings for any node using a queue. You would take a tree with the source node, and enter all the children into an queue (all children of the same parent are siblings). Then, dequeue all elements and check to make sure they all have the same parent. If they do, then they’re siblings. Otherwise, they are not.**

Text

Description automatically generated

**Question 4 (30 points):** Consider the same graph mentioned in Question 1. Now,

I. Apply DFS considering 1 as the source node. If there are multiple options at any step, pick the node with smaller index. Draw the DFS tree. Also, mention all the back-edges. **BE next to an edge stands for “back edge” indicating a cycle in the graph.**

Diagram

Description automatically generated

II. Repeat part II considering 6 as the source node. **Same as part I, BE indicates a back edge.**

**Diagram, shape, schematic

Description automatically generated**