Name: Shreenidhi Deepak Pai

Student ID: 1002232249

DAA HANDS ON 13

Depth First Search:

Graph Representation:

- The graph is represented as an adjacency list, where each node points to a list of its neighbors.
- For example, 's': ['z'] means there is a directed edge from s to z.

DFS Function:

- The function recursively visits each node and its unvisited neighbors.
- A visited set is used to ensure nodes are not revisited, avoiding infinite loops in cyclic graphs.

Starting Node:

- The DFS traversal begins at node 's' and explores as far as possible along each branch before backtracking.

Kruskal's Algorithm:

Input Graph:

- Each edge is represented as a tuple (u, v, weight).
- Example: ('a', 'b', 4) means an edge exists between nodes a and b with a weight of 4.

Sorting the Edges:

- The edges are sorted by their weights to process the lightest edge first.

Union-Find (Disjoint Set):

- find(parent, node): Determines the root of the node using path compression.
- union(parent, rank, u, v): Combines two disjoint sets into one.

Building the MST:

- Process edges in increasing weight order.
- Add an edge to the MST if it does not form a cycle (checked using the find function).

Topological Sort:

Graph Representation:

- The directed graph is represented using an adjacency list.
- Example: 'm': ['q', 'r', 'x'] means m has directed edges to q, r, and x.

DFS Function:

- This function visits all the neighbors of a node recursively.
- After visiting all neighbors, the node is added to the stack.

Topological Sort:

- Iterate over all nodes in the graph. If a node is not visited, call the DFS function.
- After all nodes are processed, reverse the stack to get the topological order.

Output:

- The topological sort order is determined by the reverse of the stack's content.