ADDIS ABABA UNIVERSITY

INSTITUTE OF TECHNOLOGY

SCHOOL OF ELECTRICAL AND COMPUTER ENGINEERING

(COMPUTER STREAM)

Data Structure

Assignment 1

Name	ID
1. Ebisa Adugna ………………………………………	UGR/7063/12

Submitted to: Dr.Menore Tekela
Submission Date: June 5,2023

1. Pseudo-code

1.2 pseudocode for DFS and BFS algorithm

```
- Method dfs(startVertex):
    - Initialize a visited array of size numVertices with all elements set to   false
    - Create an empty stack
    - Push startVertex onto the stack and mark it as visited
    - While the stack is not empty:
        - Pop the top vertex from the stack
        - Print the vertex
        - For each adjacent vertex of the current vertex:
            - If it is not visited:
                - Push it onto the stack and mark it as visited

- Method bfs(startVertex):
    - Initialize a visited array of size numVertices with all elements set to false
    - Create an empty queue
    - Enqueue startVertex and mark it as visited
    - While the queue is not empty:
        - Dequeue the front vertex from the queue
        - Print the vertex
        - For each adjacent vertex of the current vertex:
            - If it is not visited:
                - Enqueue it and mark it as visited
```

1.2 Pseudocode for Prim's algorithm

```
Function primMST(graph, V):
    - Create a priority queue pq to store edges with their weights (using min-heap)
    - Create a vector minWeight to store the minimum weight for each vertex (initialized
with INT_MAX)
    - Create a vector parent to store the parent of each vertex in the MST (initialized with
-1)
    - Create a vector visited to track visited vertices (initialized with false)

    - Start with vertex 0
    - Push the pair (0, startVertex) into pq and set minWeight[startVertex] = 0
    - While pq is not empty:
        - Pop the top element u from pq
        - Mark vertex u as visited

        - For each adjacent vertex v of u:
            - If v is not visited and the weight of the edge (u, v) is less than minWeight[v]:
                - Update minWeight[v] with the weight of the edge (u, v)
                - Set parent[v] = u
                - Push the pair (minWeight[v], v) into pq

    - Print the MST path and shortest distance
```

1.3 Pseudocode for Kruskal's algorithm

```
Function kruskalMST(edges, V):
    - Sort the edges in ascending order of weight using compareEdges function
    - Initialize an empty vector mst to store the minimum spanning tree
    - Create a DisjointSet object ds with V vertices
    - Initialize totalWeight to 0

    - For each edge in edges:
        - Get the source and destination vertices of the edge
        - Find the root vertices of the source and destination using ds.find()
        - If the root vertices are different:
            - Add the edge to the mst vector
            - Add the weight of the edge to totalWeight
            - Union the sets containing the source and destination vertices using
ds.unionSets()

    - Print the MST path and total weight
```

1.4 Pseudocode for Kruskal's algorithm

```
function Dijkstra(graph, startVertex):
    for each vertex v in graph:
        v.dist = infinity
        v.known = false

    startVertex.dist = 0

    while there is an unknown vertex:
        v = closest unknown vertex
        v.known = true

        for each vertex w adjacent to v:
            if w.known is false:
                Cost From V to W = cost of edge from v to w

                if w.dist > v.dist + cost From V to W:
                    w.dist = v.dist + cost From V to W
                    w.path = v
```