CS 1203: Data Structures **Semester:** Monsoon 2023

Assignment #4

Instructor: Subhamoy Mitra
Due: November 10, 2023

Student Name: Varsha Baisane
varsha.baisane_ug25#@ashoka.edu.in

Question 2

Yes, the idea of storing path information in an array can be applied to both directed and undirected graphs. In the code, the implementation of dijkstra's algorithm, the graph is represented as as an adjacency matrix, and the algorithm works well for directed graphs as well.

In a directed graph, the edges entail a direction telling which vertex goes to whom. The adjacency matrix graph[u][v] represents the weight of the edge from vertex u to vertex v. The algorithm implemented thus checks the edges in both directions, and the graph representation remains the same regardless of whether it is directed or undirected.

As a result of which, the given implementation will work for directed graphs. The parent array will still store the predecessor of each node on the shortest path from the source node, and the printSolution function as defined in the code will correctly display the shortest distances and paths for directed graphs.

Question 3

The time complexity of the dijkstra's implementation is $O(n^2)$, where n is the number of vertices present in the graph. This is result of using an adjacency matrix, and for each vertex, the algorithm will have to iterate through all other vertices to find the minimum distance. The auxiliary space complexity is thus $O(n)$ for the dist, sptSet, and parent arrays.

To improve the time complexity, especially for sparse graphs, we can use a priority queue (min-heap) to efficiently extract the minimum distance vertex. This modified implementation further reduces the time complexity to $O((n + u) * log(n))$, where E is the number of edges in the graph and log n is the result of extracting elements using priority queues. The space complexity remains $O(n)$.