ASSIGNMENT 4

**Name: Soham Makarand Limaye**

**Roll No: 222038**

**PRN: 22111058**

**Batch: B2**

**Aim:**:A customer wants to travel from source A to destination B. He books a cab from source A to reach destination B. Calculate a shortest path by avoiding real time traffic to reach destination B

**Objective**:

1)To understand the basic concept of oops

2)To understand graph data structure 3)to understand the dijkstra algorithm

**Theory :**

1. Dijkstra’s algorithm allows us to find the shortest path between any two vertices of a graph.
2. It differs from the minimum spanning tree because the shortest distance between two vertices might not include all the vertices of the graph.
3. Djikstra used this property in the opposite direction i.e we overestimate the distance of each vertex from the starting vertex. Then we visit each node and its neighbors to find the shortest subpath to those neighbors.

**Code :**

public class Main { public static void dijkstra(int[][] graph, int source) { int count = graph.length; boolean[] visitedVertex = new boolean[count]; int[] distance = new int[count]; for (int i = 0; i < count; i++) { visitedVertex[i] = false; distance[i] =

Integer.MAX\_VALUE;

}

distance[source] = 0; for (int i = 0; i < count; i++)

{ int u = findMinDistance(distance, visitedVertex); visitedVertex[u] = true; for (int v =

0; v < count; v++) {

if (!visitedVertex[v] && graph[u][v] != 0 && (distance[u] +

graph[u][v] < distance[v])) { distance[v] = distance[u] + graph[u][v];

}

}

}

for (int i = 0; i < distance.length; i++) {

System.out.println(String.format("Distance from %s to %s is %s", source, i, distance[i]));

}

}

private static int findMinDistance(int[] distance, boolean[]

visitedVertex) { int minDistance =

Integer.MAX\_VALUE; int minDistanceVertex = -1;

for (int i = 0; i < distance.length; i++) { if

(!visitedVertex[i] && distance[i] < minDistance)

{ minDistance = distance[i]; minDistanceVertex = i;

}

}

return minDistanceVertex;

}

public static void main(String[] args) {

int graph[][] = new int[][] { { 0, 1, 1, 2, 0, 1, 1 }, { 1, 0, 2, 0, 1, 3, 1 }, { 1, 2, 1, 3, 3, 0, 1 },

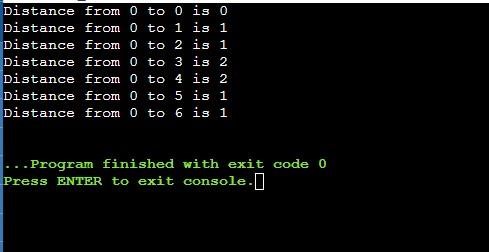
{ 2, 3, 1, 7, 7, 0, 1 }, { 1, 0, 3, 3, 5, 2, 0 }, { 0, 3, 0, 7, 2, 0, 1 }, { 3,

0, 3, 1, 7, 1, 0 } }; Main aj = new Main(); aj.dijkstra(graph, 0);

}

}

**Output :**



**Conclusion:**

Djikstra’s implemented successfully.