**Programming Project 3**

Readme

The project is creating directional graph from taking input from users.

The operations that can be performed on graph are as shown below

1. Create a Directed Graph using the inputs taken from console.

2. Perform DFS Traversal on the Graph.

3. Perform BFS Traversal on the Graph.

4. Find Shortest Path from Source to all vertices using Dijkstra’s shortest path

Algorithm.

5. Detect if there is a Cycle in the graph.

6. Check if the graph is Bipartite or not.

7. Check if Graph is a tree or not

8. Exit

User instructions to execute the program

1. User need to choose option 1 “Create a Directed Graph using the inputs taken from console.”
2. User need to enter total number of vertex in graph
3. User need to enter total number of edges in graph
4. User need to enter edges in the format as shown SOURCE DESTINATION WEIGHT

0 1 4

0 🡪 Source Node

1 🡪 Destination Node

4 🡪 Weight of edge

The input format is as shown below

1. Create a Directed Graph using the inputs taken from console.

2. Perform DFS Traversal on the Graph.

3. Perform BFS Traversal on the Graph.

4. Find Shortest Path from Source to all vertices using Dijkstra’s shortest path

Algorithm.

5. Detect if there is a Cycle in the graph.

6. Check if the graph is Bipartite or not.

7. Check if Graph is a tree or not

8. Exit

Select choice

1

Enter the total number of Vetex in Undirected Graph

7

Enter the total number of Edges in Undirected Graph

9

Enter the of Edges in Undirected Graph in source destination weight

0 1 4

3 2 5

3 4 5

2 1 8

4 6 2

6 4 6

2 5 1

3 5 4

2 4 9

The graph values will get printed in console after entering edge values.

1. For DFS and BFS traversal, user need to input source node from which graph needs to be traversed.
2. For finding shortest path from vertex using Dijkstra’s shortest path Algorithm user need to input source vertex to find distances to destination.
3. User can choose respective options from console to check if graph is tree or bipartite or is there any cycle present in the graph.
4. The program will not be ended until user select exit option.

Techniques and Data Structure used in program

1. There is main class name Graph which contain all the function of program. Each node of graph is formed using Integer. There is inner class named Edge which takes source destination and weight of edge. The program takes input of edge from user and insert it into graph. The graph is stored using adjacency list. Adjacency list is created which Linked List of object Edge. Adjacency Matrix of is also created for the graph using two-dimensional array.
2. The array of Boolean datatype is used to calculate BFS and DFS traversal of the graph. The visited nodes are marked as true and stored in graph.
3. Dijkstra shortest path algorithm 🡪 We are using adjacency matrix of graph as input for the function. There is one array that stores shortest distance from source. The we call minDistance function that pick minimum distance vertex from set of vertices not yet processed. Then we update distance value of the adjacent vertices of the picked vertex. Print the constructed distance array. If there is no path exist between two vertex then No path exist will be printed
4. Cycle 🡪 isCycle() function is called to find if there is cycle in the graph. There are three HashSet that keep track of Visited and Unvisited Vertex. Initially all vertex in NotVisited set. After visiting the vertex it will be stored in visited HashSet. Then we check the neighbors of the vertex. If the vertex is present in visited set then cycle is found in the graph.
5. Tree 🡪 The directed graph is tree when all the vertex are reachable and there is no cycle present in the graph. We are calling isRepeateUtil() function that uses visited[] array and parent to detect cycle in subgraph reachable from vertex v.
6. Bipartite 🡪 We are using adjacency matrix as an input to this function. Bipartite graph can be divided into two independent set such that for every edge (u, v), either u belongs to U and v to V, or u belongs to V and v to U. We can also say that there is no edge that connects vertices of same set. The project contains queue created using Linked list that store vertex number and enqueue source vertex for BFS traversal. The function detects the self loop in the graph. The function result true or false.
7. The ArrayOutOfBound Exception is cached using try catch block.