**Lab Assignment 04**

**Task-01**

A. In this task, I have to make an adjacency matrix. So, at first, I make an empty matrix in which row and column are as same as vertex numbers. Then I take the value of the vertex from the given vertices, find the position in the matrix, and input the weight at the desired position. In the end, I print the matrix as it is shown in the output.
B. In this task, I have to make an adjacency list. So, I take a dictionary where I store one vertex in the key and the connected vertex with the vertex in the key, I wrote it in the value along the weight of their connection. Then I print the adjacency list as it is shown in the output.

**Task-02**

In this task, I have to find out the paths which is similar to the BFS graph traversal. So, at first, I make an adjacency list for undirected and unweighted from the given information and then I write a code for queue. After this, I follow the given algorithm of BFS where I get the path which I store in the list. In the end, I print the path.

**Task-03**

In this task, I have to find out the paths which is similar to the DFS graph traversal. So, at first, I make an adjacency list for undirected and unweighted graphs from the given information. Then I follow the DFS algorithm which is given. From there I get the path and later I print the path.

**Task-04**

In this task, I have to find cycle in the graph. Firstly, I make an adjacency list for directed and unweighted graph from the given information. Then I take all the vertex in a list and from there I make a new adjacency list “g” which contain all the vertex of the graph with their directed connection. Then I write a function for color

initialization where I initialize all the color as 0. After this, I write a DFS function where I try to find a cycle in the graph by seeing the color of the vertex. If I see any vertex with color 1, it means, we find a cycle.

**Task-05**

In this task, I have to find the shortest path from the given graph. At first, I make an adjacency list for undirected and unweighted graph from the given information. Then I write a code of queue which is used in the BFS graph traversal. To get a time I store all the value in the “t” variable of dictionary, and I also store all the vertex predecessor in the dictionary called pred. After this, I store the destination in a list and print it with time.

**Task-06**

In this task, I make an adjacency matrix and then I run a nested loop along the row and column. Then I see, in the cell where I am now is “ . ” or “ D ”, if it is true then I call the diamond search function. The diamond search function I get amount of diamond I found which I compare with the diamond which I stored. After all this, I get the desired answer what I want.