

## Experiment no. 9

Aim: Implement graph using adjacency list or matrix and perform DFS or BFS.

Theory:

Algorithms:

Creation of Adjacency list:

1. Declare an array of pointers to a link list having a data field (to store vertex no) and a forward pointer. The no of array of pointer would equal the total no of vertices in the graph.
2. Take the edge set from the user. If for eg. vertex 1 is connected to vertex 2 & 3 in the graph, the 1<sup>st</sup> location of the array of pointers would point to 2 nodes, one having the data 2 and the other having data 3.
3. In this way construct the entire adjacency list.

DFS (Depth First Search).

1. The start vertex is visited. Next an ~~un~~ unvisited vertex  $w$  adjacent to  $v$  is selected & a DFS from  $w$  initiated.
2. When a vertex  $u$  is reached such that all its adjacent vertices have been visited, we back up to the last vertex visited which has an unvisited vertex  $w$  adjacent to it and initiate a DFS search from  $w$ .
3. The search terminates when no unvisited vertex can be reached from any of the visited ones.

BFS (Breadth First Search).

1. Starting at vertex  $v$  and marking it as visited, BFS differs from DFS in that all unvisited vertices adjacent to  $v$  are visited next.
2. Then unvisited vertices adjacent to these vertices are visited & so on.
3. A queue is used to store vertices as they are visited so that later search can be initiated from those vertices.

Test conditions:

Enter the graph with 8 vertices & 10 edges  $(1, 2), (1, 3), (2, 4), (2, 5), (3, 6), (3, 7), (4, 8), (5, 8), (6, 8), (7, 8)$ .

The order of the vertices visited by DFS is: 1, 2, 4, 8, 5, 6, 3, 7

The order of the vertices visited by BFS is: 1, 2, 3, 4, 5, 6, 7, 8

Input:

The no. of vertices & the edge set of the graph.

Output:

The order of vertices visited in both DFS & BFS.