**Experiment No 7**

Name:- Anuj Rajendra Mane

ROll No:- 65

Div:-A

Subject:- Data Structures

Title: Implementation of Graph Traversal Methods.

**Problem Statements:**

**1) Write programs for implementation of graph traversals by applying BFS**

#include<stdio.h>

int graph[20][20], queue[20], visited[20], n;

int front = 0, rear = -1;

int i, j;

void bfs(int v){

for(i = 1; i<=n; i++){

if(graph[v][i] == 1 && !visited[i])

queue[++rear] = i;

}

if(front <=rear){

visited[queue[front]] = 1;

bfs(queue[front++]);

}

}

int main(){

int v;

printf("Enter no of vertices: ");

scanf("%d", &n);

for(i = 0; i<=n;i++){

queue[i]= 0;

visited[i]=0;

}

printf("\nEnter the graph data in matrix form: \n");

for(i = 1; i<=n;i++){

for(j= 1; j<= n; j++){

scanf("%d", &graph[i][j]);

}

}

printf("\nEnter the starting vertex: ");

scanf("%d", &v);

bfs(v);

printf("\nThe bfs is : ");

for(i = 1; i<=n;i++){

if(visited[i])

printf("%d\t", i);

}

}

**2) Write programs for implementation of graph traversals by applying DFS**

#include<stdio.h>

void DFS(int);

int G[1000][1000], visited[1000], n; // n is no of vertices and graph is sorted in array G[1000][1000]

void DFS(int i)

{

int j;

printf("\n%d", i);

visited[i] = 1;

for(j=0; j<n; j++) {

if(!visited[j] && G[i][j]==1) {

DFS(j);

}

}

}

int main()

{

int i, j;

printf("Enter number of vertices:");

scanf("%d", &n); // Read the adjecency matrix

printf("\nEnter adjecency matrix of the graph:");

for(i=0; i<n; i++) {

for(j=0; j<n; j++) {

scanf("%d", &G[i][j]);

}

}

// Visited is initialized to zero

for(i=0; i<n; i++) {

visited[i] = 0;

}

DFS(0);

}