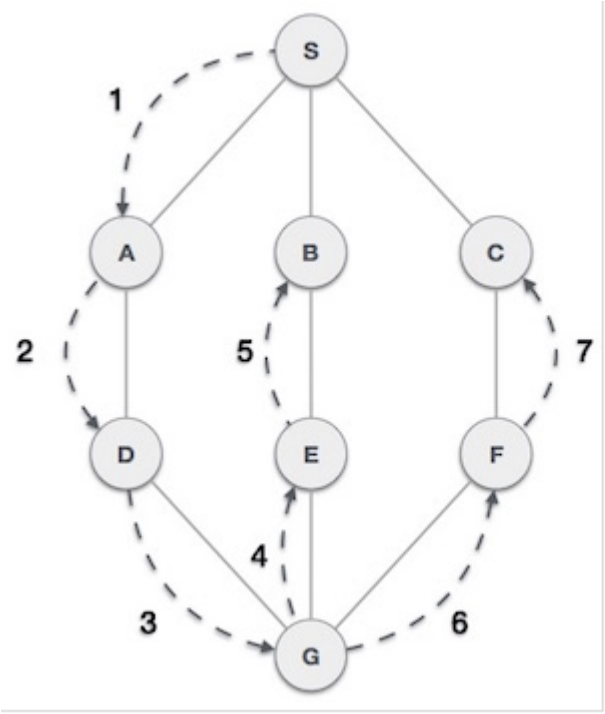
EXPERIMENT NO. 10

Implementation of Depth First Search Graph Traversal

Aim: Write A Program To implement depth first search Algorithm.

Theory: Depth First Search (DFS) algorithm traverses a graph in a depthward motion and uses a stack to remember to get the next vertex to start a search when a dead end occurs in any iteration.



Algorithm:

A standard DFS implementation puts each vertex of the graph into one of two categories:

1. Visited
2. Not Visited

The purpose of the algorithm is to mark each vertex as visited while avoiding cycles.

The DFS algorithm works as follows:

1. Start by putting any one of the graph's vertices on top of a stack.
2. Take the top item of the stack and add it to the visited list.

3. Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the top of the stack.
4. Keep repeating steps 2 and 3 until the stack is empty.

Program:

```
#include <stdio.h>
#include <conio.h>
int v, adj[10][10], visited[10];
void dfs(int k);
void main()
{
int i, j;
clrscr();
printf("Enter the number of vertices in the graph: "); scanf("%d", &v);
for (i = 0; i < v; i++)
{
for (j = 0; j < v; j++)
{
printf("Is vertex %d adjacent to vertex %d?", j + 1, i + 1);
printf(" Enter 1 if yes and 0 if no: ");
scanf("%d", &adj[i][j]);
}
}
for (i = 0; i < v; i++)
{
visited[i] = 0;
} printf("Depth first Search of the graph is as shown:\n"); for (i = 0; i < v; i++)
{
if (visited[i] == 0)
{
dfs(i);
}
}
getch();
}
void dfs(int k)
{
```

```
int j;
visited[k] = 1;
printf("%d ", k + 1);
for (j = 0; j < v; j++) // Start from 0
{
if (adj[k][j] = 1 && visited[j]=0)
{
dfs(j);
}
}
}
```

Output:

```
Enter the number of vertices in the graph:4
Is vertex 1 adjacent to vertex 1?Enter 1 if yes and 0 if no:1
Is vertex 2 adjacent to vertex 1?Enter 1 if yes and 0 if no:0
Is vertex 3 adjacent to vertex 1?Enter 1 if yes and 0 if no:1
Is vertex 4 adjacent to vertex 1?Enter 1 if yes and 0 if no:0
Is vertex 1 adjacent to vertex 2?Enter 1 if yes and 0 if no:1
Is vertex 2 adjacent to vertex 2?Enter 1 if yes and 0 if no:0
Is vertex 3 adjacent to vertex 2?Enter 1 if yes and 0 if no:1
Is vertex 4 adjacent to vertex 2?Enter 1 if yes and 0 if no:1
Is vertex 1 adjacent to vertex 3?Enter 1 if yes and 0 if no:1
Is vertex 2 adjacent to vertex 3?Enter 1 if yes and 0 if no:0
Is vertex 3 adjacent to vertex 3?Enter 1 if yes and 0 if no:1
Is vertex 4 adjacent to vertex 3?Enter 1 if yes and 0 if no:1
Is vertex 1 adjacent to vertex 4?Enter 1 if yes and 0 if no:1
Is vertex 2 adjacent to vertex 4?Enter 1 if yes and 0 if no:1
Is vertex 3 adjacent to vertex 4?Enter 1 if yes and 0 if no:1
Is vertex 4 adjacent to vertex 4?Enter 1 if yes and 0 if no:1
Depth first search of the graph is as shown:
1342
```

Conclusion:

The Depth First Search (DFS) algorithm effectively explores all vertices of a graph by following each path to its deepest point before backtracking. In this experiment, DFS was implemented using a recursive approach, marking vertices as visited and traversing adjacent, unvisited vertices. The algorithm efficiently avoids cycles and ensures every vertex is visited exactly once, demonstrating its utility in graph traversal and search problems.