Exp. No
Date :

# Depth First Search

## Aim

TO write a python program for Depth first search.

## Algorithm

Step 1: Initialize visited set:

i) create an empty set visited to keep track of visited node

Step 2: Create the graph

i) Ask the user the number of nodes in graph

ii) For each node, ask the users to input the node's neighbour and store the information in directory graph

Step 3: DFS function

i) Define a recursive function

* check is node is visited

-) If the node is not in the visited then set proceed

* visit the node

-) Print the node and add it to visited set

* Recursively visit neighbours

→ For each neighbour in graph[nodes] call dfs[visited, graph, neighbour] recursively

Step 4: Start DFS

i) Ask the user to input the starting node

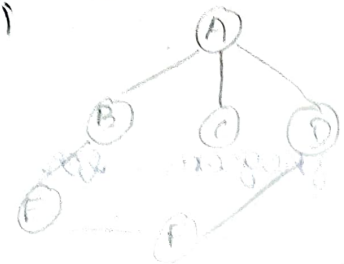
ii) call the dfs() function

Step 5: End DFS

i) The DFS traversal is complete when all reachable nodes from the start_node have been visited.

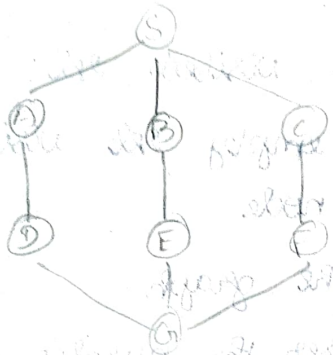
Example

1)




A
A B
A B E
A B E F
A B E F D
A B E F D C

2)




S
S A
S A D
S A D G
S A D G E
S A D G E B
S A D G E B F
S A D G E B F C

code:-

```
def dfs(visited, graph, node):
    if node is not in visited:
        print(node)
        visited.add(node)
        for neighbour in graph[node]
            dfs(visited, graph, neighbour)
def create_graph():
    graph={}
    num_nodes = int(input("Enter no of nodes
                    in graph:"))
    for _ in range(num_nodes):
        node = input("Enter the node:")
        neighbours = input("Enter the neighbour
                    of {node} separated by space:").
                    split()
        graph[node] = neighbours
    return graph
```

```
visited = set()
graph = create_graph()
start_node = input("Enter the starting node:")
Print("Following is the depth first search")
dfs(visited, graph, start_node)
```

Output:-

Enter the number of nodes in the graph: 6
Enter the node: a
Enter the neighbours of a separated by space:
b c d
Enter the node: b
Enter the neighbours of b separated by space: a e
Enter the node: e
Enter the neighbors of e separated by space: b f
Enter the node: f
Enter the neighbors of f separated by space: e d
Enter the node: d
Enter the neighbors of d separated by space: f d
Enter the node: c
Enter the neighbors of c separated by space: a
Enter the starting node for DFS: a
Following is the depth first search

a
b
e
f
d
c

Result

Thus the DFS program is successfully executed and output is verified.