Exp. No:

Date :

Depth First Search.

AIM:

To implement DFS algorithm to traverse a graph starting from a given node. and visiting all reachable nodes in depthward exploration.

ALGORITHM :

Step - 1 : Start

Step - 2 : Initialize a set visited to keep track of visited node.

Step - 3 : Start at the given starting node.

Step - 4 : Mark the node as visited by adding it to visited set.

Step - 5 : Process current node by printing (or) performing another operation on it.

Step - 6 : If neighbour node not visited, recursively apply DFS algorithm.

Step - 7 : Return to previous node once all adjacent node have been visited

Step - 8 : Repeat process.

Step - 9 : Stop.

Code:

```
def dfs(graph, start, visited = None):
    if visited is None:
        visited = set()
    visited.add(start)
    Print(start, end = ' ')
    for neighbor in graph[start]:
        if neighbour not in visited:
            dfs(graph, neighbor, visited)
    return visited
if __name__ == "__main__":
    graph = {
        'A': ['B','C'],
        'B': ['A','D','E'],
        'C': ['A','F'], 'D' = ['B'], 'E': ['B';F'], 'F': ['C','E']
    }
Print("DFS Traversal starting from node 'A':")
dfs[graph,'A')
```

Output:

```
DFS Traversal starting from node 'A':
A B D E F C
```

RESULT:

① The program has been successfully executed and the output is verified.