Ex: 2

Aim:

To implement Depth first Search using python

ALGORITHM:

Step1: Start

Step2: Create an empty set to keep track of visited node

Step3: Ask the user for the number of nodes in graph. For each node, ask the user to input the node's neighbour and store the information in directory graph.

Step4: Define a recursive function dfs(visited, graph node)

* Check if node is visited if the node is not in the visited then proceed
* Visit the node
  * print the node and add it the visited

Then proceed

or visit the node

* Print the node and add it to the visited
* Recursively visit neighbours
* for each neighbor in graph call dfs recursively

Step5: Ask the user to input the starting node. Case the dfs function with the visited set, graph and start node

Step6: The DFS traversal is complete when all reachable nodes from the start node have been visited

CODE:

```
def dfs(visited, graph, node):
    if node is not in visited:
        print(node)
        visit add(node)
        for neighbor in graph[node]:
            dfs(visited, graph, neighbour)
def create_graph():
    graph = {}
num_nodes = int(input("Enter no of nodes:"))
for _ in range(num_nodes):
    node = input("Enter the node:")
    neighbours = input("Enter the neighbours of (node)
    -Separated by spaces:).split()
    graph[node] = neighbours
    return graph
visited = set()
graph = create_graph()
start_nod = input("Enter the start node")
print("following is the DFS")
dfs(visited, graph, start_node)
```

OUTPUT

```
number nodes: 8
    node: S
        neighbour separated by space: abc
    node a
    neighbour seperated by space: sd
    node d
    neighbour separated by space: DEF
```

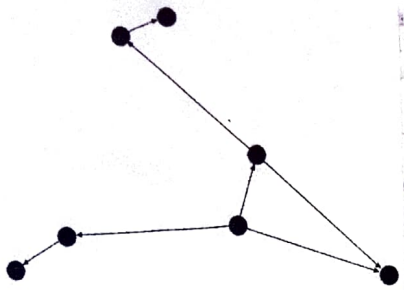
following is the DFS:

SDA AB DFEC, G.

Date :

OUTPUT :

```
Following is DFS from (starting from vertex 4)
A B D F E C G
```



RESULT:

Thus, Depth first search is implemented and the output is verified.