EX.NO:

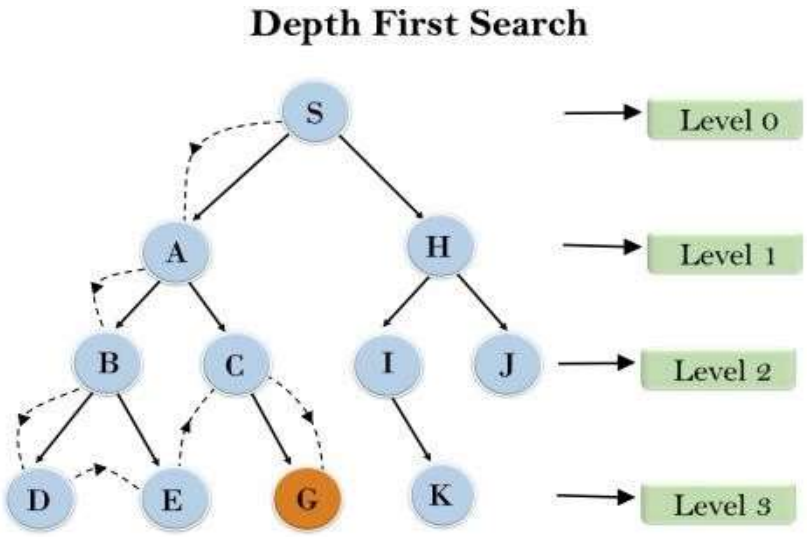
DATE:

DEPTH-FIRST SEARCH

AIM :

To implement a depth-first search problem using Python.

- Depth-first search (DFS) algorithm or searching technique starts with the root node of graph G, and then travel deeper and deeper until we find the goal node or the node which has no children by visiting different node of the tree.
- The algorithm, then backtracks or returns back from the dead end or last node towards the most recent node that is yet to be completely unexplored.
- The data structure (DS) which is being used in DFS Depth-first search is stack. The process is quite similar to the BFS algorithm.
- In DFS, the edges that go to an unvisited node are called discovery edges while the edges that go to an already visited node are called block edges.



CODE:

```python
# Function to perform DFS
def dfs(visited, graph, node):
    if node not in visited:
        print(node)
        visited.add(node)
        for neighbour in graph[node]:
            dfs(visited, graph, neighbour)

# Function to take input from the user and create the graph
def create_graph():
    graph = {}
    num_nodes = int(input("Enter the number of nodes in the graph: "))

    for _ in range(num_nodes):
        node = input("Enter the node: ")
        neighbours = input(f"Enter the neighbors of {node} separated by space: ").split()
        graph[node] = neighbours

    return graph

# Driver Code
visited = set()
graph = create_graph()
start_node = input("Enter the starting node for DFS: ")

print("Following is the Depth-First Search")
dfs(visited, graph, start_node)
```

OUTPUT:

```
Enter the number of nodes in the graph: 8
Enter the node: s
Enter the neighbors of s separated by space: a b c
Enter the node: a
Enter the neighbors of a separated by space: s d
Enter the node: d
Enter the neighbors of d separated by space: a g
Enter the node: g
Enter the neighbors of g separated by space: d e f
Enter the node: e
Enter the neighbors of e separated by space: g b
Enter the node: b
Enter the neighbors of b separated by space: e s
Enter the node: c
Enter the neighbors of c separated by space: s f
Enter the node: f
Enter the neighbors of f separated by space: c g
Enter the starting node for DFS: s
Following is the Depth-First Search
s
a
d
g
e
b
f
c
```

RESULT:

Thus Program is Executed Successfully And Output is Verified.