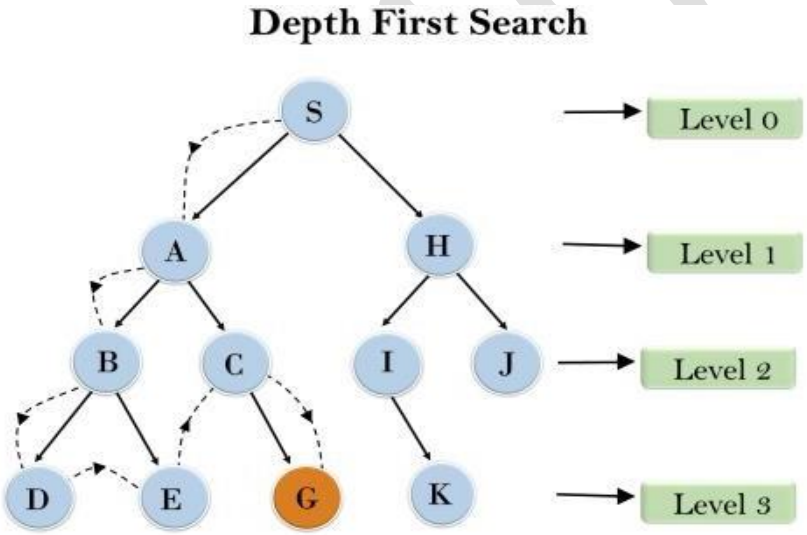
EX.NO: 2 **DATE: 16 - 08 - 2024**

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.

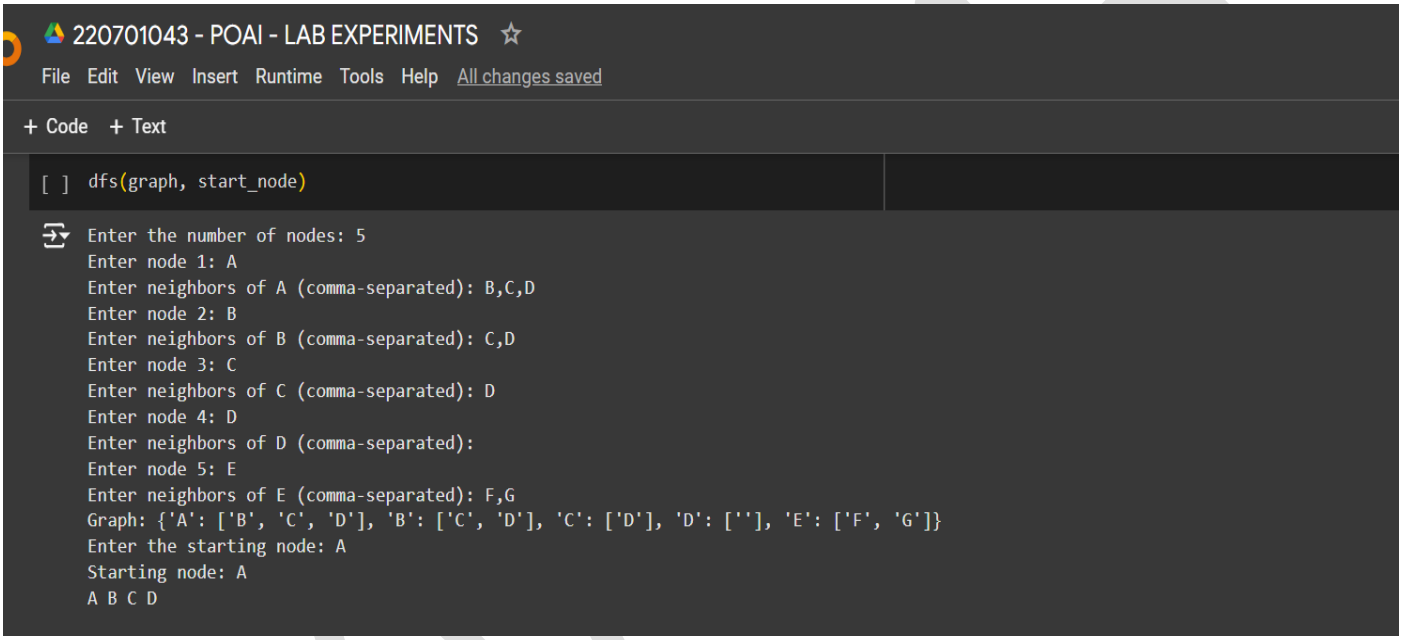


PROGRAM:

```
def dfs(graph, start, visited=None):
    if visited is None:
        visited = set()
    visited.add(start)
    print(start, end=" ")
    for neighbour in graph.get(start, []):
        if neighbour not in visited:
            dfs(graph, neighbour, visited)
num_nodes = int(input("Enter the number of nodes: "))
graph = {}
for i in range(num_nodes):
```

```
    node = input("Enter node " + str(i+1) + ": ").strip()
    neighbors = input("Enter neighbors of " + node + " (comma-separated): ").strip().split(',')
    neighbors = [n.strip() for n in neighbors]
    graph[node] = neighbors
print("Graph:", graph)
start_node = input("Enter the starting node: ").strip()
print("Starting node:", start_node)
dfs(graph, start_node)
```

OUTPUT:



RESULT:

Thus, a searching technique using Depth-First search algorithm was implemented successfully.