

EX.NO:2

DATE:4/9/2024

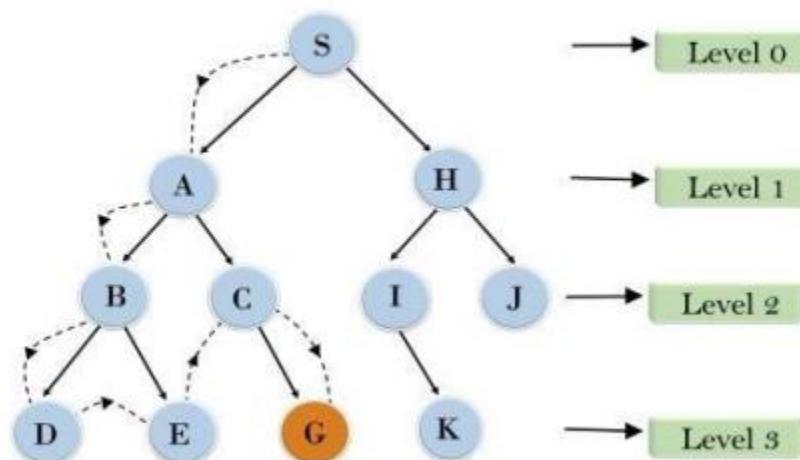
Reg.no:220701037

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

Depth First Search



CODE:

```
def dfs_recursive(graph, start, visited=None):

    if visited is None:
        visited = set()
    visited.add(start)
    print(start)

    for neighbor in graph[start]:
        if neighbor not in visited:
            dfs_recursive(graph, neighbor, visited)

graph = {
    'A': ['B', 'C'],
    'B': ['A', 'D', 'E'],
    'C': ['A', 'F'],
    'D': ['B'],
    'E': ['B', 'F'],
    'F': ['C', 'E']
}

print("DFS Recursive:")
dfs_recursive(graph, 'A')

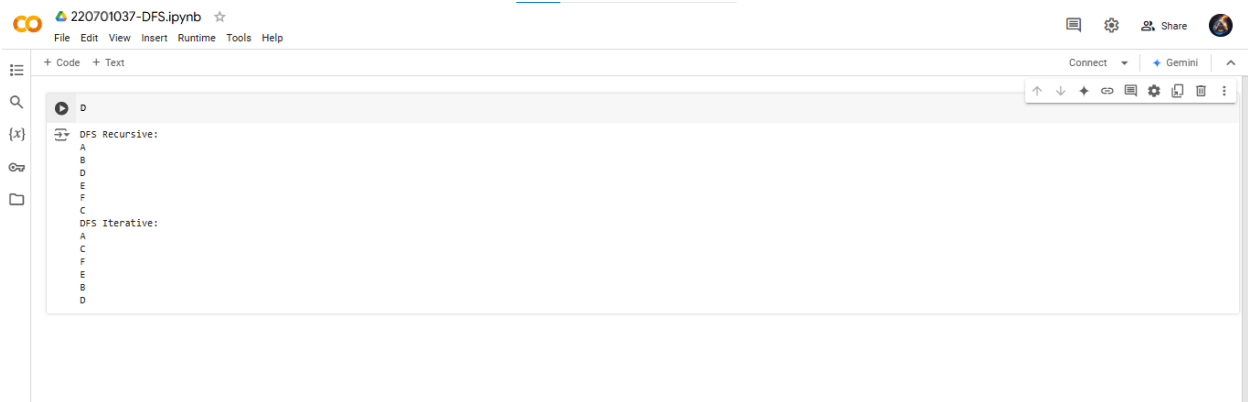
def dfs_iterative(graph, start):
    visited = set()
    stack = [start]

    while stack:
        vertex = stack.pop()
        if vertex not in visited:
            print(vertex)
            visited.add(vertex)
            stack.extend(neighbor for neighbor in graph[vertex] if
neighbor not in visited)

graph = {
    'A': ['B', 'C'],
    'B': ['A', 'D', 'E'],
    'C': ['A', 'F'],
    'D': ['B'],
    'E': ['B', 'F'],
    'F': ['C', 'E']
}

print("DFS Iterative:")
dfs_iterative(graph, 'A')
```

OUTPUT:



The image shows a Jupyter Notebook interface with a file named '220701037-DFS.ipynb'. The notebook contains two code cells. The first cell, titled 'D', shows the output of a recursive DFS algorithm, listing nodes A, B, D, E, F, and C in that order. The second cell, titled 'DFS Iterative:', shows the output of an iterative DFS algorithm, listing nodes A, C, F, E, B, and D in that order. The interface includes a top menu bar with options like File, Edit, View, Insert, Runtime, Tools, and Help. A left sidebar shows the notebook's structure, and a right sidebar shows the current cell's content. A large, light gray watermark '220701037' is visible across the bottom half of the image.

```
D
DFS Recursive:
A
B
D
E
F
C
DFS Iterative:
A
C
F
E
B
D
```