

***Name: IMRAN MAJEED***

***Roll No: SU92-BSAIM-F24-212***

### What the code does:

It performs a **Depth-First Search (DFS)** on a tree-like structure (represented as a dictionary) to find if a **goal node** exists, starting from a **start node**.

### The Tree Structure:

This dictionary represents a tree, where each key is a node, and its value is a list of its children.

tree = {

"A": ['B', 'C'],

"B": ['D', 'E'],

"C": ['F'],

"D": [],

"E": [],

"F": ['G'],

"G": []

}

A

/ \

B C

/ \ \

D E F

\

G

**Variables:**

visited = list() # This keeps track of nodes that have been visited

start = "A" # The node to start DFS from

goal = "D" # The node we are trying to find

The DFS Function:

def dfs(start, goal):

if goal not in visited:

visited.append(start)

for i in tree[start]:

dfs(i, goal)

else:

print('exist')

1. **Check if goal is not visited**:
2. if goal not in visited:
   * This is intended to stop the search once the goal has been visited.
3. **Visit the current node**:
4. visited.append(start)
   * Adds the current node to the visited list.
5. **Recursively visit each child of the current node**:
6. for i in tree[start]:
7. dfs(i, goal)
   * Calls DFS on each child node.
8. **If the goal was already visited**:
9. else:
10. print('exist')
    * Just prints 'exist' if the goal is already in the visited list.

**OUTPUT :**

