

AI LAB EXP-4
IMPLEMENTATION OF DFS AND BFS

Date: 08-02-2022

Name: Vinoth S

Reg No: RA1911030010103

Github link: <https://github.com/vk1308>

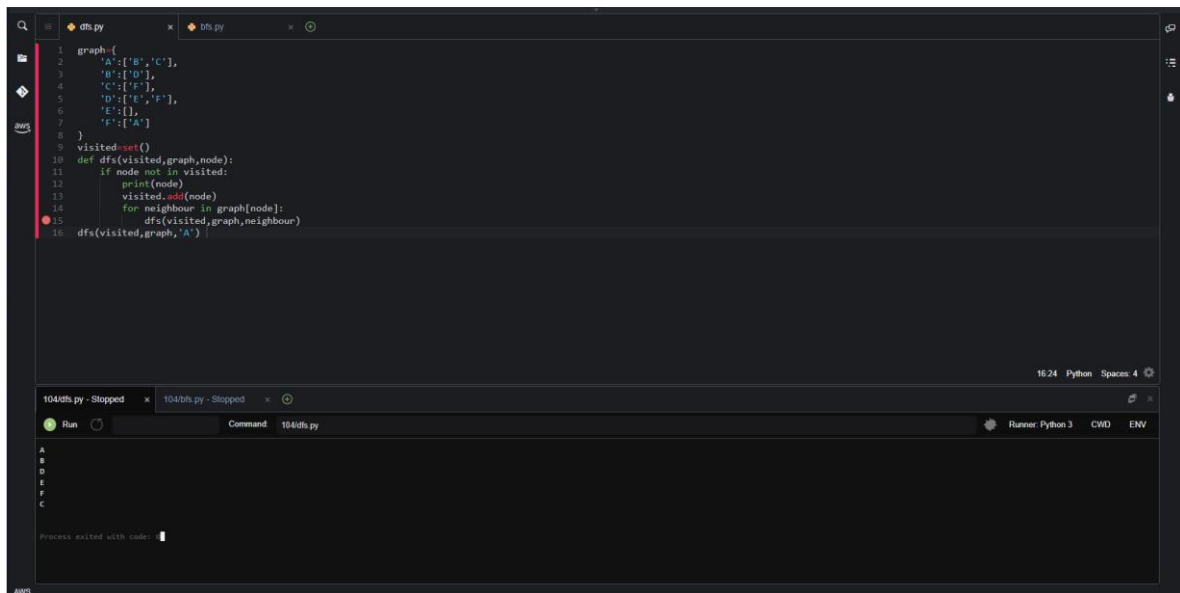
AIM:

To implement BFS(Breadth First Search) and DFS(Depth First Search) using Python.

DFS CODE:

```
graph={
'A':['B','C'],
'B':['D'],
'C':['F'],
'D':['E','F'],
'E':[],
'F':['A']
}
visited=set()
def dfs(visited,graph,node):
if node not in visited:
print(node)
visited.add(node)
for neighbour in graph[node]:
dfs(visited,graph,neighbour)
dfs(visited,graph,'A')
```

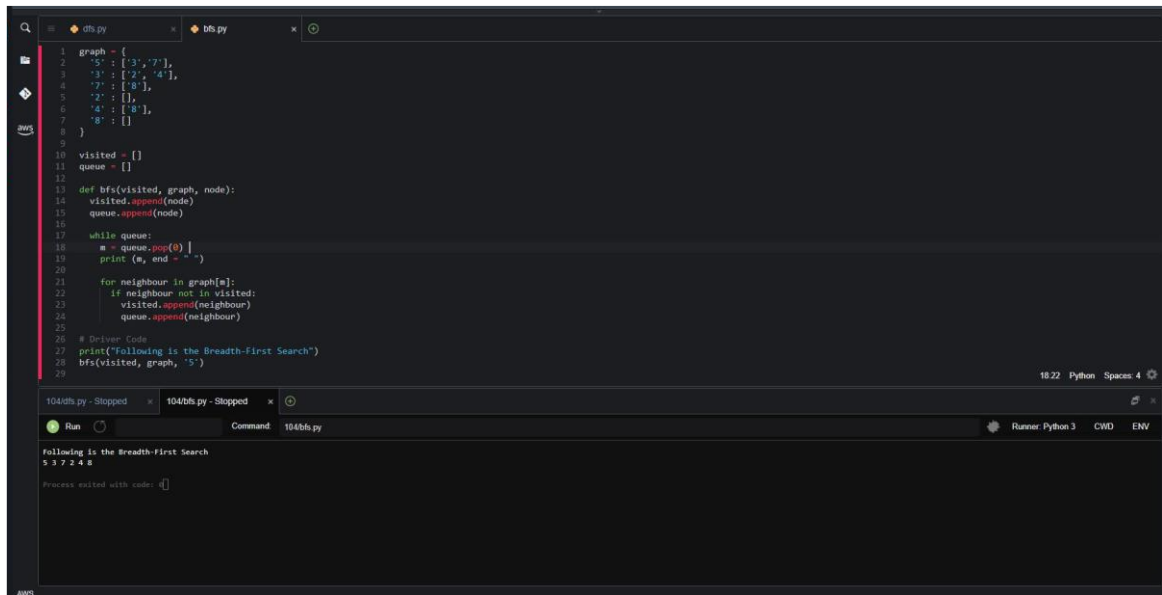
OUTPUT SCREENSHOT:

A screenshot of a code editor with two tabs: 'dfs.py' and 'bfs.py'. The 'dfs.py' tab is active, showing a Python script for Depth-First Search (DFS) on a graph. The graph is defined as a dictionary where keys are nodes and values are lists of neighbors. The script includes a recursive function 'dfs' that prints the nodes in the order they are visited. The output window at the bottom shows the nodes A, B, D, E, F, and C in that order. The status bar at the bottom indicates '104dfs.py - Stopped', 'Command: 104dfs.py', 'Runner: Python 3', 'CMD', and 'ENV'.

BFS CODE:

```
graph = {
'5' : ['3','7'],
'3' : ['2', '4'],
'7' : ['8'],
'2' : [],
'4' : ['8'],
'8' : []
}
visited = []
queue = []
def bfs(visited, graph, node):
visited.append(node)
queue.append(node)
while queue:
m = queue.pop(0)
print (m, end = " ")
for neighbour in graph[m]:
if neighbour not in visited:
visited.append(neighbour)
queue.append(neighbour)
# Driver Code
print("Following is the Breadth-First Search")
bfs(visited, graph, '5')
```

OUTPUT SCREENSHOT:



The screenshot shows a code editor with two tabs: 'dfs.py' and 'bfs.py'. The 'bfs.py' tab is active, displaying the following Python code:

```
1 graph = {
2     '5': ['3', '7'],
3     '3': ['2', '4'],
4     '7': ['8'],
5     '2': [],
6     '4': ['8'],
7     '8': []
8 }
9
10 visited = []
11 queue = []
12
13 def bfs(visited, graph, node):
14     visited.append(node)
15     queue.append(node)
16
17     while queue:
18         n = queue.pop(0)
19         print(n, end = " ")
20
21         for neighbour in graph[n]:
22             if neighbour not in visited:
23                 visited.append(neighbour)
24                 queue.append(neighbour)
25
26 # Driver Code
27 print("Following is the Breadth-First Search")
28 bfs(visited, graph, '5')
29
```

Below the code editor, there is a terminal window showing the output of the BFS algorithm:

```
Following is the Breadth-First Search
5 3 7 2 4 8
Process exited with code: 0
```

RESULT:

Hence DFS and BFS are implemented using python in an AWS environment.