Name: Richard Nadar
Reg no: RA1911030010109
Date: 8-2-22

EXP 4: Implementation of BFS and DFS

CODE:

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

visited_bfs = []
queue = []

def bfs(visited_bfs, graph, node):
  visited_bfs.append(node)
  queue.append(node)

  while queue:
    s = queue.pop(0)
    print (s, end = " ")

    for neighbour in graph[s]:
      if neighbour not in visited_bfs:
```

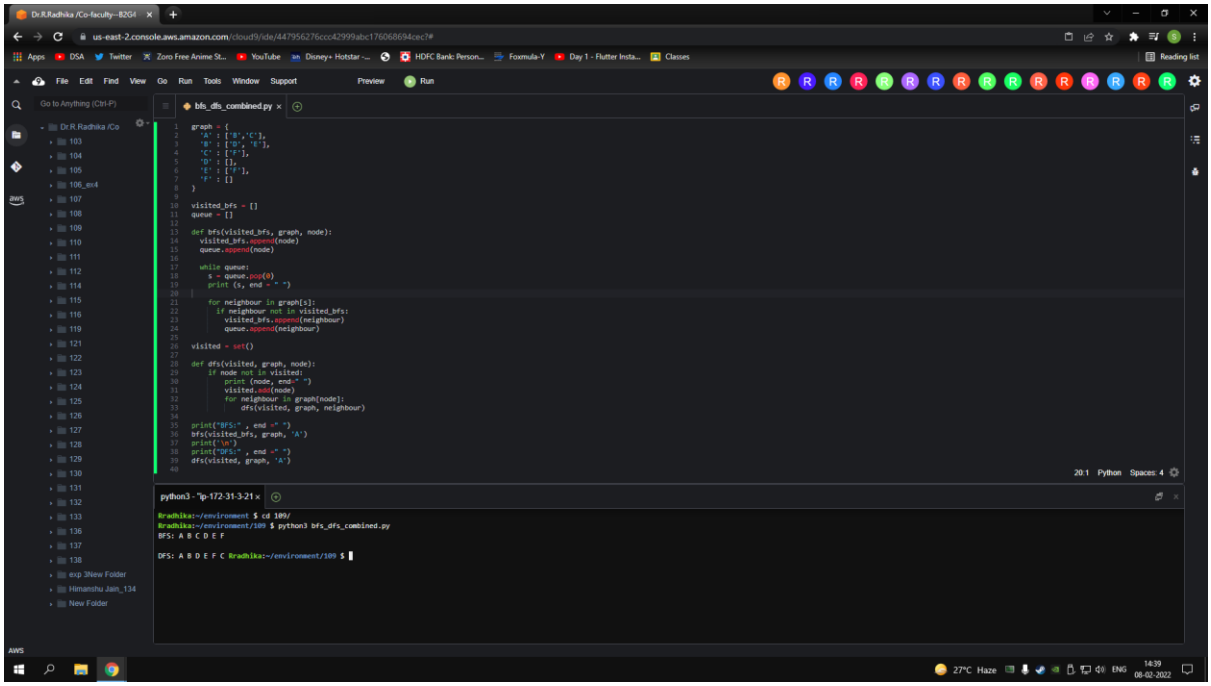
```
        visited_bfs.append(neighbour)
        queue.append(neighbour)

visited = set()

def dfs(visited, graph, node):
    if node not in visited:
        print (node, end=" ")
        visited.add(node)
        for neighbour in graph[node]:
            dfs(visited, graph, neighbour)

print("BFS:" , end =" ")
bfs(visited_bfs, graph, 'A')
print('\n')
print("DFS:" , end =" ")
dfs(visited, graph, 'A')
```

OUTPUT:


bfs_dfs_combined.py
graph = {
'A' : ['B','C'],
'B' : ['D', 'E'],
'C' : ['F'],
'D' : [],
'E' : ['F'],
'F' : []
}
visited_bfs = []
queue = []
def bfs(visited_bfs, graph, node):
visited_bfs.append(node)
queue.append(node)
while queue:
s = queue.pop(0)
print (s, end = " ")
for neighbour in graph[s]:
if neighbour not in visited_bfs:
visited_bfs.append(neighbour)
queue.append(neighbour)
visited = set()
def dfs(visited, graph, node):
if node not in visited:
print (node, end=" ")
visited.add(node)
for neighbour in graph[node]:
dfs(visited, graph, neighbour)
print("BFS:" , end =" ")
bfs(visited_bfs, graph, 'A')
print("\n")
print("DFS:" , end =" ")
dfs(visited, graph, 'A')
python3 - "ip-172-31-3-21
Rradhika:~/environment $ cd 109/
Rradhika:~/environment/109 $ python3 bfs_dfs_combined.py
BFS: A B C D E F
DFS: A B D E F C Rradhika:~/environment/109 $