TASK-1A BREADTH FIRST SEARCH

PROGRAM

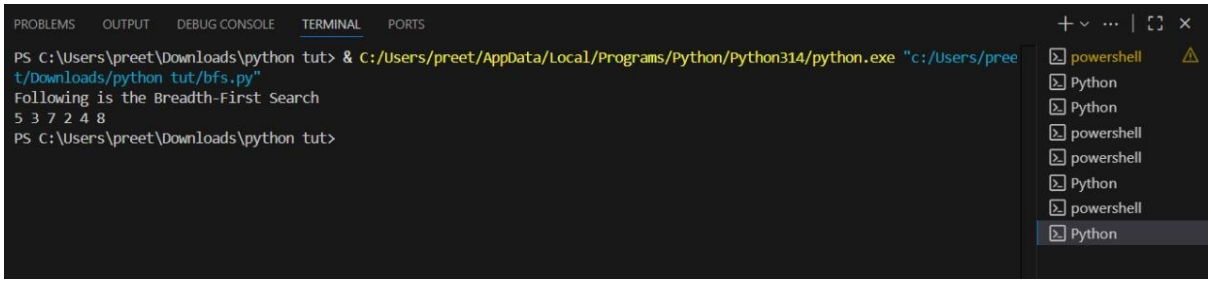
```
graph = {
    '5': ['3', '7'],
    '3': ['2', '4'],
    '7': ['8'],
    '2': [],
    '4': ['8'],
    '8': []
}

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)

print("Following is the Breadth-First Search")
bfs(visited, graph, '5')
```

OUTPUT



PROGRAM

```
# Depth First Search (DFS) Implementation in Python


# Using a dictionary to represent the graph
graph = {
    'A': ['B', 'C'],
    'B': ['D', 'E'],
    'C': ['F'],
    'D': [],
    'E': ['F'],
    'F': []
}


# Set to keep track of visited nodes
visited = set()


def dfs(visited, graph, node):
    if node not in visited:
        print(node, end=" ")        # Print the visited node
        visited.add(node)           # Mark node as visited
        for neighbour in graph[node]:
            dfs(visited, graph, neighbour)  # Recursive call


# Driver code
```

```
print("Depth First Search traversal starting from node A:")
dfs(visited, graph, 'A')
```

OUTPUT

```
Output                                                    Clear
Depth First Search traversal starting from node A:
A B D E F C
=== Code Execution Successful ===
```