Problem Statement : Traverse an Graph Using BFS Algorithm in Python Programming Language.

Objective : Implement BFS Algorithm and Traverse a Graph Using Python Programming Language

Source Code & Output :

main.py

```python
from collections import deque
graph = {
    'A': ['B', 'C'],
    'B': ['A', 'D', 'E'],
    'C': ['A', 'F'],
    'D': ['B'],
    'E': ['B', 'F'],
    'F': ['C', 'E']
}

def bfs(graph, start):
    visited = set()
    queue = deque()
    queue.append(start)
    visited.add(start)
    while queue:
        node = queue.popleft()
        print(node, end=' ')

        for neighbor in graph[node]:
            if neighbor not in visited:
                queue.append(neighbor)
                visited.add(neighbor)

print("BFS Traversal:")
bfs(graph, 'A')
```

input

```
BFS Traversal:
A B C D E F

...Program finished with exit code 0
Press ENTER to exit console.
```

Problem Statement : Traverse an Graph Using DFS Algorithm in Python Programming Language.

Objective : Implement DFS Algorithm and Traverse a Graph Using Python Programming Language

Source Code & Output :

main.py

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

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

visited = set()
print("DFS Traversal:")
dfs(graph, 'A', visited)
```

input

```
DFS Traversal:
A B D E F C

...Program finished with exit code 0
Press ENTER to exit console.
```