Question:7

BREADTH FIRST SEARCH

AIM

To implement Breadth First Search algorithm using Python

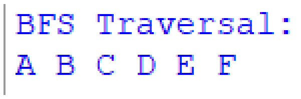
ALGORITHM

```
1. Import the `deque` class from the `collections` module to use a double-ended queue.
2. Define the `bfs` function that takes two parameters: the graph as a dictionary and the starting node.
3. Initialize an empty set called `visited` to keep track of visited nodes.
4. Initialize a queue (`deque`) with the starting node `start`, and mark it as visited.
5. While the queue is not empty:
   a. Dequeue a node `node` from the queue.
   b. Print the dequeued node (or perform any desired operation on it).
   c. Iterate over the neighbors of the dequeued node:
      - If a neighbor `neighbor` has not been visited:
         - Mark it as visited.
         - Enqueue the neighbor into the queue.
6. If there are no more nodes in the queue, the BFS traversal is complete.
7. In the `if __name__ == "__main__":` block:
   a. Define a graph as a dictionary where the keys represent nodes and the values represent their neighbors.
   b. Print a message indicating the start of BFS traversal.
   c. Call the `bfs` function with the defined graph and the starting node `'A'`.
```

CODE

```
from collections import deque
def bfs(graph, start):
    visited = set()
    queue = deque([start])
    visited.add(start)
    while queue:
        node = queue.popleft()
        print(node, end=" ")
        for neighbor in graph[node]:
            if neighbor not in visited:
                visited.add(neighbor)
                queue.append(neighbor)
if __name__ == "__main__":
    graph = {
        'A': ['B', 'C'],
        'B': ['A', 'D', 'E'],
        'C': ['A', 'F'],
        'D': ['B'],
        'E': ['B', 'F'],
        'F': ['C', 'E']
    }
    print("BFS Traversal:")
    bfs(graph, 'A')
```

OUTPUT



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