PROGRAM-7

BFS PROBLEM

AIM:-

To write and execute the python program for the BFS program.

PROCEDURE:-

- **Graph Representation**:
  - The Graph class represents a graph using the adjacency list representation. It initializes an empty defaultdict where each key represents a vertex and its corresponding value is a list of neighboring vertices.
- **Adding Edges**:
  - The add_edge method allows adding edges to the graph by appending vertices to the adjacency list of their respective source vertices.
- **Breadth-First Search (BFS)**:
  - The bfs method performs a breadth-first search traversal starting from a given start node.
  - It initializes a set visited to keep track of visited nodes and a deque queue to store nodes to be visited.
  - It starts from the start node, adds it to the visited set, and enqueues it in the queue.
  - While the queue is not empty, it dequeues a node, prints it, and explores its neighbors. If a neighbor has not been visited, it is enqueued and marked as visited.
- **Example Usage**:
  - Create an instance of the Graph class.
  - Add edges to the graph using the add_edge method.
  - Call the bfs method with a starting node to perform BFS traversal from that node..

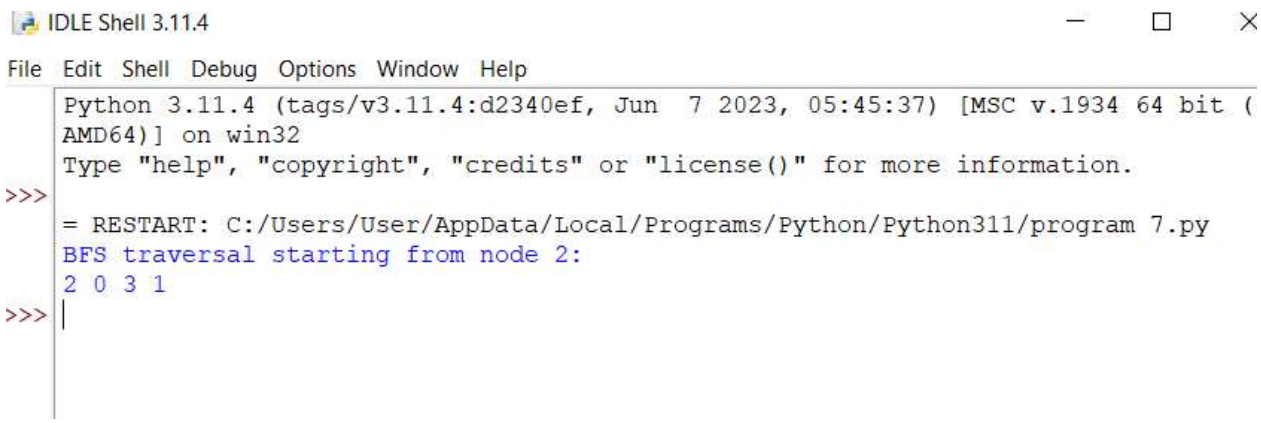
CODING:-

```
from collections import defaultdict, deque
```

```
class Graph:
    def __init__(self):
        self.graph = defaultdict(list)
    def add_edge(self, u, v):
        self.graph[u].append(v)
    def bfs(self, start):
        visited = set()
        queue = deque([start])
        visited.add(start)
        while queue:
            node = queue.popleft()
            print(node, end=' ')
            for neighbor in self.graph[node]:
                if neighbor not in visited:
                    queue.append(neighbor)
                    visited.add(neighbor)

# Example usage:
g = Graph()
g.add_edge(0, 1)
g.add_edge(0, 2)
```

```
g.add_edge(1, 2)

g.add_edge(2, 0)

g.add_edge(2, 3)

g.add_edge(3, 3)

print("BFS traversal starting from node 2:")

g.bfs(2)
```

OUTPUT:-



RESULT:-

Hence the program has been successfully executed and verified.