

**Department of Software Engineering**

**Faculty of Computer Science & Information Technology**

**The Superior University, Lahore**

**Name:** MUHAMMAD HAMZA ALI

**Roll No:** SU92-BSAIM-S24-032

**Section:** 3A

**Subject:** ARTIFICIAL INTELLIGENCE(LAB)

**Task No:** Lab-Task 6(Task-2)

**Task-6**

**BFS With Queue**

**1. Introduction:**

This report provides an analysis of the implementation of the Breadth-First Search (BFS) algorithm using a queue-based approach in Python. The task involves creating a binary tree and traversing it level by level using BFS.

**2. Features:**

To enhance the BFS functionality, we can modify the implementation to track the level of each node and calculate its depth. This can be achieved by storing tuples in the queue, where each tuple contains a node, its corresponding level, and its depth.

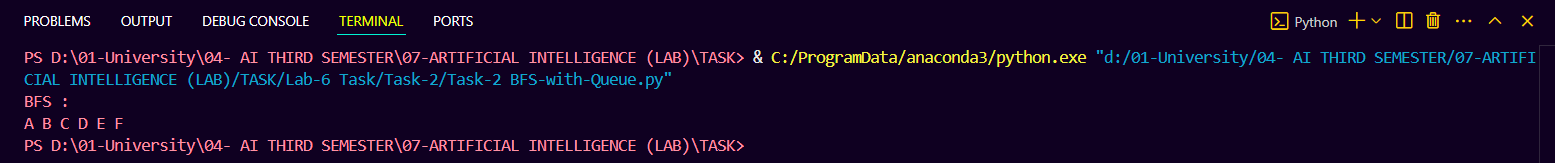
**3. Implementation Details:**

The program defines a Node class and a bfs() function to perform BFS traversal. A sample binary tree is constructed, and BFS is applied to print the nodes in level order.

**4. User Interaction:**

* The implementation correctly follows BFS principles.
* The use of a Python list for queue operations results in 0(n) complexity for popping the first element. Using collections.deque would optimize this to 0(1).
* The function assumes a non-empty tree, handling an empty input would improve robustness.

**5. Output:**



**6. Conclusion:**

The BFS implementation successfully traverses a binary tree level by level. With minor optimizations, such as using deque, the efficiency can be further improved. The newly added level-tracking feature enhances the clarity of the output by providing additional structural information about the tree.