**Name : Ali Azhar**

**Roll No : SU92-BSAIM-S24-026**

**Section : 3A**

**Subject : Artificial Intelligence**

**Documentation of Task 1**

**“BFS without Queue and Node”**

**Graph Representation :**

The graph is implemented as a dictionary where each key represents a node, and the corresponding value is a list of its adjacent nodes.

**BFS Function :**

The BFS function performs **Breadth-First Search (BFS)** traversal starting from the given root node, exploring nodes level by level without using a queue.

**Visited List :**

A visited list is used to track the nodes that have been explored, preventing revisits and infinite loops.

**Level Tracking :**

A level list keeps track of the nodes at the current level, while a next\_level list stores the nodes to be visited in the next iteration.

**Goal Node Validation :**

If the user enters the root node as the goal, a message is displayed instructing them to choose a different goal node.

**Traversal Execution :**

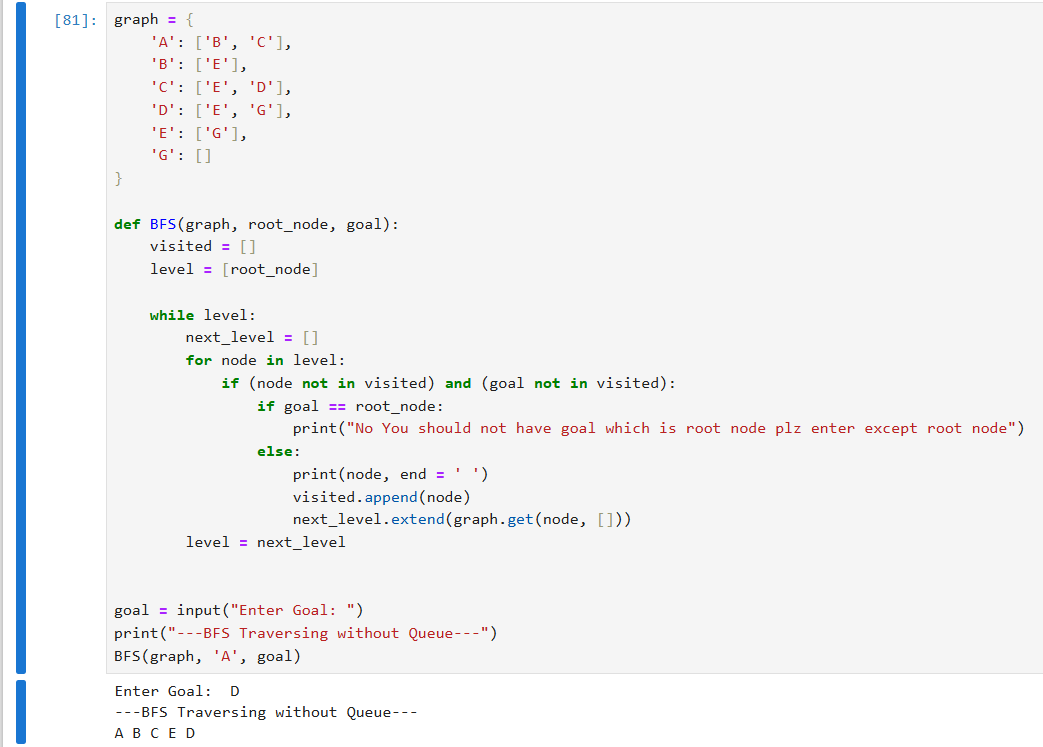
Nodes are printed as they are visited, and their neighbors are added to next\_level, ensuring a **level-order traversal** of the graph.

**User Input for Goal Node :**

The user is prompted to enter a goal node, which the function then searches for while traversing the graph.

**Output :**

The traversal path is displayed, showing the order in which nodes are visited during the BFS process.



**Documentation of Task 2**

**“BFS with Queue and Node”**

### ****Graph Representation :****

The graph is implemented as a dictionary where each key represents a node, and the corresponding value is a list of its adjacent nodes.

### ****BFS Function :****

The BFS function performs **Breadth-First Search (BFS)** traversal starting from the given root node, exploring nodes level by level using a queue.

### ****Queue Implementation :****

A **queue** (list) is used to store nodes in the order they should be visited. The first node is dequeued, processed, and its neighbors are enqueued.

### ****Visited List :****

A visited list is used to track nodes that have already been explored, preventing revisits and infinite loops.

### ****Goal Node Validation :****

If the user enters the root node as the goal, a message is displayed instructing them to choose a different goal node.

### ****Traversal Execution :****

Nodes are printed as they are visited, and their neighbors are added to the queue, ensuring a **level-order traversal** of the graph.

### ****User Input for Goal Node :****

The user is prompted to enter a goal node, which the function then searches for while traversing the graph.

### ****Output :****

The traversal path is displayed, showing the order in which nodes are visited during the BFS process.

