**Name: Usman Ul Haq**

**Roll# BSAIM-035**

**AI LAB TASKs**

**Documentation**

**BFS without Queue & without Node**

This program represents a graph using an adjacency list and performs **Breadth-First Search (BFS)** using a recursive approach instead of the traditional queue-based method.

**How It Works**

1. **Graph Representation:**
   * The graph is defined with a fixed number of vertices.
   * Connections (edges) between vertices are stored in an adjacency list.
2. **Adding Connections:**
   * Two vertices can be connected using the add\_edge method.
   * The connections are **bidirectional**, meaning if vertex A is connected to B, then B is also connected to A.
3. **Breadth-First Search (BFS) Without a Queue:**
   * The traversal starts from a given node.
   * Instead of using a queue, the function calls itself recursively to explore the next level of nodes.
   * Nodes are marked as **visited** once they are processed to avoid revisiting.
   * The process continues until all reachable nodes are visited.

**Example**

If the graph has the following connections:

0 - 1

0 - 2

1 - 3

2 - 4

Starting BFS from node 0 will output:

sql

CopyEdit

BFS without Queue & without Node:

0 1 2 3 4

This shows the nodes visited level by level.

**Key Features**

* Uses **recursion** instead of a queue for BFS.
* Works on **undirected graphs** (both directions are stored).
* Ensures each node is visited only once.

**2. BFS with Queue & Node**

This program represents a graph using nodes and edges and performs Breadth-First Search (BFS) using a queue-based approach.

How It Works

1. Graph Representation:
   * Each element in the graph is represented as a node with a unique value.
   * Each node maintains a list of neighbors (connected nodes).
   * The graph stores all nodes in a dictionary for quick access.
2. Adding Connections:
   * The add\_edge method connects two nodes by adding them to each other's neighbor list.
   * If a node does not exist, it is created automatically.
   * The graph is undirected, meaning connections work in both directions.
3. Breadth-First Search (BFS) with a Queue:
   * The traversal begins from a starting node.
   * A queue is used to keep track of nodes that need to be visited.
   * Nodes are processed in the order they were discovered.
   * Each node is marked as visited to prevent revisiting.
   * The process continues until all reachable nodes are visited.

Example Execution

If the following connections exist in the graph:

0 - 1

0 - 2

1 - 3

2 - 4

Starting BFS from node 0 produces:

csharp

BFS with Queue & Node:

0 1 2 3 4

This output represents the order in which nodes are visited.

Key Features

* Uses nodes instead of just numbers to represent graph elements.
* Uses a queue to manage BFS traversal.
* Works on undirected graphs (both directions are stored).
* Ensures each node is visited only once using a set.