**BFS – Graph Traversal**

Name: AYESHA SIKANDAR  
Roll No.: 044  
Section: BSDS 3A

**1. BFS using Queue**

**Code Explanation:**

I created a function bfs\_with\_queue() which performs Breadth-First Search on a graph using a queue.  
The graph is represented as a dictionary, where each key is a node and its value is a list of connected neighbors.  
I use a deque from the collections module as a queue to keep track of nodes to visit, and a set visited to store nodes that have already been visited.  
Inside a while loop, I remove nodes from the front of the queue, print them, and add all their unvisited neighbors to the queue.  
The loop continues until the queue is empty, ensuring a level-by-level traversal of the graph.

**How I Made It:**

I used basic Python features like dictionaries, loops, sets, and deque.  
I used a queue to explore nodes level by level.  
Visited nodes are tracked using a set to avoid revisiting.  
Nodes are printed as they are visited to show the BFS order.

**Why I Made It:**

I made this to understand BFS traversal in graphs and practice using queues in Python.  
It helps visualize how graphs are explored level by level, useful for shortest-path algorithms and network exploration.

**2. BFS without Queue**

**Code Explanation:**

I created a function bfs\_without\_queue() which performs Breadth-First Search without using a deque.  
The graph is represented as a dictionary, with each key as a node and its value as a list of neighbors.  
I use a list as a queue, and a set visited to track visited nodes.  
Inside a while loop, I remove the first element of the list using pop(0), print it, and append all unvisited neighbors to the end of the list.  
The loop continues until the list is empty, which also ensures a level-by-level traversal of the graph.

**How I Made It:**

I used basic Python features like dictionaries, loops, sets, and lists.  
I used a list to simulate a queue and maintain FIFO order.  
Visited nodes are tracked using a set, and nodes are printed to show the BFS order.

**Why I Made It:**

I made this to understand BFS without using deque, and to practice managing the queue manually with a list.  
It helps visualize level-order traversal while understanding how visited nodes and neighbors are handled in BFS.