# NAME:

M NAUMAN ANWAR

# Breadth-First Search (BFS) Documentation

## BFS with Queue

Breadth-First Search (BFS) with Queue is the standard implementation of BFS. In this approach, a queue (FIFO - First In First Out) is used to process nodes one by one. The algorithm starts by inserting the starting node into the queue and marking it as visited. Then, while the queue is not empty, the algorithm repeatedly removes the front element, visits it, and inserts all its unvisited neighbors into the queue.  
  
This ensures that nodes are explored in breadth-first order (level by level).

### Steps:

1. Initialize an empty queue and a visited list.  
2. Insert the start node into both queue and visited list.  
3. While the queue is not empty:  
 - Remove the first element from the queue.  
 - Visit this element and add it to the traversal order.  
 - Add all unvisited neighbors of this element to the queue.  
4. Continue until the queue becomes empty.

### Use Case:

This is the most widely used BFS method and is commonly taught in textbooks. It is efficient and easy to understand.



## BFS without Queue

BFS can also be implemented without explicitly using a queue. Instead, two lists (or arrays) are used: one to keep track of the current level and another for the next level. The algorithm processes all nodes in the current level and collects their unvisited neighbors into the next level. After finishing one level, the next level becomes the current level, and the process repeats.  
  
This approach also performs a breadth-first traversal but does so level by level.

### Steps:

1. Initialize a list for the current level with the start node.  
2. Mark the start node as visited.  
3. While the current level list is not empty:  
 - For each element in the current level:  
 \* Visit it and add it to the traversal order.  
 \* Add all unvisited neighbors to the next level list.  
 - Set current level = next level and clear the next level list.  
4. Continue until there are no more nodes to process.

### Use Case:

This method is less common and is mainly used as an alternative to avoid explicit queue usage. It is useful for understanding BFS from a level-by-level perspective.

## Applications of BFS

- Finding the shortest path in unweighted graphs  
- Social network analysis (finding friends of friends)  
- Web crawling (exploring websites level by level)  
- AI and game development (searching state spaces)



## Conclusion

Both BFS with Queue and BFS without Queue achieve the same result: a breadth-first traversal of a graph. The queue-based method is the standard and most efficient approach, while the list-based (without queue) method provides an alternative perspective by focusing on level-by-level traversal.