It starts by initializing an array called **visited** of size **numNodes** to keep track of which nodes have been visited during the BFS traversal. All elements in the array are initialized to **false** at the beginning, indicating that no node has been visited yet.

A black screen with white text

Description automatically generated

The function also creates a queue data structure (commonly implemented as an array or linked list) to store the nodes that need to be visited. It starts by adding the **startVertex** to the queue as the first node to visit.

A black screen with white text and numbers

Description automatically generated

The **parent** array is used to store the parent vertex of each vertex during the BFS traversal. It helps in reconstructing the shortest path later. The parent of a vertex is the node from which it was discovered during the traversal.

A black screen with white text

Description automatically generated

The BFS starts by marking the **startVertex** as visited and sets its distance to 0 in the **parent** array, indicating that it is the starting point of the traversal.

A black background with white text

Description automatically generated

While the queue is not empty, the BFS continues to explore the graph:

a. Dequeue (remove) a vertex from the front of the queue. This vertex becomes the current vertex, and we visit it.

b. Iterate over the adjacency list of the current vertex to find its neighbors (nodes connected to the current vertex).

c. For each unvisited neighbor, mark it as visited, enqueue it into the queue, and set its parent as the current vertex. This means we have discovered a new node from the current node, and we'll explore it later.

d. The BFS keeps visiting neighbors in a "breadth-first" manner, which means it explores all the nodes at the current level before moving on to the next level.

The BFS continues until the queue is empty or until the **destinationVertex** is found. If the **destinationVertex** is reached, it means the shortest path has been found, and the BFS can stop.

A screen shot of a computer program

Description automatically generated

After the BFS is completed, the function reconstructs the shortest path from **startVertex** to **destinationVertex** using the **parent** array. It creates a **path** array and traverses the parent links from the **destinationVertex** back to the **startVertex**, recording the vertices in reverse order.

A screen shot of a computer program

Description automatically generated

Finally, the **path** array is reversed to get the correct order of vertices from **startVertex** to **destinationVertex**, and this reversed array is returned as the result, representing the shortest path.

A computer screen shot of text

Description automatically generated