Topological Sorting using BFS (Kahn's Algorithm) - Java

```
import java.util.*;

public class TopologicalSortBFS {
    public static void topologicalSort(int V, List<List<Integer>> adj) {
        int[] inDegree = new int[V]; // Array to store in-degrees
        Queue<Integer> queue = new LinkedList<>();
        List<Integer> topoOrder = new ArrayList<>();

        // Calculate in-degree of each node
        for (int i = 0; i < V; i++) {
            for (int neighbor : adj.get(i)) {
                inDegree[neighbor]++;
            }
        }

        // Add nodes with in-degree 0 to the queue
        for (int i = 0; i < V; i++) {
            if (inDegree[i] == 0) {
                queue.add(i);
            }
        }

        // Process queue
        while (!queue.isEmpty()) {
            int node = queue.poll();
            topoOrder.add(node);

            // Reduce in-degree of neighbors
            for (int neighbor : adj.get(node)) {
                inDegree[neighbor]--;
                if (inDegree[neighbor] == 0) {
                    queue.add(neighbor);
                }
            }
        }

        // Print topological order
        for (int node : topoOrder) {
            System.out.print(node + " ");
        }
    }

    public static void main(String[] args) {
        int V = 6;
        List<List<Integer>> adj = new ArrayList<>();

        for (int i = 0; i < V; i++) {
            adj.add(new ArrayList<>());
        }
```

```
        // Sample DAG edges
        adj.get(5).add(2);
        adj.get(5).add(0);
        adj.get(4).add(0);
        adj.get(4).add(1);
        adj.get(2).add(3);
        adj.get(3).add(1);

        System.out.println("Topological Sort:");
        topologicalSort(V, adj);
    }
}
```