You're implementing the mergeKLists method to merge **K sorted linked lists** into one sorted list. The optimal approach is to use a **min-heap (priority queue)** to efficiently get the smallest current node across all lists.

Here’s the **correct and complete implementation**, including a fix to your ListNode class:

import java.util.PriorityQueue;

class Solution {

public ListNode mergeKLists(ListNode[] lists) {

// Define the priority queue with a comparator to sort nodes by their values

PriorityQueue<ListNode> pq = new PriorityQueue<>((a, b) -> a.val - b.val);

// Add the head of each list to the priority queue

for (ListNode list : lists) {

if (list != null) {

pq.add(list);

}

}

// Dummy node to build the merged list

ListNode dummy = new ListNode(0);

ListNode current = dummy;

// While the heap is not empty, pop the smallest node and process its next

while (!pq.isEmpty()) {

ListNode node = pq.poll();

current.next = node;

current = current.next;

if (node.next != null) {

pq.add(node.next);

}

}

return dummy.next;

}

}