Merge k Sorted Lists

This is the classic problem which generalizes merging two sorted linked lists; the optimal approach is using a min-heap (priority queue).

Key Idea:
- Use a min-heap to always extract the node with the smallest value.
- Push the first node of each list into the heap initially.
- Repeatedly pop the smallest node and push its next into the heap (if exist).
- Build the merged list incrementally.

Steps:
1. Create a min-heap (priority queue) ordered by node value.
2. Push all head nodes of the lists into the heap (if not null).
3. Initialize a dummy node and maintain a tail pointer.
4. While the heap is not empty,
   - Pop the smallest node.
   - Attach it to the merged list.
   - Push the popped node's next into the heap if it exist.
5. Return dummy->next.

Complexity:
- Let N = total number of nodes across all lists, k = number of lists.
- Time: $O(N \log k)$ (each node push/pop costs $\log k$).
- Space: $O(k)$ (heap size)