Problem 1

Time Complexity Proof:

- Each element is pushed and popped from the heap at most once.
- Heap operations (push/pop) take O (log K).
- Since there are K×N elements, the total time complexity is **O (K N log K)**.

Possible Optimizations:

- If K is small, a simple merge using two-pointer technique may be faster.
- If the arrays are stored in a way that allows direct sorted merging (e.g., a balanced BST), it could improve efficiency.

Problem 2:

Time Complexity Proof:

- We iterate through the array once (O(N).
- We use a single extra pointer (O (1) space complexity).
- Overall complexity is **O(N)**.

Possible Optimizations:

- If modifications to the original array are not allowed, we can return a new list instead.
- If the array is extremely large and memory is a concern, in-place modifications save space.