23. Merge k Sorted Lists

Hard

Merge *k* sorted linked lists and return it as one sorted list. Analyze and describe its complexity.

**Example:**

**Input:**  
[  
 1->4->5,  
 1->3->4,  
 2->6  
]  
**Output:** 1->1->2->3->4->4->5->6

/\*\*

\* Definition for singly-linked list.

\* struct ListNode {

\* int val;

\* ListNode \*next;

\* ListNode(int x) : val(x), next(NULL) {}

\* };

\*/

class Solution {

public:

ListNode\* mergeKLists(vector<ListNode\*>& lists) {

int s=lists.size();

if(s==0) return NULL;

ListNode\* dummy=(ListNode\*)malloc(sizeof(struct ListNode));

ListNode\* tail=dummy;

while(true){

int min\_index=0;

ListNode\* nextone=lists[0];

for(int i=1;i<s;i++){

ListNode\* curr=lists[i];

if(nextone==NULL||(nextone!=NULL&&curr!=NULL&&nextone->val>curr->val)){

min\_index=i;

nextone=curr;

}

}

if(nextone==NULL) break;

tail->next=nextone;

tail=tail->next;

lists[min\_index]=lists[min\_index]->next;

}

return dummy->next;

}

};

Success

[Details](https://leetcode.com/submissions/detail/205185522/)

Runtime: 280 ms, faster than 12.93% of C++ online submissions for Merge k Sorted Lists.

Memory Usage: 1.8 MB, less than 94.72% of C++ online submissions for Merge k Sorted Lists.