Problem: Sam Height <HSBC>

Sam among his friends wants to go to watch a movie in Armstord Cinema. There is something special about Armstord cinema whenever people come in the group here. They will get seats accordingly their heights. Sam as a curious guy always wants to sit in the middle as cinema has the best view from the middle. Now, Sam as the leader of his group decides who will join him for the movie. Initially, he has N−1 friends with him (N including him).

You are given N−1 numbers that represent the heights of Sam's friends. You are given the height of Sam as well.

Now, Sam can do two operations:
1. He can call a new friend of height H.
2. He can cancel any of his friend invitations.

Each operation will cost him a unit time.
He wants to do this as soon as possible.

Input format

- The first line contains T, where T is the test case.
- Each test case contains two lines,
    - The first line contains two space-separated integer N, S where N is the total number of Sam's friend and S is Sam height.
    - The second line contains N space-separated integers that represent the height of Sam's friend.

Output format
Print the required answer for each test case in a new line.

Constraints
$1 \le T \le 100$

$1 \le N \le 10^5$

$1 \le Ar[i] \le 10^9$

Note:

- Sam should always sit in middle and there's an equal number of persons in his left and right.
- Height of Sam is always unique.

Sample Input

```
2
3 2
4 3 1
1 5
6
```

Sample Output

```
1
1
```

Time Limit: 2

Memory Limit: 256

Explanation

In first test case :
We can cancel invitation of person of height 4 (Cost = 1)
In second Test Case:
We can invite person with height 4 (Cost =1)

Task 1:

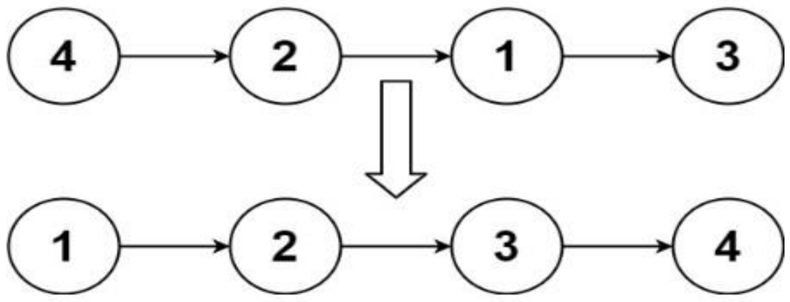
Complete the following code:

```
#include <bits/stdc++.h>
using namespace std;

int main() {


    // --- your code goes here
    cout<<"\n";
}
```

Problem: 148. Sort List [leetcode]

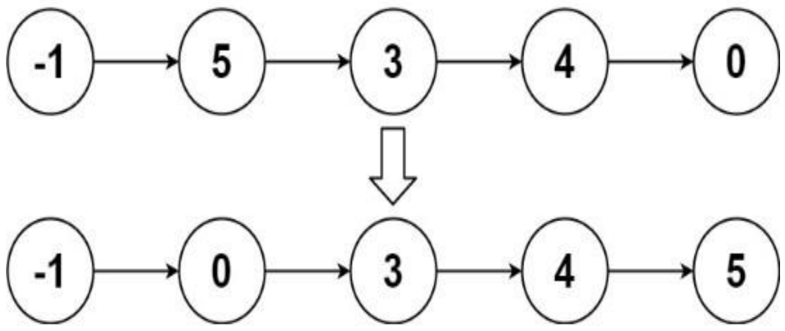
Given the head of a linked list, return *the list after sorting it in **ascending order***.

Example 1:



```
Input: head = [4,2,1,3]
Output: [1,2,3,4]
```

Example 2:



```
Input: head = [-1,5,3,4,0]
Output: [-1,0,3,4,5]
```

Example 3:

```
Input: head = []
Output: []
```

Constraints:

- The number of nodes in the list is in the range [0, 5 * 10^4].
- -10^5 <= Node.val <= 10^5

Follow up: Can you sort the linked list in O(n logn) time and O(1) memory (i.e. constant space)

Constraints

- $1 \le T \le 100$
- $1 \le N \le 100$
- $-10^9 \le A_i \le 10^9$

Input

3

9

-2 1 -3 4 -1 2 1 -5 4

1

1

5

5 4 -1 7 8

Output

6

1

23

Task 2:

Complete the following code:

```
#include <bits/stdc++.h>
using namespace std;

/**
 * Definition for singly-linked list.
 * class ListNode {
 *     int val;
 *     ListNode *next;
 *     ListNode() : val(0), next(nullptr) {}
 *     ListNode(int x) : val(x), next(nullptr) {}
 *     ListNode(int x, ListNode *next) : val(x), next(next) {}
 * };
 */
class Solution {
public:
    ListNode* sortList(ListNode* head) {
        if (!head || !head->next) return head;

        // Find the middle using slow and fast pointers
        ListNode* slow = head;
        ListNode* fast = head->next;
        while (fast && fast->next) {
            slow = slow->next;
            fast = fast->next->next;
        }

        ListNode* mid = slow->next;
        slow->next = nullptr;

        // Recursively split and merge
        ListNode* left = sortList(head);
        ListNode* right = sortList(mid);

        return merge(left, right);
    }

    ListNode* merge(ListNode* l1, ListNode* l2) {
        ListNode dummy(0);
        ListNode* tail = &dummy;

        while (l1 && l2) {
            if (l1->val < l2->val) {
                tail->next = l1;
                l1 = l1->next;
            } else {
                tail->next = l2;
                l2 = l2->next;
            }
            tail = tail->next;
        }
        tail->next = l1 ? l1 : l2;
        return dummy.next;
    }
};
```