Zonal Computing Olympiad 2019

Singing Tournament (SINGTOUR)

The Siruseri Singing Championship is going to start, and Lavanya wants to figure out the outcome before the tournament even begins! Looking at past tournaments, she realizes that the judges care only about the pitches that the singers can sing in, and so she devises a method through which she can accurately predict the outcome of a match between any two singers.

She represents various pitches as integers and has assigned a lower limit and an upper limit for each singer, which corresponds to their vocal range. For any singer, the lower limit will always be less than the upper limit. If a singer has lower limit L and upper limit U ($L < U$), it means that this particular singer can sing in all the pitches between L and U, that is they can sing in the pitches $\{L, L+1, L+2, \ldots, U\}$.

The lower bounds and upper bounds of all the singers are distinct. When two singers S_i and S_j with bounds (L_i, U_i) and (L_j, U_j) compete against each other, S_i wins if they can sing in every pitch that S_j can sing in, and some more pitches. Similarly, S_j wins if they can sing in every pitch that S_i can sing in, and some more pitches. If neither of those two conditions are met, the match ends up as a draw.

N singers are competing in the tournament. Each singer competes in N-1 matches, one match against each of the other singers. The winner of a match scores 2 points, and the loser gets no points. But in case of a draw, both the singers get 1 point each.

You are given the lower and upper bounds of all the N singers. You need to output the total scores of each of the N singers at the end of the tournament.

Input

- The first line contains a single integer, T, which is the number of testcases. The description of each testcase follows.
- The first line of every testcase contains a single integer, N, which is the number of singers.
- N lines follow, the i-th of which contains two integers: L_i and U_i, which correspond to the lower bound and upper bound of the i-th singer.

Output

For each testcase output a single line containing N integers, the i-th of which should be score of the i-th singer at the end of the tournament.

Constraints

- $1 \leq T \leq 5$
- $2 \leq N \leq 10^5$
- $1 \leq L_i < U_i \leq 10^9$
- All the $2 * N$ integers (lower bounds and upper bounds) are distinct.

Subtasks

Subtask #1 (15 points): $1 \leq N \leq 10^3$

Subtask #2 (25 points):

- $1 \leq N \leq 10^5$
- It is guaranteed that no match ends in a draw.

Subtask #3 (60 points): Original constraints.

Sample Input

```
2
3
10 20
13 18
15 19
3
10 22
13 21
15 20
```

Sample Output

```
4 1 1
4 2 0
```

Explanation

Testcase 1: There are three singers, with the lower bounds and upper bounds as (10, 20), (13, 18) and (15, 19).

When the first singer and second singer compete against each other in a match, we see that the second singer can sing in the pitches {13, 14, 15, 16, 17, 18}. Whereas the first singer can sing in the pitches {10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20}. So, we see that the first singer can sing everything that the second singer can, and also some other pitches. Hence the first singer wins this match, and gets 2 points. The second singer gets no points from this match.

When the first singer and third singer compete against each other in a match, we see that the third singer can sing in the pitches {15, 16, 17, 18, 19}. So again, we see that the first singer can sing everything that the third singer can. Hence the first singer wins this match, and gets 2 points. The third singer gets no points from this match.

When the second singer and third singer compete against each other in a match, we see that the second singer can sing in the pitches {13, 14, 15, 16, 17, 18}, whereas the third singer can sing in the pitches {15, 16, 17, 18, 19}. In particular, the second singer can sing in the pitch 14, which the third singer cannot sing in. And the third singer can sing in the pitch 19, which the second singer cannot sing in. So neither of the two conditions are met, and hence this match ends in a draw. Both the second and third singer get 1 point each.

Thus at the end of the tournament, the total score of first player is 2 + 2 = 4. Total score of the second player is 0 + 1 = 1. Total score of the third player is 0 + 1 = 1. Hence the output is 4 1 1

Testcase 2: There are three singers, with the lower bounds and upper bounds as (10, 22), (13, 21) and (15, 20).

We see that the first singer wins against both second and third singers. And the second singer wins against the third singer. So the final total scores are (2 + 2), (0 + 2), (0 + 0), which is 4 2 0. Note that this would be a valid testcase in Subtask 2, because no match ends in a draw.

UpDown Sequences (UPDOWSEQ)

A sequence of integers $(a_1, a_2, \dots, a_k)$ is said to be UpDown, if these inequalities hold true:

- $a_1 \le a_2$
- $a_2 \ge a_3$
- $a_3 \le a_4$ and so on.

That is, every even-indexed element should be at least as large as its adjacent elements. And every odd-indexed element should be at most as large as its adjacent elements. Formally, $a_{2i} \ge a_{2i+1}$ and $a_{2i+1} \le a_{2i+2}$, for all valid positions.

A subsegment is a consecutive portion of a sequence. That is, a subsegment of $(b_1, b_2, \dots, b_k)$ will be of the form $(b_i, b_{i+1}, \dots, b_j)$, for some i and j.

You are given a sequence $(s_1, s_2, \dots, s_n)$. You can insert at most one integer anywhere in this sequence. It could be any integer. After inserting an integer (or choosing not to), suppose you have the new sequence $(t_1, t_2, \dots, t_m)$. Note that m will either be n+1 or n. You want to maximize the length of the longest subsegment of $(t_1, t_2, \dots, t_m)$ which is UpDown, and output the length of that.

Input

- The first line contains a single integer, T, which is the number of testcases. The description of each testcase follows.
- The first line of every testcase contains a single integer, n, which is the number of integers in the original sequence.
- The second line contains n integers: $s_1, s_2, \dots, s_n$, which forms the original sequence.

Output

For each testcase output a single line containing one integer, which should be the length of the longest UpDown subsegment that you can get after inserting at most one integer.

Constraints

- $1 \le T \le 2$
- $1 \le n \le 10^6$
- $1 \le s_i \le 10^9$

Subtasks

Subtask #1 (20 points): $1 \le n \le 100$

Subtask #2 (10 points): $1 \le n \le 10000$

Subtask #3 (70 points): Original constraints

Sample Input

```
2
7
100 1 10 3 20 25 24
5
3 3 2 4 1
```

Sample Output

```
7
6
```

Explanation

Testcase 1: The original sequence is (100, 1, 10, 3, 20, 25, 24). Suppose you insert the element 5 between the elements 20 and 25, you will get the new sequence (100, 1, 10, 3, 20, 5, 25, 24). The longest UpDown subsegment of this sequence is (1, 10, 3, 20, 5, 25, 24), whose length is 7. You can check that you cannot do any better, and hence the answer is 7.

Testcase 2: The original sequence is (3, 3, 2, 4, 1). Suppose you insert the element 4 at the end, you will get the new sequence (3, 3, 2, 4, 1, 4). This entire sequence is UpDown, and so the longest UpDown subsegment of this sequence is (3, 3, 2, 4, 1, 4), whose length is 6. You can check that you cannot do any better, and hence the answer is 6.

Qualifying Cutoff: 25 marks and above (all students, no separate cutoffs for different categories)