F2. Cycling (Hard Version)

time limit per test: 5 seconds
memory limit per test: 1024 megabytes

This is the hard version of the problem. The difference between the versions is that in this version, $1 \le n \le 10^6$ and you need to output the answer for each prefix. You can hack only if you solved all versions of this problem.

Leo works as a programmer in the city center, and his lover teaches at a high school in the suburbs. Every weekend, Leo would ride his bike to the suburbs to spend a nice weekend with his lover.

There are n cyclists riding in front of Leo on this road right now. They are numbered $1, 2, \dots, n$ from front to back. Initially, Leo is behind the n-th cyclist. The i-th cyclist has an agility value a_i.

Leo wants to get ahead of the 1-st cyclist. Leo can take the following actions as many times as he wants:

- Assuming that the first person in front of Leo is cyclist i, he can go in front of cyclist i for a cost of a_i. This puts him behind cyclist $i-1$.
- Using his super powers, swap a_i and a_j $(1 \le i < j \le n)$ for a cost of $(j-i)$.

Leo wants to know the minimum cost to get in front of the 1-st cyclist.

In addition, he wants to know the answer for each $1 \le i \le n$, $[a_1, a_2, \dots, a_i]$ as the original array. The problems of different i are independent. To be more specific, in the i-th problem, Leo starts behind the i-th cyclist instead of the n-th cyclist, and cyclists numbered $i+1, i+2, \dots, n$ are not present.

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 10^4$). The description of the test cases follows.

The first line of each test case contains a positive integer n $(1 \le n \le 10^6)$, representing the number of the cyclists.

The second line of each test case contains n integers $a_1, \dots, a_n$ $(1 \le a_i \le 10^9)$.

It is guaranteed that the sum of n over all test cases does not exceed 10^6.

Output

For each test case, print n integers, the answers for the array $[a_1, a_2, \dots, a_i]$ for each $i = 1, 2, \dots n$ in this order.

Example

input Copy

```
4
3
1 2 4
4
1 1 1 1
2
1 2
4
4 1 3 2
```

output Copy

```
1 3 7
1 2 3 4
1 3
4 3 6 8
```

Note

In the first test case, one possible way to move from the position behind the n-th cyclist to the position in front of the 1-st cyclist is:

- Leo swaps a_2 $(i = 2)$ and a_3 $(j = 3)$, then the array becomes $[1, 4, 2]$; it costs $j - i = 3 - 2 = 1$.
- Leo is behind the 3-rd cyclist and moves behind the 2-nd cyclist; it costs $a_3 = 2$.
- Leo swaps a_1 $(i = 1)$ and a_2 $(j = 2)$, then the array becomes $[4, 1, 2]$; it costs $j - i = 2 - 1 = 1$.
- Leo is behind the 2-nd cyclist and moves behind the 1-st cyclist; it costs $a_2 = 1$.
- Leo swaps a_1 $(i = 1)$ and a_2 $(j = 2)$, then the array becomes $[1, 4, 2]$; it costs $j - i = 2 - 1 = 1$.
- Leo moves ahead of the 1-st cyclist; it costs $a_1 = 1$.

So the total cost is $1 + 2 + 1 + 1 + 1 + 1 = 7$. It can be proved that 7 is the minimum cost.

In the second test case, to move ahead of the 1-st cyclist from the position behind the n-th cyclist, Leo should not swap anyone's agility value. The total cost is $1 + 1 + 1 + 1 = 4$.