Many Many Heads 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

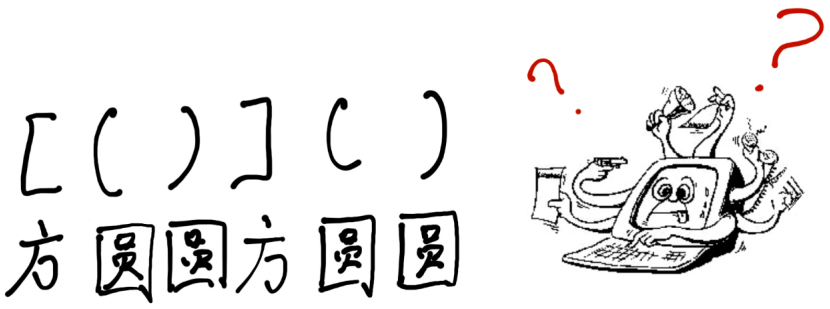
Multi-Heads Cup, or MHC for short, is a worldwide programming contest made for the participants with **many many heads**. In 2023, the chief judge of this event, Little Cyan Fish, solved a tricky authentication problem using many types of brackets.

That was the first time Little Cyan Fish saw contestants with many many heads. And this time, Little Cyan Fish brings another bracket problem to his best friend who has many many heads. Little Cyan Fish has n types of brackets in his hand, and each type of bracket is divided into left brackets and right brackets. For convenience, we use $\texttt{L}^i$ to denote the i-th type of left bracket, and $\texttt{R}^i$ to denote the i-th type of right bracket.

"Hey, don't forget," Little Cyan Fish thought, "I have introduced to you what a balanced bracket sequence is before!" To ensure you understand the concept of a balanced bracket sequence, Little Cyan Fish prepared the following formal definition of a balanced bracket sequence:

- ε (an empty string) is a balanced bracket sequence.
- If A is a balanced bracket sequence, (A) is also a balanced bracket sequence.
- If A and B are balanced bracket sequences, then AB is also a balanced bracket sequence.

For example, "`()`", "`()()`" and "`(())`" are balanced bracket sequences, but "`)(`", "`(`" and "`))`" are not.



Now, Little Cyan Fish gives you a bracket sequence S of length n, containing n types of brackets. Unfortunately, Little Cyan Fish forgot the types of brackets at some positions, and also forgot the directions of brackets at these positions. The memory of these positions has become blurred and is represented by `?` by Little Cyan Fish.

Little Cyan Fish is very curious, for all $1 \le l \le r \le n$, how many pairs of (l, r) correspond to substrings $S[l \dots r]$, such that there exists a way to fill `?` with some type of bracket in some direction, such that for each $1 \le i \le n$, we have:

- Extract all brackets of type i (i.e. all $\texttt{L}^i$ and $\texttt{R}^i$), the resulting bracket sequence (containing only the i-th type of bracket) is a balanced bracket sequence.

For example, if we use "()" to denote the first type of bracket and "[]" to denote the second type of bracket, the bracket string `([?)` satisfies the above condition, because we can replace `?` with `]`. `(?)]` also satisfies the above condition, because we can replace `?` with `[`.

Little Cyan Fish wants you to calculate the number of all valid pairs (l, r).

Input

There are multiple test cases. The first line of the input contains a single integer T ($1 \leq T$), indicating the number of test cases.

For each test case, the first line of the input contains an integer n ($1 \leq n \leq 2 \times 10^5$), indicating the length of the bracket string.

The next line contains n integers $x_1, x_2, \ldots, x_n$ ($-n \leq x_i \leq n$). These n integers describe the information of the bracket string, where:

- If $x_i > 0$, then the i-th position represents the x_i-th type of left bracket (i.e. $\texttt{L}^{x_i}$).
- If $x_i < 0$, then the i-th position represents the $-x_i$-th type of right bracket (i.e. $\texttt{R}^{-x_i}$).
- If $x_i = 0$, it means Little Cyan Fish forgot the information at this position (i.e. `?`).

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5.

Output

For each test case, output a single line containing one integer, indicating the answer.

Example

standard input	standard output
4 4 1 2 -1 -2 4 1 0 -2 0 6 1 2 3 -3 -2 -1 8 1 0 0 3 0 0 0 -2	1 3 3 14