Basic Substring Structure

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

After writing the paper *Faster Algorithms for Internal Dictionary Queries*, Little Cyan Fish and Kiwihadron decided to write this problem.

Let $\mathrm{lcp}(s,t)$ be the length of the longest common prefix of two strings $s = s_1 s_2 \ldots s_n$ and $t = t_1 t_2 \ldots t_m$, which is defined as the maximum integer k such that $0 \le k \le \min(n, m)$ and $s_1 s_2 \ldots s_k$ equals $t_1 t_2 \ldots t_k$.

Little Cyan Fish gives you a non-empty string $s = s_1 s_2 \ldots s_n$. Let $f(s) = \sum_{i=1}^{n} \mathrm{lcp}(s, \mathrm{suf}(s,i))$, where $\mathrm{suf}(s,i)$ is the suffix of s starting from s_i (i.e. $\mathrm{suf}(s,i) = s_i s_{i+1} \ldots s_n$). Note that in this problem, the alphabet contains n letters, not just 26.

For each $i = 1, 2, \cdots, n$, you are asked to answer the following query: if you MUST change s_i to another different character c ($c \ne s_i$), choose the best character c and calculate the maximum value of $f(s^{(i)})$, where $s^{(i)} = s_1 \ldots s_{i-1} c s_{i+1} \ldots s_n$.

Input

There are multiple test cases. The first line of the input contains an integer T indicating the number of test cases. For each test case:

The first line contains an integer n ($2 \le n \le 2 \times 10^5$) indicating the length of the string.

The second line contains n integers $s_1, s_2, \ldots, s_n$ ($1 \le s_i \le n$) where s_i indicates that the i-th character of the string is the s_i-th letter in the alphabet.

It's guaranteed that the sum of n over all test cases doesn't exceed 2×10^5.

Output

Let $m(i)$ be the maximum value of $f(s^{(i)})$. To decrease the size of output, for each test case output one line containing one integer which is $\sum_{i=1}^{n} (m(i) \oplus i)$, where $\oplus$ is the bitwise exclusive or operator.

Example

standard input	standard output
2 4 2 1 1 2 12 1 1 4 5 1 4 1 9 1 9 8 10	15 217

Note

For the first sample test case, let's first calculate the value of $m(1)$.

- If you change s_1 to 1, then $f(s^{(1)}) = 4 + 2 + 1 + 0 = 7$.
- If you change s_1 to 3 or 4, then $f(s^{(1)}) = 4 + 0 + 0 + 0 = 4$.

So $m(1) = 7$.

Similarly, $m(2) = 6$, $m(3) = 6$ and $m(4) = 4$. So the answer is $(7 \oplus 1) + (6 \oplus 2) + (6 \oplus 3) + (4 \oplus 4) = 15$.