Left Shifting 2

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

Given a string consisting of lower-cased English letters, we say the string is beautiful if all neighboring characters are different. For example, `icpc` and `kunming` are beautiful but `hello` is not, because its 3-rd and 4-th characters are the same.

Given a string $S = s_0 s_1 \cdots s_{n-1}$ of length n consisting of lower-cased English letters, let $f(S, d)$ be the string obtained by shifting S to the left d times. That is $f(S, d) = s_{(d+0) \bmod n} s_{(d+1) \bmod n} \cdots s_{(d+n-1) \bmod n}$.

Let $g(S, d)$ be the smallest number of operations needed to make string $f(S, d)$ beautiful. In each operation, you can change one character in $f(S, d)$ to any lower-cased English letter.

Find a non-negative integer d which minimizes $g(S, d)$ and output the minimized value.

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 and only line contains a string $s_0 s_1 \cdots s_{n-1}$ ($1 \le n \le 5 \times 10^5$) consisting only of lower-cased English letters.

It's guaranteed that the sum of n of all test cases will not exceed 5×10^5.

Output

For each test case, output one line containing one integer, indicating the smallest possible $g(S, d)$.

Example

standard input	standard output
3 abccbbbbd abcde x	2 0 0

Note

For the first sample test case, consider $d = 5$. We have $f(S, 5) =$ `bbbdabccb`. For this string, we can change its 2-nd character to `x` and its 8-th character to `y`, so the string will become `bxbdabcyb`, which is a beautiful string.