Cute String Merging

Time limit: 2.0s **Memory limit:** 256M

Marcus loves strings, especially cute strings and ones consisting of the letters `a`, `b`, and `c`. He recently received such a string (which we'll call S), and to make it as cute as possible he will perform zero or more merge operations on it to make it shorter (Marcus thinks short strings are very cute).

A merge operation is defined as follows:

Let s be the current state of Marcus's string (its length is denoted as $|s|$).

- Select an index i $(1 \leq i < |s|)$ such that $s_i \neq s_{i+1}$.
- Remove the letters s_i and s_{i+1} and replace them with the letter that is neither one of those two out of `a`, `b`, and `c` (i.e. if you removed the letters `a` and `b` you would replace them with the letter `c`).

Marcus managed to shorten his string quite a bit, but he wants you to verify his answer by helping him find the shortest length he can make his string after some number (possibly zero) of merge operations.

Lastly, to ensure the integrity of your solution, each input file will contain T separate test cases.

Constraints

For all subtasks:

$1 \leq |S|, T \leq 10^6$

S will only contain the letters `a`, `b`, and `c`.

The sum of $|S|$ over all test cases will be at most 10^6.

Subtask 1 [20%]

The sum of $|S|$ over all test cases will be at most 600.

Subtask 2 [30%]

The sum of $|S|$ over all test cases will be at most $5\,000$.

Subtask 3 [50%]

No additional constraints.

Input Specification

The first line will contain T, the number of test cases. T cases then follow.

Each case consists of a single line, the string S in that test.

Output Specification

For each test case, output the shortest length that S can reach after some number of merge operations.

Sample Input

```
3
abcabc
cccc
ab
```

Sample Output

```
2
4
1
```

Explanation

Here is an example sequence of merge operations in the first test case that yields an optimal answer (the portion in brackets denotes the indices being merged):

- `ab[ca]bc` -> `abbbc`
- `abb[bc]` -> `abba`
- `ab[ba]` -> `abc`
- `a[bc]` -> `aa`
- `aa`

In the second case, no merge operations are possible so the answer is simply the length of the string.

In the last test case, `ab` can be merged into `c`.