



Palindromic Partitions

Time Limit: 10 s Memory Limit: 128 MB

A *partition* of a string s is a set of one or more non-overlapping non-empty substrings of s (call them $a_1, a_2, a_3, \dots, a_d$), such that s is their concatenation: $s = a_1 + a_2 + a_3 + \dots + a_d$. We call these substrings "*chunks*" and define the *length* of such a partition to be the number of chunks, d .

We can represent the partition of a string by writing each chunk in parentheses. For example, the string "decode" can be partitioned as (d)(ec)(ode) or (d)(e)(c)(od)(e) or (decod)(e) or (decode) or (de)(code) or a number of other ways.

A partition is *palindromic* if its chunks form a palindrome when we consider each chunk as an atomic unit. For example, the only palindromic partitions of "decode" are (de)(co)(de) and (decode). This also illustrates that every word has a trivial palindromic partition of length one.

Your task is to compute the maximal possible number of chunks in palindromic partition.

Input

The input starts with the number of test cases t in the first line. The following t lines describe individual test cases consisting of a single word s , containing only lowercase letters of the English alphabet. There are no spaces in the input.

Output

For every testcase output a single number: the length of the longest palindromic partition of the input word s .

Constraints

Let us denote the length of the input string s with n .

- $1 \leq t \leq 10$
- $1 \leq n \leq 10^6$

Subtask 1 (15 points)

- $n \leq 30$

Subtask 2 (20 points)

- $n \leq 300$

Subtask 3 (25 points)

- $n \leq 10\,000$



Subtask 4 (40 points)

- no additional constraints

Example

| Input | Output |
|----------|--------|
| 4 | 3 |
| bonobo | 5 |
| deleted | 7 |
| racecar | 1 |
| racecars | |