Palindrome Index

Given a string, S, of lowercase letters, determine the index of the character whose removal will make S a palindrome. If S is already a palindrome or no such character exists, then print -1. There will always be a valid solution, and any correct answer is acceptable. For example, if $S =$ *"bcbc"*, we can either remove *'b'* at index 0 or *'c'* at index 3.

Input Format

The first line contains an integer, T, denoting the number of test cases.
Each line i of the T subsequent lines (where $0 \leq i < T$) describes a test case in the form of a single string, S_i.

Constraints

- $1 \leq T \leq 20$
- $1 \leq |S| \leq 10^5 + 5$
- All characters are lowercase English letters.

Output Format

Print an integer denoting the *zero-indexed* position of the character that makes S not a palindrome; if S is already a palindrome or no such character exists, print -1.

Sample Input

```
3
aaab
baa
aaa
```

Sample Output

```
3
0
-1
```

Explanation

Test Case 1: "aaab"
Removing *'b'* at index 3 results in a palindrome, so we print 3 on a new line.

Test Case 2: "baa"
Removing *'b'* at index 0 results in a palindrome, so we print 0 on a new line.

Test Case 3: "aaa"
This string is already a palindrome, so we print -1; however, 0, 1, and 2 are also all acceptable answers, as the string will still be a palindrome if any one of the characters at those indices are removed.

Note: The custom checker logic for this challenge is available here.