Kalindrome Strings (kalindrome)

William loves strings and their properties, and he is especially fond of palindromes. He likes palindromes so much, that he started reading lots of research papers on them and finally stumbled upon a new kind of property that generalizes the palindrome concept: the *kalindrome*.

We say that a string is a *kalindrome* if, for some positive integer K, it's possible to split the string in many substrings, each of them K characters long, so that they can be "read" from left to right in the same way as right to left. More precisely: if we concatenate the substrings starting from the last one, we will recreate the original string.

For example: the word "banaba" is a kalindrome, because if we choose $K = 2$ then we obtain the substrings: **ba**, **na**, **ba**, which can indeed be concatenated starting from the last one, to reconstruct the original word.

You might have noticed that, by this definition, any string is a kalindrome! That's true, in fact we could choose K to be equal to the string's length, and we would then obtain a single substring which would of course respect the definition! However, William is interested to know **what's the smallest K for a given string that would prove it is a kalindrome**.

Help him by writing a program that automates this test!

☞ Among the attachments of this task you may find a template file `kalindrome.*` with a sample incomplete implementation.

Input

The first line contains the only integer N, the length of the string to be tested. The second line contains a string S.

Output

You need to write a single line with an integer: the smallest positive integer K that proves the kalindromeness of the string S.

Constraints

- $1 \le N \le 1\,000\,000$.
- The string S is N-characters long and only contains lowercase English letters.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

– **Subtask 1** (0 points) Examples.

– **Subtask 2** (40 points) The value of K is always either 1 or N.

– **Subtask 3** (40 points) $N \le 1000$.

– **Subtask 4** (20 points) No additional limitations.

Examples

input	output

`6`
`banaba`
	`2`

`12`
`eszyciyciesz`
	`3`

Explanation

In the **first sample case**, we can split the string in three 2-characters long substrings: **ba**, **na**, **ba**.

In the **second sample case**, we can split the string in four 3-characters long substrings: **esz**, **yci**, **yci**, **esz**.