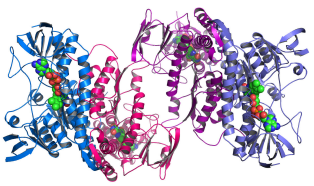
D Dividing DNA

Time limit: 2s



CC BY-NC-SA 2.0 by Argonne National Laboratory on Flickr

At the Bacteria And Protein Centre, you own a large collection of DNA. In fact, new strands of DNA come in all the time. To organise the vast amount of data, you identify each piece by its unique substrings: substrings that do not already occur in the database.

Your database can quickly determine whether a given piece of DNA occurs as a substring in the database or not. Naturally, if a certain DNA string is found in the database, it also contains all its substrings.

You now want to determine the *uniqueness* of a given piece of DNA: the maximal number of disjoint substrings it contains that are absent from the database.

You are given the length n of the query string $q_1 \dots q_n$, and you can repeatedly ask the database whether it contains the substring $q_i \dots q_{j-1}$.

As an example, consider the first sample interaction. In this case, the database contains strings "`TGC`" and "`CT`", and the query string is "`CTGCAA`". It has uniqueness 3, because it can be split into the new substrings "`CTGC`", "`A`", and "`A`". The new substring "`CTGC`" cannot be split up further: for example, the subdivision "`CT`" and "`GC`" is not allowed, because both substrings occur (possibly as substrings) in the database. Note that the actual characters in the string are not used in the interaction.

You may use at most $2n$ queries to the database.

Interaction

This is an interactive problem. Your submission will be run against an *interactor*, which reads from the standard output of your submission and writes to the standard input of your submission. This interaction needs to follow a specific protocol:

The interactor first sends one line with an integer n ($1 \leq n \leq 10\,000$), the length of the DNA piece for which you need to compute the uniqueness.

Then, your program should make at most $2n$ queries. Each query is made by printing one line of the form "`?` i j" ($0 \leq i < j \leq n$), indicating that you want to query whether the substring starting at position i and ending at position $j-1$ occurs in the database. The interactor will respond with either "`present`" or "`absent`", indicating whether the substring was found in the database.

When you have determined the uniqueness x of the piece of DNA, print one line of the form "`!` x", after which the interaction will stop. Printing the answer does not count as a query.

Make sure you flush the buffer after each write.

A testing tool is provided to help you develop your solution.

Using more than $2n$ queries will result in a wrong answer.

Read	Sample Interaction 1	Write
6		
	? 4 6	
absent		
	? 4 5	
absent		
	? 5 6	
absent		
	? 0 1	
present		
	? 0 2	
present		
	? 2 4	
present		
	? 1 4	
present		
	? 0 4	
absent		
	! 3	

Read	Sample Interaction 2	Write
10		
	? 0 10	
absent		
	? 0 9	
present		
	? 1 10	
present		
	! 1	