Problem A. The Walking Adam

Input file: `adam.in`
Output file: `standard output`

Adam has just started learning how to walk (with some help from his brother Omar), and he falls down a lot. In order to balance himself, he raises his hands up in the air (that's a true story), and once he puts his hands down, he falls.

You are given a string, each character represents a step he walks, if that character is 'U' that means his hands are up in this step, if this character is 'D' that means his hands are down and he fell down in this step. Your task is to count how many steps he will walk before falling down for the first time.

Input

Your program will be tested on one or more test cases. The first line of the input will be a single integer T representing the number of test cases.

Each test case will consist of a single line, containing a non-empty string of at most 100 characters, and each character is either 'U' or 'D'. The characters from left to right represent Adam's steps in the order he walks them.

Output

For each test case print a single line containing the number of steps that Adam will walk before falling down, or the length of the string if he won't fall down.

Example

adam.in	standard output
3 UUUDU DDD UU	3 0 2

Note

In the first test case, he falls down after 3 steps.

In the second test case, he falls down before making any steps.

In the third test case, he doesn't fall down at all.

Problem B. Duplicate Files

Input file: `files.in`
Output file: `standard output`

Many of us have this problem on their computer, where we make several copies of the exact same files, which uses more memory. So you finally decided to write a program to remove all duplicate files.

Here's how your computer works. Whenever you create a new file, it gets a unique ID (when you make a copy of an existing file, the new copy gets a new ID). The IDs are relative to the time, so older files get smaller IDs (but the IDs are not necessary sequential). Also each file has a name, but multiple files can have the same name, and when 2 or more files get the same name, this means they are all exactly the same and they are just identical copies of the one of them with the smallest ID.

You are given the list of all files with their names and IDs, your task is to delete all duplicates and just keep the oldest copy of each file.

Input

Your program will be tested on one or more test cases. The first line of the input will be a single integer $\boldsymbol{T}$ representing the number of test cases. Followed by $\boldsymbol{T}$ test cases.

Each test case starts with a line containing an integer $\boldsymbol{N}$ ($1 \leq \boldsymbol{N} \leq 10^5$) representing the number of files followed by $\boldsymbol{N}$ lines, each line will contain the file name followed by a space then the file ID. The file name is a non-empty string of at most 10 lower case English letters, and the ID is a positive integer which is at most 10^5. All IDs will be distinct within each test case.

Output

For each test case print a single line containing the IDs of the files which won't get deleted, the IDs should be sorted in increasing order and separated by a single space.

Example

files.in	standard output
2 2 aaa 6 aa 5 3 file 3 file 2 file 1	5 6 1

Problem C. Self Describing Numbers

Input file: `self.in`
Output file: `standard output`

In one of the classes, Dr. Edsger explained that an integer is said to be "self-describing"if it has the property that: the first digit represents the number of zeros in the integer; the second digit represents the number of ones in the integer, and so on.

For example, 1210 is a self-describing integer since:

- the first digit is 1 and there is 1 zero in the number &
- the second digit is 2 and there are 2 ones in the number &
- the third digit is 1 and there is 1 two in the number &
- the fourth digit is 0 and there are 0 threes in the number.

At the end of the class, Dr. Edsger gave a small homework: create a program to check whether an integer is self-describing or not.

Input

The first line of the input contains an integer T, the number of test cases.

Each test case consists of a single integer N ($0 \leq N < 10^{10}$), without leading zeros.

Output

For each test case, print a single line containing the answer which is either "self-describing"or "not self-describing"without the quotes.

Example

self.in	standard output
2 1210 2017	self-describing not self-describing

Problem D. Guess Number

Input file: `standard input`
Output file: `standard output`

We have a hidden number N, where $1 \leq N \leq 10^9$, Your is task to find it.

You guess a number X and we respond with three types of responses, "$>$" if $N > X$, "$<$" if $N < X$, or "$=$". if the response is N equals X , which means your guess was correct and then, you shout print "! N" and terminate the program immediately.

If you make more than 30 guesses, you will receive a Wrong Answer verdict.

Interaction Protocol

After printing a number X do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

Example

standard input	standard output
< > > =	11 6 9 10 ! 10