# F. Coffee Varieties (easy version)

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

*This is the easy version of the problem. You can find the hard version in the Div. 1 contest. Both versions only differ in the number of times you can ask your friend to taste coffee.*

**This is an interactive problem.**

You're considering moving to another city, where one of your friends already lives. There are $n$ cafés in this city, where $n$ is a power of two. The $i$-th café produces a single variety of coffee $a_i$.

As you're a coffee-lover, before deciding to move or not, **you want to know the number $d$ of distinct varieties of coffees** produced in this city.

You don't know the values $a_1, \ldots, a_n$. Fortunately, your friend has a memory of size $k$, where $k$ is a power of two.

Once per day, you can ask him to taste a cup of coffee produced by the café $c$, and he will tell you if he tasted a similar coffee during the last $k$ days.

You can also ask him to take a medication that will reset his memory. He will forget all previous cups of coffee tasted. You can reset his memory at most $30\,000$ times.

More formally, the memory of your friend is a queue $S$. Doing a query on café $c$ will:

- Tell you if $a_c$ is in $S$;
- Add $a_c$ at the back of $S$;
- If $|S| > k$, pop the front element of $S$.

Doing a reset request will pop all elements out of $S$.

Your friend can taste at most $\frac{2n^2}{k}$ cups of coffee in total. Find the diversity $d$ (number of distinct values in the array $a$).

Note that asking your friend to reset his memory **does not count** towards the number of times you ask your friend to taste a cup of coffee.

In some test cases the behavior of the interactor **is adaptive**. It means that the array $a$ may be **not fixed** before the start of the interaction and may **depend on your queries**. It is guaranteed that at any moment of the interaction, there is at least one array $a$ consistent with all the answers given so far.

## Input

The first line contains two integers $n$ and $k$ ($1 \le k \le n \le 1024$, $k$ and $n$ are powers of two).

It is guaranteed that $\frac{2n^2}{k} \le 20\,000$.

## Interaction

You begin the interaction by reading $n$ and $k$.

- To ask your friend to taste a cup of coffee produced by the café $c$, in a separate line output
  $? \ \ c$

  Where $c$ must satisfy $1 \le c \le n$. Don't forget to flush, to get the answer.

  In response, you will receive a single letter $\mathtt{Y}$ (yes) or $\mathtt{N}$ (no), telling you if variety $a_c$ is one of the last $k$ varieties of coffee in his memory.

- To reset the memory of your friend, in a separate line output the single letter $\mathtt{R}$ **in upper case**. You can do this operation at most $30\,000$ times.
- When you determine the number $d$ of different coffee varieties, output
  $! \ \ d$

In case your query is invalid, you asked more than $\frac{2n^2}{k}$ queries of type $?$ or you asked more than $30\,000$ queries of type $\mathtt{R}$, the program will print the letter $\mathtt{E}$ and will finish interaction. You will receive a **Wrong Answer** verdict. Make sure to

### → Problem tags

graphs interactive *2700

No tag edit access

### → Contest materials

- Announcement (en)
- Tutorial (en)

exit immediately to avoid getting other verdicts.

After printing a query 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.

### Hack format

The first line should contain the word `fixed`

The second line should contain two integers $n$ and $k$, separated by space ($1 \le k \le n \le 1024$, $k$ and $n$ are powers of two).

It must hold that $\frac{2n^2}{k} \le 20\,000$.

The third line should contain $n$ integers $a_1, a_2, \ldots, a_n$, separated by spaces ($1 \le a_i \le n$).

### Examples

**input** Copy

```
4 2
N
N
Y
N
N
N
N
```

**output** Copy

```
? 1
? 2
? 3
? 4
R
? 4
? 1
? 2
! 3
```

input

Copy

```
8 8
N
N
N
N
Y
Y
```

output

Copy

```
? 2
? 6
? 4
? 5
? 2
? 5
! 6
```

### Note

In the first example, the array is $a = [1, 4, 1, 3]$. The city produces $3$ different varieties of coffee ($1$, $3$ and $4$).

The successive varieties of coffee tasted by your friend are $1, 4, \mathbf{1}, 3, 3, 1, 4$ (bold answers correspond to `Y` answers). Note that between the two `? 4` asks, there is a reset memory request `R`, so the answer to the second `? 4` ask is `N`. Had there been no reset memory request, the answer to the second `? 4` ask is `Y`.

In the second example, the array is $a = [1, 2, 3, 4, 5, 6, 6, 6]$. The city produces $6$ different varieties of coffee.

The successive varieties of coffee tasted by your friend are $2, 6, 4, 5, \mathbf{2}, \mathbf{5}$.