# Codeforces Round #371 (Div. 1)

## A. Sonya and Queries

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

Today Sonya learned about long integers and invited all her friends to share the fun. Sonya has an initially empty multiset with integers. Friends give her $t$ queries, each of one of the following type:

1. $+ \ a_i$ — add non-negative integer $a_i$ to the multiset. Note, that she has a multiset, thus there may be many occurrences of the same integer.
2. $- \ a_i$ — delete a single occurrence of non-negative integer $a_i$ from the multiset. It's guaranteed, that there is at least one $a_i$ in the multiset.
3. $? \ s$ — count the number of integers in the multiset (with repetitions) that match some pattern $s$ consisting of $0$ and $1$. In the pattern, $0$ stands for the even digits, while $1$ stands for the odd. Integer $x$ matches the pattern $s$, if the parity of the $i$-th from the right digit in decimal notation matches the $i$-th from the right digit of the pattern. If the pattern is shorter than this integer, it's supplemented with $0$-s from the left. Similarly, if the integer is shorter than the pattern its decimal notation is supplemented with the $0$-s from the left.

For example, if the pattern is $s = 010$, than integers $92$, $2212$, $50$ and $414$ match the pattern, while integers $3$, $110$, $25$ and $1030$ do not.

### Input

The first line of the input contains an integer $t$ ($1 \le t \le 100\,000$) — the number of operation Sonya has to perform.

Next $t$ lines provide the descriptions of the queries in order they appear in the input file. The $i$-th row starts with a character $c_i$ — the type of the corresponding operation. If $c_i$ is equal to '+' or '−' then it's followed by a space and an integer $a_i$ ($0 \le a_i < 10^{18}$) given without leading zeroes (unless it's 0). If $c_i$ equals '?' then it's followed by a space and a sequence of zeroes and onse, giving the pattern of length no more than $18$.

It's guaranteed that there will be at least one query of type '?'.

It's guaranteed that any time some integer is removed from the multiset, there will be at least one occurrence of this integer in it.

### Output

For each query of the third type print the number of integers matching the given pattern. Each integer is counted as many times, as it appears in the multiset at this moment of time.

**Examples**

input

```
12
+ 1
+ 241
? 1
+ 361
- 241
? 0101
+ 101
? 101
- 101
? 101
+ 4000
? 0
```

output

```
2
1
2
1
1
```

input

```
4
+ 200
+ 200
- 200
? 0
```

output

```
1
```

### Note

Consider the integers matching the patterns from the queries of the third type. Queries are numbered in the order they appear in the input.

1. 1 and 241.
2. 361.
3. 101 and 361.
4. 361.
5. 4000.

# B. Searching Rectangles

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

Filya just learned new geometry object — rectangle. He is given a field consisting of $n \times n$ unit cells. Rows are numbered from bottom to top with integer from $1$ to $n$. Columns are numbered from left to right with integers from $1$ to $n$. Cell, located at the intersection of the row $r$ and column $c$ is denoted as $(r, c)$. Filya has painted two rectangles, such that their sides are parallel to coordinate axes and each cell lies fully inside or fully outside each of them. Moreover, no cell lies in both rectangles.

Later, hedgehog Filya became interested in the location of his rectangles but was unable to find the sheet of paper they were painted on. They were taken by Sonya and now she wants to play a little game with Filya. He tells her a query rectangle and she replies with the number of initial rectangles that lie **fully inside** the given query rectangle. The query rectangle should match the same conditions as initial rectangles. Rectangle lies fully inside the query if each o its cells lies inside the query.

Filya knows Sonya really well, so is sure that if he asks more than $200$ questions she will stop to reply.

## Input

The first line of the input contains an integer $n$ ($2 \le n \le 2^{16}$) — size of the field.

For each query an integer between $0$ and $2$ is returned — the number of initial rectangles that lie fully inside the query rectangle.

## Output

To make a query you have to print "`?` $x_1$ $y_1$ $x_2$ $y_2$" (without quotes) ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le n$), where $(x_1, y_1)$ stands for the position of the bottom left cell of the query and $(x_2, y_2)$ stands for the up right cell of the query. You are allowed to ask no more than $200$ queries. After each query you should perform "flush" operation and read the answer.

In case you suppose you've already determined the location of two rectangles (or run out of queries) you should print "`!` $x_{11}$ $y_{11}$ $x_{12}$ $y_{12}$ $x_{21}$ $y_{21}$ $x_{22}$ $y_{22}$" (without quotes), where first four integers describe the bottom left and up right cells of the first rectangle, and following four describe the corresponding cells of the second rectangle. You can print the rectangles in an arbitrary order. After you have printed the answer, print the end of the line and perform "flush". Your program should terminate immediately after it print the answer.

## Interaction

To flush you can use (just after printing an integer and end-of-line):

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

You will get the `Wrong Answer` verdict if you ask more than $200$ queries, or if you print an incorrect coordinates.

You will get the `Idleness Limit Exceeded` verdict if you don't print anything (but you should) or if you forget about flushing the output (more info below).

**Hacking.**

The first line should contain an integer $n$ ($2 \le n \le 2^{16}$).

The second line should contain four integers $x_1, y_1, x_2, y_2$ ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le n$) — the description of the first rectangle.

The third line contains the description of the second rectangle in the similar way.

**Example**

| input |
| --- |
| 5<br>2<br>1<br>0<br>1<br>1<br>1<br>0<br>1 |

| output |
| --- |
| ? 1 1 5 5<br>? 1 1 3 3<br>? 1 1 3 1<br>? 2 2 2 2<br>? 3 3 5 5<br>? 3 3 3 5<br>? 3 3 3 4<br>? 3 4 3 5 |

```
! 2 2 2 2 3 4 3 5
```

# C. Sonya and Problem Wihtout a Legend

time limit per test: 5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Sonya was unable to think of a story for this problem, so here comes the formal description.

You are given the array containing $n$ positive integers. At one turn you can pick any element and increase or decrease it by $1$. The goal is the make the array strictly increasing by making the minimum possible number of operations. You are allowed to change elements in any way, they can become negative or equal to $0$.

## Input

The first line of the input contains a single integer $n$ ($1 \le n \le 3000$) — the length of the array.

Next line contains $n$ integer $a_i$ ($1 \le a_i \le 10^9$).

## Output

Print the minimum number of operation required to make the array strictly increasing.

**Examples**

| input |
| --- |
| 7<br>2 1 5 11 5 9 11 |
| output |
| 9 |

| input |
| --- |
| 5<br>5 4 3 2 1 |
| output |
| 12 |

## Note

In the first sample, the array is going to look as follows:

$2\ 3\ 5\ 6\ 7\ 9\ 11$

$|2 - 2| + |1 - 3| + |5 - 5| + |11 - 6| + |5 - 7| + |9 - 9| + |11 - 11| = 9$

And for the second sample:

$1\ 2\ 3\ 4\ 5$

$|5 - 1| + |4 - 2| + |3 - 3| + |2 - 4| + |1 - 5| = 12$

# D. Animals and Puzzle

time limit per test: 5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Owl Sonya gave a huge lake puzzle of size $n \times m$ to hedgehog Filya as a birthday present. Friends immediately started to assemble the puzzle, but some parts of it turned out to be empty — there was no picture on them. Parts with picture on it are denoted by $1$, while empty parts are denoted by $0$. Rows of the puzzle are numbered from top to bottom with integers from $1$ to $n$, while columns are numbered from left to right with integers from $1$ to $m$.

Animals decided to complete the picture and play with it, as it might be even more fun! Owl and hedgehog ask each other some queries. Each query is provided by four integers $x_1, y_1, x_2, y_2$ which define the rectangle, where $(x_1, y_1)$ stands for the coordinates of the up left cell of the rectangle, while $(x_2, y_2)$ stands for the coordinates of the bottom right cell. The answer to the query is the size of the maximum **square** consisting of picture parts only (only parts denoted by $1$) and located fully inside the query rectangle.

Help Sonya and Filya answer $t$ queries.

## Input

The first line of the input contains two integers $n$ and $m$ ($1 \le n, m \le 1000$) — sizes of the puzzle.

Each of the following $n$ lines contains $m$ integers $a_{ij}$. Each of them is equal to $1$ if the corresponding cell contains a picture and $0$ if it's empty.

Next line contains an integer $t$ ($1 \le t \le 1\,000\,000$) — the number of queries.

Then follow $t$ lines with queries' descriptions. Each of them contains four integers $x_1, y_1, x_2, y_2$ ($1 \le x_1 \le x_2 \le n$, $1 \le y_1 \le y_2 \le m$) — coordinates of the up left and bottom right cells of the query rectangle.

## Output

Print $t$ lines. The $i$-th of them should contain the maximum size of the square consisting of $1$-s and lying fully inside the query rectangle.

**Example**

input

```
3 4
1 1 0 1
0 1 1 0
0 1 1 0
5
1 1 2 3
2 1 3 2
3 2 3 4
1 1 3 4
1 2 3 4
```

output

```
1
1
1
2
2
```

# E. Sonya Partymaker

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Owl Sonya decided to become a partymaker. To train for this role she gather all her owl friends in the country house. There are $m$ chairs located in a circle and consequently numbered with integers from $1$ to $m$. Thus, chairs $i$ and $i + 1$ are neighbouring for all $i$ from $1$ to $m$ - $1$. Chairs $1$ and $m$ are also neighbouring. Some chairs are occupied by her friends. There are $n$ friends in total. No two friends occupy the same chair. Rules are the following:

1. Each participant removes from the game the chair he is currently sitting on.
2. Each of the participants choose a direction that she will follow: clockwise (indices increase, from $m$ goes to $1$) and counter-clockwise (indices decrease, from $1$ goes to $m$). This direction may coincide or be different for any pair of owls.
3. Each turn all guests move one step in the chosen directions. If some guest move to the position with a chair there, he removes this chair from the game.
4. Game ends if there are no more chairs left in the game.

Owls are very busy and want to get rid of the game as soon as possible. They cooperate to pick the direction. Your goal is to find the minimum number o moves required to finish the game.

## Input

The first line of the input contains a single integer $m$ ($1 \le m \le 10^9$) — the length of the circle.

The second line contains a single integer $n$ ($1 \le n \le 100\,000$) — the number of friends.

Last line contains an increasing sequence of $n$ integers $a_i$ ($1 \le a_i \le m$) — initial positions of all owls.

## Output

Print the minimum number of move required to finish the game. Note, that $0$ also may be an answer.

**Examples**

| input |
| --- |
| 6<br>3<br>1 3 5 |
| output |
| 1 |

| input |
| --- |
| 6<br>2<br>1 6 |
| output |
| 2 |

| input |
| --- |
| 406<br>6<br>1 2 3 204 205 206 |
| output |
| 100 |

## Note

In the first sample, it's possible if all owls will move clockwise, i.e. in the direction of increasing indices.

In the sample, first owl has to move clockwise, while the second — counterclockwise.

In the third sample, the first and the fourth owls should move counterclockwise, while the third and the sixth — clockwise. The second and the firth may move in any direction.