A. A+B Again?

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

Given a two-digit positive integer n, find the sum of its digits.

Input

The first line contains an integer t ($1 \le t \le 90$) — the number of test cases.

The only line of each test case contains a single two-digit positive integer n ($10 \le n \le 99$).

Output

For each test case, output a single integer — the sum of the digits of n.

Standard Input	Standard Output
8 77 21 40 34 19 84 10 99	14 3 4 7 10 12 1 18

,

B. Card Game

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   256 megabytes
```

Suneet and Slavic play a card game. The rules of the game are as follows:

- Each card has an integer value between 1 and 10.
- Each player receives 2 cards which are face-down (so a player doesn't know their cards).
- The game is turn-based and consists **exactly of two turns**. In a round, both players pick a **random unflipped** card and flip it. The player who flipped a card with a strictly greater number wins the round. In case of equality, no one wins the round.
- A player wins a game if he wins the most number of rounds (i.e. strictly greater than the other player). In case of equality, no one wins the game.

Since Suneet and Slavic aren't best friends, you need to calculate the number of ways the game could happen that Suneet would end up as the winner.

For a better understanding, please check the notes section.

Input

The first line contains an integer t ($1 \leq t \leq 10^4$) — the number of test cases.

The first and only line of each test case contains 4 integers a_1, a_2, b_1, b_2 ($1 \leq a_1, a_2, b_1, b_2 \leq 10$) where a_1 and a_2 represent the cards Suneet has, and b_1 and b_2 represent the cards Slavic has, respectively.

Output

For each test case, output a single integer — the number of games Suneet would win considering all possible games.

Standard Input	Standard Output
5 3 8 2 6 1 1 1 1 10 10 2 2 1 1 10 10 3 8 7 2	2 0 4 0 2

Note

Consider the first test case when Slavic starts with the cards that have the values 2 and 6, and Suneet starts with cards that have the values 3 and 8. The game could happen in 4 different ways:

- Suneet flips 3 and Slavic flips 2. Suneet wins the first round. Then, Suneet flips 8 and Slavic flips 6. Suneet wins the second round as well. Since Suneet won 2 rounds, he wins the game.
- Suneet flips 3 and Slavic flips 6. Slavic wins the first round. Then, Suneet flips 8 and Slavic flips 2. Suneet wins the second round. Nobody wins since both players won an equal amount of rounds.
- Suneet flips 8 and Slavic flips 6. Suneet wins the first round. Then, Suneet flips 3 and Slavic flips 2. Suneet wins the second round as well. Since Suneet won 2 rounds, he wins the game.

- Suneet flips 8 and Slavic flips 2. Suneet wins the first round. Then, Suneet flips 3 and Slavic flips 6. Slavic wins the round. Nobody wins since both players won an equal amount of rounds.

,

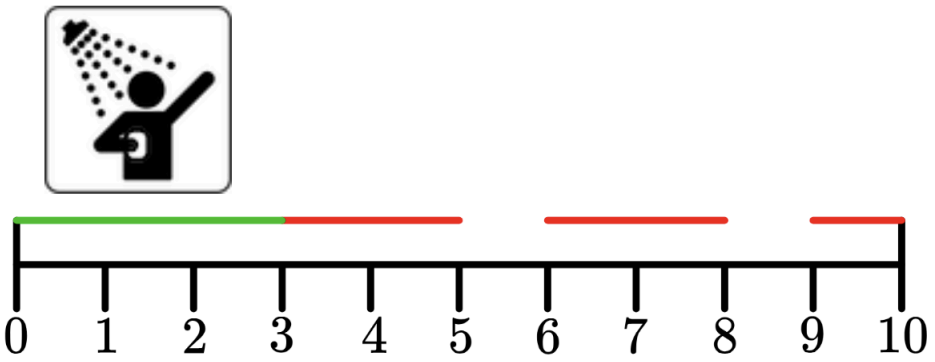
C. Showering

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	256 megabytes

As a computer science student, Alex faces a hard challenge — showering. He tries to shower daily, but despite his best efforts there are always challenges. He takes s minutes to shower and a day only has m minutes!

He already has n tasks planned for the day. Task i is represented as an interval (l_i, r_i), which means that Alex is busy and can not take a shower in that time interval (at any point in time strictly between l_i and r_i). **No two tasks overlap.**

Given all n time intervals, will Alex be able to shower that day? In other words, will Alex have a free time interval of length at least s?



In the first test case, Alex can shower for the first 3 minutes of the day and not miss any of the tasks.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains three integers n, s, and m ($1 \le n \le 2 \cdot 10^5$; $1 \le s, m \le 10^9$) — the number of time intervals Alex already has planned, the amount of time Alex takes to take a shower, and the amount of minutes a day has.

Then n lines follow, the i-th of which contains two integers l_i and r_i ($0 \le l_i < r_i \le m$) — the time interval of the i-th task. No two tasks overlap.

Additional constraint on the input: $l_i > r_{i-1}$ for every $i > 1$.

The sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case output "YES" (without quotes) if Alex can take a shower for that given test case, and "NO" (also without quotes) otherwise.

You can output "YES" and "NO" in any case (for example, strings "yEs", "yes", and "Yes" will be recognized as a positive response).

Standard Input	Standard Output

4 3 3 10 3 5 6 8 9 10 3 3 10 1 2 3 5 6 7 3 3 10 1 2 3 5 6 8 3 4 10 1 2 6 7 8 9	YES YES NO YES

,

D. Slavic's Exam

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

Slavic has a very tough exam and needs your help in order to pass it. Here is the question he is struggling with:

There exists a string s, which consists of lowercase English letters and possibly zero or more "?".

Slavic is asked to change each "?" to a lowercase English letter such that string t becomes a subsequence (not necessarily continuous) of the string s.

Output any such string, or say that it is impossible in case no string that respects the conditions exists.

Input

The first line contains a single integer T ($1 \leq T \leq 10^4$) — the number of test cases.

The first line of each test case contains a single string s ($1 \leq |s| \leq 2 \cdot 10^5$, and s consists only of lowercase English letters and "?"-s) – the original string you have.

The second line of each test case contains a single string t ($1 \leq |t| \leq |s|$, and t consists only of lowercase English letters) – the string that should be a subsequence of string s.

The sum of $|s|$ over all test cases doesn't exceed $2 \cdot 10^5$, where $|x|$ denotes the length of the string x.

Output

For each test case, if no such string exists as described in the statement, output "NO" (without quotes).

Otherwise, output "YES" (without quotes). Then, output one line — the string that respects all conditions.

You can output "YES" and "NO" in any case (for example, strings "yEs", "yes", and "Yes" will be recognized as a positive response).

If multiple answers are possible, you can output any of them.

Standard Input	Standard Output
5 ????? xbx ab??e abcde ayy?x a ab??e dac paiu mom	YES xabax YES abcde YES ayyyx NO NO

,

E. Triple Operations

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 256 megabytes

On the board Ivy wrote down all integers from l to r, inclusive.

In an operation, she does the following:

- pick two numbers x and y on the board, erase them, and in their place write the numbers $3x$ and $\lfloor \frac{y}{3} \rfloor$. (Here $\lfloor \bullet \rfloor$ denotes rounding down to the nearest integer).

What is the minimum number of operations Ivy needs to make all numbers on the board equal 0? We have a proof that this is always possible.

Input

The first line contains an integer t ($1 \le t \le 10^4$) — the number of test cases.

The only line of each test case contains two integers l and r ($1 \le l < r \le 2 \cdot 10^5$).

Output

For each test case, output a single integer — the minimum number of operations needed to make all numbers on the board equal 0.

Standard Input	Standard Output
4 1 3 2 4 199999 200000 19 84	5 6 36 263

Note

In the first test case, we can perform 5 operations as follows:

$$1, 2, 3 \xrightarrow[x=1,\, y=2]{} 3, 0, 3 \xrightarrow[x=0,\, y=3]{} 1, 0, 3 \xrightarrow[x=0,\, y=3]{} 1, 0, 1 \xrightarrow[x=0,\, y=1]{} 0, 0, 1 \xrightarrow[x=0,\, y=1]{} 0, 0, 0.$$

,

F. Expected Median

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 256 megabytes

Arul has a **binary** array* a of length n.

He will take all subsequences† of length k (k is odd) of this array and find their median.‡

What is the sum of all these values?

As this sum can be very large, output it modulo $10^9 + 7$. In other words, print the remainder of this sum when divided by $10^9 + 7$.

*A binary array is an array consisting only of zeros and ones.

†An array b is a subsequence of an array a if b can be obtained from a by the deletion of several (possibly, zero or all) elements. Subsequences **don't** have to be contiguous.

‡The median of an array of odd length k is the $\frac{k+1}{2}$-th element when sorted.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains two integers n and k ($1 \le k \le n \le 2 \cdot 10^5$, k **is odd**) — the length of the array and the length of the subsequence, respectively.

The second line of each test case contains n integers a_i ($0 \le a_i \le 1$) — the elements of the array.

It is guaranteed that sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output

For each test case, print the sum modulo $10^9 + 7$.

Standard Input	Standard Output
8 4 3 1 0 0 1 5 1 1 1 1 1 1 5 5 0 1 0 1 0 6 3 1 0 1 0 1 1 4 3 1 0 1 1 5 3 1 0 1 1 0 2 1 0 0 34 17	2 5 0 16 4 7 0 333606206

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	

Note

In the first test case, there are four subsequences of $[1, 0, 0, 1]$ with length $k = 3$:

- $[1, 0, 0]$: median $= 0$.
- $[1, 0, 1]$: median $= 1$.
- $[1, 0, 1]$: median $= 1$.
- $[0, 0, 1]$: median $= 0$.

The sum of the results is $0 + 1 + 1 + 0 = 2$.
In the second test case, all subsequences of length 1 have median 1, so the answer is 5.

,

G1. Ruler (easy version)

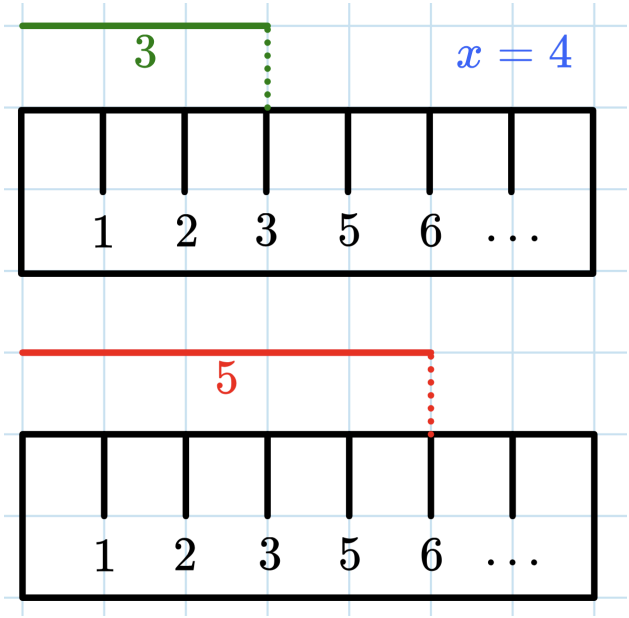
```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   256 megabytes
```

This is the easy version of the problem. The only difference between the two versions is that in this version, you can make at most $\mathbf{10}$ *queries*.

This is an interactive problem. If you are unsure how interactive problems work, then it is recommended to read the guide for participants.

We have a secret ruler that is missing one number x ($2 \le x \le 999$). When you measure an object of length y, the ruler reports the following values:

- If $y < x$, the ruler (correctly) measures the object as having length y.
- If $y \ge x$, the ruler incorrectly measures the object as having length $y + 1$.



The ruler above is missing the number 4, so it correctly measures the first segment as length 3 but incorrectly measures the second segment as length 6 even though it is actually 5.

You need to find the value of x. To do that, you can make queries of the following form:

- ? a b — in response, we will measure the side lengths of an $a \times b$ rectangle with our ruler and multiply the results, reporting the measured area of the rectangle back to you. For example, if $x = 4$ and you query a 3×5 rectangle, we will measure its side lengths as 3×6 and report 18 back to you.

Find the value of x. You can ask at most $\mathbf{10}$ queries.

Input

Each test contains multiple test cases. The first line of input contains a single integer t ($1 \le t \le 1000$) — the number of test cases.

Interaction

There is no initial input for each test case. You should begin the interaction by asking a query.

To make a query, output a single line of the form ? a b ($1 \le a, b \le 1000$). In response, you will be told the measured area of the rectangle, according to our secret ruler.

When you are ready to print the answer, output a single line of the form `!` x ($2 \le x \le 999$). After that, proceed to process the next test case or terminate the program if it was the last test case. Printing the answer does not count as a query.

The interactor is **not** adaptive, meaning that the answer is known before the participant asks the queries and doesn't depend on the queries asked by the participant.

If your program makes more than 10 queries for one set of input data, makes an invalid query, or prints the wrong value of x, then **the response to the query will be** -1. After receiving such a response, your program should immediately terminate to receive the verdict `Wrong Answer`. Otherwise, you can get an arbitrary verdict because your solution will continue to read from a closed stream.

After printing a query do not forget to output the end of line and flush the output. Otherwise, you may get `Idleness limit exceeded` verdict. 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 the documentation for other languages.

Hacks

To make a hack, use the following format.

The first line should contain a single integer t ($1 \le t \le 1000$) — the number of test cases.

The only line of each test case should contain a single integer x ($2 \le x \le 999$) — the missing number on the ruler.

Standard Input	Standard Output
2 18 25 9999	? 3 5 ? 4 4 ! 4 ? 99 100 ! 100

Note

In the first test, the interaction proceeds as follows.

Solution	Jury	Explanation
	2	There are 2 test cases.
? 3 5	18	Secretly, the jury picked $x = 4$. The solution requests the 3×5 rectangle, and the jury responds with $3 \times 6 = 18$, as described in the statement.
? 4 4	25	The solution requests the 4×4 rectangle, which the jury measures as 5×5 and responds with 25.

`! 4`		The solution has somehow determined that $x = 4$, and outputs it. Since the output is correct, the jury continues to the next test case.
`? 99 100`	`1`	Secretly, the jury picked $x = 100$. The solution requests the 99×100 rectangle, which the jury measures as 99×101 and responds with 9999.
`! 100`		The solution has somehow determined that $x = 100$, and outputs it. Since the output is correct and there are no more test cases, the jury and the solution exit.

Note that the line breaks in the example input and output are for the sake of clarity, and do not occur in the real interaction.

,

G2. Ruler (hard version)

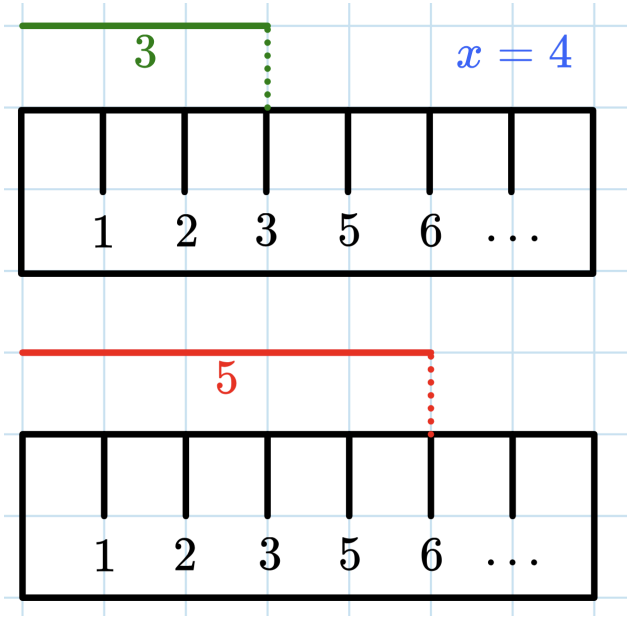
Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

This is the hard version of the problem. The only difference between the two versions is that in this version, you can make at most **7** *queries*.

This is an interactive problem. If you are unsure how interactive problems work, then it is recommended to read the guide for participants.

We have a secret ruler that is missing one number x ($2 \le x \le 999$). When you measure an object of length y, the ruler reports the following values:

- If $y < x$, the ruler (correctly) measures the object as having length y.
- If $y \ge x$, the ruler incorrectly measures the object as having length $y + 1$.



The ruler above is missing the number 4, so it correctly measures the first segment as length 3 but incorrectly measures the second segment as length 6 even though it is actually 5.

You need to find the value of x. To do that, you can make queries of the following form:

- **?** a b — in response, we will measure the side lengths of an $a \times b$ rectangle with our ruler and multiply the results, reporting the measured area of the rectangle back to you. For example, if $x = 4$ and you query a 3×5 rectangle, we will measure its side lengths as 3×6 and report 18 back to you.

Find the value of x. You can ask at most **7** queries.

Input

Each test contains multiple test cases. The first line of input contains a single integer t ($1 \le t \le 1000$) — the number of test cases.

Interaction

There is no initial input for each test case. You should begin the interaction by asking a query.

To make a query, output a single line of the form **?** a b ($1 \le a, b \le 1000$). In response, you will be told the measured area of the rectangle, according to our secret ruler.

When you are ready to print the answer, output a single line of the form `!` x ($2 \le x \le 999$). After that, proceed to process the next test case or terminate the program if it was the last test case. Printing the answer does not count as a query.

The interactor is **not** adaptive, meaning that the answer is known before the participant asks the queries and doesn't depend on the queries asked by the participant.

If your program makes more than 7 queries for one set of input data, makes an invalid query, or prints the wrong value of x, then **the response to the query will be** -1. After receiving such a response, your program should immediately terminate to receive the verdict `Wrong Answer`. Otherwise, you can get an arbitrary verdict because your solution will continue to read from a closed stream.

After printing a query do not forget to output the end of line and flush the output. Otherwise, you may get `Idleness limit exceeded` verdict. 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 the documentation for other languages.

Hacks

To make a hack, use the following format.

The first line should contain a single integer t ($1 \le t \le 1000$) — the number of test cases.

The only line of each test case should contain a single integer x ($2 \le x \le 999$) — the missing number on the ruler.

Standard Input	Standard Output
2 18 25 9999	? 3 5 ? 4 4 ! 4 ? 99 100 ! 100

Note

In the first test, the interaction proceeds as follows.

Solution	Jury	Explanation
	2	There are 2 test cases.
? 3 5	18	Secretly, the jury picked $x = 4$. The solution requests the 3×5 rectangle, and the jury responds with $3 \times 6 = 18$, as described in the statement.
? 4 4	25	The solution requests the 4×4 rectangle, which the jury measures as 5×5 and responds with 25.

`! 4`		The solution has somehow determined that $x = 4$, and outputs it. Since the output is correct, the jury continues to the next test case.
`? 99 100`	`1`	Secretly, the jury picked $x = 100$. The solution requests the 99×100 rectangle, which the jury measures as 99×101 and responds with 9999.
`! 100`		The solution has somehow determined that $x = 100$, and outputs it. Since the output is correct and there are no more test cases, the jury and the solution exit.

Note that the line breaks in the example input and output are for the sake of clarity, and do not occur in the real interaction.