A. Quintomania

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

Boris Notkin composes melodies. He represents them as a sequence of notes, where each note is encoded as an integer from 0 to 127 inclusive. The interval between two notes a and b is equal to $|a - b|$ semitones.

Boris considers a melody *perfect* if the interval between each two adjacent notes is either 5 semitones or 7 semitones.

After composing his latest melodies, he enthusiastically shows you his collection of works. Help Boris Notkin understand whether his melodies are *perfect*.

Input

The first line contains an integer t ($1 \le t \le 1000$) — the number of melodies.

Each melody is described by two lines.

The first line contains an integer n ($2 \le n \le 50$) — the number of notes in the melody.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 127$) — the notes of the melody.

Output

For each melody, output "YES", if it is *perfect*; otherwise, output "NO".

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

Standard Input	Standard Output
8 2 114 109 2 17 10 3 76 83 88 8 38 45 38 80 85 92 99 106 5 63 58 65 58 65 8 117 124 48 53 48 43 54 49 5 95 102 107 114 121 10 72 77 82 75 70 75 68 75 68 75	YES YES YES NO YES NO YES YES

,

B. Startup

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

Arseniy came up with another business plan — to sell soda from a vending machine! For this, he purchased a machine with n shelves, as well as k bottles, where the i-th bottle is characterized by the brand index b_i and the cost c_i.

You can place any number of bottles on each shelf, but all bottles on the same shelf must be of the same brand.

Arseniy knows that all the bottles he puts on the shelves of the machine will be sold. Therefore, he asked you to calculate the **maximum** amount he can earn.

Input

The first line contains one 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 ($1 \le n, k \le 2 \cdot 10^5$), where n is the number of shelves in the machine, and k is the number of bottles available to Arseniy.

The next k lines contain two integers b_i and c_i ($1 \le b_i \le k, 1 \le c_i \le 1000$) — the brand and cost of the i-th bottle.

It is also guaranteed that the sum of n across all test cases does not exceed $2 \cdot 10^5$ and that the sum of k across all test cases also does not exceed $2 \cdot 10^5$.

Output

For each test case, output one integer — the maximum amount that Arseniy can earn.

Standard Input	Standard Output
4 3 3 2 6 2 7 1 15 1 3 2 6 2 7 1 15 6 2 1 7 2 5 190000 1 1 1000	28 15 12 1000

Note

In the first test case, Arseniy has 3 shelves in the vending machine. He can place, for example, two bottles of the brand 2 on the first shelf and a bottle of the brand 1 on the second shelf. Then the total cost of the bottles

would be $6 + 7 + 15 = 28$.

In the second test case, he has only one shelf. It is not difficult to show that the optimal option is to place a bottle of the brand 1 on it. Then the total cost will be 15.

In the third test case, he has as many as 6 shelves, so he can place all available bottles with a total cost of $7 + 5 = 12$.

,

C. Anya and 1100

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

While rummaging through things in a distant drawer, Anya found a beautiful string s consisting only of zeros and ones.

Now she wants to make it even more beautiful by performing q operations on it.

Each operation is described by two integers i ($1 \le i \le |s|$) and v ($v \in \{0, 1\}$) and means that the i-th character of the string is assigned the value v (that is, the assignment $s_i = v$ is performed).

But Anya loves the number 1100, so after each query, she asks you to tell her whether the substring "1100" is present in her string (i.e. there exist such $1 \le i \le |s| - 3$ that $s_i s_{i+1} s_{i+2} s_{i+3} =$ 1100).

Input

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

The first line of the test case contains the string s ($1 \le |s| \le 2 \cdot 10^5$), consisting only of the characters "0" and "1". Here $|s|$ denotes the length of the string s.

The next line contains an integer q ($1 \le q \le 2 \cdot 10^5$) — the number of queries.

The following q lines contain two integers i ($1 \le i \le |s|$) and v ($v \in \{0, 1\}$), describing the query.

It is guaranteed that the sum of $|s|$ across all test cases does not exceed $2 \cdot 10^5$. It is also guaranteed that the sum of q across all test cases does not exceed $2 \cdot 10^5$.

Output

For each query, output "YES", if "1100" is present in Anya's string; otherwise, output "NO".

You can output the answer in any case (upper or lower). For example, the strings "yEs", "yes", "Yes", and "YES" will be recognized as positive responses.

Standard Input	Standard Output
4 100 4 1 1 2 0 2 0 3 1 1100000 3 6 1 7 1 4 1 111010 4	NO NO NO NO YES YES NO NO YES YES YES NO NO

1 1 5 0 4 1 5 0 0100 4 3 1 1 1 2 0 2 1	NO NO

,

D. I Love 1543

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

One morning, Polycarp woke up and realized that 1543 is the most favorite number in his life.

The first thing that Polycarp saw that day as soon as he opened his eyes was a large wall carpet of size n by m cells; n and m are even integers. Each cell contains one of the digits from 0 to 9.

Polycarp became curious about how many times the number 1543 would appear in all layers* of the carpet when traversed **clockwise**.

*The first layer of a carpet of size $n \times m$ is defined as a closed strip of length $2 \cdot (n + m - 2)$ and thickness of 1 element, surrounding its outer part. Each subsequent layer is defined as the first layer of the carpet obtained by removing all previous layers from the original carpet.

Input

The first line of the input contains a single integer t ($1 \le t \le 100$) — the number of test cases. The following lines describe the test cases.

The first line of each test case contains a pair of numbers n and m ($2 \le n, m \le 10^3$, n, m — even integers).

This is followed by n lines of length m, consisting of digits from 0 to 9 — the description of the carpet.

It is guaranteed that the sum of $n \cdot m$ across all test cases does not exceed 10^6.

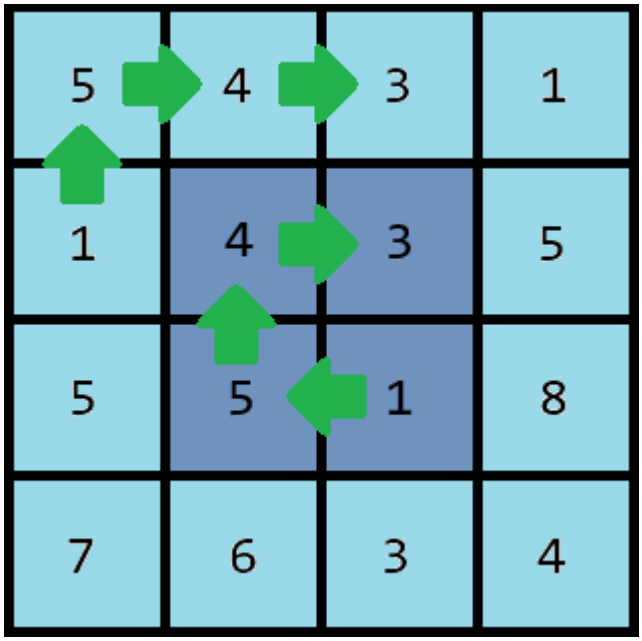
Output

For each test case, output a single number — the total number of times 1543 appears in all layers of the carpet in the order of traversal **clockwise**.

Standard Input	Standard Output
8 2 4 1543 7777 2 4 7154 8903 2 4 3451 8888 2 2 54 13 2 2 51 43 2 6 432015	1 1 0 1 0 2 2 2

512034 4 4 5431 1435 5518 7634 6 4 5432 1152 4542 2432 2302 5942	

Note



Occurrences of 1543 in the seventh example. Different layers are colored in different colors.

,

E. Reverse the Rivers

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

A conspiracy of ancient sages, who decided to redirect rivers for their own convenience, has put the world on the brink. But before implementing their grand plan, they decided to carefully think through their strategy — that's what sages do.

There are n countries, each with exactly k regions. For the j-th region of the i-th country, they calculated the value $a_{i,j}$, which reflects the amount of water in it.

The sages intend to create channels between the j-th region of the i-th country and the j-th region of the $(i+1)$-th country for all $1 \le i \le (n-1)$ and for all $1 \le j \le k$.

Since all n countries are on a large slope, water flows towards the country with the highest number. According to the sages' predictions, after the channel system is created, the new value of the j-th region of the i-th country will be $b_{i,j} = a_{1,j} | a_{2,j} | \ldots | a_{i,j}$, where $|$ denotes the bitwise "OR" operation.

After the redistribution of water, the sages aim to choose the most suitable country for living, so they will send you q queries for consideration.

Each query will contain m requirements.

Each requirement contains three parameters: the region number r, the sign o (either "$<$" or "$>$"), and the value c. If o = "$<$", then in the r-th region of the country you choose, the new value must be strictly less than the limit c, and if o = "$>$", it must be strictly greater.

In other words, the chosen country i must satisfy all m requirements. If in the current requirement o = "$<$", then it must hold that $b_{i,r} < c$, and if o = "$>$", then $b_{i,r} > c$.

In response to each query, you should output a single integer — the number of the suitable country. If there are multiple such countries, output the smallest one. If no such country exists, output -1.

Input

The first line contains three integers n, k, and q ($1 \le n, k, q \le 10^5$) — the number of countries, regions, and queries, respectively.

Next, there are n lines, where the i-th line contains k integers $a_{i,1}, a_{i,2}, \ldots, a_{i,k}$ ($1 \le a_{i,j} \le 10^9$), where $a_{i,j}$ is the value of the j-th region of the i-th country.

Then, q queries are described.

The first line of each query contains a single integer m ($1 \le m \le 10^5$) — the number of requirements.

Then follow m lines, each containing an integer r, a character o, and an integer c ($1 \le r \le k$, $0 \le c \le 2 \cdot 10^9$), where r and c are the region number and the value, and o is either "$<$" or "$>$" — the sign.

It is guaranteed that $n \cdot k$ does not exceed 10^5 and that the sum of m across all queries also does not exceed 10^5.

Output

For each query, output a single integer on a new line — the smallest number of the suitable country, or -1 if no such country exists.

Standard Input	Standard Output
3 4 4 1 3 5 9 4 6 5 3 2 1 2 7 3 1 > 4 2 < 8 1 < 6 2 1 < 8 2 > 8 1 3 > 5 2 4 > 8 1 < 8	2 -1 3 1

Note

In the example, the initial values of the regions are as follows:

1	3	5	9
4	6	5	3
2	1	2	7

After creating the channels, the new values will look like this:

1	3	5	9
$1\|4$	$3\|6$	$5\|5$	$9\|3$
$1\|4\|2$	$3\|6\|1$	$5\|5\|2$	$9\|3\|7$

$\downarrow$

1	3	5	9
5	7	5	11
7	7	7	15

In the first query, it is necessary to output the minimum country number (i.e., row) where, after the redistribution of water in the first region (i.e., column), the new value will be greater than four and less than six, and in the second region it will be less than eight. Only the country with number 2 meets these requirements.

In the second query, there are no countries that meet the specified requirements.

In the third query, only the country with number 3 is suitable.

In the fourth query, all three countries meet the conditions, so the answer is the smallest number 1.

,

F. XORificator 3000

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

Alice has been giving gifts to Bob for many years, and she knows that what he enjoys the most is performing bitwise XOR of *interesting* integers. Bob considers a positive integer x to be *interesting* if it satisfies $x \not\equiv k (\bmod 2^i)$. Therefore, this year for his birthday, she gifted him a super-powerful "XORificator 3000", the latest model.

Bob was very pleased with the gift, as it allowed him to instantly compute the XOR of all *interesting* integers in any range from l to r, inclusive. After all, what else does a person need for happiness? Unfortunately, the device was so powerful that at one point it performed XOR with itself and disappeared. Bob was very upset, and to cheer him up, Alice asked you to write your version of the "XORificator".

Input

The first line of input contains a single integer t ($1 \le t \le 10^4$) — the number of XOR queries on the segment. The following t lines contain the queries, each consisting of the integers l, r, i, k ($1 \le l \le r \le 10^{18}$, $0 \le i \le 30$, $0 \le k < 2^i$).

Output

For each query, output a single integer — the XOR of all integers x in the range $[l, r]$ such that $x \not\equiv k \pmod{2^i}$.

Standard Input	Standard Output
6 1 3 1 0 2 28 3 7 15 43 1 0 57 2007 1 0 1010 1993 2 2 1 1000000000 30 1543	2 2 13 0 4 1000000519

Note

In the first query, the *interesting* integers in the range $[1, 3]$ are 1 and 3, so the answer will be $1 \oplus 3 = 2$.

,

G. Library of Magic

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

This is an interactive problem.

The Department of Supernatural Phenomena at the Oxenfurt Academy has opened the Library of Magic, which contains the works of the greatest sorcerers of Redania — n ($3 \le n \le 10^{18}$) types of books, numbered from 1 to n. Each book's type number is indicated on its spine. Moreover, each type of book is stored in the library in exactly two copies! And you have been appointed as the librarian.

One night, you wake up to a strange noise and see a creature leaving the building through a window. Three thick tomes of different colors were sticking out of the mysterious thief's backpack. Before you start searching for them, you decide to compute the numbers a, b, and c written on the spines of these books. All three numbers are **distinct**.

So, you have an unordered set of tomes, which includes one tome with each of the pairwise distinct numbers a, b, and c, and two tomes for all numbers from 1 to n, except for a, b, and c. You want to find these values a, b, and c.

Since you are not working in a simple library, but in the Library of Magic, you can only use one spell in the form of a query to check the presence of books in their place:

- "`xor l r`" — **Bitwise XOR query** with parameters l and r. Let k be the number of such tomes in the library whose numbers are greater than or equal to l and less than or equal to r. You will receive the result of the computation $v_1 \oplus v_2 \oplus \ldots \oplus v_k$, where $v_1 \ldots v_k$ are the numbers on the spines of these tomes, and $\oplus$ denotes the operation of bitwise exclusive OR.

Since your magical abilities as a librarian are severely limited, you can make no more than 150 queries.

Input

The first line of input contains an integer t ($1 \le t \le 300$) — the number of test cases.

The first line of each test case contains a single integer n ($3 \le n \le 10^{18}$) — the number of types of tomes.

Interaction

The interaction for each test case begins with reading the integer n.

Then you can make up to 150 queries.

To make a query, output a string in the format "`xor l r`" (without quotes) ($1 \le l \le r \le n$). After each query, read an integer — the answer to your query.

To report the answer, output a string in the format "`ans a b c`" (without quotes), where a, b, and c are the numbers you found as the answer to the problem. You can output them in any order.

The interactor is **not** adaptive, which means that the answer is known before the participant makes queries and does not depend on the queries made by the participant.

After making 150 queries, the answer to any other query will be -1. Upon receiving such an answer, terminate the program to receive a verdict of "WA" (`Wrong answer`).

After outputting a query, do not forget to output a newline and flush the output buffer. Otherwise, you will receive a verdict of "`IL`" (`Idleness limit exceeded`). To flush the buffer, use:

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

Hacks

To make a hack, use the following format.

The first line should contain a single integer t ($1 \le t \le 300$) — the number of test cases.

The only line of each test case should contain four integers n, a, b, and c ($3 \le n \le 10^{18}$, $1 \le a, b, c \le n$) — the number of books in the library and the numbers of the stolen tomes. The numbers a, b, and c must be **distinct**.

Standard Input	Standard Output
2 6 0 2 3 5 3	xor 1 1 xor 2 2 xor 3 3 xor 4 6 ans 2 3 5 ans 1 2 3

Note

In the first test case, the books in the library after the theft look like this:

Now consider the answers to the queries:

- For the query "`xor 1 1`", you receive the result $1 \oplus 1 = 0$. Two tomes satisfy the condition specified in the query — both with the number 1.
- For the query "`xor 2 2`", you receive the result 2, as only one tome satisfies the specified condition.
- For the query "`xor 3 3`", you receive the result 3.
- For the query "`xor 4 6`", you receive the result $4 \oplus 6 \oplus 4 \oplus 5 \oplus 6 = 5$.

In the second test case, there are only 3 types of books, and it is easy to guess that the missing ones have the numbers 1, 2, and 3.