A. Cirno's Perfect Bitmasks Classroom

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

Even if it's a really easy question, she won't be able to answer it

— Perfect Memento in Strict Sense

Cirno's perfect bitmasks classroom has just started!

Cirno gave her students a positive integer x. As an assignment, her students need to find the **minimum positive** integer y, which satisfies the following two conditions:

$$x\ \texttt{and}\ y > 0$$

$$x\ \texttt{xor}\ y > 0$$

Where **and** is the bitwise AND operation, and **xor** is the bitwise XOR operation.

Among the students was Mystia, who was truly baffled by all these new operators. Please help her!

Input

The first line of input contains a single integer t ($1 \le t \le 10^3$) — the number of input test cases.

For each test case, the only line of input contains one integer x ($1 \le x \le 2^{30}$).

Output

For each test case, print a single integer — the minimum number of y.

Example

input

```
7
1
2
5
9
16
114514
1000000
```

output

```
3
3
1
1
17
2
64
```

Note

Test case 1:

$1\ \texttt{and}\ 3 = 1 > 0$, $1\ \texttt{xor}\ 3 = 2 > 0$.

Test case 2:

$2\ \texttt{and}\ 3 = 2 > 0$, $2\ \texttt{xor}\ 3 = 1 > 0$.

B. Patchouli's Magical Talisman

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

She is skilled in all kinds of magics, and is keen on inventing new one.

—Perfect Memento in Strict Sense

Patchouli is making a magical talisman. She initially has n magical tokens. Their magical power can be represented with **positive** integers $a_1, a_2, \ldots, a_n$.

Patchouli may perform the following two operations on the tokens.

- **Fusion:** Patchouli chooses two tokens, removes them, and creates a new token with magical power equal to the sum of the two chosen tokens.
- **Reduction:** Patchouli chooses a token with an **even** value of magical power x, removes it and creates a new token with magical power equal to $\frac{x}{2}$.

Tokens are more effective when their magical powers are **odd** values. Please help Patchouli to find the minimum number of operations she needs to make magical powers of all tokens **odd** values.

Input

Each test contains multiple test cases.

The first line contains a single integer t ($1 \le t \le 10^3$) — the number of test cases. The description of the test cases follows.

For each test case, the first line contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the initial number of tokens.

The second line contains n intergers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the initial magical power of the n tokens.

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

Output

For each test case, print a single integer — the minimum number of operations Patchouli needs to make all tokens have an **odd** value of magical power.

It can be shown that under such restrictions the required sequence of operations exists.

Example

input

```
4
2
1 9
3
1 1 2
3
2 4 8
3
1049600 33792 1280
```

output

```
0
1
3
10
```

Note

Test case 1:

a consists solely of odd numbers initially.

Test case 2:

Choose the tokens with magical power of 1 and 2 and perform Fusion. Now $a = [1, 3]$, both are odd numbers.

Test case 3:

Choose the tokens with magical power of 2 and 8 and perform Fusion. Now $a = [4, 10]$.

Choose the token with magical power of 10 and perform Reduction. Now $a = [4, 5]$.

Choose the tokens with magical power of 4 and 5 and perform Fusion. Now $a = [9]$, and 9 is an odd number.

It can be shown that you can not make all the magical powers **odd** numbers in less than 3 moves, so the answer is 3.

C. Manipulating History

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

As a human, she can erase history of its entirety. As a *Bai Ze (Hakutaku)*, she can create history out of nothingness.

—*Perfect Memento in Strict Sense*

Keine has the ability to manipulate history.

The history of Gensokyo is a string s **of length 1 initially**. To fix the chaos caused by Yukari, she needs to do the following operations n times, for the i-th time:

- She chooses a **non-empty substring** t_{2i-1} of s.
- She replaces t_{2i-1} with a **non-empty** string, t_{2i}. Note that the lengths of strings t_{2i-1} and t_{2i} can be different.

Note that if t_{2i-1} occurs more than once in s, **exactly one** of them will be replaced.

For example, let $s =$"marisa", $t_{2i-1} =$"a", and $t_{2i} =$"z". After the operation, s becomes "mzrisa" or "marisz".

After n operations, Keine got the final string and an operation sequence t of length $2n$. Just as Keine thinks she has finished, Yukari appears again and shuffles the order of t. Worse still, Keine forgets the initial history.

Help Keine find the initial history of Gensokyo!

Recall that a substring is a sequence of consecutive characters of the string. For example, for string "abc" its substrings are: "ab", "c", "bc" and some others. But the following strings are not its substring: "ac", "cba", "acb".

Hacks

You cannot make hacks in this problem.

Input

Each test contains multiple test cases. The first line contains a single integer T ($1 \le T \le 10^3$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains a single integer n ($1 \le n < 10^5$) — the number of operations.

The next $2n$ lines contains one **non-empty** string t_i — the i-th string of the shuffled sequence t.

The next line contains one **non-empty** string s — the final string.

It is guaranteed that the total length of given strings (including t_i and s) over all test cases does not exceed $2 \cdot 10^5$. All given strings consist of lowercase English letters only.

It is guaranteed that the initial string exists. It can be shown that the initial string is unique.

Output

For each test case, print the initial string in one line.

Example

input

```
2
2
a
ab
b
cd
acd
3
z
a
a
aa
yakumo
ran
yakumoran
```

output

```
a
z
```

Note

Test case 1:

Initially s is "a".

- In the first operation, Keine chooses "a", and replaces it with "ab". s becomes "ab".
- In the second operation, Keine chooses "b", and replaces it with "cd". s becomes "acd".

So the final string is "acd", and $t =$["a", "ab", "b", "cd"] before being shuffled.

Test case 2:

Initially s is "z".

- In the first operation, Keine chooses "z", and replaces it with "aa". s becomes "aa".
- In the second operation, Keine chooses "a", and replaces it with "ran". s becomes "aran".
- In the third operation, Keine chooses "a", and replaces it with "yakumo". s becomes "yakumoran".

So the final string is "yakumoran", and $t =$["z", "aa", "a", "ran", "a", "yakumo"] before being shuffled.

D. The Enchanted Forest

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The enchanted forest got its name from the magical mushrooms growing here. They may cause illusions and generally should not be approached.

—Perfect Memento in Strict Sense

Marisa comes to pick mushrooms in the Enchanted Forest.

The Enchanted forest can be represented by n points on the X-axis numbered 1 through n. Before Marisa started, her friend, Patchouli, used magic to detect the initial number of mushroom on each point, represented by $a_1, a_2, \ldots, a_n$.

Marisa can start out at **any** point in the forest on minute 0. Each minute, the followings happen in order:

- She moves from point x to y ($|x-y| \le 1$, possibly $y = x$).
- She collects all mushrooms on point y.
- A new mushroom appears on each point in the forest.

Note that she **cannot** collect mushrooms on minute 0.

Now, Marisa wants to know the maximum number of mushrooms she can pick after k minutes.

Input

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

The first line of each test case contains two integers n, k ($1 \le n \le 2 \cdot 10^5$, $1 \le k \le 10^9$) — the number of positions with mushrooms and the time Marisa has, respectively.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the initial number of mushrooms on point $1, 2, \ldots, n$.

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

Output

For each test case, print the maximum number of mushrooms Marisa can pick after k minutes.

Example

input

```
4
5 2
5 6 1 2 3
5 7
5 6 1 2 3
1 2
999999
5 70000
1000000000 1000000000 1000000000 1000000000 1000000000
```

output

```
12
37
1000000
5000349985
```

Note

Test case 1:

Marisa can start at $x = 2$. In the first minute, she moves to $x = 1$ and collect 5 mushrooms. The number of mushrooms will be $[1, 7, 2, 3, 4]$. In the second minute, she moves to $x = 2$ and collects 7 mushrooms. The numbers of mushrooms will be $[2, 1, 3, 4, 5]$. After 2 minutes, Marisa collects 12 mushrooms.

It can be shown that it is impossible to collect more than 12 mushrooms.

Test case 2:

This is one of her possible moving path:

$2 \rightarrow 3 \rightarrow 2 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$

It can be shown that it is impossible to collect more than 37 mushrooms.

E. Railway System

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

As for the technology in the outside world, it is really too advanced for Gensokyo to even look up to.

—Yasaka Kanako, *Symposium of Post-mysticism*

This is an interactive problem.

Under the direct supervision of Kanako and the Moriya Shrine, the railway system of Gensokyo is finally finished. GSKR (Gensokyo Railways) consists of n stations with m bidirectional tracks connecting them. The i-th track has length l_i ($1 \le l_i \le 10^6$). Due to budget limits, the railway system **may not be connected**, though there may be more than one track between two stations.

The *value* of a railway system is defined as the total length of its all tracks. The *maximum (or minimum) capacity* of a railway system is defined as the maximum (or minimum) value among all of the currently functional system's full spanning forest.

In brief, full spanning forest of a graph is a spanning forest with the same connectivity as the given graph.

Kanako has a simulator only able to process no more than $2m$ queries. The input of the simulator is a string s of length m, consisting of characters `0` and/or `1`. The simulator will assume the i-th track functional if $s_i =$ `1`. The device will then tell Kanako the maximum capacity of the system in the simulated state.

Kanako wants to know the the minimum capacity of the system with all tracks functional with the help of the simulator.

The structure of the railway system is fixed in advance. In other words, the interactor is not adaptive.

Input

The first and only line of input contains two integers n, m ($2 \le n \le 200$, $1 \le m \le 500$) — the number of stations and tracks.

Interaction

Begin the interaction by reading n, m.

To make a query, print "`?` s" (without quotes, s is a string of length m, consisting of characters `0` and/or `1`). Then you should read our response from standard input — the maximum capacity of the system in the simulated state.

If your program has made an invalid query or has run out of tries, the interactor will terminate immediately and your program will get a verdict `Wrong answer`.

To give the final answer, print "`!` L" (without the quotes, L is the minimum capacity of the system with all tracks functional). Note that giving this answer is not counted towards the limit of $2m$ queries.

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.

Hacks

The first line of input must contain two integers n, m ($2 \le n \le 200$, $1 \le m \le 500$) — the number of stations and tracks.

The next m lines of input must contain exactly 3 space-separated integers u_i, v_i, l_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $1 \le l_i \le 10^6$) — the endpoints and the length of the i-th track.

Example

input

```
5 4

0

5

9

7
```

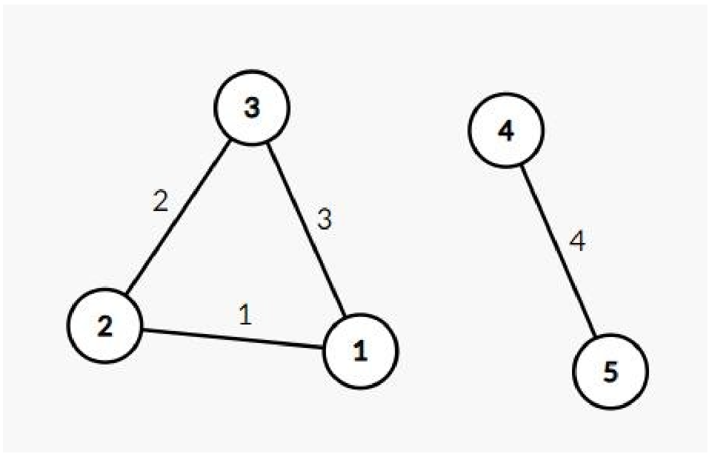
output

```
? 0000

? 1110

? 1111

? 1101
```

```
! 7
```

Note

Here is the graph of the example, satisfying $l_i = i$.



F. Sanae and Giant Robot

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

Is it really?! The robot only existing in my imagination?! The Colossal Walking Robot?!!

— Kochiya Sanae

Sanae made a giant robot — Hisoutensoku, but something is wrong with it. To make matters worse, Sanae can not figure out how to stop it, and she is forced to fix it on-the-fly.
The state of a robot can be represented by an array of integers of length n. Initially, the robot is at state a. She wishes to turn it into state b.

As a great programmer, Sanae knows the art of copy-and-paste. In one operation, she can choose some segment from given segments, copy the segment from b and paste it into **the same place** of the robot, replacing the original state there. However, she has to ensure that the sum of a **does not change** after each copy operation in case the robot go haywire. Formally, Sanae can choose segment $[l, r]$ and assign $a_i = b_i$ ($l \le i \le r$) if $\sum\limits_{i=1}^{n} a_i$ does not change after the operation.

Determine whether it is possible for Sanae to successfully turn the robot from the initial state a to the desired state b with any (possibly, zero) operations.

Input

Each test contains multiple test cases. The first line contains a single integer t ($1 \le t \le 2 \cdot 10^4$) — the number of test cases. The descriptions of the test cases follow.

The first line of each test case contains two integers n, m ($2 \le n \le 2 \cdot 10^5$, $1 \le m \le 2 \cdot 10^5$) — the length of a, b and the number of segments.

The second line contains n intergers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) — the initial state a.

The third line contains n intergers $b_1, b_2, \ldots, b_n$ ($1 \le b_i \le 10^9$) — the desired state b.

Then m lines follow, the i-th line contains two intergers l_i, r_i ($1 \le l_i < r_i \le n$) — the segments that can be copy-pasted by Sanae.

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

Output

For each test case, print "YES" (without quotes) if a can be turned into b, or "NO" (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).

Example

input

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

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

output

```
YES
NO
```

Note

Test case 1:

One possible way of turning a to b:

First, select $[1, 3]$. After the operation, $a = [3, 2, 5, 2, 3]$.

Then, select $[2, 5]$. After the operation, $a = [3, 2, 5, 4, 1] = b$.

Test case 2:

It can be shown that it is impossible to turn a into b.