A. 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:

$2 \to 3 \to 2 \to 1 \to 2 \to 3 \to 4 \to 5$

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

B. 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

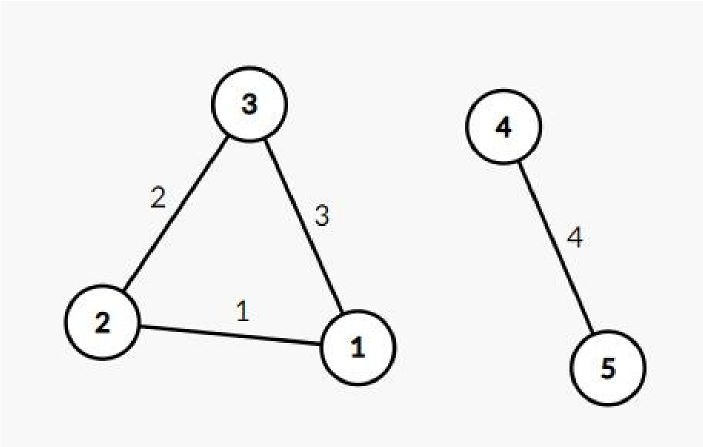
input
5 4
0
5
9
7

output

```
? 0000

? 1110

? 1111

? 1101

! 7
```

Note

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



C. 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.

D. Cute number

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

> Ran is especially skilled in computation and mathematics. It is said that she can do unimaginable calculation work in an instant.
>
> —*Perfect Memento in Strict Sense*

Ran Yakumo is a cute girl who loves creating cute Maths problems.

Let $f(x)$ be the minimal square number **strictly** greater than x, and $g(x)$ be the maximal square number less than or equal to x. For example, $f(1) = f(2) = g(4) = g(8) = 4$.

A positive integer x is *cute* if $x - g(x) < f(x) - x$. For example, $1, 5, 11$ are cute integers, while $3, 8, 15$ are not.

Ran gives you an array a of length n. She wants you to find the smallest non-negative integer k such that $a_i + k$ is a cute number for any element of a.

Input

The first line contains one integer n ($1 \le n \le 10^6$) — the length of a.

The second line contains n intergers $a_1, a_2, \dots, a_n$ ($1 \le a_1 \le a_2 \le \dots \le a_n \le 2 \cdot 10^6$) — the array a.

Output

Print a single interger k — the answer.

Examples

input

```
4
1 3 8 10
```

output

```
1
```

input

```
5
2 3 8 9 11
```

output

```
8
```

input

```
8
```

```
1 2 3 4 5 6 7 8
```

output

```
48
```

Note

Test case 1:

3 is not cute integer, so $k \neq 0$.

$2, 4, 9, 11$ are cute integers, so $k = 1$.

E. Become Big For Me

time limit per test: 3 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Come, let's build a world where even the weak are not forgotten!

—Kijin Seija, *Double Dealing Characters*

Shinmyoumaru has a mallet that can turn objects bigger or smaller. She is testing it out on a sequence a and a number v whose initial value is 1. She wants to make $v = \gcd\limits_{i \neq j}\{a_i \cdot a_j\}$ by **no more than** 10^5 operations ($\gcd\limits_{i \neq j}\{a_i \cdot a_j\}$ denotes the $\gcd$ of all products of two distinct elements of the sequence a).

In each operation, she picks a subsequence b of a, and does one of the followings:

- **Enlarge**: $v = v \cdot \operatorname{lcm}(b)$
- **Reduce**: $v = \frac{v}{\operatorname{lcm}(b)}$

Note that she does **not** need to guarantee that v is an integer, that is, v does **not** need to be a multiple of $\operatorname{lcm}(b)$ when performing Reduce.

Moreover, she wants to guarantee that the total length of b chosen over the operations does not exceed 10^6. Fine a possible operation sequence for her. **You don't need to minimize anything**.

Input

The first line contains a single integer n ($2 \leq n \leq 10^5$) — the size of sequence a.

The second line contains n integers $a_1, a_2, \cdots, a_n$ ($1 \leq a_i \leq 10^6$) — the sequence a.

It can be shown that the answer exists.

Output

The first line contains a non-negative integer k ($0 \leq k \leq 10^5$) — the number of operations.

The following k lines contains several integers. For each line, the first two integers f ($f \in \{0, 1\}$) and p ($1 \leq p \leq n$) stand for the option you choose (0 for Enlarge and 1 for Reduce) and the length of b. The other p integers of the line $i_1, i_2, \ldots, i_p$ ($1 \leq i_1 < i_2 < \ldots < i_p \leq n$) represents the indexes of the subsequence. Formally, $b_j = a_{i_j}$.

Examples

input

```
3
6 10 15
```

output

```
1
0 3 1 2 3
```

input

```
4
2 4 8 16
```

output

```
2
0 1 4
1 1 1
```

Note

Test case 1:

$\gcd\limits_{i \neq j}\{a_i \cdot a_j\} = \gcd\{60, 90, 150\} = 30$.

Perform $v = v \cdot \text{lcm}\{a_1, a_2, a_3\} = 30$.

Test case 2:

$\gcd\limits_{i \neq j}\{a_i \cdot a_j\} = 8$.

Perform $v = v \cdot \text{lcm}\{a_4\} = 16$.

Perform $v = \frac{v}{\text{lcm}\{a_1\}} = 8$.

F. Koishi's Unconscious Permutation

time limit per test: 12 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

As she closed the Satori's eye that could read minds, Koishi gained the ability to live in unconsciousness. Even she herself does not know what she is up to.

— Subterranean Animism

Koishi is unconsciously permuting n numbers: $1, 2, \ldots, n$.

She thinks the permutation p is **beautiful** if $s = \sum\limits_{i=1}^{n-1} [p_i + 1 = p_{i+1}]$. $[x]$ equals to 1 if x holds, or 0 otherwise.

For each $k \in [0, n-1]$, she wants to know the number of beautiful permutations of length n satisfying $k = \sum\limits_{i=1}^{n-1} [p_i < p_{i+1}]$.

Input

There is one line containing two intergers n ($1 \leq n \leq 250\,000$) and s ($0 \leq s < n$).

Output

Print one line with n intergers. The i-th integers represents the answer of $k = i - 1$, modulo 998244353.

Examples

input
```
2 0
```
output
```
1 0
```

input
```
4 1
```
output
```
0 3 6 0
```

input
```
8 3
```
output
```
0 0 0 35 770 980 70 0
```

Note

Let $f(p) = \sum\limits_{i=1}^{n-1} [p_i < p_{i+1}]$.

Testcase 1:

$[2, 1]$ is the only beautiful permutation. And $f([2, 1]) = 0$.

Testcase 2:

Beautiful permutations:

$[1, 2, 4, 3]$, $[1, 3, 4, 2]$, $[1, 4, 2, 3]$, $[2, 1, 3, 4]$, $[2, 3, 1, 4]$, $[3, 1, 2, 4]$, $[3, 4, 2, 1]$, $[4, 2, 3, 1]$, $[4, 3, 1, 2]$. The first six of them satisfy $f(p) = 2$, while others satisfy $f(p) = 1$.