Educational Codeforces Round 23

A. Treasure Hunt

1 second, 256 megabytes

Captain Bill the Hummingbird and his crew recieved an interesting challenge offer. Some stranger gave them a map, potion of teleportation and said that only this potion might help them to reach the treasure.

Bottle with potion has two values x and y written on it. These values define four moves which can be performed using the potion:

- $(a, b) \to (a + x, b + y)$
- $(a, b) \to (a + x, b - y)$
- $(a, b) \to (a - x, b + y)$
- $(a, b) \to (a - x, b - y)$

Map shows that the position of Captain Bill the Hummingbird is (x_1, y_1) and the position of the treasure is (x_2, y_2).

You task is to tell Captain Bill the Hummingbird whether he should accept this challenge or decline. If it is possible for Captain to reach the treasure using the potion then output "YES", otherwise "NO" (without quotes).

The potion can be used infinite amount of times.

Input

The first line contains four integer numbers x_1, y_1, x_2, y_2 ($-10^5 \le x_1, y_1, x_2, y_2 \le 10^5$) — positions of Captain Bill the Hummingbird and treasure respectively.

The second line contains two integer numbers x, y ($1 \le x, y \le 10^5$) — values on the potion bottle.

Output

Print "YES" if it is possible for Captain to reach the treasure using the potion, otherwise print "NO" (without quotes).

input

```
0 0 0 6
2 3
```

output

```
YES
```

input

```
1 1 3 6
1 5
```

output

```
NO
```

In the first example there exists such sequence of moves:

1. $(0, 0) \to (2, 3)$ — the first type of move
2. $(2, 3) \to (0, 6)$ — the third type of move

B. Makes And The Product

2 seconds, 256 megabytes

After returning from the army Makes received a gift — an array a consisting of n positive integer numbers. He hadn't been solving problems for a long time, so he became interested to answer a particular question: how many triples of indices (i, j, k) ($i < j < k$), such that $a_i \cdot a_j \cdot a_k$ is minimum possible, are there in the array? Help him with it!

Input

The first line of input contains a positive integer number n ($3 \le n \le 10^5$) — the number of elements in array a. The second line contains n positive integer numbers a_i ($1 \le a_i \le 10^9$) — the elements of a given array.

Output

Print one number — the quantity of triples (i, j, k) such that i, j and k are pairwise distinct and $a_i \cdot a_j \cdot a_k$ is minimum possible.

input

```
4
1 1 1 1
```

output
4

input
5 1 3 2 3 4
output
2

input
6 1 3 3 1 3 2
output
1

In the first example Makes always chooses three ones out of four, and the number of ways to choose them is 4.

In the second example a triple of numbers $(1, 2, 3)$ is chosen (numbers, not indices). Since there are two ways to choose an element 3, then the answer is 2.

In the third example a triple of numbers $(1, 1, 2)$ is chosen, and there's only one way to choose indices.

C. Really Big Numbers

1 second, 256 megabytes

Ivan likes to learn different things about numbers, but he is especially interested in *really big* numbers. Ivan thinks that a positive integer number x is *really big* if the difference between x and the sum of its digits (in decimal representation) is not less than s. To prove that these numbers may have different special properties, he wants to know how rare (or not rare) they are — in fact, he needs to calculate the quantity of *really big* numbers that are not greater than n.

Ivan tried to do the calculations himself, but soon realized that it's too difficult for him. So he asked you to help him in calculations.

Input

The first (and the only) line contains two integers n and s ($1 \le n, s \le 10^{18}$).

Output

Print one integer — the quantity of *really big* numbers that are not greater than n.

input
12 1
output
3

input
25 20
output
0

input
10 9
output
1

In the first example numbers 10, 11 and 12 are *really big*.

In the second example there are no *really big* numbers that are not greater than 25 (in fact, the first *really big* number is 30: $30 - 3 \ge 20$).

In the third example 10 is the only *really big* number ($10 - 1 \ge 9$).

D. Imbalanced Array

2 seconds, 256 megabytes

You are given an array a consisting of n elements. The *imbalance value* of some subsegment of this array is the difference between the maximum and minimum element from this segment. The *imbalance value* of the array is the sum of *imbalance values* of all subsegments of this array.

For example, the *imbalance value* of array $[1, 4, 1]$ is 9, because there are 6 different subsegments of this array:

- $[1]$ (from index 1 to index 1), *imbalance value* is 0;
- $[1, 4]$ (from index 1 to index 2), *imbalance value* is 3;
- $[1, 4, 1]$ (from index 1 to index 3), *imbalance value* is 3;
- $[4]$ (from index 2 to index 2), *imbalance value* is 0;
- $[4, 1]$ (from index 2 to index 3), *imbalance value* is 3;
- $[1]$ (from index 3 to index 3), *imbalance value* is 0;

You have to determine the *imbalance value* of the array a.

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

The second line contains n integers $a_1, a_2 \dots a_n$ ($1 \le a_i \le 10^6$) — elements of the array.

Output
Print one integer — the *imbalance value* of a.

input
3 1 4 1
output
9

E. Choosing The Commander

2 seconds, 256 megabytes

As you might remember from the previous round, Vova is currently playing a strategic game known as Rage of Empires.

Vova managed to build a large army, but forgot about the main person in the army - the commander. So he tries to hire a commander, and he wants to choose the person who will be respected by warriors.

Each warrior is represented by his personality — an integer number p_i. Each commander has two characteristics — his personality p_j and leadership l_j (both are integer numbers). Warrior i *respects* commander j only if $p_i \oplus p_j < l_j$ ($x \oplus y$ is the bitwise excluding OR of x and y).

Initially Vova's army is empty. There are three different types of events that can happen with the army:

- 1 p_i — one warrior with personality p_i joins Vova's army;
- 2 p_i — one warrior with personality p_i leaves Vova's army;
- 3 p_i l_i — Vova tries to hire a commander with personality p_i and leadership l_i.

For each event of the third type Vova wants to know how many warriors (counting only those who joined the army and haven't left yet) *respect* the commander he tries to hire.

Input
The first line contains one integer q ($1 \le q \le 100000$) — the number of events.

Then q lines follow. Each line describes the event:

- 1 p_i ($1 \le p_i \le 10^8$) — one warrior with personality p_i joins Vova's army;
- 2 p_i ($1 \le p_i \le 10^8$) — one warrior with personality p_i leaves Vova's army (it is guaranteed that there is at least one such warrior in Vova's army by this moment);
- 3 p_i l_i ($1 \le p_i, l_i \le 10^8$) — Vova tries to hire a commander with personality p_i and leadership l_i. There is at least one event of this type.

Output
For each event of the third type print one integer — the number of warriors who *respect* the commander Vova tries to hire in the event.

input

```
5
1 3
1 4
3 6 3
2 4
3 6 3
```

output

```
1
0
```

In the example the army consists of two warriors with personalities 3 and 4 after first two events. Then Vova tries to hire a commander with personality 6 and leadership 3, and only one warrior respects him ($4 \oplus 6 = 2$, and $2 < 3$, but $3 \oplus 6 = 5$, and $5 \geq 3$). Then warrior with personality 4 leaves, and when Vova tries to hire that commander again, there are no warriors who respect him.

F. MEX Queries

2 seconds, 256 megabytes

You are given a set of integer numbers, initially it is empty. You should perform n queries.

There are three different types of queries:

- `1` l r — Add all missing numbers from the interval $[l, r]$
- `2` l r — Remove all present numbers from the interval $[l, r]$
- `3` l r — Invert the interval $[l, r]$ — add all missing and remove all present numbers from the interval $[l, r]$

After each query you should output *MEX* of the set — the smallest positive (*MEX* ≥ 1) integer number which is not presented in the set.

Input

The first line contains one integer number n ($1 \leq n \leq 10^5$).

Next n lines contain three integer numbers t, l, r ($1 \leq t \leq 3, 1 \leq l \leq r \leq 10^{18}$) — type of the query, left and right bounds.

Output

Print *MEX* of the set after each query.

input

```
3
1 3 4
3 1 6
2 1 3
```

output

```
1
3
1
```

input

```
4
1 1 3
3 5 6
2 4 4
3 1 6
```

output

```
4
4
4
1
```

Here are contents of the set after each query in the first example:

1. $\{3, 4\}$ — the interval $[3, 4]$ is added
2. $\{1, 2, 5, 6\}$ — numbers $\{3, 4\}$ from the interval $[1, 6]$ got deleted and all the others are added
3. $\{5, 6\}$ — numbers $\{1, 2\}$ got deleted