Codeforces Round #334 (Div. 1)

A. Alternative Thinking

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

Kevin has just recevied his disappointing results on the USA Identification of Cows Olympiad (USAICO) in the form of a binary string of length n. Each character of Kevin's string represents Kevin's score on one of the n questions of the olympiad—`'1'` for a correctly identified cow and `'0'` otherwise.

However, all is not lost. Kevin is a big proponent of alternative thinking and believes that his score, instead of being the sum of his points, should be the length of the longest alternating subsequence of his string. Here, we define an *alternating subsequence* of a string as a **not-necessarily contiguous** subsequence where no two consecutive elements are equal. For example, $\{0, 1, 0, 1\}$, $\{1, 0, 1\}$, and $\{1, 0, 1, 0\}$ are alternating sequences, while $\{1, 0, 0\}$ and $\{0, 1, 0, 1, 1\}$ are not.

Kevin, being the sneaky little puffball that he is, is willing to hack into the USAICO databases to improve his score. In order to be subtle, he decides that he will flip exactly one substring—that is, take a **contiguous** non-empty substring of his score and change all `'0'`s in that substring to `'1'`s and vice versa. After such an operation, Kevin wants to know the length of the longest possible alternating subsequence that his string could have.

Input

The first line contains the number of questions on the olympiad n ($1 \leq n \leq 100\,000$).

The following line contains a binary string of length n representing Kevin's results on the USAICO.

Output

Output a single integer, the length of the longest possible alternating subsequence that Kevin can create in his string after flipping a single substring.

Sample test(s)

input
8 10000011
output
5

input
2 01
output
2

Note

In the first sample, Kevin can flip the bolded substring '`100`**`00`**`011`' and turn his string into '`10011011`', which has an alternating subsequence of length 5: '**`1`**`0`**`01`**`1`**`01`**`1`'.

In the second sample, Kevin can flip the entire string and still have the same score.

B. Moodular Arithmetic

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

As behooves any intelligent schoolboy, Kevin Sun is studying psycowlogy, cowculus, and cryptcowgraphy at the Bovinia State University (BGU) under Farmer Ivan. During his Mathematics of Olympiads (MoO) class, Kevin was confronted with a weird functional equation and needs your help. For two fixed integers k and p, where p is an odd prime number, the functional equation states that

$$f(kx \mod p) \equiv k \cdot f(x) \mod p$$

for some function $f : \{0, 1, 2, \cdots, p-1\} \to \{0, 1, 2, \cdots, p-1\}$. (This equation should hold for any integer x in the range 0 to p - 1, inclusive.)

It turns out that f can actually be many different functions. Instead of finding a solution, Kevin wants you to count the number of distinct functions f that satisfy this equation. Since the answer may be very large, you should print your result modulo $10^9 + 7$.

Input

The input consists of two space-separated integers p and k ($3 \le p \le 1\,000\,000$, $0 \le k \le p$ - 1) on a single line. It is guaranteed that p is an odd prime number.

Output

Print a single integer, the number of distinct functions f modulo $10^9 + 7$.

Sample test(s)

input
3 2
output
3

input
5 4
output
25

Note

In the first sample, $p = 3$ and $k = 2$. The following functions work:

1. $f(0) = 0, f(1) = 1, f(2) = 2$.
2. $f(0) = 0, f(1) = 2, f(2) = 1$.
3. $f(0) = f(1) = f(2) = 0$.

C. Lieges of Legendre

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

Kevin and Nicky Sun have invented a new game called Lieges of Legendre. In this game, two players take turns modifying the game state with Kevin moving first. Initially, the game is set up so that there are n piles of cows, with the i-th pile containing a_i cows. During each player's turn, that player calls upon the power of Sunlight, and uses it to either:

1. Remove a single cow from a chosen non-empty pile.
2. Choose a pile of cows with even size $2 \cdot x$ ($x > 0$), and replace it with k piles of x cows each.

The player who removes the last cow wins. Given n, k, and a sequence $a_1, a_2, ..., a_n$, help Kevin and Nicky find the winner, given that both sides play in optimal way.

Input

The first line of the input contains two space-separated integers n and k ($1 \leq n \leq 100\,000$, $1 \leq k \leq 10^9$).

The second line contains n integers, $a_1, a_2, \ldots a_n$ ($1 \leq a_i \leq 10^9$) describing the initial state of the game.

Output

Output the name of the winning player, either "`Kevin`" or "`Nicky`" (without quotes).

Sample test(s)

input
2 1 3 4
output
Kevin

input
1 2 3
output
Nicky

Note

In the second sample, Nicky can win in the following way: Kevin moves first and is forced to remove a cow, so the pile contains two cows after his move. Next, Nicky replaces this pile of size 2 with two piles of size 1. So the game state is now two piles of size 1. Kevin then removes one of the remaining cows and Nicky wins by removing the other.

D. Ruminations on Ruminants

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Kevin Sun is ruminating on the origin of cows while standing at the origin of the Cartesian plane. He notices n lines $\ell_1, \ell_2, \cdots \ell_n$ on the plane, each representable by an equation of the form $ax + by = c$. He also observes that no two lines are parallel and that no three lines pass through the same point.

For each triple (i, j, k) such that $1 \le i < j < k \le n$, Kevin considers the triangle formed by the three lines ℓ_i, ℓ_j, ℓ_k. He calls a triangle *original* if the circumcircle of that triangle passes through the origin. Since Kevin believes that the circles of bovine life are tied directly to such triangles, he wants to know the number of original triangles formed by unordered triples of distinct lines.

Recall that the *circumcircle* of a triangle is the circle which passes through all the vertices of that triangle.

Input

The first line of the input contains a single integer n ($3 \le n \le 2000$), the number of lines.

The next n lines describe lines $\ell_1, \ell_2, \cdots \ell_n$. The i-th of these lines contains three space-separated integers a_i, b_i, c_i ($|a_i|, |b_i|, |c_i| \le 10\,000, a_i^2 + b_i^2 > 0$), representing the equation $a_i x + b_i y = c_i$ of line ℓ_i.

Output

Print a single integer, the number of triples (i, j, k) with $i < j < k$ such that lines ℓ_i, ℓ_j, ℓ_k form an original triangle.

Sample test(s)

input

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

output

```
2
```

input

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

output

```
1
```

Note

Note that in the first sample, some of the lines pass through the origin.

In the second sample, there is exactly one triple of lines: $y = 1, x + y = 2, x - y = -2$. The triangle they form has vertices $(0, 2), (1, 1), (-1, 1)$. The circumcircle of this triangle has equation $x^2 + (y - 1)^2 = 1$. This indeed passes through $(0, 0)$.

E. Pastoral Oddities

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

In the land of Bovinia there are n pastures, but no paths connecting the pastures. Of course, this is a terrible situation, so Kevin Sun is planning to rectify it by constructing m undirected paths connecting pairs of distinct pastures. To make transportation more efficient, he also plans to pave some of these new paths.

Kevin is very particular about certain aspects of path-paving. Since he loves odd numbers, he wants each pasture to have an odd number of paved paths connected to it. Thus we call a paving *sunny* if each pasture is incident to an odd number of paved paths. He also enjoys short paths more than long paths, so he would like the longest paved path to be as short as possible. After adding each path, Kevin wants to know if a sunny paving exists for the paths of Bovinia, and if at least one does, the minimum possible length of the longest path in such a paving. Note that "longest path" here means maximum-weight edge.

Input

The first line contains two integers n ($2 \le n \le 100\,000$) and m ($1 \le m \le 300\,000$), denoting the number of pastures and paths, respectively. The next m lines each contain three integers a_i, b_i and l_i, describing the i-th path. The i-th path connects pastures a_i and b_i ($1 \le a_i, b_i \le n$; $a_i \ne b_i$) and has length l_i ($1 \le l_i \le 10^9$). Paths are given in the order in which they are constructed.

Output

Output m lines. The i-th line should contain a single integer denoting the minimum possible length of the longest path (maximum-weight edge) in a sunny paving using only the first i paths. If Kevin cannot pave a set of paths so that each pasture is incident to an odd number of paved paths, output -1.

Note that the paving is only hypothetical—your answer after adding the i-th path should not be affected by any of your previous answers.

Sample test(s)

input

```
4 4
1 3 4
2 4 8
1 2 2
3 4 3
```

output

```
-1
8
8
3
```

input

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

output

```
-1
-1
```

input

```
4 10
2 1 987
3 2 829
4 1 768
4 2 608
3 4 593
3 2 488
4 2 334
2 1 204
1 3 114
1 4 39
```

output

```
-1
-1
829
829
768
768
768
488
334
204
```

Note

For the first sample, these are the paths that Kevin should pave after building the i-th path:

1. No set of paths works.
2. Paths 1 (length 4) and 2 (length 8).
3. Paths 1 (length 4) and 2 (length 8).
4. Paths 3 (length 2) and 4 (length 3).

In the second sample, there never exists a paving that makes Kevin happy.
