Codeforces Round #336 (Div. 1)

A. Chain Reaction

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

There are n beacons located at distinct positions on a number line. The i-th beacon has position a_i and power level b_i. When the i-th beacon is activated, it destroys all beacons to its left (direction of decreasing coordinates) within distance b_i inclusive. The beacon itself is not destroyed however. Saitama will activate the beacons one at a time from right to left. If a beacon is destroyed, it cannot be activated.

Saitama wants Genos to add a beacon **strictly to the right** of all the existing beacons, with any position and any power level, such that the least possible number of beacons are destroyed. Note that Genos's placement of the beacon means it will be the first beacon activated. Help Genos by finding the minimum number of beacons that could be destroyed.

Input

The first line of input contains a single integer n ($1 \le n \le 100\,000$) — the initial number of beacons.

The i-th of next n lines contains two integers a_i and b_i ($0 \le a_i \le 1\,000\,000$, $1 \le b_i \le 1\,000\,000$) — the position and power level of the i-th beacon respectively. No two beacons will have the same position, so $a_i \neq a_j$ if $i \neq j$.

Output

Print a single integer — the minimum number of beacons that could be destroyed if exactly one beacon is added.

Sample test(s)

input
4 1 9 3 1 6 1 7 4
output
1

input
7 1 1 2 1 3 1 4 1 5 1 6 1 7 1
output
3

Note

For the first sample case, the minimum number of beacons destroyed is 1. One way to achieve this is to place a beacon at position 9 with power level 2.

For the second sample case, the minimum number of beacons destroyed is 3. One way to achieve this is to place a beacon at position 1337 with power level 42.

B. Zuma

time limit per test: 2 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Genos recently installed the game Zuma on his phone. In Zuma there exists a line of n gemstones, the i-th of which has color c_i. The goal of the game is to destroy all the gemstones in the line as quickly as possible.

In one second, Genos is able to choose exactly one continuous substring of colored gemstones that is a palindrome and remove it from the line. After the substring is removed, the remaining gemstones shift to form a solid line again. What is the minimum number of seconds needed to destroy the entire line?

Let us remind, that the string (or substring) is called *palindrome*, if it reads same backwards or forward. In our case this means the color of the first gemstone is equal to the color of the last one, the color of the second gemstone is equal to the color of the next to last and so on.

Input

The first line of input contains a single integer n $(1 \le n \le 500)$ — the number of gemstones.

The second line contains n space-separated integers, the i-th of which is c_i $(1 \le c_i \le n)$ — the color of the i-th gemstone in a line.

Output

Print a single integer — the minimum number of seconds needed to destroy the entire line.

Sample test(s)

input
3 1 2 1
output
1

input
3 1 2 3
output
3

input
7 1 4 4 2 3 2 1
output
2

Note

In the first sample, Genos can destroy the entire line in one second.

In the second sample, Genos can only destroy one gemstone at a time, so destroying three gemstones takes three seconds.

In the third sample, to achieve the optimal time of two seconds, destroy palindrome `4 4` first and then destroy palindrome `1 2 3 2 1`.

C. Marbles

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

In the spirit of the holidays, Saitama has given Genos two grid paths of length n (a weird gift even by Saitama's standards). A grid path is an ordered sequence of neighbouring squares in an infinite grid. Two squares are neighbouring if they share a side.

One example of a grid path is $(0, 0) \rightarrow (0, 1) \rightarrow (0, 2) \rightarrow (1, 2) \rightarrow (1, 1) \rightarrow (0, 1) \rightarrow (- 1, 1)$. Note that squares in this sequence might be repeated, i.e. path has self intersections.

Movement within a grid path is restricted to adjacent squares within the sequence. That is, from the i-th square, one can **only move** to the $(i - 1)$-th or $(i + 1)$-th squares of this path. Note that there is only a single valid move from the first and last squares of a grid path. Also note, that even if there is some j-th square of the path that coincides with the i-th square, only moves to $(i - 1)$-th and $(i + 1)$-th squares are available. For example, from the second square in the above sequence, one can only move to either the first or third squares.

To ensure that movement is not ambiguous, the two grid paths will not have an alternating sequence of three squares. For example, a contiguous subsequence $(0, 0) \rightarrow (0, 1) \rightarrow (0, 0)$ **cannot occur** in a valid grid path.

One marble is placed on the first square of each grid path. Genos wants to get both marbles to the last square of each grid path. However, there is a catch. Whenever he moves one marble, the other marble will copy its movement if possible. For instance, if one marble moves east, then the other marble will *try* and move east as well. By *try*, we mean if moving east is a valid move, then the marble will move east.

Moving north increases the second coordinate by 1, while moving south decreases it by 1. Similarly, moving east increases first coordinate by 1, while moving west decreases it.

Given these two valid grid paths, Genos wants to know if it is possible to move both marbles to the ends of their respective paths. That is, if it is possible to move the marbles such that both marbles rest on the last square of their respective paths.

Input

The first line of the input contains a single integer n $(2 \leq n \leq 1\,000\,000)$ — the length of the paths.

The second line of the input contains a string consisting of $n - 1$ characters (each of which is either 'N', 'E', 'S', or 'W') — the first grid path. The characters can be thought of as the sequence of moves needed to traverse the grid path. For example, the example path in the problem statement can be expressed by the string "NNESWW".

The third line of the input contains a string of $n - 1$ characters (each of which is either 'N', 'E', 'S', or 'W') — the second grid path.

Output

Print "YES" (without quotes) if it is possible for both marbles to be at the end position at the same time. Print "NO" (without quotes) otherwise. In both cases, the answer is case-insensitive.

Sample test(s)

input
7 NNESWW SWSWSW
output
YES

input
3 NN SS
output
NO

Note

In the first sample, the first grid path is the one described in the statement. Moreover, the following sequence of moves will get both marbles to the end: NNESWWSWSW.

In the second sample, no sequence of moves can get both marbles to the end.

D. Power Tree

time limit per test: 3.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Genos and Saitama went shopping for Christmas trees. However, a different type of tree caught their attention, the exalted Power Tree.

A Power Tree starts out as a single root vertex indexed 1. A Power Tree grows through a magical phenomenon known as an update. In an *update*, a single vertex is added to the tree as a child of some other vertex.

Every vertex in the tree (the root and all the added vertices) has some value v_i associated with it. The *power* of a vertex is defined as the strength of the multiset composed of the value associated with this vertex (v_i) and the *powers* of its direct children. The *strength* of a multiset is defined as the sum of all elements in the **multiset** multiplied by the number of elements in it. Or in other words for some **multiset** S:

$$Strength(S) = |S| \cdot \sum_{d \in S} d$$

Saitama knows the *updates* that will be performed on the tree, so he decided to test Genos by asking him queries about the tree during its growth cycle.

An update is of the form $1\ p\ v$, and adds a new vertex with value v as a child of vertex p.

A query is of the form $2\ u$, and asks for the power of vertex u.

Please help Genos respond to these queries modulo $10^9 + 7$.

Input

The first line of the input contains two space separated integers v_1 and q ($1 \le v_1 < 10^9$, $1 \le q \le 200\,000$) — the value of vertex 1 and the total number of updates and queries respectively.

The next q lines contain the updates and queries. Each of them has one of the following forms:

- $1\ p_i\ v_i$, if these line describes an update. The index of the added vertex is equal to the smallest positive integer not yet used as an index in the tree. It is guaranteed that p_i is some already existing vertex and $1 \le v_i < 10^9$.
- $2\ u_i$, if these line describes a query. It is guaranteed u_i will exist in the tree.

It is guaranteed that the input will contain at least one query.

Output

For each query, print out the power of the given vertex modulo $10^9 + 7$.

Sample test(s)

input

```
2 5
1 1 3
1 2 5
1 3 7
1 4 11
2 1
```

output

```
344
```

input

```
5 5
1 1 4
1 2 3
2 2
1 2 7
2 1
```

output

```
14
94
```

Note

For the first sample case, after all the updates the graph will have vertices labelled in the following manner: 1 — 2 — 3 — 4 — 5

These vertices will have corresponding values: 2 — 3 — 5 — 7 — 11

And corresponding powers: 344 — 170 — 82 — 36 — 11

E. Cross Sum

time limit per test: 7 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Genos has been given n distinct lines on the Cartesian plane. Let $\mathcal{I}$ be a list of intersection points of these lines. A single point might appear multiple times in this list if it is the intersection of multiple pairs of lines. The order of the list does not matter.

Given a query point (p, q), let $\mathcal{D}$ be the corresponding list of distances of all points in $\mathcal{I}$ to the query point. Distance here refers to euclidean distance. As a refresher, the euclidean distance between two points (x_1, y_1) and (x_2, y_2) is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.

Genos is given a point (p, q) and a positive integer m. He is asked to find the sum of the m smallest elements in $\mathcal{D}$. Duplicate elements in $\mathcal{D}$ are treated as separate elements. Genos is intimidated by Div1 E problems so he asked for your help.

Input

The first line of the input contains a single integer n $(2 \le n \le 50\,000)$ — the number of lines.

The second line contains three integers x, y and m $(|x|, |y| \le 1\,000\,000, 1 \le m \le min(3 \cdot 10^7, |\mathcal{D}|))$ — the encoded coordinates of the query point and the integer m from the statement above. The query point (p, q) is obtained as $(\frac{x}{1000}, \frac{y}{1000})$. In other words, divide x and y by 1000 to get the actual query point. $|\mathcal{D}|$ denotes the length of the list $\mathcal{D}$ and it is guaranteed that $|\mathcal{D}| > 0$.

Each of the next n lines contains two integers a_i and b_i $(|a_i|, |b_i| \le 1\,000\,000)$ — the parameters for a line of the form: $y = \frac{a_i}{1000}x + \frac{b_i}{1000}$. It is guaranteed that no two lines are the same, that is $(a_i, b_i) \ne (a_j, b_j)$ if $i \ne j$.

Output

Print a single real number, the sum of m smallest elements of $\mathcal{D}$. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

To clarify, let's assume that your answer is a and the answer of the jury is b. The checker program will consider your answer correct if $\frac{|a-b|}{max(1,b)} \le 10^{-6}$.

Sample test(s)

input

```
4
1000 1000 3
1000 0
-1000 0
0 5000
0 -5000
```

output

```
14.282170363
```

input

```
2
-1000000 -1000000 1
1000000 -1000000
999999 1000000
```

output

```
2000001000.999999500
```

input

```
3
-1000 1000 3
1000 0
-1000 2000
2000 -1000
```

output

```
6.000000000
```

input

```
5
-303667 189976 10
-638 116487
-581 44337
1231 -756844
1427 -44097
8271 -838417
```

output

```
12953.274911829
```

Note

In the first sample, the three closest points have distances $\sqrt{2}, 4\sqrt{2},$ and $2\sqrt{13}$.

In the second sample, the two lines $y = 1000x$ - 1000 and $y = \frac{999999}{1000}x + 1000$ intersect at $(2000000, 1999999000)$. This point has a distance of $1000\sqrt{4000004004001}$ from $(-1000, -1000)$.

In the third sample, the three lines all intersect at the point $(1, 1)$. This intersection point is present three times in $\mathcal{I}$ since it is the intersection of three pairs of lines. Since the distance between the intersection point and the query point is 2, the answer is three times that or 6.
