A. 2Char

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

Andrew often reads articles in his favorite magazine 2Char. The main feature of these articles is that each of them uses at most two distinct letters. Andrew decided to send an article to the magazine, but as he hasn't written any article, he just decided to take a random one from magazine 26Char. However, before sending it to the magazine 2Char, he needs to adapt the text to the format of the journal. To do so, he removes some words from the chosen article, in such a way that the remaining text can be written using no more than two distinct letters.

Since the payment depends from the number of non-space characters in the article, Andrew wants to keep the words with the maximum total length.

Input

The first line of the input contains number n ($1 \le n \le 100$) — the number of words in the article chosen by Andrew. Following are n lines, each of them contains one word. All the words consist only of small English letters and their total length doesn't exceed 1000. The words are not guaranteed to be distinct, in this case you are allowed to use a word in the article as many times as it appears in the input.

Output

Print a single integer — the maximum possible total length of words in Andrew's article.

Examples

input

```
4
abb
cacc
aaa
bbb
```

output

```
9
```

input

```
5
a
a
bcbcb
cdecdecdecdecdecde
aaaa
```

output

```
6
```

Note

In the first sample the optimal way to choose words is {'abb', 'aaa', 'bbb'}.

In the second sample the word 'cdecdecdecdecdecde' consists of three distinct letters, and thus cannot be used in the article. The optimal answer is {'a', 'a', 'aaaa'}.

B. Anton and Lines

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

The teacher gave Anton a large geometry homework, but he didn't do it (as usual) as he participated in a regular round on Codeforces. In the task he was given a set of n lines defined by the equations $y = k_i \cdot x + b_i$. It was necessary to determine whether there is at least one point of intersection of two of these lines, that lays strictly inside the strip between $x_1 < x_2$. In other words, is it true that there are $1 \le i < j \le n$ and x', y', such that:

- $y' = k_i * x' + b_i$, that is, point (x', y') belongs to the line number i;
- $y' = k_j * x' + b_j$, that is, point (x', y') belongs to the line number j;
- $x_1 < x' < x_2$, that is, point (x', y') lies inside the strip bounded by $x_1 < x_2$.

You can't leave Anton in trouble, can you? Write a program that solves the given task.

Input

The first line of the input contains an integer n ($2 \le n \le 100\,000$) — the number of lines in the task given to Anton. The second line contains integers x_1 and x_2 ($-1\,000\,000 \le x_1 < x_2 \le 1\,000\,000$) defining the strip inside which you need to find a point of intersection of at least two lines.

The following n lines contain integers k_i, b_i ($-1\,000\,000 \le k_i, b_i \le 1\,000\,000$) — the descriptions of the lines. It is guaranteed that all lines are pairwise distinct, that is, for any two $i \ne j$ it is true that either $k_i \ne k_j$, or $b_i \ne b_j$.

Output

Print "Yes" (without quotes), if there is at least one intersection of two distinct lines, located strictly inside the strip. Otherwise print "No" (without quotes).

Examples

input

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

output

```
NO
```

input

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

output

```
YES
```

input

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

output

```
YES
```

input

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

output

```
NO
```

Note

In the first sample there are intersections located on the border of the strip, but there are no intersections located strictly inside it.

C. Beautiful Function

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

Every day Ruslan tried to count sheep to fall asleep, but this didn't help. Now he has found a more interesting thing to do. First, he thinks of some set of circles on a plane, and then tries to choose a beautiful set of points, such that there is at least one point from the set inside or on the border of each of the imagined circles.

Yesterday Ruslan tried to solve this problem for the case when the set of points is considered beautiful if it is given as $(x_t = f(t), y_t = g(t))$, where argument t takes all integer values from 0 to 50. Moreover, $f(t)$ and $g(t)$ should be *correct* functions.

Assume that $w(t)$ and $h(t)$ are some correct functions, and c is an integer ranging from 0 to 50. The function $s(t)$ is correct if it's obtained by one of the following rules:

1. $s(t) = abs(w(t))$, where $abs(x)$ means taking the absolute value of a number x, i.e. $|x|$;
2. $s(t) = (w(t) + h(t))$;
3. $s(t) = (w(t) - h(t))$;
4. $s(t) = (w(t) * h(t))$, where $*$ means multiplication, i.e. $(w(t) \cdot h(t))$;
5. $s(t) = c$;
6. $s(t) = t$;

Yesterday Ruslan thought on and on, but he could not cope with the task. Now he asks you to write a program that computes the appropriate $f(t)$ and $g(t)$ for any set of at most 50 circles.

In each of the functions $f(t)$ and $g(t)$ you are allowed to use no more than 50 multiplications. The length of any function should not exceed $100 \cdot n$ characters. The function **should not contain spaces.**

Ruslan can't keep big numbers in his memory, so you should choose $f(t)$ and $g(t)$, such that for all integer t from 0 to 50 value of $f(t)$ and $g(t)$ and all the intermediate calculations won't exceed 10^9 by their absolute value.

Input

The first line of the input contains number n ($1 \le n \le 50$) — the number of circles Ruslan thinks of. Next follow n lines, each of them containing three integers x_i, y_i and r_i ($0 \le x_i, y_i \le 50$, $2 \le r_i \le 50$) — the coordinates of the center and the raduis of the i-th circle.

Output

In the first line print a correct function $f(t)$. In the second line print a correct function $g(t)$. The set of the points $(x_t = f(t), y_t = g(t))$ ($0 \le t \le 50$) must satisfy the condition, that there is at least one point inside or on the border of each of the circles, Ruslan thinks of at the beginning.

Examples

input

```
3
0 10 4
10 0 4
20 10 4
```

output

```
t
abs((t-10))
```

Note

Correct functions:

1. 10
2. $(1+2)$
3. $((t-3)+(t*4))$
4. $abs((t-10))$
5. $(abs((((23-t)*(t*t))+((45+12)*(t*t))))*((5*t)+((12*t)-13)))$
6. $abs((t-(abs((t*31))+14))))$

Incorrect functions:

1. $3+5+7$ (not enough brackets, it should be $((3+5)+7)$ or $(3+(5+7)))$
2. $abs(t-3)$ (not enough brackets, it should be $abs((t-3))$
3. $2+(2-3$ (one bracket too many)

4. $1(t+5)$ (no arithmetic operation between 1 and the bracket)
5. $5000*5000$ (the number exceeds the maximum)

The picture shows one of the possible solutions

D. Happy Tree Party

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bogdan has a birthday today and mom gave him a tree consisting of n vertecies. For every edge of the tree i, some number x_i was written on it. In case you forget, a tree is a connected non-directed graph without cycles. After the present was granted, m guests consecutively come to Bogdan's party. When the i-th guest comes, he performs exactly one of the two possible operations:

1. Chooses some number y_i, and two vertecies a_i and b_i. After that, he moves along the edges of the tree from vertex a_i to vertex b_i using the shortest path (of course, such a path is unique in the tree). Every time he moves along some edge j, he replaces his current number y_i by $y_i = \lfloor \frac{y_i}{x_j} \rfloor$, that is, by the result of integer division y_i div x_j.
2. Chooses some edge p_i and replaces the value written in it x_{p_i} by some positive integer $c_i < x_{p_i}$.

As Bogdan cares about his guests, he decided to ease the process. Write a program that performs all the operations requested by guests and outputs the resulting value y_i for each i of the first type.

Input

The first line of the input contains integers, n and m ($2 \le n \le 200\,000$, $1 \le m \le 200\,000$) — the number of vertecies in the tree granted to Bogdan by his mom and the number of guests that came to the party respectively.

Next $n - 1$ lines contain the description of the edges. The i-th of these lines contains three integers u_i, v_i and x_i ($1 \le u_i, v_i \le n$, $u_i \ne v_i$, $1 \le x_i \le 10^{18}$), denoting an edge that connects vertecies u_i and v_i, with the number x_i initially written on it.

The following m lines describe operations, requested by Bogdan's guests. Each description contains three or four integers and has one of the two possible forms:

- $1\ a_i\ b_i\ y_i$ corresponds to a guest, who chooses the operation of the first type.
- $2\ p_i\ c_i$ corresponds to a guests, who chooses the operation of the second type.

It is guaranteed that all the queries are correct, namely $1 \le a_i, b_i \le n$, $1 \le p_i \le n - 1$, $1 \le y_i \le 10^{18}$ and $1 \le c_i < x_{p_i}$, where x_{p_i} represents a number written on edge p_i at this particular moment of time that is not necessarily equal to the initial value x_{p_i}, as some decreases may have already been applied to it. The edges are numbered from 1 to $n - 1$ in the order they appear in the input.

Output

For each guest who chooses the operation of the first type, print the result of processing the value y_i through the path from a_i to b_i.

Examples

input

```
6 6
1 2 1
1 3 7
1 4 4
2 5 5
2 6 2
1 4 6 17
2 3 2
1 4 6 17
1 5 5 20
2 4 1
1 5 1 3
```

output

```
2
4
20
3
```

input

```
5 4
1 2 7
1 3 3
3 4 2
3 5 5
1 4 2 100
1 5 4 1
2 2 2
1 1 3 4
```

output

```
2
0
2
```

Note

Initially the tree looks like this:

The response to the first query is: $\lfloor \frac{\lfloor \frac{17}{4} \rfloor}{2} \rfloor = 2$

After the third edge is changed, the tree looks like this:

The response to the second query is: $\lfloor \frac{\lfloor \frac{17}{2} \rfloor}{2} \rfloor = 4$

In the third query the initial and final vertex coincide, that is, the answer will be the initial number 20.

After the change in the fourth edge the tree looks like this:

In the last query the answer will be: $\lfloor \frac{\lfloor \frac{3}{1} \rfloor}{1} \rfloor = 3$

E. Strange Calculation and Cats

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

Gosha's universe is a table consisting of n rows and m columns. Both the rows and the columns are numbered with consecutive integers starting with 1. We will use (r, c) to denote a cell located in the row r and column c.

Gosha is often invited somewhere. Every time he gets an invitation, he first calculates the number of ways to get to this place, and only then he goes. Gosha's house is located in the cell $(1, 1)$.

At any moment of time, Gosha moves from the cell he is currently located in to a cell adjacent to it (two cells are adjacent if they share a common side). Of course, the movement is possible only if such a cell exists, i.e. Gosha will not go beyond the boundaries of the table. Thus, from the cell (r, c) he is able to make a move to one of the cells $(r - 1, c)$, $(r, c - 1)$, $(r + 1, c)$, $(r, c + 1)$. Also, Ghosha can skip a move and stay in the current cell (r, c).

Besides the love of strange calculations, Gosha is allergic to cats, so he never goes to the cell that has a cat in it. Gosha knows exactly where and when he will be invited and the schedule of cats travelling along the table. Formally, he has q records, the i-th of them has one of the following forms:

- 1, x_i, y_i, t_i — Gosha is invited to come to cell (x_i, y_i) at the moment of time t_i. It is guaranteed that there is no cat inside cell (x_i, y_i) at this moment of time.
- 2, x_i, y_i, t_i — at the moment t_i a cat appears in cell (x_i, y_i). It is guaranteed that no other cat is located in this cell (x_i, y_i) at that moment of time.
- 3, x_i, y_i, t_i — at the moment t_i a cat leaves cell (x_i, y_i). It is guaranteed that there is cat located in the cell (x_i, y_i).

Gosha plans to accept only one invitation, but he has not yet decided, which particular one. In order to make this decision, he asks you to calculate for each of the invitations i the number of ways to get to the cell (x_i, y_i) at the moment t_i. For every invitation, assume that Gosha he starts moving from cell $(1, 1)$ at the moment 1.

Moving between two neighboring cells takes Gosha exactly one unit of tim. In particular, this means that Gosha can come into the cell only if a cat sitting in it leaves the moment when Gosha begins his movement from the neighboring cell, and if none of the cats comes to the cell at the time when Gosha is in it.

Two ways to go from cell $(1, 1)$ to cell (x, y) at time t are considered distinct if for at least one moment of time from 1 to t Gosha's positions are distinct for the two ways at this moment. Note, that during this travel Gosha is allowed to visit both $(1, 1)$ and (x, y) multiple times. Since the number of ways can be quite large, print it modulo $10^9 + 7$.

Input

The first line of the input contains three positive integers n, m and q ($1 \le n \cdot m \le 20$, $1 \le q \le 10\,000$) — the number of rows and columns in the table and the number of events respectively.

Next q lines describe the events, each description contains four integers tp_i, x_i, y_i and t_i ($1 \le tp \le 3$, $1 \le x \le n$, $1 \le y \le m$, $2 \le t \le 10^9$) — the type of the event (1 if Gosha gets an invitation, 2 if a cat comes to the cell and 3 if a cat leaves the cell), the coordinates of the cell where the action takes place and the moment of time at which the action takes place respectively.

It is guaranteed that the queries are given in the chronological order, i.e. $t_i < t_{i+1}$.

Output

For each invitation i (that is, $tp_i = 1$) calculate the number of ways to get to cell (x_i, y_i) at the moment of time t_i. Respond to the invitations chronologically, that is, in the order they appear in the input.

Examples

input

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

output

```
5
```

input

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

output

```
2
42
```

input

```
4 5 5
2 2 5 3
2 2 4 6
3 2 4 9
1 4 4 13
1 4 4 15
```

output

```
490902
10598759
```

Note

Explanation of the first sample. Each picture specifies the number of ways to arrive at the cell at the appropriate time. (X stands for a cell blocked at this particular moment of time)

1	0	0

Time moment 1.

1	1	0

Time moment 2.

2	X	1

Time moment 3.

2	X	1

Time moment 4.

2	0	1

Time moment 5.

2	3	1

Time moment 6.

5	6	4

Time moment 7.