Educational Codeforces Round 125 (Rated for Div. 2)

A. Integer Moves

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

There's a chip in the point $(0, 0)$ of the coordinate plane. In one operation, you can move the chip from some point (x_1, y_1) to some point (x_2, y_2) if the Euclidean distance between these two points is an **integer** (i.e. $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ is integer).

Your task is to determine the minimum number of operations required to move the chip from the point $(0, 0)$ to the point (x, y).

Input

The first line contains a single integer t ($1 \le t \le 3000$) — number of test cases.

The single line of each test case contains two integers x and y ($0 \le x, y \le 50$) — the coordinates of the destination point.

Output

For each test case, print one integer — the minimum number of operations required to move the chip from the point $(0, 0)$ to the point (x, y).

Example

input

```
3
8 6
0 0
9 15
```

output

```
1
0
2
```

Note

In the first example, one operation $(0, 0) \to (8, 6)$ is enough. $\sqrt{(0-8)^2 + (0-6)^2} = \sqrt{64 + 36} = \sqrt{100} = 10$ is an integer.

In the second example, the chip is already at the destination point.

In the third example, the chip can be moved as follows: $(0, 0) \to (5, 12) \to (9, 15)$. $\sqrt{(0-5)^2 + (0-12)^2} = \sqrt{25 + 144} = \sqrt{169} = 13$ and $\sqrt{(5-9)^2 + (12-15)^2} = \sqrt{16 + 9} = \sqrt{25} = 5$ are integers.

B. XY Sequence

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

You are given four integers n, B, x and y. You should build a sequence $a_0, a_1, a_2, \dots, a_n$ where $a_0 = 0$ and for each $i \ge 1$ you can choose:

- either $a_i = a_{i-1} + x$
- or $a_i = a_{i-1} - y$.

Your goal is to build such a sequence a that $a_i \le B$ for all i and $\sum\limits_{i=0}^{n} a_i$ is maximum possible.

Input

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases. Next t cases follow.

The first and only line of each test case contains four integers n, B, x and y ($1 \le n \le 2 \cdot 10^5$; $1 \le B, x, y \le 10^9$).

It's guaranteed that the total sum of n doesn't exceed $2 \cdot 10^5$.

Output

For each test case, print one integer — the maximum possible $\sum\limits_{i=0}^{n} a_i$.

Example

input

```
3
5 100 1 30
7 1000000000 1000000000 1000000000
4 1 7 3
```

output

```
15
4000000000
-10
```

Note

In the first test case, the optimal sequence a is $[0, 1, 2, 3, 4, 5]$.

In the second test case, the optimal sequence a is $[0, 10^9, 0, 10^9, 0, 10^9, 0, 10^9]$.

In the third test case, the optimal sequence a is $[0, -3, -6, 1, -2]$.

C. Bracket Sequence Deletion

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

You are given a bracket sequence consisting of n characters '(' and/or)'. You perform several operations with it.

During one operation, you choose the **shortest** prefix of this string (some amount of first characters of the string) that is **good** and remove it from the string.

The prefix is considered **good** if one of the following two conditions is satisfied:

- this prefix is a regular bracket sequence;
- this prefix is a palindrome of length **at least two**.

A bracket sequence is called regular if it is possible to obtain a correct arithmetic expression by inserting characters '+' and '1' into this sequence. For example, sequences (())(), () and (()(())) are regular, while)(, (() and (()))(are not.

The bracket sequence is called palindrome if it reads the same back and forth. For example, the bracket sequences)), ((and)(() are palindromes, while bracket sequences (),)(and))(are not palindromes.

You stop performing the operations when it's not possible to find a **good** prefix. Your task is to find the number of operations you will perform on the given string and the number of remaining characters in the string.

You have to answer t independent test cases.

Input

The first line of the input contains one integer t ($1 \le t \le 10^4$) — the number of test cases. The next $2t$ lines describe test cases.

The first line of the test case contains one integer n ($1 \le n \le 5 \cdot 10^5$) — the length of the bracket sequence.

The second line of the test case contains n characters '(' and/or ')' — the bracket sequence itself.

It is guaranteed that the sum of n over all test cases do not exceed $5 \cdot 10^5$ ($\sum n \le 5 \cdot 10^5$).

Output

For each test case, print two integers c and r — the number of operations you will perform on the given bracket sequence and the number of characters that remain in the string after performing all operations.

Example

input

```
5
2
()
3
())
4
((((
5
)((()
6
)((()(
```

output

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

D. For Gamers. By Gamers.

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

Monocarp is playing a strategy game. In the game, he recruits a squad to fight monsters. Before each battle, Monocarp has C coins to spend on his squad.

Before each battle starts, his squad is empty. Monocarp chooses **one type of units** and recruits no more units of that type than he can recruit with C coins.

There are n types of units. Every unit type has three parameters:

- c_i — the cost of recruiting one unit of the i-th type;
- d_i — the damage that one unit of the i-th type deals in a second;
- h_i — the amount of health of one unit of the i-th type.

Monocarp has to face m monsters. Every monster has two parameters:

- D_j — the damage that the j-th monster deals in a second;
- H_j — the amount of health the j-th monster has.

Monocarp has to fight only the j-th monster during the j-th battle. He wants all his recruited units to stay alive. Both Monocarp's squad and the monster attack continuously (not once per second) and at the same time. Thus, Monocarp wins the battle if and only if his squad kills the monster strictly faster than the monster kills one of his units. The time is compared with no rounding.

For each monster, Monocarp wants to know the minimum amount of coins he has to spend to kill that monster. If this amount is greater than C, then report that it's impossible to kill that monster.

Input

The first line contains two integers n and C ($1 \le n \le 3 \cdot 10^5$; $1 \le C \le 10^6$) — the number of types of units and the amount of coins Monocarp has before each battle.

The i-th of the next n lines contains three integers c_i, d_i and h_i ($1 \le c_i \le C$; $1 \le d_i, h_i \le 10^6$).

The next line contains a single integer m ($1 \le m \le 3 \cdot 10^5$) — the number of monsters that Monocarp has to face.

The j-th of the next m lines contains two integers D_j and H_j ($1 \le D_j \le 10^6$; $1 \le H_j \le 10^{12}$).

Output

Print m integers. For each monster, print the minimum amount of coins Monocarp has to spend to kill that monster. If this amount is greater than C, then print -1.

Examples

input

```
3 10
3 4 6
5 5 5
10 3 4
3
8 3
5 4
10 15
```

output

```
5 3 -1
```

input

```
5 15
14 10 3
9 2 2
10 4 3
7 3 5
4 3 1
6
11 2
1 1
4 7
2 1
1 14
3 3
```

output

```
14 4 14 4 7 7
```

input

```
5 13
```

```
13 1 9
6 4 5
12 18 4
9 13 2
5 4 5
2
16 3
6 2
```

output

```
12 5
```

Note

Consider the first monster of the first example.

Monocarp can't recruit one unit of the first type, because it will take both him and the monster 0.75 seconds to kill each other. He can recruit two units for the cost of 6 coins and kill the monster in 0.375 second.

Monocarp can recruit one unit of the second type, because he kills the monster in 0.6 seconds, and the monster kills him in 0.625 seconds. The unit is faster. Thus, 5 coins is enough.

Monocarp will need at least three units of the third type to kill the first monster, that will cost 30 coins.

Monocarp will spend the least coins if he chooses the second type of units and recruits one unit of that type.

E. Star MST

time limit per test: 6 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

In this problem, we will consider **complete** undirected graphs consisting of n vertices with weighted edges. The weight of each edge is an integer from 1 to k.

An undirected graph is considered **beautiful** if the sum of weights of all edges incident to vertex 1 is equal to the weight of MST in the graph. MST is the minimum spanning tree — a tree consisting of $n-1$ edges of the graph, which connects all n vertices and has the minimum sum of weights among all such trees; the weight of MST is the sum of weights of all edges in it.

Calculate the number of **complete beautiful** graphs having exactly n vertices and the weights of edges from 1 to k. Since the answer might be large, print it modulo 998244353.

Input

The only line contains two integers n and k ($2 \le n \le 250$; $1 \le k \le 250$).

Output

Print one integer — the number of **complete beautiful** graphs having exactly n vertices and the weights of edges from 1 to k. Since the answer might be large, print it modulo 998244353.

Examples

input

```
3 2
```

output

```
5
```

input

```
4 4
```

output

```
571
```

input

```
6 9
```

output

```
310640163
```

input

```
42 13
```

output

```
136246935
```

F. Words on Tree

time limit per test: 9 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

You are given a tree consisting of n vertices, and q triples (x_i, y_i, s_i), where x_i and y_i are integers from 1 to n, and s_i is a string **with length equal to the number of vertices on the simple path from x_i to y_i**.

You want to write a lowercase Latin letter on each vertex in such a way that, for each of q given triples, at least one of the following conditions holds:

- if you write out the letters on the vertices on the simple path from x_i to y_i in the order they appear on this path, you get the string s_i;
- if you write out the letters on the vertices on the simple path from y_i to x_i in the order they appear on this path, you get the string s_i.

Find any possible way to write a letter on each vertex to meet these constraints, or report that it is impossible.

Input

The first line contains two integers n and q ($2 \le n \le 4 \cdot 10^5$; $1 \le q \le 4 \cdot 10^5$) — the number of vertices in the tree and the number of triples, respectively.

Then $n-1$ lines follow; the i-th of them contains two integers u_i and v_i ($1 \le u_i, v_i \le n$; $u_i \ne v_i$) — the endpoints of the i-th edge. These edges form a tree.

Then q lines follow; the j-th of them contains two integers x_j and y_j, and a string s_j consisting of lowercase Latin letters. The length of s_j is equal to the number of vertices on the simple path between x_j and y_j.

Additional constraint on the input: $\sum_{j=1}^{q} |s_j| \le 4 \cdot 10^5$.

Output

If there is no way to meet the conditions on all triples, print NO. Otherwise, print YES in the first line, and a string of n lowercase Latin letters in the second line; the i-th character of the string should be the letter you write on the i-th vertex. If there are multiple answers, print any of them.

Examples

input
```
3 2
2 3
2 1
2 1 ab
2 3 bc
```
output
```
YES
abc
```

input
```
3 2
2 3
2 1
2 1 ab
2 3 cd
```
output
```
NO
```

input
```
10 10
1 2
1 3
1 4
1 5
1 6
1 7
1 8
1 9
1 10
1 2 ab
1 3 ab
1 4 ab
1 5 ab
1 6 ab
1 7 ab
1 8 ab
1 9 ab
1 10 ab
10 2 aba
```

output

```
YES
baaaaaaaaa
```

input

```
10 10
1 2
1 3
1 4
1 5
1 6
1 7
1 8
1 9
1 10
1 2 ab
1 3 ab
1 4 aa
1 5 ab
1 6 ab
1 7 ab
1 8 ab
1 9 ab
1 10 ab
10 2 aba
```

output

```
NO
```