A. Metro

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

Alice has a birthday today, so she invited home her best friend Bob. Now Bob needs to find a way to commute to the Alice's home.

In the city in which Alice and Bob live, the first metro line is being built. This metro line contains n stations numbered from 1 to n. Bob lives near the station with number 1, while Alice lives near the station with number s. The metro line has two tracks. Trains on the first track go from the station 1 to the station n and trains on the second track go in reverse direction. Just after the train arrives to the end of its track, it goes to the depot immediately, so it is impossible to travel on it after that.

Some stations are not yet open at all and some are only partially open — for each station and for each track it is known whether the station is closed for that track or not. If a station is closed for some track, all trains going in this track's direction pass the station without stopping on it.

When the Bob got the information on opened and closed stations, he found that traveling by metro may be unexpectedly complicated. Help Bob determine whether he can travel to the Alice's home by metro or he should search for some other transport.

Input

The first line contains two integers n and s ($2 \le s \le n \le 1000$) — the number of stations in the metro and the number of the station where Alice's home is located. Bob lives at station 1.

Next lines describe information about closed and open stations.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($a_i = 0$ or $a_i = 1$). If $a_i = 1$, then the i-th station is open on the first track (that is, in the direction of increasing station numbers). Otherwise the station is closed on the first track.

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($b_i = 0$ or $b_i = 1$). If $b_i = 1$, then the i-th station is open on the second track (that is, in the direction of decreasing station numbers). Otherwise the station is closed on the second track.

Output

Print "YES" (quotes for clarity) if Bob will be able to commute to the Alice's home by metro and "NO" (quotes for clarity) otherwise.

You can print each letter in any case (upper or lower).

Examples

input

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

output

```
YES
```

input

```
5 4
1 0 0 0 1
0 1 1 1 1
```

output

```
YES
```

input

```
5 2
0 1 1 1 1
1 1 1 1 1
```

output

```
NO
```

Note

In the first example, all stations are opened, so Bob can simply travel to the station with number 3.

In the second example, Bob should travel to the station 5 first, switch to the second track and travel to the station 4 then.

In the third example, Bob simply can't enter the train going in the direction of Alice's home.

B. Alice and Hairdresser

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

Alice's hair is growing by leaps and bounds. Maybe the cause of it is the excess of vitamins, or maybe it is some black magic...

To prevent this, Alice decided to go to the hairdresser. She wants for her hair length to be at most l centimeters after haircut, where l is her favorite number. Suppose, that the Alice's head is a straight line on which n hairlines grow. Let's number them from 1 to n. With one swing of the scissors the hairdresser can shorten all hairlines on any segment to the length l, given that **all** hairlines on that segment had length **strictly greater** than l. The hairdresser wants to complete his job as fast as possible, so he will make the least possible number of swings of scissors, since each swing of scissors takes one second.

Alice hasn't decided yet when she would go to the hairdresser, so she asked you to calculate how much time the haircut would take depending on the time she would go to the hairdresser. In particular, you need to process queries of two types:

- 0 — Alice asks how much time the haircut would take if she would go to the hairdresser now.
- 1 p d — p-th hairline grows by d centimeters.

Note, that in the request 0 Alice is interested in hypothetical scenario of taking a haircut now, so no hairlines change their length.

Input

The first line contains three integers n, m and l ($1 \le n, m \le 100\,000$, $1 \le l \le 10^9$) — the number of hairlines, the number of requests and the favorite number of Alice.

The second line contains n integers a_i ($1 \le a_i \le 10^9$) — the initial lengths of all hairlines of Alice.

Each of the following m lines contains a request in the format described in the statement.

The request description starts with an integer t_i. If $t_i = 0$, then you need to find the time the haircut would take. Otherwise, $t_i = 1$ and in this moment one hairline grows. The rest of the line than contains two more integers: p_i and d_i ($1 \le p_i \le n$, $1 \le d_i \le 10^9$) — the number of the hairline and the length it grows by.

Output

For each query of type 0 print the time the haircut would take.

Example

input

```
4 7 3
1 2 3 4
0
1 2 3
0
1 1 3
0
1 3 1
0
```

output

```
1
2
2
1
```

Note

Consider the first example:

- Initially lengths of hairlines are equal to $1, 2, 3, 4$ and only 4-th hairline is longer $l = 3$, and hairdresser can cut it in 1 second.
- Then Alice's second hairline grows, the lengths of hairlines are now equal to $1, 5, 3, 4$
- Now haircut takes two seonds: two swings are required: for the 4-th hairline and for the 2-nd.
- Then Alice's first hairline grows, the lengths of hairlines are now equal to $4, 5, 3, 4$
- The haircut still takes two seconds: with one swing hairdresser can cut 4-th hairline and with one more swing cut the segment from 1-st to 2-nd hairline.
- Then Alice's third hairline grows, the lengths of hairlines are now equal to $4, 5, 4, 4$
- Now haircut takes only one second: with one swing it is possible to cut the segment from 1-st hairline to the 4-th.

C. Lucky Days

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

Bob and Alice are often participating in various programming competitions. Like many competitive programmers, Alice and Bob have good and bad days. They noticed, that their lucky and unlucky days are repeating with some period. For example, for Alice days $[l_a; r_a]$ are lucky, then there are some unlucky days: $[r_a + 1; l_a + t_a - 1]$, and then there are lucky days again: $[l_a + t_a; r_a + t_a]$ and so on. In other words, the day is lucky for Alice if it lies in the segment $[l_a + kt_a; r_a + kt_a]$ for some non-negative integer k.

The Bob's lucky day have similar structure, however the parameters of his sequence are different: l_b, r_b, t_b. So a day is a lucky for Bob if it lies in a segment $[l_b + kt_b; r_b + kt_b]$, for some non-negative integer k.

Alice and Bob want to participate in team competitions together and so they want to find out what is the largest possible number of consecutive days, which are lucky for both Alice and Bob.

Input

The first line contains three integers l_a, r_a, t_a ($0 \le l_a \le r_a \le t_a - 1, 2 \le t_a \le 10^9$) and describes Alice's lucky days.

The second line contains three integers l_b, r_b, t_b ($0 \le l_b \le r_b \le t_b - 1, 2 \le t_b \le 10^9$) and describes Bob's lucky days.

It is guaranteed that both Alice and Bob have some unlucky days.

Output

Print one integer: the maximum number of days in the row that are lucky for both Alice and Bob.

Examples

input

```
0 2 5
1 3 5
```

output

```
2
```

input

```
0 1 3
2 3 6
```

output

```
1
```

Note

The graphs below correspond to the two sample tests and show the lucky and unlucky days of Alice and Bob as well as the possible solutions for these tests.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
Alice	+	+	+	−	−	+	+	+	−	−	+	+	+	−	−	…
Bob	−	+	+	+	−	−	+	+	+	−	−	+	+	+	−	…

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	
Alice	+	+	−	+	+	−	+	+	−	+	+	−	+	+	−	…
Bob	−	−	+	+	−	−	−	−	+	+	−	−	−	−	+	…

D. Refactoring

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

Alice has written a program and now tries to improve its readability. One of the ways to improve readability is to give sensible names to the variables, so now Alice wants to rename some variables in her program. In her IDE there is a command called "massive refactoring", which can replace names of many variable in just one run. To use it, Alice needs to select two strings s and t and after that for each variable the following algorithm is performed: if the variable's name contains s as a substring, then the first (and only first) occurrence of s is replaced with t. If the name doesn't contain s, then this variable's name stays the same.

The list of variables is known and for each variable both the initial name and the name Alice wants this variable change to are known. Moreover, for each variable the lengths of the initial name and the target name are equal (otherwise the alignment of the code could become broken). You need to perform renaming of all variables in **exactly one** run of the massive refactoring command or determine that it is impossible.

Input

The first line contains the only integer n ($1 \le n \le 3000$) — the number of variables in Alice's program.

The following n lines contain the initial names of variables $w_1, w_2, \ldots, w_n$, one per line. After that, n more lines go, the i-th of them contains the target name w'_i for the i-th variable. It is guaranteed that $1 \le |w_i| = |w'_i| \le 3000$.

It is guaranteed that there is **at least one** variable having its target name **different** from the initial name. Both initial and target names consist of lowercase English letters only. For each variable the length of its initial name is equal to the length of its target name.

Output

If it is impossible to rename all variables with one call of "massive refactoring", print "NO" (quotes for clarity).

Otherwise, on the first line print "YES" (quotes for clarity) and on the following lines print s and t ($1 \le |s|, |t| \le 5000$), which should be used for replacement. Strings s and t should consist only of lowercase letters of English alphabet.

If there are multiple ways to perform a "massive refactoring", you can use any of them.

Examples

input

```
1
topforces
codecoder
```

output

```
YES
topforces
codecoder
```

input

```
3
bab
cac
cdc
bdb
cdc
cdc
```

output

```
YES
a
d
```

input

```
2
you
shal
not
pass
```

output

```
NO
```

E. Segments on the Line

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

You are a given a list of integers $a_1, a_2, \ldots, a_n$ and s of its segments $[l_j; r_j]$ (where $1 \le l_j \le r_j \le n$).

You need to select exactly m segments in such a way that the k-th order statistic of the multiset of a_i, where i is contained in at least one segment, is the smallest possible. If it's impossible to select a set of m segments in such a way that the multiset contains at least k elements, print -1.

The k-th order statistic of a multiset is the value of the k-th element after sorting the multiset in non-descending order.

Input

The first line contains four integers n, s, m and k ($1 \le m \le s \le 1500$, $1 \le k \le n \le 1500$) — the size of the list, the number of segments, the number of segments to choose and the statistic number.

The second line contains n integers a_i ($1 \le a_i \le 10^9$) — the values of the numbers in the list.

Each of the next s lines contains two integers l_i and r_i ($1 \le l_i \le r_i \le n$) — the endpoints of the segments.

It is possible that some segments coincide.

Output

Print exactly one integer — the smallest possible k-th order statistic, or -1 if it's impossible to choose segments in a way that the multiset contains at least k elements.

Examples

input

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

output

```
2
```

input

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

output

```
1
```

input

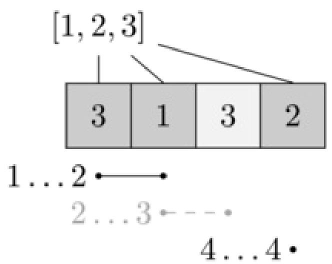
```
5 3 3 5
5 5 2 1 1
1 2
2 3
3 4
```

output

```
-1
```

Note

In the first example, one possible solution is to choose the first and the third segment. Together they will cover three elements of the list (all, except for the third one). This way the 2-nd order statistic for the covered elements is 2.



F. Tree and XOR

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

You are given a connected undirected graph without cycles (that is, a tree) of n vertices, moreover, there is a non-negative integer written on every edge.

Consider all pairs of vertices (v, u) (that is, there are exactly n^2 such pairs) and for each pair calculate the bitwise exclusive or (`xor`) of all integers on edges of the simple path between v and u. If the path consists of one vertex only, then xor of all integers on edges of this path is equal to 0.

Suppose we sorted the resulting n^2 values in non-decreasing order. You need to find the k-th of them.

The definition of `xor` is as follows.

Given two integers x and y, consider their binary representations (possibly with leading zeros): $x_k \dots x_2 x_1 x_0$ and $y_k \dots y_2 y_1 y_0$ (where k is any number so that all bits of x and y can be represented). Here, x_i is the i-th bit of the number x and y_i is the i-th bit of the number y. Let $r = x \oplus y$ be the result of the `xor` operation of x and y. Then r is defined as $r_k \dots r_2 r_1 r_0$ where:

$$r_i = \begin{cases} 1, & \text{if } x_i \neq y_i \\ 0, & \text{if } x_i = y_i \end{cases}$$

Input

The first line contains two integers n and k ($2 \le n \le 10^6$, $1 \le k \le n^2$) — the number of vertices in the tree and the number of path in the list with non-decreasing order.

Each of the following $n - 1$ lines contains two integers p_i and w_i ($1 \le p_i \le i$, $0 \le w_i < 2^{62}$) — the ancestor of vertex $i + 1$ and the weight of the corresponding edge.

Output

Print one integer: k-th smallest xor of a path in the tree.

Examples

input

```
2 1
1 3
```

output

```
0
```

input

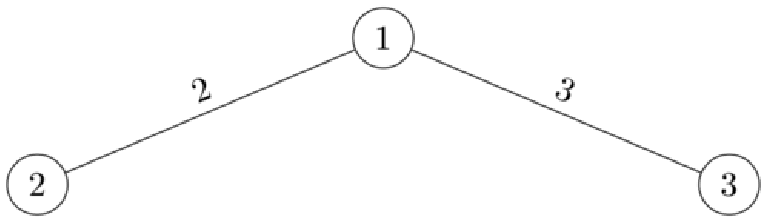
```
3 6
1 2
1 3
```

output

```
2
```

Note

The tree in the second sample test looks like this:



For such a tree in total 9 paths exist:

1. $1 \to 1$ of value 0
2. $2 \to 2$ of value 0
3. $3 \to 3$ of value 0
4. $2 \to 3$ (goes through 1) of value $1 = 2 \oplus 3$
5. $3 \to 2$ (goes through 1) of value $1 = 2 \oplus 3$
6. $1 \to 2$ of value 2
7. $2 \to 1$ of value 2
8. $1 \to 3$ of value 3
9. $3 \to 1$ of value 3

G. Jellyfish Nightmare

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

Bob has put on some weight recently. In order to lose weight a bit, Bob has decided to swim regularly in the pool. However, the day before he went to the pool for the first time he had a weird dream. In this dream Bob was swimming along one of the pool's lanes, but also there were some jellyfish swimming around him. It's worth mentioning that jellyfish have always been one of Bob's deepest childhood fears.

Let us assume the following physical model for Bob's dream.

1. The pool's lane is an area of the plane between lines $x = 0$ and $x = w$. Bob is not allowed to swim outside of the lane, but he may touch its bounding lines if he wants.
2. The jellyfish are very small, but in Bob's dream they are extremely swift. Each jellyfish has its area of activity around it. Those areas are circles of various radii, with the jellyfish sitting in their centers. The areas of activity of two jellyfish may overlap and one area of activity may even be fully contained within another one.
3. Bob has a shape of a convex polygon.
4. Unfortunately, Bob's excess weight has made him very clumsy, and as a result he can't rotate his body while swimming. So he swims in a parallel translation pattern. However at any given moment of time he can choose any direction of his movement.
5. Whenever Bob swims into a jellyfish's activity area, it will immediately notice him and sting him very painfully. We assume that Bob has swum into the activity area if at some moment of time the intersection of his body with the jellyfish's activity area had a positive area (for example, if they only touch, the jellyfish does not notice Bob).
6. Once a jellyfish stung Bob, it happily swims away and no longer poses any threat to Bob.

Bob wants to swim the lane to its end and get stung the least possible number of times. He will start swimming on the line $y = -h$, and finish on the line $y = h$ where $h = 10^{10}$.

Input

The first line contains two integers n and w ($3 \le n \le 200, 1 \le w \le 30000$) — the number of vertices in the polygon that constitutes the Bob's shape and the width of the swimming pool lane.

Each of the next n lines contains two integers x_i and y_i $(0 \le x_i \le w, 0 \le y_i \le 30000)$ — the coordinates of corresponding vertex of the polygon. The vertices in the polygon are given in counterclockwise order. It is guaranteed that the given polygon is strictly convex.

The next line contains an only integer m $(0 \le m \le 200)$ — the number of the jellyfish in the pool.

Each of the next m lines contains three integers (x_i, y_i, r_i $(0 \le x_i \le w, 0 \le y_i \le 30000, 1 \le r_i \le 30000)$ — coordinates of the i-th jellyfish in the pool and the radius of her activity. It is guaranteed, that no two jellyfish are located in the same point.

Output

Output a single integer — the least possible number of jellyfish that will sting Bob.

Examples

input

```
4 4
0 0
2 0
2 2
0 2
3
1 1 1
3 5 1
1 9 1
```

output

```
0
```

input

```
4 6
0 0
3 0
3 3
0 3
3
1 0 1
4 2 2
3 6 1
```

output

```
2
```

input

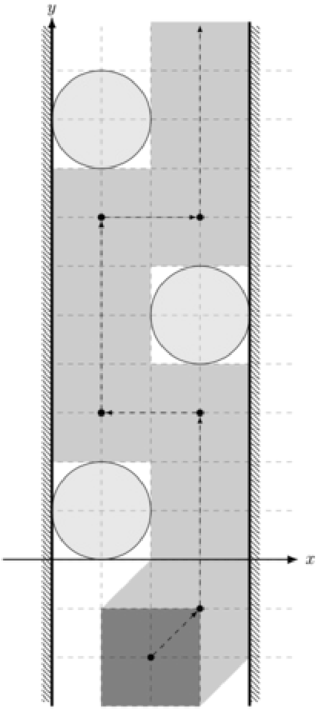
```
4 2
0 0
1 0
1 1
0 1
2
1 1 1
1 3 1
```

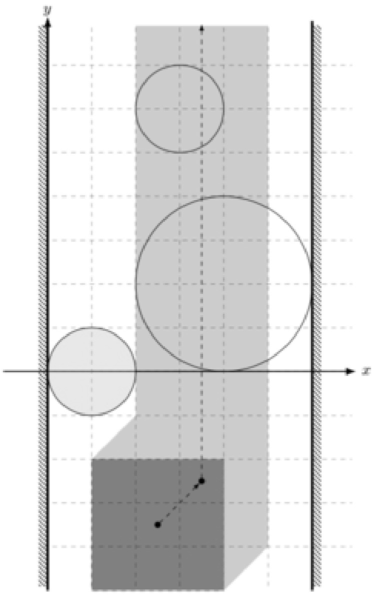
output

```
2
```

Note

Visualization of the possible solutions to the first and the second sample test cases are below:


y
x


y
x