Codeforces Round #512 (Div. 1, based on Technocup 2019 Elimination Round 1)

A. Vasya and Triangle

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

Vasya has got three integers n, m and k. He'd like to find three integer points (x_1, y_1), (x_2, y_2), (x_3, y_3), such that $0 \le x_1, x_2, x_3 \le n$, $0 \le y_1, y_2, y_3 \le m$ and the area of the triangle formed by these points is equal to $\frac{nm}{k}$.

Help Vasya! Find such points (if it's possible). If there are multiple solutions, print any of them.

Input

The single line contains three integers n, m, k ($1 \le n, m \le 10^9$, $2 \le k \le 10^9$).

Output

If there are no such points, print "NO".

Otherwise print "YES" in the first line. The next three lines should contain integers x_i, y_i — coordinates of the points, one point per line. If there are multiple solutions, print any of them.

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

Examples

input

```
4 3 3
```

output

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

input

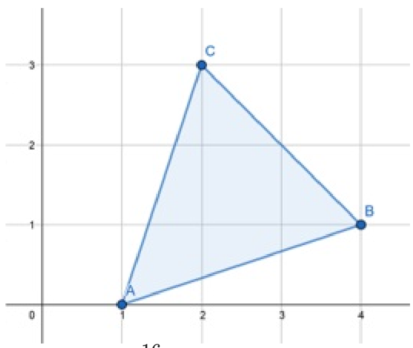
```
4 4 7
```

output

```
NO
```

Note

In the first example area of the triangle should be equal to $\frac{nm}{k} = 4$. The triangle mentioned in the output is pictured below:



In the second example there is no triangle with area $\frac{nm}{k} = \frac{16}{7}$.

B. Vasya and Good Sequences

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

Vasya has a sequence a consisting of n integers $a_1, a_2, \dots, a_n$. Vasya may pefrom the following operation: choose some number from the sequence and swap any pair of bits in its binary representation. For example, Vasya can transform number 6 $(\dots 00000000110_2)$ into 3 $(\dots 00000000011_2)$, 12 $(\dots 000000001100_2)$, 1026 $(\dots 10000000010_2)$ and many others. Vasya can use this operation any (possibly zero) number of times on any number from the sequence.

Vasya names a sequence as *good* one, if, using operation mentioned above, he can obtain the sequence with bitwise exclusive or of all elements equal to 0.

For the given sequence $a_1, a_2, \ldots, a_n$ Vasya'd like to calculate number of integer pairs (l, r) such that $1 \le l \le r \le n$ and sequence $a_l, a_{l+1}, \ldots, a_r$ is good.

Input

The first line contains a single integer n ($1 \le n \le 3 \cdot 10^5$) — length of the sequence.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{18}$) — the sequence a.

Output

Print one integer — the number of pairs (l, r) such that $1 \le l \le r \le n$ and the sequence $a_l, a_{l+1}, \ldots, a_r$ is good.

Examples

input

```
3
6 7 14
```

output

```
2
```

input

```
4
1 2 1 16
```

output

```
4
```

Note

In the first example pairs $(2, 3)$ and $(1, 3)$ are valid. Pair $(2, 3)$ is valid since $a_2 = 7 \to 11$, $a_3 = 14 \to 11$ and $11 \oplus 11 = 0$, where $\oplus$ — bitwise exclusive or. Pair $(1, 3)$ is valid since $a_1 = 6 \to 3$, $a_2 = 7 \to 13$, $a_3 = 14 \to 14$ and $3 \oplus 13 \oplus 14 = 0$.

In the second example pairs $(1, 2)$, $(2, 3)$, $(3, 4)$ and $(1, 4)$ are valid.

C. Putting Boxes Together

time limit per test: 2.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

There is an infinite line consisting of cells. There are n boxes in some cells of this line. The i-th box stands in the cell a_i and has weight w_i. All a_i are distinct, moreover, $a_{i-1} < a_i$ holds for all valid i.

You would like to put together some boxes. Putting together boxes with *indices* in the segment $[l, r]$ means that you will move some of them in such a way that their *positions* will form some segment $[x, x + (r - l)]$.

In one step you can move any box to a neighboring cell if it isn't occupied by another box (i.e. you can choose i and change a_i by 1, all positions should remain distinct). You spend w_i units of energy moving the box i by one cell. You can move any box any number of times, in arbitrary order.

Sometimes weights of some boxes change, so you have queries of two types:

1. $id\ nw$ — weight w_{id} of the box id becomes nw.
2. $l\ r$ — you should compute the minimum total energy needed to put together boxes with indices in $[l, r]$. Since the answer can be rather big, print the remainder it gives when divided by $1\,000\,000\,007 = 10^9 + 7$. Note that the boxes are not moved during the query, you only should compute the answer.

Note that you should minimize the answer, not its remainder modulo $10^9 + 7$. So if you have two possible answers $2 \cdot 10^9 + 13$ and $2 \cdot 10^9 + 14$, you should choose the first one and print $10^9 + 6$, even though the remainder of the second answer is 0.

Input

The first line contains two integers n and q ($1 \le n, q \le 2 \cdot 10^5$) — the number of boxes and the number of queries.

The second line contains n integers $a_1, a_2, \ldots a_n$ ($1 \le a_i \le 10^9$) — the positions of the boxes. All a_i are distinct, $a_{i-1} < a_i$ holds for all valid i.

The third line contains n integers $w_1, w_2, \ldots w_n$ ($1 \le w_i \le 10^9$) — the initial weights of the boxes.

Next q lines describe queries, one query per line.

Each query is described in a single line, containing two integers x and y. If $x < 0$, then this query is of the first type, where $id = -x$, $nw = y$ ($1 \le id \le n$, $1 \le nw \le 10^9$). If $x > 0$, then the query is of the second type, where $l = x$ and $r = y$ ($1 \le l_j \le r_j \le n$). x

can not be equal to 0.

Output

For each query of the second type print the answer on a separate line. Since answer can be large, print the remainder it gives when divided by $1\,000\,000\,007 = 10^9 + 7$.

Example

input

```
5 8
1 2 6 7 10
1 1 1 1 2
1 1
1 5
1 3
3 5
-3 5
-1 10
1 4
2 5
```

output

```
0
10
3
4
18
7
```

Note

Let's go through queries of the example:

1. $1\ 1$ — there is only one box so we don't need to move anything.
2. $1\ 5$ — we can move boxes to segment $[4, 8]$: $1 \cdot |1 - 4| + 1 \cdot |2 - 5| + 1 \cdot |6 - 6| + 1 \cdot |7 - 7| + 2 \cdot |10 - 8| = 10$.
3. $1\ 3$ — we can move boxes to segment $[1, 3]$.
4. $3\ 5$ — we can move boxes to segment $[7, 9]$.
5. $-3\ 5$ — w_3 is changed from 1 to 5.
6. $-1\ 10$ — w_1 is changed from 1 to 10. The weights are now equal to $w = [10, 1, 5, 1, 2]$.
7. $1\ 4$ — we can move boxes to segment $[1, 4]$.
8. $2\ 5$ — we can move boxes to segment $[5, 8]$.

D. Linear Congruential Generator

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

You are given a tuple generator $f^{(k)} = (f_1^{(k)}, f_2^{(k)}, \dots, f_n^{(k)})$, where $f_i^{(k)} = (a_i \cdot f_i^{(k-1)} + b_i) \bmod p_i$ and $f^{(0)} = (x_1, x_2, \dots, x_n)$. Here $x \bmod y$ denotes the remainder of x when divided by y. All p_i are primes.

One can see that with fixed sequences x_i, y_i, a_i the tuples $f^{(k)}$ starting from some index will repeat tuples with smaller indices. Calculate the maximum number of different tuples (from all $f^{(k)}$ for $k \ge 0$) that can be produced by this generator, if x_i, a_i, b_i are integers in the range $[0, p_i - 1]$ and can be chosen arbitrary. The answer can be large, so print the remainder it gives when divided by $10^9 + 7$

Input

The first line contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of elements in the tuple.

The second line contains n space separated prime numbers — the modules $p_1, p_2, \dots, p_n$ ($2 \le p_i \le 2 \cdot 10^6$).

Output

Print one integer — the maximum number of different tuples modulo $10^9 + 7$.

Examples

input

```
4
2 3 5 7
```

output

```
210
```

input

```
3
5 3 3
```

output

```
30
```

Note

In the first example we can choose next parameters: $a = [1, 1, 1, 1]$, $b = [1, 1, 1, 1]$, $x = [0, 0, 0, 0]$, then $f_i^{(k)} = k \bmod p_i$.

In the second example we can choose next parameters: $a = [1, 1, 2]$, $b = [1, 1, 0]$, $x = [0, 0, 1]$.

E. Euler tour

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

Euler is a little, cute squirrel. When the autumn comes, he collects some reserves for winter. The interesting fact is that Euler likes to collect acorns in a specific way. A tree can be described as n acorns connected by $n - 1$ branches, such that there is exactly one way between each pair of acorns. Let's enumerate the acorns from 1 to n.

The squirrel chooses one acorn (not necessary with number 1) as a start, and visits them in a way called "Euler tour" (see notes), collecting each acorn when he visits it for the last time.

Today morning Kate was observing Euler. She took a sheet of paper and wrote down consecutive indices of acorns on his path. Unfortunately, during her way to home it started raining and some of numbers became illegible. Now the girl is very sad, because she has to present the observations to her teacher.

"Maybe if I guess the lacking numbers, I'll be able to do it!" she thought. Help her and restore any valid Euler tour of some tree or tell that she must have made a mistake.

Input

The first line contains a single integer n ($1 \leq n \leq 5 \cdot 10^5$), denoting the number of acorns in the tree.

The second line contains $2n - 1$ integers $a_1, a_2, \ldots, a_{2n-1}$ ($0 \leq a_i \leq n$) — the Euler tour of the tree that Kate wrote down. 0 means an illegible number, which has to be guessed.

Output

If there is no Euler tour satisfying the given description, output "no" in the first line.

Otherwise, on the first line output "yes", and then in the second line print the Euler tour which satisfy the given description.

Any valid Euler tour will be accepted, since the teacher doesn't know how exactly the initial tree looks.

Examples

input

```
2
1 0 0
```

output

```
yes
1 2 1
```

input

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

output

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

input

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

output

```
no
```

Note

An Euler tour of a tree with n acorns is a sequence of $2n - 1$ indices of acorns. such that each acorn occurs at least once, the first and the last acorns are same and each two consecutive acorns are directly connected with a branch.