Educational Codeforces Round 45 (Rated for Div. 2)

A. Commentary Boxes

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

Berland Football Cup starts really soon! Commentators from all over the world come to the event.

Organizers have already built n commentary boxes. m regional delegations will come to the Cup. Every delegation should get the *same number* of the commentary boxes. If any box is left unoccupied then the delegations will be upset. *So each box should be occupied by exactly one delegation.*

If n is not divisible by m, it is impossible to distribute the boxes to the delegations at the moment.

Organizers can build a new commentary box paying a burles and demolish a commentary box paying b burles. They can both build and demolish boxes arbitrary number of times (each time paying a corresponding fee). It is allowed to demolish all the existing boxes.

What is the minimal amount of burles organizers should pay to satisfy all the delegations (i.e. to make the number of the boxes be divisible by m)?

Input

The only line contains four integer numbers n, m, a and b ($1 \le n, m \le 10^{12}$, $1 \le a, b \le 100$), where n is the initial number of the commentary boxes, m is the number of delegations to come, a is the fee to build a box and b is the fee to demolish a box.

Output

Output the minimal amount of burles organizers should pay to satisfy all the delegations (i.e. to make the number of the boxes be divisible by m). It is allowed that the final number of the boxes is equal to 0.

Examples

input
```
9 7 3 8
```
output
```
15
```

input
```
2 7 3 7
```
output
```
14
```

input
```
30 6 17 19
```
output
```
0
```

Note

In the first example organizers can build 5 boxes to make the total of 14 paying 3 burles for the each of them.

In the second example organizers can demolish 2 boxes to make the total of 0 paying 7 burles for the each of them.

In the third example organizers are already able to distribute all the boxes equally among the delegations, each one get 5 boxes.

B. Micro-World

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

You have a Petri dish with bacteria and you are preparing to dive into the harsh micro-world. But, unfortunately, you don't have any microscope nearby, so you can't watch them.

You know that you have n bacteria in the Petri dish and size of the i-th bacteria is a_i. Also you know intergalactic positive integer constant $$$K$$$.

The $$$i$$$-th bacteria can swallow the $$$j$$$-th bacteria if and only if $$$a_i > a_j$$$ and $$$a_i \le a_j + K$$$. The $$$j$$$-th bacteria disappear, but the $$$i$$$-th bacteria doesn't change its size. The bacteria can perform multiple swallows. On each swallow operation any bacteria $$$i$$$ can swallow any bacteria $$$j$$$ if $$$a_i > a_j$$$ and $$$a_i \le a_j + K$$$. The swallow operations go one after another.

For example, the sequence of bacteria sizes $$$a=[101, 53, 42, 102, 101, 55, 54]$$$ and $$$K=1$$$. The one of possible sequences of swallows is: $$$[101, 53, 42, 102, \underline{101}, 55, 54]$$$ $$$\to$$$ $$$[101, \underline{53}, 42, 102, 55, 54]$$$ $$$\to$$$ $$$[\underline{101}, 42, 102, 55, 54]$$$ $$$\to$$$ $$$[42, 102, 55, \underline{54}]$$$ $$$\to$$$ $$$[42, 102, 55]$$$. In total there are $$$3$$$ bacteria remained in the Petri dish.

Since you don't have a microscope, you can only guess, what the minimal possible number of bacteria can remain in your Petri dish when you finally will find any microscope.

Input

The first line contains two space separated positive integers $$$n$$$ and $$$K$$$ ($$$1 \le n \le 2 \cdot 10^5$$$, $$$1 \le K \le 10^6$$$) — number of bacteria and intergalactic constant $$$K$$$.

The second line contains $$$n$$$ space separated integers $$$a_1, a_2, \dots, a_n$$$ ($$$1 \le a_i \le 10^6$$$) — sizes of bacteria you have.

Output

Print the only integer — minimal possible number of bacteria can remain.

Examples

input

```
7 1
101 53 42 102 101 55 54
```

output

```
3
```

input

```
6 5
20 15 10 15 20 25
```

output

```
1
```

input

```
7 1000000
1 1 1 1 1 1 1
```

output

```
7
```

Note

The first example is clarified in the problem statement.

In the second example an optimal possible sequence of swallows is: $$$[20, 15, 10, 15, \underline{20}, 25]$$$ $$$\to$$$ $$$[20, 15, 10, \underline{15}, 25]$$$ $$$\to$$$ $$$[20, 15, \underline{10}, 25]$$$ $$$\to$$$ $$$[20, \underline{15}, 25]$$$ $$$\to$$$ $$$[\underline{20}, 25]$$$ $$$\to$$$ $$$[25]$$$.

In the third example no bacteria can swallow any other bacteria.

C. Bracket Sequences Concatenation Problem

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

A bracket sequence is a string containing only characters "(" and ")".

A regular bracket sequence is a bracket sequence that can be transformed into a correct arithmetic expression by inserting characters "1" and "+" between the original characters of the sequence. For example, bracket sequences "()()", "(())" are regular (the resulting expressions are: "(1)+(1)", "((1+1)+1)"), and ")(" and "(" are not.

You are given $$$n$$$ bracket sequences $$$s_1, s_2, \dots , s_n$$$. Calculate the number of pairs $$$i, j \, (1 \le i, j \le n)$$$ such that the bracket sequence $$$s_i + s_j$$$ is a regular bracket sequence. Operation $$$+$$$ means concatenation i.e. "()(" + ")()" = "()()()".

If $$$s_i + s_j$$$ and $$$s_j + s_i$$$ are regular bracket sequences and $$$i \ne j$$$, then both pairs $$$(i, j)$$$ and $$$(j, i)$$$ must be counted in the answer. Also, if $$$s_i + s_i$$$ is a regular bracket sequence, the pair $$$(i, i)$$$ must be counted in the answer.

Input

The first line contains one integer $$$n \, (1 \le n \le 3 \cdot 10^5)$$$ — the number of bracket sequences. The following $$$n$$$ lines contain bracket sequences — **non-empty** strings consisting only of characters "(" and ")". The sum of lengths of all bracket sequences does not exceed

$3 \cdot 10^5$.

Output

In the single line print a single integer — the number of pairs $i, j \, (1 \le i, j \le n)$ such that the bracket sequence $s_i + s_j$ is a regular bracket sequence.

Examples

input

```
3
)
()
(
```

output

```
2
```

input

```
2
()
()
```

output

```
4
```

Note

In the first example, suitable pairs are $(3, 1)$ and $(2, 2)$.

In the second example, any pair is suitable, namely $(1, 1), (1, 2), (2, 1), (2, 2)$.

D. Graph And Its Complement

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

Given three numbers n, a, b. You need to find an adjacency matrix of such an undirected graph that the number of components in it is equal to a, and the number of components in its complement is b. The matrix must be symmetric, and all digits on the main diagonal must be zeroes.

In an undirected graph loops (edges from a vertex to itself) are not allowed. It can be at most one edge between a pair of vertices.

The adjacency matrix of an undirected graph is a square matrix of size n consisting only of "0" and "1", where n is the number of vertices of the graph and the i-th row and the i-th column correspond to the i-th vertex of the graph. The cell (i,j) of the adjacency matrix contains 1 if and only if the i-th and j-th vertices in the graph are connected by an edge.

A connected component is a set of vertices X such that for every two vertices from this set there exists at least one path in the graph connecting this pair of vertices, but adding any other vertex to X violates this rule.

The complement or inverse of a graph G is a graph H on the same vertices such that two distinct vertices of H are adjacent if and only if they are not adjacent in G.

Input

In a single line, three numbers are given $n, a, b \,(1 \le n \le 1000, 1 \le a, b \le n)$: is the number of vertexes of the graph, the required number of connectivity components in it, and the required amount of the connectivity component in it's complement.

Output

If there is no graph that satisfies these constraints on a single line, print "NO" (without quotes).

Otherwise, on the first line, print "YES"(without quotes). In each of the next n lines, output n digits such that j-th digit of i-th line must be 1 if and only if there is an edge between vertices i and j in G (and 0 otherwise). Note that the matrix must be symmetric, and all digits on the main diagonal must be zeroes.

If there are several matrices that satisfy the conditions — output any of them.

Examples

input

```
3 1 2
```

output

```
YES
001
001
110
```

input

```
3 3 3
```

output

```
NO
```

E. Post Lamps

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

Adilbek's house is located on a street which can be represented as the OX axis. This street is really dark, so Adilbek wants to install some post lamps to illuminate it. Street has n positions to install lamps, they correspond to the integer numbers from 0 to $n - 1$ on the OX axis. However, some positions are blocked and no post lamp can be placed there.

There are post lamps of different types which differ only by their power. When placed in position x, post lamp of power l illuminates the segment $[x; x + l]$. The power of each post lamp is always a positive integer number.

The post lamp shop provides an infinite amount of lamps of each type from power 1 to power k. Though each customer is only allowed to order post lamps of *exactly one* type. Post lamps of power l cost a_l each.

What is the minimal total cost of the post lamps of *exactly one* type Adilbek can buy to illuminate the entire segment $[0; n]$ of the street? If some lamps illuminate any other segment of the street, Adilbek does not care, so, for example, he may place a lamp of power 3 in position $n - 1$ (even though its illumination zone doesn't completely belong to segment $[0; n]$).

Input

The first line contains three integer numbers n, m and k ($1 \le k \le n \le 10^6$, $0 \le m \le n$) — the length of the segment of the street Adilbek wants to illuminate, the number of the blocked positions and the maximum power of the post lamp available.

The second line contains m integer numbers $s_1, s_2, \dots, s_m$ ($0 \le s_1 < s_2 < \dots s_m < n$) — the blocked positions.

The third line contains k integer numbers $a_1, a_2, \dots, a_k$ ($1 \le a_i \le 10^6$) — the costs of the post lamps.

Output

Print the minimal total cost of the post lamps of *exactly one* type Adilbek can buy to illuminate the entire segment $[0; n]$ of the street.

If illumintaing the entire segment $[0; n]$ is impossible, print `-1`.

Examples

input

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

output

```
6
```

input

```
4 3 4
1 2 3
1 10 100 1000
```

output

```
1000
```

input

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

output

```
-1
```

input

```
7 4 3
2 4 5 6
3 14 15
```

output

```
-1
```

F. Flow Control

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You have to handle a very complex water distribution system. The system consists of n junctions and m pipes, i-th pipe connects junctions x_i and y_i.

The only thing you can do is adjusting the pipes. You have to choose m integer numbers f_1, f_2, ..., f_m and use them as pipe settings. i-th pipe will distribute f_i units of water per second from junction x_i to junction y_i (if f_i is negative, then the pipe will distribute $|f_i|$ units of water per second from junction y_i to junction x_i). It is allowed to set f_i to any integer from $-2 \cdot 10^9$ to $2 \cdot 10^9$.

In order for the system to work properly, there are some constraints: for every $i \in [1, n]$, i-th junction has a number s_i associated with it meaning that the difference between incoming and outcoming flow for i-th junction must be **exactly** s_i (if s_i is not negative, then i-th junction must receive s_i units of water per second; if it is negative, then i-th junction must transfer $|s_i|$ units of water per second to other junctions).

Can you choose the integers f_1, f_2, ..., f_m in such a way that all requirements on incoming and outcoming flows are satisfied?

Input

The first line contains an integer n ($1 \le n \le 2 \cdot 10^5$) — the number of junctions.

The second line contains n integers $s_1, s_2, \dots, s_n$ ($-10^4 \le s_i \le 10^4$) — constraints for the junctions.

The third line contains an integer m ($0 \le m \le 2 \cdot 10^5$) — the number of pipes.

i-th of the next m lines contains two integers x_i and y_i ($1 \le x_i, y_i \le n$, $x_i \ne y_i$) — the description of i-th pipe. It is guaranteed that each unordered pair (x, y) will appear no more than once in the input (it means that there won't be any pairs (x, y) or (y, x) after the first occurrence of (x, y)). It is guaranteed that for each pair of junctions there exists a path along the pipes connecting them.

Output

If you can choose such integer numbers $f_1, f_2, \dots, f_m$ in such a way that all requirements on incoming and outcoming flows are satisfied, then output "`Possible`" in the first line. Then output m lines, i-th line should contain f_i — the chosen setting numbers for the pipes. Pipes are numbered in order they appear in the input.

Otherwise output "`Impossible`" in the only line.

Examples

input

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

output

```
Possible
4
-6
8
-7
7
```

input

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

output

```
Impossible
```

G. GCD Counting

time limit per test: 4.5 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are given a tree consisting of $$$n$$$ vertices. A number is written on each vertex; the number on vertex $$$i$$$ is equal to $$$a_i$$$.

Let's denote the function $$$g(x, y)$$$ as the greatest common divisor of the numbers written on the vertices belonging to the simple path from vertex $$$x$$$ to vertex $$$y$$$ (including these two vertices).

For every integer from $$$1$$$ to $$$2 \cdot 10^5$$$ you have to count the number of pairs $$$(x, y)$$$ $$$(1 \le x \le y \le n)$$$ such that $$$g(x, y)$$$ is equal to this number.

Input

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

The second line contains $$$n$$$ integers $$$a_1$$$, $$$a_2$$$, ..., $$$a_n$$$ $$$(1 \le a_i \le 2 \cdot 10^5)$$$ — the numbers written on vertices.

Then $$$n - 1$$$ lines follow, each containing two integers $$$x$$$ and $$$y$$$ $$$(1 \le x, y \le n, x \ne y)$$$ denoting an edge connecting vertex $$$x$$$ with vertex $$$y$$$. It is guaranteed that these edges form a tree.

Output

For every integer $$$i$$$ from $$$1$$$ to $$$2 \cdot 10^5$$$ do the following: if there is no pair $$$(x, y)$$$ such that $$$x \le y$$$ and $$$g(x, y) = i$$$, don't output anything. Otherwise output two integers: $$$i$$$ and the number of aforementioned pairs. You have to consider the values of $$$i$$$ in ascending order.

See the examples for better understanding.

Examples

input

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

output

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

input

```
6
1 2 4 8 16 32
1 6
6 3
3 4
4 2
6 5
```

output

```
1 6
2 5
4 6
8 1
16 2
32 1
```

input

```
4
9 16 144 6
1 3
2 3
4 3
```

output

```
1 1
2 1
3 1
6 2
9 2
16 2
144 1
```