A. Classroom Watch

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Eighth-grader Vova is on duty today in the class. After classes, he went into the office to wash the board, and found on it the number n. He asked what is this number and the teacher of mathematics Inna Petrovna answered Vova that n is the answer to the arithmetic task for first-graders. In the textbook, a certain **positive integer** x was given. The task was to add x to the sum of the digits of the number x written in decimal numeral system.

Since the number n on the board was small, Vova quickly guessed which x could be in the textbook. Now he wants to get a program which will search for arbitrary values of the number n for all suitable values of x or determine that such x does not exist. Write such a program for Vova.

Input

The first line contains integer n ($1 \leq n \leq 10^9$).

Output

In the first line print one integer k — number of different values of x satisfying the condition.

In next k lines print these values in ascending order.

Examples

input

```
21
```

output

```
1
15
```

input

```
20
```

output

```
0
```

Note

In the first test case $x = 15$ there is only one variant: $15 + 1 + 5 = 21$.

In the second test case there are no such x.

B. Sorting the Coins

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Recently, Dima met with Sasha in a philatelic store, and since then they are collecting coins together. Their favorite occupation is to sort collections of coins. Sasha likes having things in order, that is why he wants his coins to be arranged in a row in such a way that firstly come coins out of circulation, and then come coins still in circulation.

For arranging coins Dima uses the following algorithm. One step of his algorithm looks like the following:

1. He looks through all the coins from left to right;
2. If he sees that the i-th coin is still in circulation, and $(i+1)$-th coin is already out of circulation, he exchanges these two coins and continues watching coins from $(i+1)$-th.

Dima repeats the procedure above until it happens that no two coins were exchanged during this procedure. Dima calls *hardness of ordering* the number of steps required for him according to the algorithm above to sort the sequence, e.g. the number of times he looks through the coins from the very beginning. For example, for the ordered sequence hardness of ordering equals one.

Today Sasha invited Dima and proposed him a game. First he puts n coins in a row, all of them are out of circulation. Then Sasha chooses one of the coins out of circulation and replaces it with a coin in circulation for n times. During this process Sasha constantly asks Dima what is the hardness of ordering of the sequence.

The task is more complicated because Dima should not touch the coins and he should determine hardness of ordering in his mind. Help Dima with this task.

Input

The first line contains single integer n $(1 \le n \le 300\ 000)$ — number of coins that Sasha puts behind Dima.

Second line contains n distinct integers $p_1, p_2, \ldots, p_n$ $(1 \le p_i \le n)$ — positions that Sasha puts coins in circulation to. At first Sasha replaces coin located at position p_1, then coin located at position p_2 and so on. Coins are numbered from left to right.

Output

Print $n+1$ numbers $a_0, a_1, \ldots, a_n$, where a_0 is a hardness of ordering at the beginning, a_1 is a hardness of ordering after the first replacement and so on.

Examples

input
```
4
1 3 4 2
```
output
```
1 2 3 2 1
```

input
```
8
6 8 3 4 7 2 1 5
```
output
```
1 2 2 3 4 3 4 5 1
```

Note

Let's denote as `O` coin out of circulation, and as `X` — coin is circulation.

At the first sample, initially in row there are coins that are not in circulation, so Dima will look through them from left to right and won't make any exchanges.

After replacement of the first coin with a coin in circulation, Dima will exchange this coin with next three times and after that he will finally look through the coins and finish the process.

`XOOO` $\rightarrow$ `OOOX`

After replacement of the third coin, Dima's actions look this way:

`XOXO` $\rightarrow$ `OXOX` $\rightarrow$ `OOXX`

After replacement of the fourth coin, Dima's actions look this way:

`XOXX` $\rightarrow$ `OXXX`

Finally, after replacement of the second coin, row becomes consisting of coins that are in circulation and Dima will look through coins from left to right without any exchanges.

C. National Property

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

You all know that the Library of Bookland is the largest library in the world. There are dozens of thousands of books in the library.

Some long and uninteresting story was removed...

The alphabet of Bookland is so large that its letters are denoted by positive integers. Each letter can be small or large, the large version of a letter x is denoted by x'. BSCII encoding, which is used everywhere in Bookland, is made in that way so that large letters are presented in the order of the numbers they are denoted by, and small letters are presented in the order of the numbers they are denoted by, but all large letters are **before** all small letters. For example, the following conditions hold: $2 < 3$, $2' < 3'$, $3' < 2$.

A word $x_1, x_2, \ldots, x_a$ is not *lexicographically* greater than $y_1, y_2, \ldots, y_b$ if one of the two following conditions holds:

- $a \le b$ and $x_1 = y_1, \ldots, x_a = y_a$, i.e. the first word is the prefix of the second word;
- there is a position $1 \le j \le min(a, b)$, such that $x_1 = y_1, \ldots, x_{j-1} = y_{j-1}$ and $x_j < y_j$, i.e. at the first position where the words differ the first word has a smaller letter than the second word has.

For example, the word "$3'$ 7 5" is before the word "2 $4'$ 6" in lexicographical order. It is said that sequence of words is in lexicographical order if each word is not lexicographically greater than the next word in the sequence.

Denis has a sequence of words consisting of small letters only. He wants to change some letters to large (let's call this process a *capitalization*) in such a way that the sequence of words is in lexicographical order. However, he soon realized that for some reason he can't change a single letter in a single word. He only can choose a letter and change all of its occurrences in **all** words to large letters. He can perform this operation any number of times with arbitrary letters of Bookland's alphabet.

Help Denis to choose which letters he needs to capitalize (make large) in order to make the sequence of words lexicographically ordered, or determine that it is impossible.

Note that some words can be **equal**.

Input

The first line contains two integers n and m ($2 \le n \le 100\,000$, $1 \le m \le 100\,000$) — the number of words and the number of letters in Bookland's alphabet, respectively. The letters of Bookland's alphabet are denoted by integers from 1 to m.

Each of the next n lines contains a description of one word in format $l_i, s_{i,1}, s_{i,2}, \ldots, s_{i,l_i}$ ($1 \le l_i \le 100\,000$, $1 \le s_{i,j} \le m$), where l_i is the length of the word, and $s_{i,j}$ is the sequence of letters in the word. The words are given in the order Denis has them in the sequence.

It is guaranteed that the total length of all words is not greater than $100\,000$.

Output

In the first line print "`Yes`" (without quotes), if it is possible to capitalize some set of letters in such a way that the sequence of words becomes lexicographically ordered. Otherwise, print "`No`" (without quotes).

If the required is possible, in the second line print k — the number of letters Denis has to capitalize (make large), and in the third line print k distinct integers — these letters. Note that you **don't need to minimize** the value k.

You can print the letters in any order. If there are multiple answers, print any of them.

Examples

input

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

output

```
Yes
2
2 3
```

input

```
6 5
2 1 2
2 1 2
3 1 2 3
2 1 5
2 4 4
2 4 4
```

output

```
Yes
```

```
0
```

input

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

output

```
No
```

Note

In the first example after Denis makes letters 2 and 3 large, the sequence looks like the following:

- $2'$
- 1
- $1\ 3'\ 2'$
- $1\ 1$

The condition $2' < 1$ holds, so the first word is not lexicographically larger than the second word. The second word is the prefix of the third word, so the are in lexicographical order. As the first letters of the third and the fourth words are the same, and $3' < 1$, then the third word is not lexicographically larger than the fourth word.

In the second example the words are in lexicographical order from the beginning, so Denis can do nothing.

In the third example there is no set of letters such that if Denis capitalizes them, the sequence becomes lexicographically ordered.

D. High Cry

time limit per test: 1 second
memory limit per test: 512 megabytes
input: standard input
output: standard output

Disclaimer: there are lots of untranslateable puns in the Russian version of the statement, so there is one more reason for you to learn Russian :)

Rick and Morty like to go to the ridge High Cry for crying loudly — there is an extraordinary echo. Recently they discovered an interesting acoustic characteristic of this ridge: if Rick and Morty begin crying simultaneously from different mountains, their cry would be heard between these mountains up to the height equal the bitwise OR of mountains they've climbed and all the mountains between them.

Bitwise OR is a binary operation which is determined the following way. Consider representation of numbers x and y in binary numeric system (probably with leading zeroes) $x = x_k \dots x_1 x_0$ and $y = y_k \dots y_1 y_0$. Then $z = x \mid y$ is defined following way: $z = z_k \dots z_1 z_0$, where $z_i = 1$, if $x_i = 1$ or $y_i = 1$, and $z_i = 0$ otherwise. In the other words, digit of bitwise OR of two numbers equals zero if and only if digits at corresponding positions is both numbers equals zero. For example bitwise OR of numbers $10 = 1010_2$ and $9 = 1001_2$ equals $11 = 1011_2$. In programming languages C/C++/Java/Python this operation is defined as «`|`», and in Pascal as «`or`».

Help Rick and Morty calculate the number of ways they can select two mountains in such a way that if they start crying from these mountains their cry will be heard above these mountains and all mountains between them. More formally you should find number of pairs l and r ($1 \le l < r \le n$) such that bitwise OR of heights of all mountains between l and r (inclusive) is larger than the height of any mountain at this interval.

Input

The first line contains integer n ($1 \le n \le 200\,000$), the number of mountains in the ridge.

Second line contains n integers a_i ($0 \le a_i \le 10^9$), the heights of mountains in order they are located in the ridge.

Output

Print the only integer, the number of ways to choose two different mountains.

Examples

input

```
5
3 2 1 6 5
```

output

```
8
```

input

```
4
3 3 3 3
```

output

```
0
```

Note

In the first test case all the ways are pairs of mountains with the numbers (numbering from one):

$$(1, 4), (1, 5), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 5)$$

In the second test case there are no such pairs because for any pair of mountains the height of cry from them is 3, and this height is equal to the height of any mountain.

E. Delivery Club

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

Petya and Vasya got employed as couriers. During the working day they are to deliver packages to n different points on the line. According to the company's internal rules, the delivery of packages must be carried out strictly in a certain order. Initially, Petya is at the point with the coordinate s_1, Vasya is at the point with the coordinate s_2, and the clients are at the points $x_1, x_2, \dots, x_n$ in the order of the required visit.

The guys agree in advance who of them will deliver the package to which of the customers, and then they act as follows. When the package for the i-th client is delivered, the one who delivers the package to the $(i + 1)$-st client is sent to the path (it can be the same person who went to the point x_i, or the other). The friend who is not busy in delivering the current package, is standing still.

To communicate with each other, the guys have got walkie-talkies. The walkie-talkies work rather poorly at great distances, so Petya and Vasya want to distribute the orders so that the maximum distance between them during the day is as low as possible. Help Petya and Vasya to minimize the maximum distance between them, observing all delivery rules.

Input

The first line contains three integers n, s_1, s_2 $(1 \le n \le 100\,000, 0 \le s_1, s_2 \le 10^9)$ — number of points of delivery and starting positions of Petya and Vasya.

The second line contains n integers $x_1, x_2, \dots, x_n$ — customers coordinates $(0 \le x_i \le 10^9)$, in the order to make a delivery.

It is guaranteed, that among the numbers $s_1, s_2, x_1, \dots, x_n$ there are no two equal.

Output

Output the only integer, minimum possible maximal distance between couriers during delivery.

Examples

input

```
2 0 10
5 6
```

output

```
10
```

input

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

output

```
1
```

input

```
1 4 5
2
```

output

```
2
```

Note

In the first test case the initial distance between the couriers is 10. This value will be the answer, for example, Petya can perform both deliveries, and Vasya will remain at the starting point.

In the second test case you can optimally act, for example, like this: Vasya delivers the package to the first customer, Petya to the second and, finally, Vasya delivers the package to the third client. With this order of delivery, the distance between the couriers will never exceed 1.

In the third test case only two variants are possible: if the delivery of a single package is carried out by Petya, the maximum distance between them will be $5 - 2 = 3$. If Vasya will deliver the package, the maximum distance is $4 - 2 = 2$. The latter method is optimal.

F. Royal Questions

time limit per test: 1.5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

In a medieval kingdom, the economic crisis is raging. Milk drops fall, Economic indicators are deteriorating every day, money from the treasury disappear. To remedy the situation, King Charles Sunnyface decided make his n sons-princes marry the brides with as big dowry as possible.

In search of candidates, the king asked neighboring kingdoms, and after a while several delegations arrived with m unmarried princesses. Receiving guests, Karl learned that the dowry of the i th princess is w_i of golden coins.

Although the action takes place in the Middle Ages, progressive ideas are widespread in society, according to which no one can force a princess to marry a prince whom she does not like. Therefore, each princess has an opportunity to choose two princes, for each of which she is ready to become a wife. The princes were less fortunate, they will obey the will of their father in the matter of choosing a bride.

Knowing the value of the dowry and the preferences of each princess, Charles wants to play weddings in such a way that the total dowry of the brides of all his sons would be as great as possible. At the same time to marry all the princes or princesses is not necessary. Each prince can marry no more than one princess, and vice versa, each princess can marry no more than one prince.

Help the king to organize the marriage of his sons in the most profitable way for the treasury.

Input

The first line contains two integers n, m ($2 \le n \le 200\ 000$, $1 \le m \le 200\ 000$) — number of princes and princesses respectively.

Each of following m lines contains three integers a_i, b_i, w_i ($1 \le a_i, b_i \le n$, $a_i \ne b_i$, $1 \le w_i \le 10\ 000$) — number of princes, which i-th princess is ready to marry and the value of her dowry.

Output

Print the only integer — the maximum number of gold coins that a king can get by playing the right weddings.

Examples

input

```
2 3
1 2 5
1 2 1
2 1 10
```

output

```
15
```

input

```
3 2
1 2 10
3 2 20
```

output

```
30
```