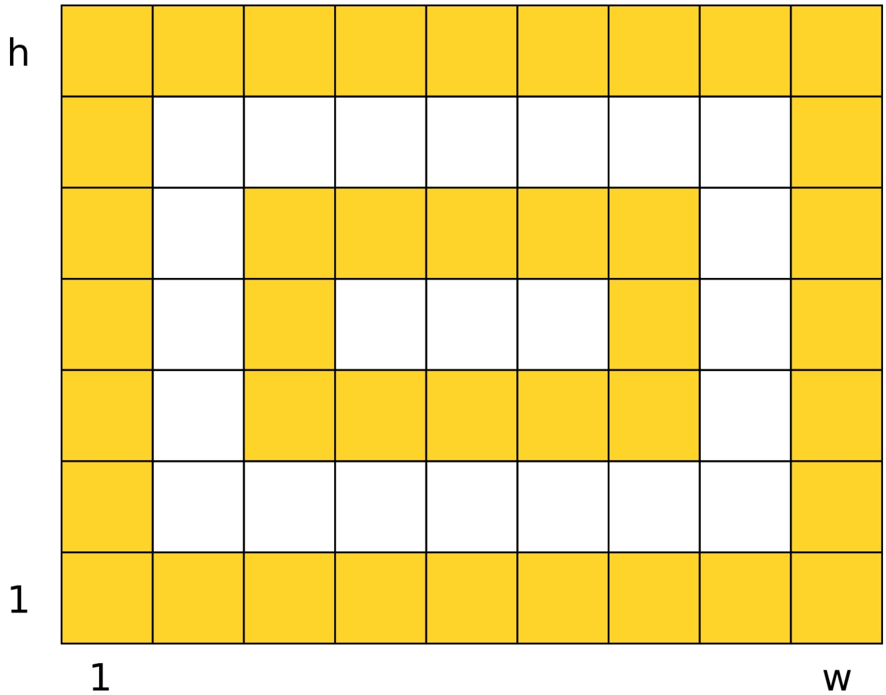
A. Golden Plate

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

You have a plate and you want to add some gilding to it. The plate is a rectangle that we split into $w \times h$ cells. There should be k gilded rings, the first one should go along the edge of the plate, the second one — 2 cells away from the edge and so on. Each ring has a width of 1 cell. Formally, the i-th of these rings should consist of all bordering cells on the inner rectangle of size $(w - 4(i-1)) \times (h - 4(i-1))$.



The picture corresponds to the third example.

Your task is to compute the number of cells to be gilded.

Input

The only line contains three integers w, h and k ($3 \le w, h \le 100$, $1 \le k \le \left\lfloor \frac{min(n,m)+1}{4} \right\rfloor$, where $\lfloor x \rfloor$ denotes the number x rounded down) — the number of rows, columns and the number of rings, respectively.

Output

Print a single positive integer — the number of cells to be gilded.

Examples

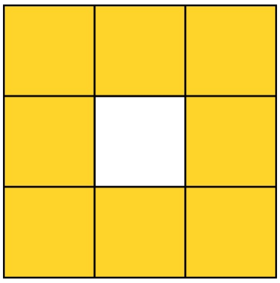
input
```
3 3 1
```
output
```
8
```

input
```
7 9 1
```
output
```
28
```

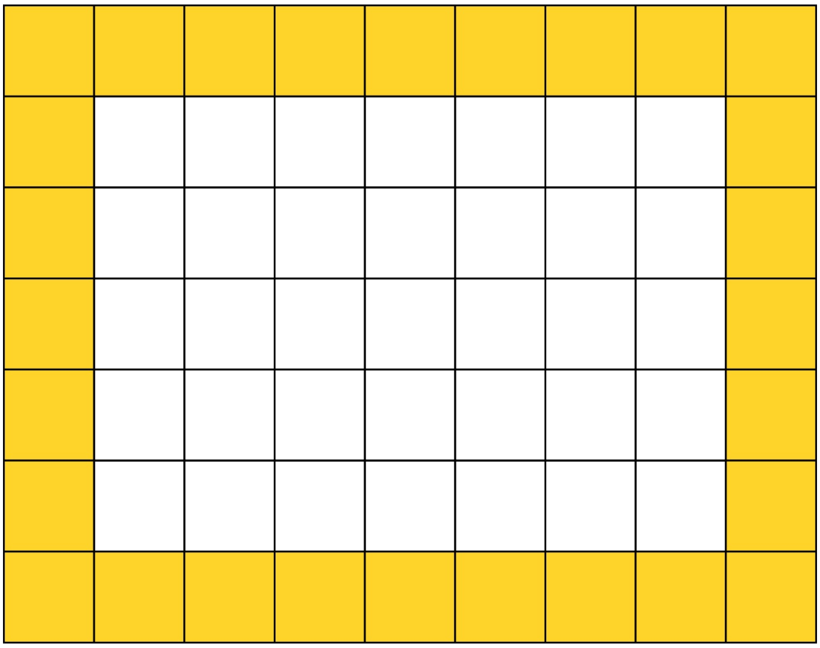
input
```
7 9 2
```
output
```
40
```

Note

The first example is shown on the picture below.



The second example is shown on the picture below.



The third example is shown in the problem description.

B. Curiosity Has No Limits

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

When Masha came to math classes today, she saw two integer sequences of length $n-1$ on the blackboard. Let's denote the elements of the first sequence as a_i ($0 \le a_i \le 3$), and the elements of the second sequence as b_i ($0 \le b_i \le 3$).

Masha became interested if or not there is an integer sequence of length n, which elements we will denote as t_i ($0 \le t_i \le 3$), so that for every i ($1 \le i \le n-1$) the following is true:

- $a_i = t_i | t_{i+1}$ (where $|$ denotes the bitwise OR operation) and
- $b_i = t_i \& t_{i+1}$ (where $\&$ denotes the bitwise AND operation).

The question appeared to be too difficult for Masha, so now she asked you to check whether such a sequence t_i of length n exists. If it exists, find such a sequence. If there are multiple such sequences, find any of them.

Input

The first line contains a single integer n ($2 \le n \le 10^5$) — the length of the sequence t_i.

The second line contains $n-1$ integers $a_1, a_2, \ldots, a_{n-1}$ ($0 \le a_i \le 3$) — the first sequence on the blackboard.

The third line contains $n-1$ integers $b_1, b_2, \ldots, b_{n-1}$ ($0 \le b_i \le 3$) — the second sequence on the blackboard.

Output

In the first line print "YES" (without quotes), if there is a sequence t_i that satisfies the conditions from the statements, and "NO" (without quotes), if there is no such sequence.

If there is such a sequence, on the second line print n integers $t_1, t_2, \ldots, t_n$ ($0 \le t_i \le 3$) — the sequence that satisfies the statements conditions.

If there are multiple answers, print any of them.

Examples

input

```
4
3 3 2
```

```
1 2 0
```

output

```
YES
1 3 2 0
```

input

```
3
1 3
3 2
```

output

```
NO
```

Note

In the first example it's easy to see that the sequence from output satisfies the given conditions:

- $t_1|t_2 = (01_2)|(11_2) = (11_2) = 3 = a_1$ and $t_1\&t_2 = (01_2)\&(11_2) = (01_2) = 1 = b_1$;
- $t_2|t_3 = (11_2)|(10_2) = (11_2) = 3 = a_2$ and $t_2\&t_3 = (11_2)\&(10_2) = (10_2) = 2 = b_2$;
- $t_3|t_4 = (10_2)|(00_2) = (10_2) = 2 = a_3$ and $t_3\&t_4 = (10_2)\&(00_2) = (00_2) = 0 = b_3$.

In the second example there is no such sequence.

C. Cram Time

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

In a galaxy far, far away Lesha the student has just got to know that he has an exam in two days. As always, he hasn't attended any single class during the previous year, so he decided to spend the remaining time wisely.

Lesha knows that today he can study for at most a hours, and he will have b hours to study tomorrow. Note that it is possible that on his planet there are more hours in a day than on Earth. Lesha knows that the quality of his knowledge will only depend on the number of lecture notes he will read. He has access to an infinite number of notes that are enumerated with positive integers, but he knows that he can read the first note in one hour, the second note in two hours and so on. In other words, Lesha can read the note with number k in k hours. Lesha can read the notes in arbitrary order, however, he can't start reading a note in the first day and finish its reading in the second day.

Thus, the student has to fully read several lecture notes today, spending at most a hours in total, and fully read several lecture notes tomorrow, spending at most b hours in total. What is the maximum number of notes Lesha can read in the remaining time? Which notes should he read in the first day, and which — in the second?

Input

The only line of input contains two integers a and b ($0 \le a, b \le 10^9$) — the number of hours Lesha has today and the number of hours Lesha has tomorrow.

Output

In the first line print a single integer n ($0 \le n \le a$) — the number of lecture notes Lesha has to read in the first day. In the second line print n distinct integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le a$), the sum of all p_i should not exceed a.

In the third line print a single integer m ($0 \le m \le b$) — the number of lecture notes Lesha has to read in the second day. In the fourth line print m distinct integers $q_1, q_2, \ldots, q_m$ ($1 \le q_i \le b$), the sum of all q_i should not exceed b.

All integers p_i and q_i should be distinct. The sum $n + m$ should be largest possible.

Examples

input

```
3 3
```

output

```
1
3
2
2 1
```

input

```
9 12
```

output

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

Note

In the first example Lesha can read the third note in 3 hours in the first day, and the first and the second notes in one and two hours correspondingly in the second day, spending 3 hours as well. Note that Lesha can make it the other way round, reading the first and the second notes in the first day and the third note in the second day.

In the second example Lesha should read the third and the sixth notes in the first day, spending 9 hours in total. In the second day Lesha should read the first, second fourth and fifth notes, spending 12 hours in total.

D. Minimum path

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a matrix of size $n \times n$ filled with lowercase English letters. You can change no more than k letters in this matrix.

Consider all paths from the upper left corner to the lower right corner that move from a cell to its neighboring cell to the right or down. Each path is associated with the string that is formed by all the letters in the cells the path visits. Thus, the length of each string is $2n - 1$.

Find the lexicographically smallest string that can be associated with a path after changing letters in at most k cells of the matrix.

A string a is lexicographically smaller than a string b, if the first different letter in a and b is smaller in a.

Input

The first line contains two integers n and k ($1 \le n \le 2000$, $0 \le k \le n^2$) — the size of the matrix and the number of letters you can change.

Each of the next n lines contains a string of n lowercase English letters denoting one row of the matrix.

Output

Output the lexicographically smallest string that can be associated with some valid path after changing no more than k letters in the matrix.

Examples

input

```
4 2
abcd
bcde
bcad
bcde
```

output

```
aaabcde
```

input

```
5 3
bwwwz
hrhdh
sepsp
sqfaf
ajbvw
```

output

```
aaaepfafw
```

input

```
7 6
ypnxnnp
pnxonpm
nxanpou
xnnpmud
nhtdudu
npmuduh
pmutsnz
```

output

```
aaaaaaadudsnz
```

Note

In the first sample test case it is possible to change letters 'b' in cells $(2, 1)$ and $(3, 1)$ to 'a', then the minimum path contains cells $(1, 1), (2, 1), (3, 1), (4, 1), (4, 2), (4, 3), (4, 4)$. The first coordinate corresponds to the row and the second coordinate corresponds to the column.

E. Triple Flips

time limit per test: 1 second
memory limit per test: 256 megabytes

input: standard input
output: standard output

You are given an array a of length n that consists of zeros and ones.

You can perform the following operation multiple times. The operation consists of two steps:

1. Choose three integers $1 \le x < y < z \le n$, that form an arithmetic progression ($y - x = z - y$).
2. Flip the values a_x, a_y, a_z (i.e. change 1 to 0, change 0 to 1).

Determine if it is possible to make all elements of the array equal to zero. If yes, print the operations that lead the the all-zero state. Your solution should not contain more than $(\lfloor \frac{n}{3} \rfloor + 12)$ operations. Here $\lfloor q \rfloor$ denotes the number q rounded down. We can show that it is possible to make all elements equal to zero in no more than this number of operations whenever it is possible to do so at all.

Input

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

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 1$) — the elements of the array.

Output

Print "YES" (without quotes) if the answer exists, otherwise print "NO" (without quotes). You can print each letter in any case (upper or lower).

If there is an answer, in the second line print an integer m ($0 \le m \le (\lfloor \frac{n}{3} \rfloor + 12)$) — the number of operations in your answer.

After that in ($i + 2$)-th line print the i-th operations — the integers x_i, y_i, z_i. You can print them in arbitrary order.

Examples

input

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

output

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

input

```
3
0 1 0
```

output

```
NO
```

Note

In the first sample the shown output corresponds to the following solution:

- `1 1 0 1 1` (initial state);
- `0 1 1 1 0` (the flipped positions are the first, the third and the fifth elements);
- `0 0 0 0 0` (the flipped positions are the second, the third and the fourth elements).

Other answers are also possible. In this test the number of operations should not exceed $\lfloor \frac{5}{3} \rfloor + 12 = 1 + 12 = 13$.

In the second sample the only available operation is to flip all the elements. This way it is only possible to obtain the arrays `0 1 0` and `1 0 1`, but it is impossible to make all elements equal to zero.

F. Familiar Operations

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

You are given two positive integers a and b. There are two possible operations:

1. multiply one of the numbers by some prime p;
2. divide one of the numbers on its prime factor p.

What is the minimum number of operations required to obtain two integers having the same number of divisors? You are given several such pairs, you need to find the answer for each of them.

Input

The first line contains a single integer t ($1 \le t \le 10^5$) — the number of pairs of integers for which you are to find the answer.

Each of the next t lines contain two integers a_i and b_i ($1 \le a_i, b_i \le 10^6$).

Output

Output t lines — the i-th of them should contain the answer for the pair a_i, b_i.

Example

input

```
8
9 10
100 17
220 70
17 19
4 18
32 20
100 32
224 385
```

output

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

Note

These are the numbers with equal number of divisors, which are optimal to obtain in the sample test case:

- $(27, 10)$, 4 divisors
- $(100, 1156)$, 9 divisors
- $(220, 140)$, 12 divisors
- $(17, 19)$, 2 divisors
- $(12, 18)$, 6 divisors
- $(50, 32)$, 6 divisors
- $(224, 1925)$, 12 divisors

Note that there can be several optimal pairs of numbers.