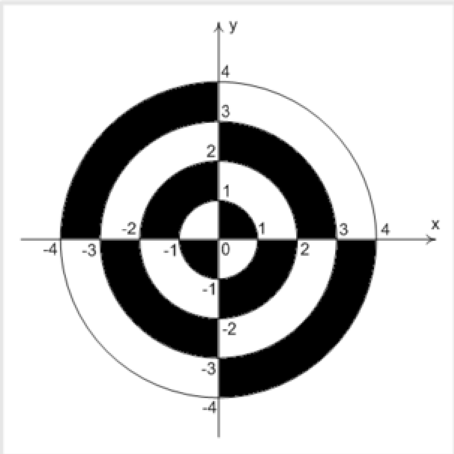
# A. Find Color

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

Not so long ago as a result of combat operations the main Berland place of interest — the magic clock — was damaged. The cannon's balls made several holes in the clock, that's why the residents are concerned about the repair. The magic clock can be represented as an **infinite** Cartesian plane, where the origin corresponds to the clock center. The clock was painted two colors as is shown in the picture:




The picture shows only the central part of the clock. This coloring naturally extends to infinity.

The balls can be taken to be points on the plane. Your task is to find the color of the area, damaged by the given ball.

All the points located on the border of one of the areas have to be considered painted black.

### Input

The first and single line contains two integers $x$ and $y$ — the coordinates of the hole made in the clock by the ball. Each of the numbers $x$ and $y$ has an absolute value that does not exceed $1000$.

### Output

Find the required color.

All the points between which and the origin of coordinates the distance is integral-value are painted black.

### Sample test(s)

input
```
-2 1
```
output
```
white
```

input
```
2 1
```
output
```
black
```

input
```
4 3
```
output
```
black
```

# B. Repaintings

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

A chessboard $n \times m$ in size is given. During the zero minute we repaint all the black squares to the 0 color. During the $i$-th minute we repaint to the $i$ color the **initially black** squares that have exactly four corner-adjacent squares painted $i - 1$ (all such squares are repainted simultaneously). This process continues ad infinitum. You have to figure out how many squares we repainted exactly $x$ times.

The upper left square of the board has to be assumed to be always black. Two squares are called corner-adjacent, if they have exactly one common point.

### Input

The first line contains integers $n$ and $m$ ($1 \le n, m \le 5000$). The second line contains integer $x$ ($1 \le x \le 10^9$).

### Output

Print how many squares will be painted exactly $x$ times.

**Sample test(s)**

input

```
3 3
1
```

output

```
4
```

input

```
3 3
2
```

output

```
1
```

input

```
1 1
1
```

output

```
1
```

# C. Berland Square

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

Last year the world's largest square was built in Berland. It is known that the square can be represented as an infinite plane with an introduced Cartesian system of coordinates. On that square two sets of concentric circles were painted. Let's call the set of concentric circles with radii $1, 2, ..., K$ and the center in the point $(z, 0)$ a $(K, z)$-set. Thus, on the square were painted a $(N, x)$-set and a $(M, y)$-set. You have to find out how many parts those sets divided the square into.

### Input

The first line contains integers $N, x, M, y$. ($1 \le N, M \le 100000$, $-100000 \le x, y \le 100000$, $x \ne y$).

### Output

Print the sought number of parts.

**Sample test(s)**

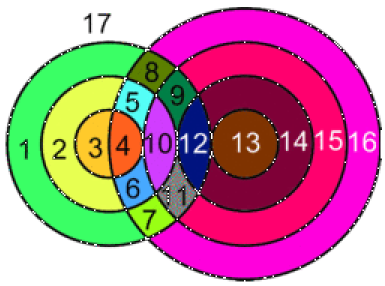
input
```
1 0 1 1
```
output
```
4
```

input
```
1 0 1 2
```
output
```
3
```

input
```
3 3 4 7
```
output
```
17
```

### Note

Picture for the third sample:



# D. Interesting Sequence

time limit per test: 3 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

Berland scientists noticed long ago that the world around them depends on Berland population. Due to persistent research in this area the scientists managed to find out that the Berland chronology starts from the moment when the first two people came to that land (it is considered to have happened in the first year). After one Berland year after the start of the chronology the population had already equaled 13 people (the second year). However, tracing the population number during the following years was an ultimately difficult task, still it was found out that if $d_i$ — the number of people in Berland in the year of $i$, then either $d_i = 12d_{i-2}$, or $d_i = 13d_{i-1} - 12d_{i-2}$. Of course no one knows how many people are living in Berland at the moment, but now we can tell if there could possibly be a year in which the country population equaled $A$. That's what we ask you to determine. Also, if possible, you have to find out in which years it could be (from the beginning of Berland chronology). Let's suppose that it could be in the years of $a_1, a_2, ..., a_k$. Then you have to define how many residents could be in the country during those years apart from the $A$ variant. Look at the examples for further explanation.

### Input

The first line contains integer $A$ $(1 \le A < 10^{300})$. It is guaranteed that the number doesn't contain leading zeros.

### Output

On the first output line print `YES`, if there could be a year in which the total population of the country equaled $A$, otherwise print `NO`.

If the answer is `YES`, then you also have to print number $k$ — the number of years in which the population could equal $A$. On the next line you have to output precisely $k$ space-separated numbers — $a_1, a_2, ..., a_k$. Those numbers have to be output in the increasing order.

On the next line you should output number $p$ — how many variants of the number of people could be in the years of $a_1, a_2, ..., a_k$, apart from the $A$ variant. On each of the next $p$ lines you have to print one number — the sought number of residents. Those number also have to go in the increasing order.

If any number (or both of them) $k$ or $p$ exceeds $1000$, then you have to print $1000$ instead of it and only the first $1000$ possible answers in the increasing order.

The numbers should have no leading zeros.

**Sample test(s)**

input
```
2
```
output
```
YES
1
1
0
```

input
```
3
```
output
```
NO
```

input
```
13
```
output
```
YES
1
2
0
```

input
```
1729
```
output
```
YES
1
4
1
156
```

# E. Number Table

time limit per test: 2 seconds
memory limit per test: 216 megabytes
input: standard input
output: standard output

As it has been found out recently, all the Berland's current economical state can be described using a simple table $n \times m$ in size. $n$ — the number of days in each Berland month, $m$ — the number of months. Thus, a table cell corresponds to a day and a month of the Berland's year. Each cell will contain either `1`, or `-1`, which means the state's gains in a particular month, on a particular day. `1` corresponds to profits, `-1` corresponds to losses. It turned out important for successful development to analyze the data on the state of the economy of the previous year, however when the treasurers referred to the archives to retrieve the data, it turned out that the table had been substantially damaged. In some table cells the number values had faded and were impossible to be deciphered. It is known that the number of cells in which the data had been preserved is strictly less than $max(n, m)$. However, there is additional information — the product of the numbers in each line and column equaled `-1`. Your task is to find out how many different tables may conform to the preserved data. As the answer to the task can be quite large, you have to find it modulo $p$.

### Input

The first line contains integers $n$ and $m$ ($1 \le n, m \le 1000$). The second line contains the integer $k$ ($0 \le k < max(n, m)$) — the number of cells in which the data had been preserved. The next $k$ lines contain the data on the state of the table in the preserved cells. Each line is of the form "$a$ $b$ $c$", where $a$ ($1 \le a \le n$) — the number of the table row, $b$ ($1 \le b \le m$) — the number of the column, $c$ — the value containing in the cell (`1` or `-1`). They are numbered starting from $1$. It is guaranteed that no two lines with same $a$ and $b$ values exist. The last line contains an integer $p$ ($2 \le p \le 10^9 + 7$).

### Output

Print the number of different tables that could conform to the preserved data modulo $p$.

#### Sample test(s)

input

```
2 2
0
100
```

output

```
2
```

input

```
2 2
1
1 1 -1
100
```

output

```
1
```