Educational Codeforces Round 63 (Rated for Div. 2)

A. Reverse a Substring

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

You are given a string s consisting of n lowercase Latin letters.

Let's define a substring as a contiguous subsegment of a string. For example, "acab" is a substring of "abacaba" (it starts in position 3 and ends in position 6), but "aa" or "d" aren't substrings of this string. So the substring of the string s from position l to position r is $s[l; r] = s_l s_{l+1} \dots s_r$.

You have to choose **exactly** one of the substrings of the given string and reverse it (i. e. make $s[l; r] = s_r s_{r-1} \dots s_l$) to obtain a string that is **less** lexicographically. Note that it **is not necessary** to obtain the minimum possible string.

If it is impossible to reverse some substring of the given string to obtain a string that is less, print "NO". Otherwise print "YES" and **any** suitable substring.

String x is lexicographically less than string y, if either x is a prefix of y (and $x \ne y$), or there exists such i ($1 \le i \le min(|x|, |y|)$), that $x_i < y_i$, and for any j ($1 \le j < i$) $x_j = y_j$. Here $|a|$ denotes the length of the string a. The lexicographic comparison of strings is implemented by operator < in modern programming languages.

Input

The first line of the input contains one integer n ($2 \le n \le 3 \cdot 10^5$) — the length of s.

The second line of the input contains the string s of length n consisting only of lowercase Latin letters.

Output

If it is impossible to reverse some substring of the given string to obtain a string which is lexicographically **less**, print "NO". Otherwise print "YES" and two indices l and r ($1 \le l < r \le n$) denoting the substring you have to reverse. If there are multiple answers, you can print any.

Examples

input

```
7
abacaba
```

output

```
YES
2 5
```

input

```
6
aabcfg
```

output

```
NO
```

Note

In the first testcase the resulting string is "aacabba".

B. Game with Telephone Numbers

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

A telephone number is a sequence of **exactly** 11 digits such that its first digit is 8.

Vasya and Petya are playing a game. Initially they have a string s of length n (n is odd) consisting of digits. Vasya makes the first move, then players alternate turns. In one move the player **must** choose a character and erase it from the current string. For example, if the current string 1121, after the player's move it may be 112, 111 or 121. The game ends when the length of string s becomes 11. If the resulting string is a telephone number, Vasya wins, otherwise Petya wins.

You have to determine if Vasya has a winning strategy (that is, if Vasya can win the game no matter which characters Petya chooses during his moves).

Input

The first line contains one integer n ($13 \le n < 10^5$, n is odd) — the length of string s.

The second line contains the string s ($|s| = n$) consisting only of decimal digits.

Output

If Vasya has a strategy that guarantees him victory, print YES.

Otherwise print NO.

Examples

input

```
13
8380011223344
```

output

```
YES
```

input

```
15
807345619350641
```

output

```
NO
```

Note

In the first example Vasya needs to erase the second character. Then Petya cannot erase a character from the remaining string 880011223344 so that it does not become a telephone number.

In the second example after Vasya's turn Petya can erase one character character 8. The resulting string can't be a telephone number, because there is no digit 8 at all.

C. Alarm Clocks Everywhere

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Ivan is going to sleep now and wants to set his alarm clock. There will be many necessary events tomorrow, the i-th of them will start during the x_i-th minute. Ivan doesn't want to skip any of the events, so he has to set his alarm clock in such a way that it rings during minutes $x_1, x_2, \dots, x_n$, so he will be awake during each of these minutes (**note that it does not matter if his alarm clock will ring during any other minute**).

Ivan can choose two properties for the alarm clock — the first minute it will ring (let's denote it as y) and the interval between two consecutive signals (let's denote it by p). After the clock is set, it will ring during minutes $y, y+p, y+2p, y+3p$ and so on.

Ivan can choose **any** minute as the first one, but he cannot choose any arbitrary value of p. He has to pick it among the given values $p_1, p_2, \dots, p_m$ (his phone does not support any other options for this setting).

So Ivan has to choose the first minute y when the alarm clock should start ringing and the interval between two consecutive signals p_j in such a way that it will ring during all given minutes $x_1, x_2, \dots, x_n$ (and it does not matter if his alarm clock will ring in any other minutes).

Your task is to tell the first minute y and the index j such that if Ivan sets his alarm clock with properties y and p_j it will ring during all given minutes $x_1, x_2, \dots, x_n$ or say that it is impossible to choose such values of the given properties. If there are multiple answers, you can print any.

Input

The first line of the input contains two integers n and m ($2 \le n \le 3 \cdot 10^5, 1 \le m \le 3 \cdot 10^5$) — the number of events and the number of possible settings for the interval between signals.

The second line of the input contains n integers $x_1, x_2, \dots, x_n$ ($1 \le x_i \le 10^{18}$), where x_i is the minute when i-th event starts. It is guaranteed that all x_i are given in increasing order (i. e. the condition $x_1 < x_2 < \dots < x_n$ holds).

The third line of the input contains m integers $p_1, p_2, \dots, p_m$ ($1 \le p_j \le 10^{18}$), where p_j is the j-th option for the interval between two consecutive signals.

Output

If it's impossible to choose such values y and j so all constraints are satisfied, print "NO" in the first line.

Otherwise print "YES" in the first line. Then print two integers y ($1 \le y \le 10^{18}$) and j ($1 \le j \le m$) in the second line, where y is the first minute Ivan's alarm clock should start ringing and j is the index of the option for the interval between two consecutive signals (options are numbered from 1 to m in the order they are given input). These values should be chosen in such a way that the alarm clock will ring during all given minutes $x_1, x_2, \dots, x_n$. If there are multiple answers, you can print any.

Examples

input

```
3 5
3 12 18
2 6 5 3 3
```

output

```
YES
3 4
```

input

```
4 2
1 5 17 19
4 5
```

output

```
NO
```

input

```
4 2
1 5 17 19
2 1
```

output

```
YES
1 1
```

D. Beautiful Array

time limit per test: 2 seconds

memory limit per test: 256 megabytes

input: standard input

output: standard output

You are given an array a consisting of n integers. Beauty of array is the maximum sum of some **consecutive subarray** of this array (this subarray may be empty). For example, the beauty of the array `[10, -5, 10, -4, 1]` is 15, and the beauty of the array `[-3, -5, -1]` is `0`.

You may choose **at most one consecutive subarray** of a and multiply all values contained in this subarray by x. You want to maximize the beauty of array after applying at most one such operation.

Input

The first line contains two integers n and x ($1 \le n \le 3 \cdot 10^5, -100 \le x \le 100$) — the length of array a and the integer x respectively.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($-10^9 \le a_i \le 10^9$) — the array a.

Output

Print one integer — the maximum possible beauty of array a after multiplying all values belonging to some consecutive subarray x.

Examples

input

```
5 -2
-3 8 -2 1 -6
```

output

```
22
```

input

```
12 -3
1 3 3 7 1 3 3 7 1 3 3 7
```

output

```
42
```

input

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

output

```
0
```

Note

In the first test case we need to multiply the subarray `[-2, 1, -6]`, and the array becomes `[-3, 8, 4, -2, 12]` with beauty 22 (`[-3, `**`8, 4, -2, 12`**`]`).

In the second test case we don't need to multiply any subarray at all.

In the third test case no matter which subarray we multiply, the beauty of array will be equal to `0`.

E. Guess the Root

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

Jury picked a polynomial $f(x) = a_0 + a_1 \cdot x + a_2 \cdot x^2 + \dots + a_k \cdot x^k$. $k \le 10$ and all a_i are integer numbers and $0 \le a_i < 10^6 + 3$. It's guaranteed that there is at least one i such that $a_i > 0$.

Now jury wants you to find such an integer x_0 that $f(x_0) \equiv 0 \mod (10^6 + 3)$ or report that there is not such x_0.

You can ask no more than 50 queries: you ask value x_q and jury tells you value $f(x_q) \mod (10^6 + 3)$.

Note that printing the answer doesn't count as a query.

Interaction

To ask a question, print "? x_q" $(0 \le x_q < 10^6 + 3)$. The judge will respond with a single integer $f(x_q) \mod (10^6 + 3)$. If you ever get a result of -1 (because you printed an invalid query), exit immediately to avoid getting other verdicts.

After printing a query do not forget to output end of line and flush the output. Otherwise, you will get `Idleness limit exceeded`. To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `flush(output)` in Pascal;
- `stdout.flush()` in Python;
- see documentation for other languages.

When you are ready to answer, print "! x_0" where x_0 is the answer or -1 if there is no such x_0.

You can ask at most 50 questions per test case.

Hack Format

To hack, use the following format.

The only line should contain 11 integers $a_0, a_1, \dots, a_{10}$ $(0 \le a_i < 10^6 + 3, \max\limits_{0 \le i \le 10} a_i > 0)$ — corresponding coefficients of the polynomial.

Examples

input

```
1000002

0
```

output

```
? 0

? 1

! 1
```

input

```
5

2

1
```

output

```
? 2

? 1

? 0

! -1
```

Note

The polynomial in the first sample is $1000002 + x^2$.

The polynomial in the second sample is $1 + x^2$.

F. Delivery Oligopoly

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

The whole delivery market of Berland is controlled by two rival companies: BerEx and BerPS. They both provide fast and reliable delivery services across all the cities of Berland.

The map of Berland can be represented as an **undirected** graph. The cities are vertices and the roads are edges between them. Each pair of cities has no more than one road between them. Each road connects different cities.

BerEx and BerPS are so competitive that for each pair of cities (v, u) they have set up their paths from v to u in such a way that **these two paths don't share a single road**. It is guaranteed that it was possible.

Now Berland government decided to cut down the road maintenance cost by abandoning some roads. Obviously, they want to maintain as little roads as possible. However, they don't want to break the entire delivery system. So BerEx and BerPS should still be able to have their paths between every pair of cities non-intersecting.

What is the minimal number of roads Berland government can maintain?

More formally, given a 2-edge connected undirected graph, what is the minimum number of edges that can be left in it so that the resulting graph is also 2-edge connected?

Input

The first line contains two integers n and m ($3 \le n \le 14$, $n \le m \le \frac{n(n-1)}{2}$) — the number of cities and the number of roads between them.

Each of the next m lines contains two integers v and u ($1 \le v, u \le n$, $v \ne u$) — the cities connected by the next road.

It is guaranteed that each pair of cities has no more than one road between them. It is guaranteed that each pair of cities have at least two paths between them that don't share a single road.

Output

The first line should contain a single integer k — the minimum number of roads Berland government can maintain so that BerEx and BerPS are still able to have their paths between every pair of cities non-intersecting.

The next k lines should contain the list of roads which are being maintained. Each line of form "v u", where v and u are cities connected by the next road.

If there are multiple lists of minimum size, print any of them. The order of roads in the list doesn't matter.

Examples

input

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

output

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

input

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

output

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

input

```
6 10
1 2
2 3
3 1
3 4
```

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

output

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

Note

Here are graphs from the examples, red edges are the maintained ones.