A. IQ Test

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

In the city of Ultima Thule job applicants are often offered an IQ test.

The test is as follows: the person gets a piece of squared paper with a 4×4 square painted on it. Some of the square's cells are painted black and others are painted white. Your task is to repaint **at most one** cell the other color so that the picture has a 2×2 square, completely consisting of cells of the same color. If the initial picture already has such a square, the person should just say so and the test will be completed.

Your task is to write a program that determines whether it is possible to pass the test. You cannot pass the test if either repainting any cell or no action doesn't result in a 2×2 square, consisting of cells of the same color.

Input

Four lines contain four characters each: the j-th character of the i-th line equals "." if the cell in the i-th row and the j-th column of the square is painted white, and "#", if the cell is black.

Output

Print "YES" (without the quotes), if the test can be passed and "NO" (without the quotes) otherwise.

Examples

input

```
####
.#..
####
....
```

output

```
YES
```

input

```
####
....
####
....
```

output

```
NO
```

Note

In the first test sample it is enough to repaint the first cell in the second row. After such repainting the required 2×2 square is on the intersection of the 1-st and 2-nd row with the 1-st and 2-nd column.

B. Pipeline

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

Vova, the Ultimate Thule new shaman, wants to build a pipeline. As there are exactly n houses in Ultimate Thule, Vova wants the city to have exactly n pipes, each such pipe should be connected to the water supply. A pipe can be connected to the water supply if there's water flowing out of it. Initially Vova has only one pipe with flowing water. Besides, Vova has several splitters.

A splitter is a construction that consists of one input (it can be connected to a water pipe) and x output pipes. When a splitter is connected to a water pipe, water flows from each output pipe. You can assume that the output pipes are ordinary pipes. For example, you can connect water supply to such pipe if there's water flowing out from it. At most one splitter can be connected to any water pipe.

The figure shows a 4-output splitter

Vova has one splitter of each kind: with 2, 3, 4, ..., k outputs. Help Vova use the minimum number of splitters to build the required pipeline or otherwise state that it's impossible.

Vova needs the pipeline to have exactly n pipes with flowing out water. Note that some of those pipes can be the output pipes of the splitters.

Input

The first line contains two space-separated integers n and k ($1 \le n \le 10^{18}$, $2 \le k \le 10^9$).

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use the `cin`, `cout` streams or the `%I64d` specifier.

Output

Print a single integer — the minimum number of splitters needed to build the pipeline. If it is impossible to build a pipeline with the given splitters, print -1.

Examples

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

input
```
5 5
```
output
```
1
```

input
```
8 4
```
output
```
-1
```

C. Lucky Permutation

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

A permutation p of size n is the sequence $p_1, p_2, \ldots, p_n$, consisting of n distinct integers, each of them is from 1 to n $(1 \le p_i \le n)$.

A lucky permutation is such permutation p, that any integer i $(1 \le i \le n)$ meets this condition $p_{p_i} = n - i + 1$.

You have integer n. Find some lucky permutation p of size n.

Input

The first line contains integer n $(1 \le n \le 10^5)$ — the required permutation size.

Output

Print "-1" (without the quotes) if the lucky permutation p of size n doesn't exist.

Otherwise, print n distinct integers $p_1, p_2, \ldots, p_n$ $(1 \le p_i \le n)$ after a space — the required permutation.

If there are multiple answers, you can print any of them.

Examples

input
```
1
```
output
```
1
```

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

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

input
```
5
```
output
```
2 5 3 1 4
```

D. Shifting

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

John Doe has found the beautiful permutation formula.

Let's take permutation $p = p_1, p_2, \ldots, p_n$. Let's define transformation f of this permutation:

$$f(p,k) = \overbrace{p_2, p_3, \ldots, p_k, p_1}, \overbrace{p_{k+2}, p_{k+3}, \ldots, p_{2k}, p_{k+1}}, \ldots, \overbrace{p_{rk+2}, p_{rk+3}, \ldots, p_n, p_{rk+1}};$$

where k $(k > 1)$ is an integer, the transformation parameter, r is such maximum integer that $rk \le n$. If $rk = n$, then elements p_{rk+1}, p_{rk+2} and so on are omitted. In other words, the described transformation of permutation p cyclically shifts to the left each consecutive block of length k and the last block with the length equal to the remainder after dividing n by k.

John Doe thinks that permutation $f(f(\ldots f(p = [1, 2, \ldots, n], 2) \ldots , n - 1), n)$ is beautiful. Unfortunately, he cannot quickly find the beautiful permutation he's interested in. That's why he asked you to help him.

Your task is to find a beautiful permutation for the given n. For clarifications, see the notes to the third sample.

Input

A single line contains integer n $(2 \le n \le 10^6)$.

Output

Print n distinct space-separated integers from 1 to n — a beautiful permutation of size n.

Examples

input

```
2
```

output

```
2 1
```

input

```
3
```

output

```
1 3 2
```

input

```
4
```

output

```
4 2 3 1
```

Note

A note to the third test sample:

- $f([1, 2, 3, 4], 2) = [2, 1, 4, 3]$
- $f([2, 1, 4, 3], 3) = [1, 4, 2, 3]$
- $f([1, 4, 2, 3], 4) = [4, 2, 3, 1]$

E. Main Sequence

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

As you know, Vova has recently become a new shaman in the city of Ultima Thule. So, he has received the shaman knowledge about the correct bracket sequences. The shamans of Ultima Thule have been using lots of different types of brackets since prehistoric times. A bracket type is a positive integer. The shamans define a correct bracket sequence as follows:

- An empty sequence is a correct bracket sequence.
- If $\{a_1, a_2, ..., a_l\}$ and $\{b_1, b_2, ..., b_k\}$ are correct bracket sequences, then sequence $\{a_1, a_2, ..., a_l, b_1, b_2, ..., b_k\}$ (their concatenation) also is a correct bracket sequence.
- If $\{a_1, a_2, ..., a_l\}$ — is a correct bracket sequence, then sequence $\{v, a_1, a_2, \dots, a_l, -v\}$ also is a correct bracket sequence, where v $(v > 0)$ is an integer.

For example, sequences $\{1, 1, -1, 2, -2, -1\}$ and $\{3, -3\}$ are correct bracket sequences, and $\{2, -3\}$ is not.

Moreover, after Vova became a shaman, he learned the *most important* correct bracket sequence $\{x_1, x_2, ..., x_n\}$, consisting of n integers. As sequence x is the most important, Vova decided to encrypt it just in case.

Encrypting consists of two sequences. The first sequence $\{p_1, p_2, ..., p_n\}$ contains types of brackets, that is, $p_i = |x_i|$ $(1 \le i \le n)$. The second sequence $\{q_1, q_2, ..., q_t\}$ contains t integers — **some positions** (possibly, not all of them), which had negative numbers in sequence $\{x_1, x_2, ..., x_n\}$.

Unfortunately, Vova forgot the main sequence. But he was lucky enough to keep the encryption: sequences $\{p_1, p_2, ..., p_n\}$ and $\{q_1, q_2, ..., q_t\}$. Help Vova restore sequence x by the encryption. If there are multiple sequences that correspond to the encryption, restore any of them. If there are no such sequences, you should tell so.

Input

The first line of the input contains integer n $(1 \le n \le 10^6)$. The second line contains n integers: $p_1, p_2, ..., p_n$ $(1 \le p_i \le 10^9)$.

The third line contains integer t $(0 \le t \le n)$, followed by t distinct integers $q_1, q_2, ..., q_t$ $(1 \le q_i \le n)$.

The numbers in each line are separated by spaces.

Output

Print a single string "NO" (without the quotes) if Vova is mistaken and a suitable sequence $\{x_1, x_2, ..., x_n\}$ doesn't exist.

Otherwise, in the first line print "YES" (without the quotes) and in the second line print n integers $x_1, x_2, ..., x_n$ $(|x_i| = p_i; x_{q_j} < 0)$. If there are multiple sequences that correspond to the encrypting, you are allowed to print any of them.

Examples

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

input
```
4
1 1 1 1
1 3
```
output
```
YES
1 1 -1 -1
```

input
```
3
1 1 1
0
```
output
```
NO
```

input

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

output

```
YES
1 2 -2 -1
```