A. Giga Tower

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

Giga Tower is the tallest and deepest building in Cyberland. There are $17\,777\,777\,777$ floors, numbered from $-8\,888\,888\,888$ to $8\,888\,888\,888$. In particular, there is floor 0 between floor -1 and floor 1. Every day, thousands of tourists come to this place to enjoy the wonderful view.

In Cyberland, it is believed that the number "8" is a lucky number (that's why Giga Tower has $8\,888\,888\,888$ floors above the ground), and, an integer is *lucky*, if and only if its decimal notation contains at least one digit "8". For example, $8, -180, 808$ are all *lucky* while $42, -10$ are not. In the Giga Tower, if you write code at a floor with lucky floor number, good luck will always be with you (Well, this round is #278, also lucky, huh?).

Tourist Henry goes to the tower to seek good luck. Now he is at the floor numbered a. He wants to find the minimum **positive** integer b, such that, if he walks b floors higher, he will arrive at a floor with a **lucky** number.

Input

The only line of input contains an integer a ($-10^9 \le a \le 10^9$).

Output

Print the minimum b in a line.

Sample test(s)

input
```
179
```
output
```
1
```

input
```
-1
```
output
```
9
```

input
```
18
```
output
```
10
```

Note

For the first sample, he has to arrive at the floor numbered 180.

For the second sample, he will arrive at 8.

Note that b should be positive, so the answer for the third sample is 10, not 0.

B. Candy Boxes

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

There is an old tradition of keeping 4 boxes of candies in the house in Cyberland. The numbers of candies are *special* if their *arithmetic mean*, their *median* and their *range* are all equal. By definition, for a set $\{x_1, x_2, x_3, x_4\}$ ($x_1 \le x_2 \le x_3 \le x_4$) *arithmetic mean* is $\frac{x_1+x_2+x_3+x_4}{4}$, *median* is $\frac{x_2+x_3}{2}$ and *range* is $x_4 - x_1$. **The arithmetic mean and median are not necessary integer.** It is well-known that if those three numbers are same, boxes will create a "debugging field" and codes in the field will have no bugs.

For example, $1, 1, 3, 3$ is the example of 4 numbers meeting the condition because their mean, median and range are all equal to 2.

Jeff has 4 special boxes of candies. However, something bad has happened! Some of the boxes could have been lost and now there are only n ($0 \le n \le 4$) boxes remaining. The i-th remaining box contains a_i candies.

Now Jeff wants to know: is there a possible way to find the number of candies of the $4 - n$ missing boxes, meeting the condition above (the mean, median and range are equal)?

Input

The first line of input contains an only integer n ($0 \le n \le 4$).

The next n lines contain integers a_i, denoting the number of candies in the i-th box ($1 \le a_i \le 500$).

Output

In the first output line, print "`YES`" if a solution exists, or print "`NO`" if there is no solution.

If a solution exists, you should output $4 - n$ more lines, each line containing an integer b, denoting the number of candies in a missing box.

All your numbers b must satisfy inequality $1 \le b \le 10^6$. It is guaranteed that if there exists a positive integer solution, you can always find such b's meeting the condition. If there are multiple answers, you are allowed to print any of them.

Given numbers a_i may follow in any order in the input, not necessary in non-decreasing.

a_i may have stood at any positions in the original set, not necessary on lowest n first positions.

Sample test(s)

input

```
2
1
1
```

output

```
YES
3
3
```

input

```
3
1
1
1
```

output

```
NO
```

input

```
4
1
2
2
3
```

output

```
YES
```

Note

For the first sample, the numbers of candies in 4 boxes can be $1, 1, 3, 3$. The arithmetic mean, the median and the range of them are all 2.

For the second sample, it's impossible to find the missing number of candies.

In the third example no box has been lost and numbers satisfy the condition.

You may output b in any order.

C. Fight the Monster

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

A monster is attacking the Cyberland!

Master Yang, a braver, is going to beat the monster. Yang and the monster each have 3 attributes: hitpoints (HP), offensive power (ATK) and defensive power (DEF).

During the battle, every second the monster's HP decrease by $max(0, ATK_Y - DEF_M)$, while Yang's HP decreases by $max(0, ATK_M - DEF_Y)$, where index Y denotes Master Yang and index M denotes monster. Both decreases happen simultaneously Once monster's $HP \le 0$ and the same time Master Yang's $HP > 0$, Master Yang wins.

Master Yang can buy attributes from the magic shop of Cyberland: h bitcoins per HP, a bitcoins per ATK, and d bitcoins per DEF.

Now Master Yang wants to know the minimum number of bitcoins he can spend in order to win.

Input

The first line contains three integers HP_Y, ATK_Y, DEF_Y, separated by a space, denoting the initial HP, ATK and DEF of Master Yang.

The second line contains three integers HP_M, ATK_M, DEF_M, separated by a space, denoting the HP, ATK and DEF of the monster.

The third line contains three integers h, a, d, separated by a space, denoting the price of 1 HP, 1 ATK and 1 DEF.

All numbers in input are **integer** and lie between 1 and 100 inclusively.

Output

The only output line should contain an integer, denoting the minimum bitcoins Master Yang should spend in order to win.

Sample test(s)

input

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

output

```
99
```

input

```
100 100 100
1 1 1
1 1 1
```

output

```
0
```

Note

For the first sample, prices for ATK and DEF are extremely high. Master Yang can buy 99 HP, then he can beat the monster with 1 HP left.

For the second sample, Master Yang is strong enough to beat the monster, so he doesn't need to buy anything.

D. Strip

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

Alexandra has a paper strip with n numbers on it. Let's call them a_i from left to right.

Now Alexandra wants to split it into some pieces (possibly 1). For each piece of strip, it must satisfy:

- Each piece should contain at least l numbers.
- The difference between the maximal and the minimal number on the piece should be at most s.

Please help Alexandra to find the minimal number of pieces meeting the condition above.

Input

The first line contains three space-separated integers n, s, l ($1 \le n \le 10^5, 0 \le s \le 10^9, 1 \le l \le 10^5$).

The second line contains n integers a_i separated by spaces ($-10^9 \le a_i \le 10^9$).

Output

Output the minimal number of strip pieces.

If there are no ways to split the strip, output -1.

Sample test(s)

input

```
7 2 2
1 3 1 2 4 1 2
```

output

```
3
```

input

```
7 2 2
1 100 1 100 1 100 1
```

output

```
-1
```

Note

For the first sample, we can split the strip into 3 pieces: $[1, 3, 1], [2, 4], [1, 2]$.

For the second sample, we can't let 1 and 100 be on the same piece, so no solution exists.

E. Prefix Product Sequence

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

Consider a sequence $[a_1, a_2, \ldots, a_n]$. Define its prefix product sequence $[a_1 \mod n, (a_1 a_2) \mod n, \cdots, (a_1 a_2 \cdots a_n) \mod n]$.

Now given n, find a permutation of $[1, 2, ..., n]$, such that its prefix product sequence is a permutation of $[0, 1, ..., n - 1]$.

Input

The only input line contains an integer n $(1 \le n \le 10^5)$.

Output

In the first output line, print "`YES`" if such sequence exists, or print "`NO`" if no such sequence exists.

If any solution exists, you should output n more lines. i-th line contains only an integer a_i. The elements of the sequence should be different positive integers no larger than n.

If there are multiple solutions, you are allowed to print any of them.

Sample test(s)

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

input
```
6
```
output
```
NO
```

Note

For the second sample, there are no valid sequences.