A. Polycarp's Pockets

1 second, 256 megabytes

Polycarp has n coins, the value of the i -th coin is a_i . Polycarp wants to distribute all the coins between his pockets, but he cannot put two coins with the same value into the same pocket.

For example, if Polycarp has got six coins represented as an array $a = [1, 2, 4, 3, 3, 2]$, he can distribute the coins into two pockets as follows: $[1, 2, 3], [2, 3, 4]$.

Polycarp wants to distribute all the coins with the minimum number of used pockets. Help him to do that.

Input

The first line of the input contains one integer n ($1 \le n \le 100$) — the number of coins.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 100$) — values of coins.

Output

Print only one integer — the minimum number of pockets Polycarp needs to distribute all the coins so no two coins with the same value are put into the same pocket.

input

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

output

```
2
```

input

```
1
100
```

output

```
1
```

B. Binary String Constructing

1 second, 256 megabytes

You are given three integers a , b and x . Your task is to construct a binary string s of length $n = a + b$ such that there are exactly a zeroes, exactly b ones and exactly x indices i (where $1 \le i < n$) such that $s_i \neq s_{i+1}$. It is guaranteed that the answer always exists.

For example, for the string "01010" there are four indices i such that $1 \le i < n$ and $s_i \neq s_{i+1}$ ($i = 1, 2, 3, 4$). For the string "111001" there are two such indices i ($i = 3, 5$).

Recall that binary string is a non-empty sequence of characters where each character is either 0 or 1.

Input

The first line of the input contains three integers a , b and x ($1 \le a, b \le 100, 1 \le x < a + b)$.

Output

Print only one string s , where s is **any** binary string satisfying conditions described above. It is guaranteed that the answer always exists.

input

```
2 2 1
```

output

```
1100
```

input

```
3 3 3
```

output

```
101100
```

input

```
5 3 6
```

output

```
01010100
```

All possible answers for the first example:

- 1100;
- 0011.

All possible answers for the second example:

- 110100;
- 101100;
- 110010;
- 100110;
- 011001;
- 001101;
- 010011;
- 001011.

C. Intense Heat

4 seconds, 256 megabytes

The heat during the last few days has been really intense. Scientists from all over the Berland study how the temperatures and weather change, and they claim that this summer is abnormally hot. But any scientific claim sounds a lot more reasonable if there are some numbers involved, so they have decided to actually calculate some value which would represent how high the temperatures are.

Mathematicians of Berland State University came up with a special *heat intensity value*. This value is calculated as follows:

Suppose we want to analyze the segment of n consecutive days. We have measured the temperatures during these n days; the temperature during i -th day equals a_i .

We denote the *average temperature* of a segment of some consecutive days as the arithmetic mean of the temperature measures during this segment of days. So, if we want to analyze the *average temperature* from day x to day y , we calculate it as $\frac{\sum\limits_{i=x}^{y} a_i}{y - x + 1}$ (note that division is performed without any rounding). The *heat intensity value* is the maximum of *average temperatures* over all segments of not less than k consecutive days. For example, if analyzing the measures $[3, 4, 1, 2]$ and $k = 3$, we are interested in segments $[3, 4, 1]$, $[4, 1, 2]$ and $[3, 4, 1, 2]$ (we want to find the maximum value of *average temperature* over these segments).

You have been hired by Berland State University to write a program that would compute the *heat intensity value* of a given period of days. Are you up to this task?

Input

The first line contains two integers n and k ($1 \le k \le n \le 5000$) — the number of days in the given period, and the minimum number of days in a segment we consider when calculating *heat intensity value*, respectively.

The second line contains n integers a_1 , a_2 , ..., a_n ($1 \le a_i \le 5000$) — the temperature measures during given n days.

Output

Print one real number — the *heat intensity value*, i. e., the maximum of *average temperatures* over all segments of not less than k consecutive days.

Your answer will be considered correct if the following condition holds: $|res - res_0| < 10^{-6}$, where res is your answer, and res_0 is the answer given by the jury's solution.

input

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

output

```
2.666666666666667
```

D. Coins and Queries

2 seconds, 256 megabytes

Polycarp has n coins, the value of the i -th coin is a_i . It is guaranteed that all the values are integer powers of 2 (i.e. $a_i = 2^d$ for some **non-negative** integer number d).

Polycarp wants to know answers on q queries. The j -th query is described as integer number b_j . The answer to the query is the minimum number of coins that is necessary to obtain the value b_j using some subset of coins (Polycarp can use only coins he has). If Polycarp can't obtain the value b_j , the answer to the j -th query is `-1`.

The queries are independent (the answer on the query doesn't affect Polycarp's coins).

Input

The first line of the input contains two integers n and q ($1 \le n, q \le 2 \cdot 10^5$) — the number of coins and the number of queries.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ — values of coins ($1 \le a_i \le 2 \cdot 10^9$). It is guaranteed that all a_i are integer powers of 2 (i.e. $a_i = 2^d$ for some **non-negative** integer number d).

The next q lines contain one integer each. The j -th line contains one integer b_j — the value of the j -th query ($1 \le b_j \le 10^9$).

Output

Print q integers ans_j . The j -th integer must be equal to the answer on the j -th query. If Polycarp can't obtain the value b_j the answer to the j -th query is `-1`.

input

```
5 4
2 4 8 2 4
8
5
14
10
```

output

```
1
-1
3
2
```

E. Tree Constructing

4 seconds, 256 megabytes

You are given three integers n , d and k .

Your task is to construct an undirected tree on n vertices with diameter d and degree of each vertex at most k , or say that it is impossible.

An undirected tree is a connected undirected graph with $n - 1$ edges.

Diameter of a tree is the maximum length of a simple path (a path in which each vertex appears at most once) between all pairs of vertices of this tree.

Degree of a vertex is the number of edges incident to this vertex (i.e. for a vertex u it is the number of edges (u, v) that belong to the tree, where v is any other vertex of a tree).

Input

The first line of the input contains three integers n , d and k ($1 \le n, d, k \le 4 \cdot 10^5$).

Output

If there is no tree satisfying the conditions above, print only one word "`NO`" (without quotes).

Otherwise in the first line print "`YES`" (without quotes), and then print $n - 1$ lines describing edges of a tree satisfying the conditions above. Vertices of the tree must be numbered from 1 to n . You can print edges and vertices connected by an edge in any order. If there are multiple answers, print any of them.1

input

```
6 3 3
```

output

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

input

```
6 2 3
```

output

```
NO
```

input

```
10 4 3
```

output

```
YES
2 9
2 10
10 3
3 1
6 10
8 2
4 3
5 6
6 7
```

input

```
8 5 3
```

output

```
YES
2 5
7 2
3 7
3 1
1 6
8 7
4 3
```

F. Abbreviation

1 second, 256 megabytes

You are given a text consisting of n space-separated words. There is exactly one space character between any pair of adjacent words. There are no spaces before the first word and no spaces after the last word. The length of text is the number of letters and spaces in it. w_i is the i -th word of text. All words consist only of lowercase Latin letters.

Let's denote a segment of words $w[i..j]$ as a sequence of words $w_i, w_{i+1}, \dots, w_j$. Two segments of words $w[i_1 .. j_1]$ and $w[i_2 .. j_2]$ are considered **equal** if $j_1 - i_1 = j_2 - i_2$, $j_1 \ge i_1$, $j_2 \ge i_2$, and for every $t \in [0, j_1 - i_1]$ $w_{i_1+t} = w_{i_2+t}$. For example, for the text "`to be or not to be`" the segments $w[1..2]$ and $w[5..6]$ are equal, they correspond to the words "`to be`".

An abbreviation is a replacement of some segments of words with their first **uppercase** letters. In order to perform an abbreviation, you have to choose **at least two** non-intersecting equal segments of words, and replace each chosen segment with the string consisting of first letters of the words in the segment (written in uppercase). For example, for the text "`a ab a a b ab a a b c`" you can replace segments of words $w[2..4]$ and $w[6..8]$ with an abbreviation "`AAA`" and obtain the text "`a AAA b AAA b c`", or you can replace segments of words $w[2..5]$ and $w[6..9]$ with an abbreviation "`AAAB`" and obtain the text "`a AAAB AAAB c`".

What is the minimum length of the text after at most one abbreviation?

Input

The first line of the input contains one integer n ($1 \le n \le 300$) — the number of words in the text.

The next line contains n space-separated words of the text $w_1, w_2, \dots, w_n$. Each word consists only of lowercase Latin letters.

It is guaranteed that the length of text does not exceed 10^5.

Output

Print one integer — the minimum length of the text after at most one abbreviation.

input

```
6
to be or not to be
```

output

```
12
```

input

```
10
a ab a a b ab a a b c
```

output

```
13
```

input

```
6
aa bb aa aa bb bb
```

output

```
11
```

In the first example you can obtain the text "`TB or not TB`".

In the second example you can obtain the text "`a AAAB AAAB c`".

In the third example you can obtain the text "`AB aa AB bb`".