A. Equation

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

Let's call a positive integer **composite** if it has at least one divisor other than 1 and itself. For example:

- the following numbers are composite: 1024, 4, 6, 9;
- the following numbers are not composite: 13, 1, 2, 3, 37.

You are given a positive integer n. Find two composite integers a, b such that $a - b = n$.

It can be proven that solution always exists.

Input

The input contains one integer n ($1 \leq n \leq 10^7$): the given integer.

Output

Print two composite integers a, b ($2 \leq a, b \leq 10^9, a - b = n$).

It can be proven, that solution always exists.

If there are several possible solutions, you can print any.

Examples

input

```
1
```

output

```
9 8
```

input

```
512
```

output

```
4608 4096
```

B. Modulo Equality

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

You are given a positive integer m and two integer sequence: $a = [a_1, a_2, \ldots, a_n]$ and $b = [b_1, b_2, \ldots, b_n]$. Both of these sequence have a length n.

Permutation is a sequence of n different positive integers from 1 to n. For example, these sequences are permutations: $[1]$, $[1, 2]$, $[2, 1]$, $[6, 7, 3, 4, 1, 2, 5]$. These are not: $[0]$, $[1, 1]$, $[2, 3]$.

You need to find the non-negative integer x, and increase all elements of a_i by x, modulo m (i.e. you want to change a_i to $(a_i + x) \bmod m$), so it would be possible to rearrange elements of a to make it equal b, among them you need to find the smallest possible x.

In other words, you need to find the smallest non-negative integer x, for which it is possible to find some permutation $p = [p_1, p_2, \ldots, p_n]$, such that for all $1 \leq i \leq n$, $(a_i + x) \bmod m = b_{p_i}$, where $y \bmod m$ — remainder of division of y by m.

For example, if $m = 3$, $a = [0, 0, 2, 1], b = [2, 0, 1, 1]$, you can choose $x = 1$, and a will be equal to $[1, 1, 0, 2]$ and you can rearrange it to make it equal $[2, 0, 1, 1]$, which is equal to b.

Input

The first line contains two integers n, m ($1 \leq n \leq 2000, 1 \leq m \leq 10^9$): number of elemens in arrays and m.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \leq a_i < m$).

The third line contains n integers $b_1, b_2, \ldots, b_n$ ($0 \leq b_i < m$).

It is guaranteed that there exists some non-negative integer x, such that it would be possible to find some permutation $p_1, p_2, \ldots, p_n$ such that $(a_i + x) \bmod m = b_{p_i}$.

Output

Print one integer, the smallest non-negative integer x, such that it would be possible to find some permutation $p_1, p_2, \ldots, p_n$ such that $(a_i + x) \bmod m = b_{p_i}$ for all $1 \le i \le n$.

Examples

input

```
4 3
0 0 2 1
2 0 1 1
```

output

```
1
```

input

```
3 2
0 0 0
1 1 1
```

output

```
1
```

input

```
5 10
0 0 0 1 2
2 1 0 0 0
```

output

```
0
```

C. Long Beautiful Integer

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

You are given an integer x of n digits $a_1, a_2, \ldots, a_n$, which make up its decimal notation in order from left to right.

Also, you are given a positive integer $k < n$.

Let's call integer $b_1, b_2, \ldots, b_m$ **beautiful** if $b_i = b_{i+k}$ for each i, such that $1 \le i \le m - k$.

You need to find the smallest **beautiful** integer y, such that $y \ge x$.

Input

The first line of input contains two integers n, k ($2 \le n \le 200\,000, 1 \le k < n$): the number of digits in x and k.

The next line of input contains n digits $a_1, a_2, \ldots, a_n$ ($a_1 \neq 0$, $0 \le a_i \le 9$): digits of x.

Output

In the first line print one integer m: the number of digits in y.

In the next line print m digits $b_1, b_2, \ldots, b_m$ ($b_1 \neq 0$, $0 \le b_i \le 9$): digits of y.

Examples

input

```
3 2
353
```

output

```
3
353
```

input

```
4 2
1234
```

output

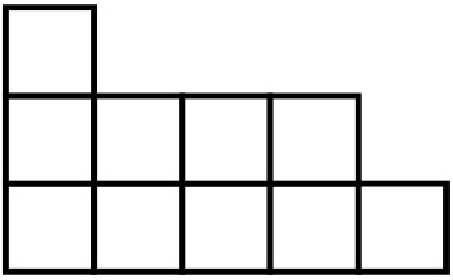
```
4
1313
```

D. Domino for Young

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

You are given a Young diagram.

Given diagram is a histogram with n columns of lengths $a_1, a_2, \ldots, a_n$ ($a_1 \geq a_2 \geq \ldots \geq a_n \geq 1$).



Young diagram for $a = [3, 2, 2, 2, 1]$.

Your goal is to find the largest number of non-overlapping dominos that you can draw inside of this histogram, a domino is a 1×2 or 2×1 rectangle.

Input

The first line of input contain one integer n ($1 \leq n \leq 300\,000$): the number of columns in the given histogram.

The next line of input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 300\,000$, $a_i \geq a_{i+1}$): the lengths of columns.

Output

Output one integer: the largest number of non-overlapping dominos that you can draw inside of the given Young diagram.

Example

input

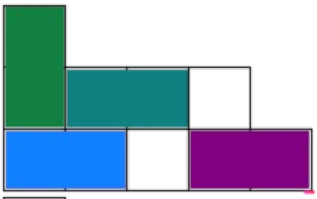
```
5
3 2 2 2 1
```

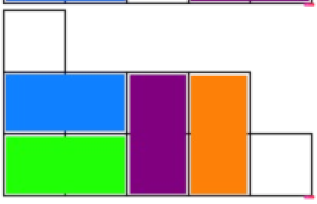
output

```
4
```

Note

Some of the possible solutions for the example:





E. K Integers

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

You are given a permutation $p_1, p_2, \ldots, p_n$.

In one move you can swap two adjacent values.

You want to perform a minimum number of moves, such that in the end there will exist a subsegment $1, 2, \ldots, k$, in other words in the end there should be an integer i, $1 \leq i \leq n-k+1$ such that $p_i = 1, p_{i+1} = 2, \ldots, p_{i+k-1} = k$.

Let $f(k)$ be the minimum number of moves that you need to make a subsegment with values $1, 2, \ldots, k$ appear in the permutation.

You need to find $f(1), f(2), \ldots, f(n)$.

Input

The first line of input contains one integer n ($1 \leq n \leq 200\,000$): the number of elements in the permutation.

The next line of input contains n integers $p_1, p_2, \ldots, p_n$: given permutation ($1 \le p_i \le n$).

Output

Print n integers, the minimum number of moves that you need to make a subsegment with values $1, 2, \ldots, k$ appear in the permutation, for $k = 1, 2, \ldots, n$.

Examples

input

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

output

```
0 1 3 6 10
```

input

```
3
1 2 3
```

output

```
0 0 0
```