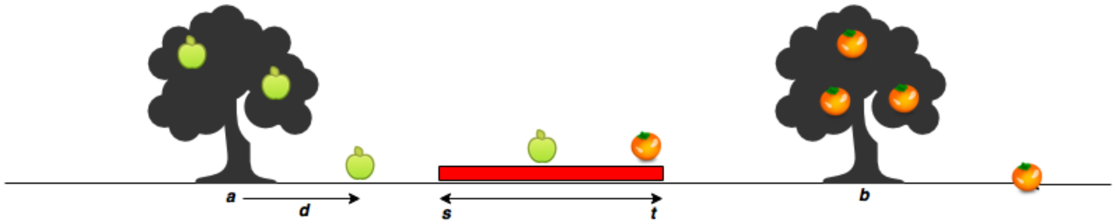
Apple and Orange

Sam's house has an apple tree and an orange tree that yield an abundance of fruit. In the diagram below, the red region denotes his house, where s is the start point, and t is the endpoint. The apple tree is to the left of his house, and the orange tree is to its right. You can assume the trees are located on a single point, where the apple tree is at point a, and the orange tree is at point b.



When a fruit falls from its tree, it lands d units of distance from its tree of origin along the x-axis. A negative value of d means the fruit fell d units to the tree's left, and a positive value of d means it falls d units to the tree's right.

Complete the function `countApplesAndOranges`,

where,

$start$ Starting point of Sam's house location.
end Ending location of Sam's house location.
loc_a Location of the Apple tree.
loc_o Location of the Orange tree.
$size_a$ Number of apples that fell from the tree.
$apples$ Distance at which each apple falls from the tree.
$size_o$ Number of oranges that fell from the tree.
$oranges$ Distance at which each orange falls from the tree.

Given the value of d for m apples and n oranges, can you determine how many apples and oranges will fall on Sam's house (i.e., in the inclusive range $[s, t]$)? Print the number of apples that fall on Sam's house as your first line of output, then print the number of oranges that fall on Sam's house as your second line of output.

Input Format

The first line contains two space-separated integers denoting the respective values of s and t.
The second line contains two space-separated integers denoting the respective values of a and b.
The third line contains two space-separated integers denoting the respective values of m and n.
The fourth line contains m space-separated integers denoting the respective distances that each apple falls from point a.
The fifth line contains n space-separated integers denoting the respective distances that each orange falls from point b.

Constraints

- $1 \leq s, t, a, b, m, n \leq 10^5$
- $-10^5 \leq d \leq 10^5$
- $a < s < t < b$

Output Format

Print two lines of output:

1. On the first line, print the number of apples that fall on Sam's house.
2. On the second line, print the number of oranges that fall on Sam's house.

Sample Input 0

```
7 11
5 15
3 2
-2 2 1
5 -6
```

Sample Output 0

```
1
1
```

Explanation 0

The first apple falls at position $5 - 2 = 3$.
The second apple falls at position $5 + 2 = 7$.
The third apple falls at position $5 + 1 = 6$.
The first orange falls at position $15 + 5 = 20$.
The second orange falls at position $15 - 6 = 9$.
Only one fruit (the second apple) falls within the region between 7 and 11, so we print 1 as our first line of output.
Only the second orange falls within the region between 7 and 11, so we print 1 as our second line of output.

Kangaroo

You are choreograhing a circus show with various animals. For one act, you are given two kangaroos on a number line ready to jump in the positive direction (i.e, toward positive infinity).

- The first kangaroo starts at location x_1 and moves at a rate of v_1 meters per jump.
- The second kangaroo starts at location x_2 and moves at a rate of v_2 meters per jump.

You have to figure out a way to get both kangaroos at the same location as part of the show.

Complete the function `kangaroo` which takes starting location and speed of both kangaroos as input, and return Yes or No appropriately. Can you determine if the kangaroos will ever land *at the same location at the same time*? The two kangaroos must land at the same location after making the same number of jumps.

Input Format

A single line of four space-separated integers denoting the respective values of x_1, v_1, x_2, and v_2.

Constraints

- $0 \leq x_1 < x_2 \leq 10000$
- $1 \leq v_1 \leq 10000$
- $1 \leq v_2 \leq 10000$

Output Format

Print `YES` if they can land on the same location at the same time; otherwise, print `NO`.

Note: The two kangaroos must land at the same location *after making the same number of jumps*.

Sample Input 0

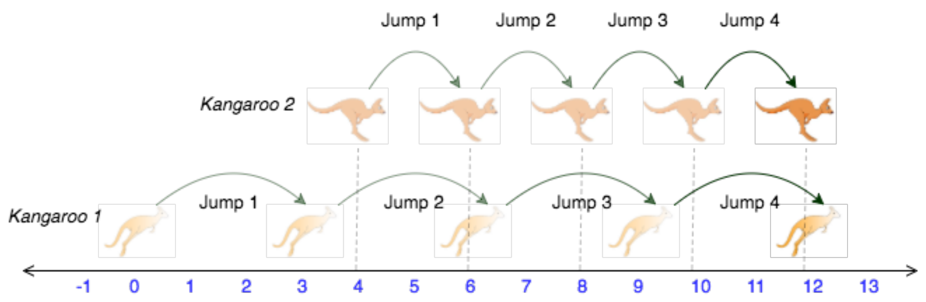
```
0 3 4 2
```

Sample Output 0

```
YES
```

Explanation 0

The two kangaroos jump through the following sequence of locations:



From the image, it is clear that the kangaroos meet at the same location (number 12 on the number line) after same number of jumps (4 jumps), and we print `YES`.

Sample Input 1

```
0 2 5 3
```

Sample Output 1

```
NO
```

Explanation 1

The second kangaroo has a starting location that is ahead (further to the right) of the first kangaroo's starting location (i.e., $x_2 > x_1$). Because the second kangaroo moves at a faster rate (meaning $v_2 > v_1$) *and* is already ahead of the first kangaroo, the first kangaroo will never be able to catch up. Thus, we print *NO*.

Between Two Sets

You will be given two arrays of integers and asked to determine all integers that satisfy the following two conditions:

1. The elements of the first array are all factors of the integer being considered
2. The integer being considered is a factor of all elements of the second array

These numbers are referred to as being *between* the two arrays. You must determine how many such numbers exist.

For example, given the arrays $a = [2, 6]$ and $b = [24, 36]$, there are two numbers between them: 6 and 12. $6\%2 = 0$, $6\%6 = 0$, $24\%6 = 0$ and $36\%6 = 0$ for the first value. Similarly, $12\%2 = 0$, $12\%6 = 0$ and $24\%12 = 0$, $36\%12 = 0$.

Function Description

Complete the *getTotalX* function in the editor below. It should return the number of integers that are betwen the sets.

getTotalX has the following parameter(s):

- *a*: an array of integers
- *b*: an array of integers

Input Format

The first line contains two space-separated integers, n and m, the number of elements in array a and the number of elements in array b.
The second line contains n distinct space-separated integers describing $a[i]$ where $0 \le i < n$.
The third line contains m distinct space-separated integers describing $b[j]$ where $0 \le j < m$.

Constraints

- $1 \le n, m \le 10$
- $1 \le a[i] \le 100$
- $1 \le b[j] \le 100$

Output Format

Print the number of integers that are considered to be *between* a and b.

Sample Input

```
2 3
2 4
16 32 96
```

Sample Output

```
3
```

Explanation

2 and 4 divide evenly into 4, 8, 12 and 16.
4, 8 and 16 divide evenly into 16, 32, 96.

4, 8 and 16 are the only three numbers for which each element of a is a factor and each is a factor of all elements of b.

Breaking the Records

Maria plays n games of college basketball in a season. Because she wants to go pro, she tracks her points scored per game sequentially in an array defined as $score = [s_0, s_1, \ldots, s_{n-1}]$. After each game i, she checks to see if score s_i breaks her record for most or least points scored so far during that season.

Given Maria's array of $scores$ for a season of n games, find and print the number of times she breaks her record for *most* and *least* points scored during the season.

Note: Assume her records for most and least points at the start of the season are the number of points scored during the first game of the season.

Input Format

The first line contains an integer denoting n (the number of games).
The second line contains n space-separated integers describing the respective values of $s_0, s_1, \ldots, s_{n-1}$.

Constraints

- $1 \leq n \leq 1000$
- $0 \leq s_i \leq 10^8$

Output Format

Print two space-seperated integers describing the respective numbers of times her best (highest) score increased and her worst (lowest) score decreased.

Sample Input 0

```
9
10 5 20 20 4 5 2 25 1
```

Sample Output 0

```
2 4
```

Explanation 0

The diagram below depicts the number of times Maria broke her best and worst records throughout the season:

Game	0	1	2	3	4	5	6	7	8
Score	10	5	20	20	4	5	2	25	1
Highest Score	10	10	20	20	20	20	20	25	25
Lowest Score	10	5	5	5	4	4	2	2	1

She broke her best record twice (after games 2 and 7) and her worst record four times (after games 1, 4, 6, and 8), so we print `2 4` as our answer. Note that she *did not* break her record for best score during game 3, as her score during that game was *not* strictly greater than her best record at the time.

Sample Input 1

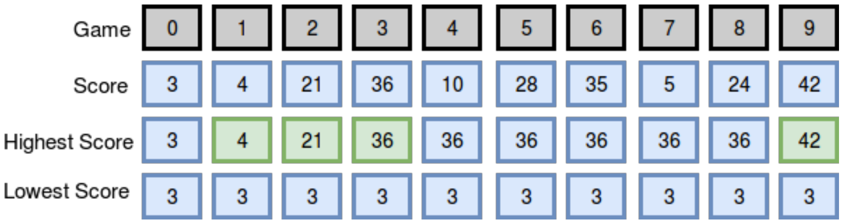
```
10
3 4 21 36 10 28 35 5 24 42
```

Sample Output 1

```
4 0
```

Explanation 1

The diagram below depicts the number of times Maria broke her best and worst records throughout the season:

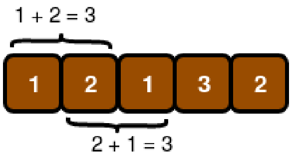


She broke her best record four times (after games 1, 2, 3, and 9) and her worst record zero times (no score during the season was lower than the one she earned during her first game), so we print `4 0` as our answer.

Birthday Chocolate

Lily has a chocolate bar with numbered squares. She wants to share it with Ron for his birthday. She decides to share a contiguous segment of the bar selected such that the sum of the integers on the squares is equal to a given value. The length of the segment will match Ron's birth month. The sum of the segments will match his birth day. You must determine how many ways she can divide the chocolate.

Consider the chocolate bar as an array of squares, $s = [1, 2, 1, 3, 2]$. She wants to find segments summing to Ron's birth day, $d = 3$ with a length equalling his birth month, $m = 2$. In this case, there are two segments meeting her criteria.



Input Format

The first line contains an integer n, the number of squares in the chocolate bar.
The second line contains n space-separated integers $s[i]$, the numbers on the chocolate squares where $0 \leq i < n$.
The third line contains two space-separated integers, d and m, Ron's birth day and his birth month.

Constraints

- $1 \leq n \leq 100$
- $1 \leq s[i] \leq 5$, where ($0 \leq i < n$)
- $1 \leq d \leq 31$
- $1 \leq m \leq 12$

Output Format

Print an integer denoting the total number of ways that Lily can portion her chocolate bar to share with Ron.

Sample Input 0

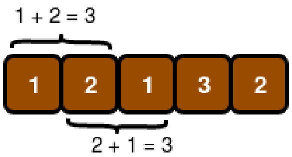
```
5
1 2 1 3 2
3 2
```

Sample Output 0

```
2
```

Explanation 0

Lily wants to give Ron $m = 2$ squares summing to $d = 3$. The following two segments meet the criteria:



Sample Input 1

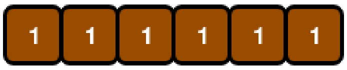
```
6
1 1 1 1 1 1
3 2
```

Sample Output 1

```
0
```

Explanation 1

Lily only wants to give Ron $m = 2$ consecutive squares of chocolate whose integers sum to $d = 3$. There are no possible pieces satisfying these constraints:



Thus, we print 0 as our answer.

Sample Input 2

```
1
4
4 1
```

Sample Output 2

```
1
```

Explanation 2

Lily only wants to give Ron $m = 1$ square of chocolate with an integer value of $d = 4$. Because the only square of chocolate in the bar satisfies this constraint, we print 1 as our answer.

Divisible Sum Pairs

You are given an array of n integers, $a_0, a_1, \ldots, a_{n-1}$, and a positive integer, k. Find and print the number of (i, j) pairs where $i < j$ and a_i + a_j is divisible by k.

Input Format

The first line contains 2 space-separated integers, n and k, respectively.
The second line contains n space-separated integers describing the respective values of $a_0, a_1, \ldots, a_{n-1}$.

Constraints

- $2 \leq n \leq 100$
- $1 \leq k \leq 100$
- $1 \leq a_i \leq 100$

Output Format

Print the number of (i, j) pairs where $i < j$ and a_i + a_j is evenly divisible by k.

Sample Input

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

Sample Output

```
5
```

Explanation

Here are the 5 valid pairs:

- $(0, 2) \rightarrow a_0 + a_2 = 1 + 2 = 3$
- $(0, 5) \rightarrow a_0 + a_5 = 1 + 2 = 3$
- $(1, 3) \rightarrow a_1 + a_3 = 3 + 6 = 9$
- $(2, 4) \rightarrow a_2 + a_4 = 2 + 1 = 3$
- $(4, 5) \rightarrow a_4 + a_5 = 1 + 2 = 3$

Migratory Birds

You have been asked to help study the population of birds migrating across the continent. Each type of bird you are interested in will be identified by an integer value. Each time a particular kind of bird is spotted, its id number will be added to your array of sightings. You would like to be able to find out which type of bird is most common given a list of sightings. Your task is to print the type number of that bird and if two or more types of birds are equally common, choose the type with the smallest ID number.

For example, assume your bird sightings are of types $arr = [1, 1, 2, 2, 3]$. There are two each of types 1 and 2, and one sighting of type 3. Pick the lower of the two types seen twice: type 1.

Function Description

Complete the *migratoryBirds* function in the editor below. It should return the lowest type number of the most frequently sighted bird.

migratoryBirds has the following parameter(s):

- *arr*: an array of integers representing types of birds sighted

Input Format

The first line contains an integer denoting n, the number of birds sighted and reported in the array arr.
The second line describes arr as n space-separated integers representing the type numbers of each bird sighted.

Constraints

- $5 \leq n \leq 2 \times 10^5$
- It is guaranteed that each type is 1, 2, 3, 4, or 5.

Output Format

Print the type number of the most common bird; if two or more types of birds are equally common, choose the type with the smallest ID number.

Sample Input 0

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

Sample Output 0

```
4
```

Explanation 0

The different types of birds occur in the following frequencies:

- Type 1: 1 bird

- Type 2: 0 birds
- Type 3: 1 bird
- Type 4: 3 birds
- Type 5: 1 bird

The type number that occurs at the highest frequency is type 4, so we print 4 as our answer.

Sample Input 1

```
11
1 2 3 4 5 4 3 2 1 3 4
```

Sample Output 1

```
3
```

Explanation 1

The different types of birds occur in the following frequencies:

- Type 1: 2
- Type 2: 2
- Type 3: 3
- Type 4: 3
- Type 5: 1

Two types have a frequency of 3, and the lower of those is type 3.

Day of the Programmer

Marie invented a Time Machine and wants to test it by time-traveling to visit Russia on the Day of the Programmer (the 256^{th} day of the year) during a year in the inclusive range from 1700 to 2700.

From 1700 to 1917, Russia's official calendar was the Julian calendar; since 1919 they used the Gregorian calendar system. The transition from the Julian to Gregorian calendar system occurred in 1918, when the next day after January 31^{st} was February 14^{th}. This means that in 1918, February 14^{th} was the 32^{nd} day of the year in Russia.

In both calendar systems, February is the only month with a variable amount of days; it has 29 days during a *leap year*, and 28 days during all other years. In the Julian calendar, leap years are divisible by 4; in the Gregorian calendar, leap years are either of the following:

- Divisible by 400.
- Divisible by 4 and *not* divisible by 100.

Given a year, y, find the date of the 256^{th} day of that year *according to the official Russian calendar during that year*. Then print it in the format `dd.mm.yyyy`, where `dd` is the two-digit day, `mm` is the two-digit month, and `yyyy` is y.

Input Format

A single integer denoting year y.

Constraints

- $1700 \le y \le 2700$

Output Format

Print the full date of *Day of the Programmer* during year y in the format `dd.mm.yyyy`, where `dd` is the two-digit day, `mm` is the two-digit month, and `yyyy` is y.

Sample Input 0

```
2017
```

Sample Output 0

```
13.09.2017
```

Explanation 0

In the year $y = 2017$, January has 31 days, February has 28 days, March has 31 days, April has 30 days, May has 31 days, June has 30 days, July has 31 days, and August has 31 days. When we sum the total number of days in the first eight months, we get $31 + 28 + 31 + 30 + 31 + 30 + 31 + 31 = 243$. Day of the Programmer is the 256^{th} day, so then calculate $256 - 243 = 13$ to determine that it falls on day 13 of the 9^{th} month (September). We then print the full date in the specified format, which is `13.09.2017`.

Sample Input 1

```
2016
```

Sample Output 1

```
12.09.2016
```

Explanation 1

Year $y = 2016$ is a leap year, so February has 29 days but all the other months have the same number of days as in 2017. When we sum the total number of days in the first eight months, we get $31 + 29 + 31 + 30 + 31 + 30 + 31 + 31 = 244$. Day of the Programmer is the 256^{th} day, so then calculate $256 - 244 = 12$ to determine that it falls on day 12 of the 9^{th} month (September). We then print the full date in the specified format, which is `12.09.2016`.

Sample Input 2

```
1800
```

Sample Output 2

```
12.09.1800
```

Explanation 2

Since 1800 is leap year. Day lies on 12 September.

Bon Appétit

Anna and Brian order n items at a restaurant, but Anna declines to eat any of the k^{th} item (where $0 \leq k < n$) due to an allergy. When the check comes, they decide to split the cost of all the items they shared; however, Brian may have forgotten that they didn't split the k^{th} item and accidentally charged Anna for it.

You are given n, k, the cost of each of the n items, and the total amount of money that Brian charged Anna for her portion of the bill. If the bill is fairly split, print `Bon Appetit`; otherwise, print the amount of money that Brian must refund to Anna.

Input Format

The first line contains two space-separated integers denoting the respective values of n (the number of items ordered) and k (the 0-based index of the item that Anna did not eat).
The second line contains n space-separated integers where each integer i denotes the cost, $c[i]$, of item i (where $0 \leq i < n$).
The third line contains an integer, $b_{charged}$, denoting the amount of money that Brian charged Anna for her share of the bill.

Constraints

- $2 \leq n \leq 10^5$
- $0 \leq k < n$
- $0 \leq c[i] \leq 10^4$
- $0 \leq b \leq \sum c[i]$

Output Format

If Brian did not overcharge Anna, print `Bon Appetit` on a new line; otherwise, print the difference (i.e., $b_{charged} - b_{actual}$) that Brian must refund to Anna (it is guaranteed that this will always be an integer).

Sample Input 0

```
4 1
3 10 2 9
12
```

Sample Output 0

```
5
```

Explanation 0

Anna didn't eat item $c[1] = 10$, but she shared the rest of the items with Brian. The total cost of the shared items is $3 + 2 + 9 = 14$ and, split in half, the cost per person is $b_{actual} = 7$. Brian charged her $b_{charged} = 12$ for her portion of the bill, which is more than the 7 dollars worth of food that she actually shared with him. Thus, we print the amount Anna was overcharged, $b_{charged} - b_{actual} = 12 - 7 = 5$, on a new line.

Sample Input 1

```
4 1
3 10 2 9
7
```

Sample Output 1

```
Bon Appetit
```

Explanation 1

Anna didn't eat item $c[1] = 10$, but she shared the rest of the items with Brian. The total cost of the shared items is $3 + 2 + 9 = 14$ and, split in half, the cost per person is $b_{actual} = 7$. Because this matches the amount, $b_{charged} = 7$, that Brian charged Anna for her portion of the bill, we print `Bon Appetit` on a new line.

Sock Merchant

John works at a clothing store. He has a large pile of socks that he must pair them by color for sale.

You will be given an array of integers representing the color of each sock. Determine how many pairs of socks with matching colors there are.

John works at a clothing store and he's going through a pile of socks to find the number of matching pairs. More specifically, he has a pile of n loose socks where each sock i is labeled with an integer, c_i, denoting its color. He wants to sell as many socks as possible, but his customers will only buy them in matching pairs. Two socks, i and j, are a single matching pair if they have the same color ($c_i = c_j$).

Given n and the color of each sock, how many pairs of socks can John sell?

Input Format

The first line contains an integer n, the number of socks.
The second line contains n space-separated integers describing the colors c_i of the socks in the pile.

Constraints

- $1 \leq n \leq 100$
- $1 \leq c_i \leq 100$ where $0 \leq i < n$

Output Format

Print the total number of *matching pairs* of socks that John can sell.

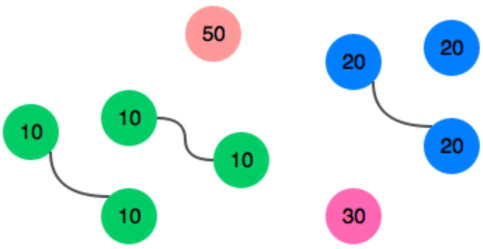
Sample Input

```
9
10 20 20 10 10 30 50 10 20
```

Sample Output

```
3
```

Explanation



John can match three pairs of socks.