Time Conversion

Given a time in 12-hour AM/PM format, convert it to military (24-hour) time.

Note: Midnight is 12:00:00AM on a 12-hour clock, and 00:00:00 on a 24-hour clock. Noon is 12:00:00PM on a 12-hour clock, and 12:00:00 on a 24-hour clock.

Input Format

A single string s containing a time in 12-hour clock format (i.e.: $\mathbf{hh:mm:ssAM}$ or $\mathbf{hh:mm:ssPM}$), where $01 \leq hh \leq 12$ and $00 \leq mm, ss \leq 59$.

Output Format

Convert and print the given time in 24-hour format, where $00 \leq hh \leq 23$.

Sample Input

```
07:05:45PM
```

Sample Output

```
19:05:45
```

Birthday Cake Candles

You are in-charge of the cake for your niece's birthday and have decided the cake will have one candle for each year of her total age. When she blows out the candles, she'll only be able to blow out the tallest ones.

For example, if your niece is turning 4 years old, and the cake will have 4 candles of height 3, 2, 1, 3, she will be able to blow out 2 candles successfully, since the tallest candle is of height 3 and there are 2 such candles.

Complete the function Given the height of each individual candle, find and print the number of candles she can successfully blow out.

Input Format

The first line contains a single integer, n, denoting the number of candles on the cake.
The second line contains n space-separated integers, where each integer i describes the height of candle i.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq height_i \leq 10^7$

Output Format

Print the number of candles the can be blown out on a new line.

Sample Input 0

```
4
3 2 1 3
```

Sample Output 0

```
2
```

Explanation 0

We have one candle of height 1, one candle of height 2, and two candles of height 3. Your niece only blows out the tallest candles, meaning the candles where $height = 3$. Because there are 2 such candles, we print 2 on a new line.

Compare the Triplets

Alice and Bob each created one problem for HackerRank. A reviewer rates the two challenges, awarding points on a scale from 1 to 100 for three categories: *problem clarity*, *originality*, and *difficulty*.

We define the rating for Alice's challenge to be the triplet $A = (a[0], a[1], a[2])$, and the rating for Bob's challenge to be the triplet $B = (b[0], b[1], b[2])$.

Your task is to find their *comparison points* by comparing $a[0]$ with $b[0]$, $a[1]$ with $b[1]$, and $a[2]$ with $b[2]$.

- If $a[i] > b[i]$, then Alice is awarded 1 point.
- If $a[i] < b[i]$, then Bob is awarded 1 point.
- If $a[i] = b[i]$, then neither person receives a point.

Comparison points is the total points a person earned.

Given a and b, determine their respective comparison points.

For example, $a = [1, 2, 3]$ and $b = [3, 2, 1]$. For elements 0, Bob is awarded a point because $a[0] < b[0]$. For the equal elements $a[1]$ and $b[1]$, no points are earned. Finally, for elements 2, $a[2] > b[2]$ so Alice receives a point. Your return array would be $[1, 1]$ with Alice's score first and Bob's second.

Function Description

Complete the function *compareTriplets* in the editor below. It must return an array of two integers, the first being Alice's score and the second being Bob's.

compareTriplets has the following parameter(s):

- *a*: an array of integers representing Alice's challenge rating
- *b*: an array of integers representing Bob's challenge rating

Input Format

The first line contains 3 space-separated integers, $a[0]$, $a[1]$, and $a[2]$, describing the respective values in triplet A.
The second line contains 3 space-separated integers, $b[0]$, $b[1]$, and $b[2]$, describing the respective values in triplet B.

Constraints

- $1 \leq a[i] \leq 100$
- $1 \leq b[i] \leq 100$

Output Format

Return an array of two integers denoting the respective comparison points earned by Alice and Bob.

Sample Input 0

```
5 6 7
3 6 10
```

Sample Output 0

```
1 1
```

Explanation 0

In this example:

- $a = (a[0], a[1], a[2]) = (5, 6, 7)$
- $b = (b[0], b[1], b[2]) = (3, 6, 10)$

Now, let's compare each individual score:

- $a[0] > b[0]$, so Alice receives 1 point.
- $a[1] = b[1]$, so nobody receives a point.
- $a[2] < b[2]$, so Bob receives 1 point.

Alice's comparison score is 1, and Bob's comparison score is 1. Thus, we return the array $[1, 1]$.

Sample Input 1

```
17 28 30
99 16 8
```

Sample Output 1

```
2 1
```

Explanation 1

Comparing the 0^{th} elements, $17 < 99$ so Bob receives a point.
Comparing the 1^{st} and 2^{nd} elements, $28 > 16$ and $30 > 8$ so Alice receives two points.
The return array is $[2, 1]$.

Diagonal Difference

Given a square matrix of size $N \times N$, calculate the absolute difference between the sums of its diagonals.

Input Format

The first line contains a single integer, N. The next N lines denote the matrix's rows, with each line containing N space-separated integers describing the columns.

Constraints

- $-100 \leq \text{Elements in the matrix} \leq 100$

Output Format

Print the absolute difference between the two sums of the matrix's diagonals as a single integer.

Sample Input

```
3
11 2 4
4 5 6
10 8 -12
```

Sample Output

```
15
```

Explanation

The primary diagonal is:

```
11
  5
    -12
```

Sum across the primary diagonal: 11 + 5 - 12 = 4

The secondary diagonal is:

```
    4
  5
10
```

Sum across the secondary diagonal: 4 + 5 + 10 = 19
Difference: |4 - 19| = 15

Note: |x| is absolute value function

Mini-Max Sum

Given five positive integers, find the minimum and maximum values that can be calculated by summing exactly four of the five integers. Then print the respective minimum and maximum values as a single line of two space-separated long integers.

For example, $arr = [1, 3, 5, 7, 9]$. Our minimum sum is $1 + 3 + 5 + 7 = 16$ and our maximum sum is $3 + 5 + 7 + 9 = 24$. We would print

```
16 24
```

Function Description

Complete the *miniMaxSum* function in the editor below. It should print two space-separated integers on one line: the minimum sum and the maximum sum of 4 of 5 elements.

miniMaxSum has the following parameter(s):

- *arr*: an array of 5 integers

Input Format

A single line of five space-separated integers.

Constraints

$1 \leq arr[i] \leq 10^9$

Output Format

Print two space-separated long integers denoting the respective minimum and maximum values that can be calculated by summing exactly *four* of the five integers. (The output can be greater than a 32 bit integer.)

Sample Input

```
1 2 3 4 5
```

Sample Output

```
10 14
```

Explanation

Our initial numbers are 1, 2, 3, 4, and 5. We can calculate the following sums using four of the five integers:

1. If we sum everything except 1, our sum is $2 + 3 + 4 + 5 = 14$.
2. If we sum everything except 2, our sum is $1 + 3 + 4 + 5 = 13$.
3. If we sum everything except 3, our sum is $1 + 2 + 4 + 5 = 12$.
4. If we sum everything except 4, our sum is $1 + 2 + 3 + 5 = 11$.
5. If we sum everything except 5, our sum is $1 + 2 + 3 + 4 = 10$.

Hints: Beware of integer overflow! Use 64-bit Integer.

Plus Minus

Given an array of integers, calculate the fractions of its elements that are *positive*, *negative*, and are *zeros*. Print the decimal value of each fraction on a new line.

Note: This challenge introduces precision problems. The test cases are scaled to six decimal places, though answers with absolute error of up to 10^{-4} are acceptable.

For example, given the array $arr = [1, 1, 0, -1, -1]$ there are 5 elements, two positive, two negative and one zero. Their ratios would be $\frac{2}{5} = 0.400000$, $\frac{2}{5} = 0.400000$ and $\frac{1}{5} = 0.200000$. It should be printed as

```
0.400000
0.400000
0.200000
```

Function Description

Complete the *plusMinus* function in the editor below. It should print out the ratio of positive, negative and zero items in the array, each on a separate line rounded to six decimals.

plusMinus has the following parameter(s):

- *arr*: an array of integers

Input Format

The first line contains an integer, n, denoting the size of the array.
The second line contains n space-separated integers describing an array of numbers $arr(arr[0], arr[1], arr[2], \ldots, arr[n-1])$.

Constraints

$0 < n \leq 100$
$-100 \leq arr[i] \leq 100$

Output Format

You must print the following 3 lines:

1. A decimal representing of the fraction of *positive* numbers in the array compared to its size.
2. A decimal representing of the fraction of *negative* numbers in the array compared to its size.
3. A decimal representing of the fraction of *zeros* in the array compared to its size.

Sample Input

```
6
-4 3 -9 0 4 1
```

Sample Output

```
0.500000
0.333333
0.166667
```

Explanation

There are 3 positive numbers, 2 negative numbers, and 1 zero in the array.

The proportions of occurrence are positive: $\frac{3}{6} = 0.500000$, negative: $\frac{2}{6} = 0.333333$ and zeros: $\frac{1}{6} = 0.166667$.

Staircase

Consider a staircase of size $n = 4$:

```
   #
  ##
 ###
####
```

Observe that its base and height are both equal to n, and the image is drawn using `#` symbols and spaces. *The last line is not preceded by any spaces.*

Write a program that prints a staircase of size n.

Input Format

A single integer, n, denoting the size of the staircase.

Output Format

Print a staircase of size n using `#` symbols and spaces.

Note: The last line must have 0 spaces in it.

Sample Input

```
6
```

Sample Output

```
     #
    ##
   ###
  ####
 #####
######
```

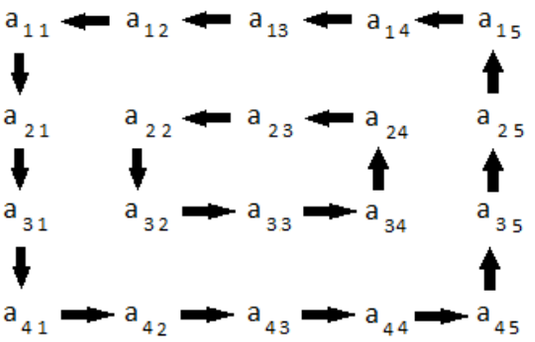
Explanation

The staircase is right-aligned, composed of `#` symbols and spaces, and has a height and width of $n = 6$.

Matrix Layer Rotation

You are given a 2D matrix, *a*, of dimension *MxN* and a positive integer *R*. You have to rotate the matrix *R* times and print the resultant matrix. Rotation should be in anti-clockwise direction.

Rotation of a *4x5* matrix is represented by the following figure. Note that in one rotation, you have to shift elements by one step only (refer sample tests for more clarity).



Matrix Rotation

It is guaranteed that the minimum of *M* and *N* will be even.

Input Format
First line contains three space separated integers, *M*, *N* and *R*, where *M* is the number of rows, *N* is number of columns in matrix, and *R* is the number of times the matrix has to be rotated.
Then *M* lines follow, where each line contains *N* space separated positive integers. These *M* lines represent the matrix.

Constraints
$2 <= M, N <= 300$
$1 <= R <= 10^9$
min(*M, N*) % 2 == 0
$1 <= a_{ij} <= 10^8$, where $i \in [1..M]$ & $j \in [1..N]$

Output Format
Print the rotated matrix.

Sample Input #00

```
4 4 1
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

Sample Output #00

```
2 3 4 8
1 7 11 12
5 6 10 16
9 13 14 15
```

Sample Input #01

```
4 4 2
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

Sample Output #01

```
3 4 8 12
2 11 10 16
1 7 6 15
5 9 13 14
```

Sample Input #02

```
5 4 7
1 2 3 4
7 8 9 10
13 14 15 16
19 20 21 22
25 26 27 28
```

Sample Output #02

```
28 27 26 25
22 9 15 19
16 8 21 13
10 14 20 7
4 3 2 1
```

Sample Input #03

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

Sample Output #03

```
1 1
1 1
```

Explanation

Sample Case #00: Here is an illustration of what happens when the matrix is rotated once.

```
 1  2  3  4      2  3  4  8
 5  6  7  8      1  7 11 12
 9 10 11 12  ->  5  6 10 16
13 14 15 16      9 13 14 15
```

Sample Case #01: Here is what happens when to the matrix after two rotations.

```
 1  2  3  4      2  3  4  8      3  4  8 12
 5  6  7  8      1  7 11 12      2 11 10 16
 9 10 11 12  ->  5  6 10 16  ->  1  7  6 15
13 14 15 16      9 13 14 15      5  9 13 14
```

Sample Case #02: Following are the intermediate states.

```
 1  2  3  4      2  3  4 10    3  4 10 16    4 10 16 22
 7  8  9 10      1  9 15 16    2 15 21 22    3 21 20 28
13 14 15 16 ->  7  8 21 22 -> 1  9 20 28 -> 2 15 14 27 ->
19 20 21 22    13 14 20 28    7  8 14 27    1  9  8 26
25 26 27 28    19 25 26 27    13 19 25 26   7 13 19 25



10 16 22 28    16 22 28 27    22 28 27 26    28 27 26 25
 4 20 14 27    10 14  8 26    16  8  9 25    22  9 15 19
 3 21  8 26 ->  4 20  9 25 -> 10 14 15 19 -> 16  8 21 13
 2 15  9 25     3 21 15 19     4 20 21 13    10 14 20  7
 1  7 13 19     2  1  7 13     3  2  1  7     4  3  2  1
```

Sample Case #03: As all elements are same, any rotation will reflect the same matrix.

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 4.

Minimum Distances

Consider an array of n integers, $A = [a_0, a_1, \ldots, a_{n-1}]$. The distance between two indices, i and j, is denoted by $d_{i,j} = |i - j|$.

Given A, find the *minimum* $d_{i,j}$ such that $a_i = a_j$ and $i \neq j$. In other words, find the minimum distance between any pair of equal elements in the array. If no such value exists, print -1.

Note: $|a|$ denotes the absolute value of a.

Input Format

The first line contains an integer, n, denoting the size of array A.
The second line contains n space-separated integers describing the respective elements in array A.

Constraints

- $1 \leq n \leq 10^3$
- $1 \leq a_i \leq 10^5$

Output Format

Print a single integer denoting the minimum $d_{i,j}$ in A; if no such value exists, print -1.

Sample Input

```
6
7 1 3 4 1 7
```

Sample Output

```
3
```

Explanation
Here, we have two options:

- a_1 and a_4 are both 1, so $d_{1,4} = |1 - 4| = 3$.
- a_0 and a_5 are both 7, so $d_{0,5} = |0 - 5| = 5$.

The answer is $min(3, 5) = 3$.

Sequence Equation

Given a sequence of n integers, $p(1), p(2), \ldots, p(n)$ where each element is distinct and satisfies $1 \leq p(x) \leq n$. For *each* x where $1 \leq x \leq n$, find any integer y such that $p(p(y)) \equiv x$ and print the value of y on a new line.

For example, assume the sequence $p = [5, 2, 1, 3, 4]$. Each value of x between 1 and 5, the length of the sequence, is analyzed as follows:

1. $x = 1 \equiv p[3], p[4] = 3$, so $p[p[4]] = 1$
2. $x = 2 \equiv p[2], p[2] = 2$, so $p[p[2]] = 2$
3. $x = 3 \equiv p[4], p[5] = 4$, so $p[p[5]] = 3$
4. $x = 4 \equiv p[5], p[1] = 5$, so $p[p[1]] = 4$
5. $x = 5 \equiv p[1], p[3] = 1$, so $p[p[3]] = 5$

The values for y are $[4, 2, 5, 1, 3]$.

Function Description

Complete the *permutationEquation* function in the editor below. It should return an array of integers that represent the values of y.

permutationEquation has the following parameter(s):

- *p*: an array of integers

Input Format

The first line contains an integer n, the number of elements in the sequence.
The second line contains n space-separated integers $p[i]$ where $1 \leq i \leq n$.

Constraints

- $1 \leq n \leq 50$
- $1 \leq p[i] \leq 50$, where $1 \leq i \leq n$.
- Each element in the sequence is distinct.

Output Format

For each x from 1 to n, print an integer denoting any valid y satisfying the equation $p(p(y)) \equiv x$ on a new line.

Sample Input 0

```
3
2 3 1
```

Sample Output 0

```
2
3
1
```

Explanation 0

Given the values of $p(1) = 2$, $p(2) = 3$, and $p(3) = 1$, we calculate and print the following values for each x from 1 to n:

1. $x = 1 \equiv p(3) = p(p(2)) = p(p(y))$, so we print the value of $y = 2$ on a new line.
2. $x = 2 \equiv p(1) = p(p(3)) = p(p(y))$, so we print the value of $y = 3$ on a new line.
3. $x = 3 \equiv p(2) = p(p(1)) = p(p(y))$, so we print the value of $y = 1$ on a new line.

Sample Input 1

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

Sample Output 1

```
1
3
5
4
2
```

Picking Numbers

Given an array of integers, find and print the maximum number of integers you can select from the array such that the absolute difference between any two of the chosen integers is ≤ 1.

Input Format

The first line contains a single integer, n, denoting the size of the array.
The second line contains n space-separated integers describing the respective values of $a_0, a_1, \ldots, a_{n-1}$.

Constraints

- $2 \leq n \leq 100$
- $0 < a_i < 100$
- The answer will be ≥ 2.

Output Format

A single integer denoting the maximum number of integers you can choose from the array such that the absolute difference between any two of the chosen integers is ≤ 1.

Sample Input 0

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

Sample Output 0

```
3
```

Explanation 0

We choose the following multiset of integers from the array: $\{4, 3, 3\}$. Each pair in the multiset has an absolute difference ≤ 1 (i.e., $|4 - 3| = 1$ and $|3 - 3| = 0$), so we print the number of chosen integers, 3, as our answer.

Sample Input 1

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

Sample Output 1

```
5
```

Explanation 1

We choose the following multiset of integers from the array: $\{1, 2, 2, 1, 2\}$. Each pair in the multiset has an absolute difference ≤ 1 (i.e., $|1 - 2| = 1$, $|1 - 1| = 0$, and $|2 - 2| = 0$), so we print the number of chosen integers, 5, as our answer.

Repeated String

Lilah has a string, s, of lowercase English letters that she repeated infinitely many times.

Given an integer, n, find and print the number of letter `a`'s in the first n letters of Lilah's infinite string.

Input Format

The first line contains a single string, s.
The second line contains an integer, n.

Constraints

- $1 \leq |s| \leq 100$
- $1 \leq n \leq 10^{12}$
- For 25% of the test cases, $n \leq 10^6$.

Output Format

Print a single integer denoting the number of letter `a`'s in the first n letters of the infinite string created by repeating s infinitely many times.

Sample Input 0

```
aba
10
```

Sample Output 0

```
7
```

Explanation 0
The first $n = 10$ letters of the infinite string are `abaabaabaa`. Because there are 7 `a`'s, we print 7 on a new line.

Sample Input 1

```
a
1000000000000
```

Sample Output 1

```
1000000000000
```

Explanation 1
Because all of the first $n = 1000000000000$ letters of the infinite string are `a`, we print 1000000000000 on a new line.

Save the Prisoner!

A jail has a number of prisoners and a number of treats to pass out to them. Their jailer decides the fairest way to divide the treats is to seat the prisoners around a circular table in sequentially numbered chairs. A chair number will be drawn from a hat. Beginning with the prisoner in that chair, one candy will be handed to each prisoner sequentially around the table until all have been distributed.

The jailer is playing a little joke, though. The last piece of candy looks like all the others, but it tastes *awful*. Determine the chair number occupied by the prisoner who will receive that candy.

For example, there are 4 prisoners and 6 pieces of candy. The prisoners arrange themselves in seats numbered 1 to 4. Let's suppose two is drawn from the hat. Prisoners receive candy at positions $2, 3, 4, 1, 2, 3$. The prisoner to be warned sits in chair number 3.

Function Description

Complete the *saveThePrisoner* function in the editor below. It should return an integer representing the chair number of the prisoner to warn.

saveThePrisoner has the following parameter(s):

- *n*: an integer, the number of prisoners
- *m*: an integer, the number of sweets
- *s*: an integer, the chair number to begin passing out sweets from

Input Format

The first line contains an integer, t, denoting the number of test cases.
The next t lines each contain 3 space-separated integers:
- n: the number of prisoners
- m: the number of sweets
- s: the chair number to start passing out treats at

Constraints

- $1 \leq t \leq 100$
- $1 \leq n \leq 10^9$
- $1 \leq m \leq 10^9$
- $1 \leq s \leq n$

Output Format

For each test case, print the chair number of the prisoner who receives the *awful treat* on a new line.

Sample Input 0

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

Sample Output 0

```
2
3
```

Explanation 0

In first query, there are $n = 5$ prisoners and $m = 2$ sweets. Distribution starts at seat number $s = 1$. Prisoners in seats numbered 1 and 2 get sweets. Warn prisoner 2.
In the second query, distribution starts at seat 2 so prisoners in seats 2 and 3 get sweets. Warn prisoner 3 .

Sample Input 1

```
2
7 19 2
3 7 3
```

Sample Output 1

```
6
3
```

Explanation 1

In the first test case, there are $n = 7$ prisoners, $m = 19$ sweets and they are passed out starting at chair $s = 2$. The candies go all around twice and there are 5 more candies passed to each prisoner from seat 2 to seat 6.

In the second test case, there are $n = 3$ prisoners, $m = 7$ candies and they are passed out starting at seat $s = 3$. They go around twice, and there is one more to pass out to the prisoner at seat 3.

Sherlock and Squares

Watson likes to challenge Sherlock's math ability. He will provide a starting and ending value describing a range of integers. Sherlock must determine the number of *square integers* within that range, inclusive of the endpoints.

Note: A square integer is an integer which is the square of an integer, e.g. $1, 4, 9, 16, 25$.

For example, the range is $a = 24$ and $b = 49$, inclusive. There are three square integers in the range: $25, 36$ and 49.

Function Description

Complete the *squares* function in the editor below. It should return an integer representing the number of square integers in the inclusive range from a to b.

squares has the following parameter(s):

- *a*: an integer, the lower range boundary
- *b*: an integer, the uppere range boundary

Input Format

The first line contains q, the number of test cases.
Each of the next q lines contains two space-separated integers denoting a and b, the starting and ending integers in the ranges.

Constraints

$1 \leq q \leq 100$
$1 \leq a \leq b \leq 10^9$

Output Format

For each test case, print the number of square integers in the range on a new line.

Sample Input

```
2
3 9
17 24
```

Sample Output

```
2
0
```

Explanation

Test Case #00: In range $[3, 9]$, 4 and 9 are the two square integers.
Test Case #01: In range $[17, 24]$, there are no square integers.

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.

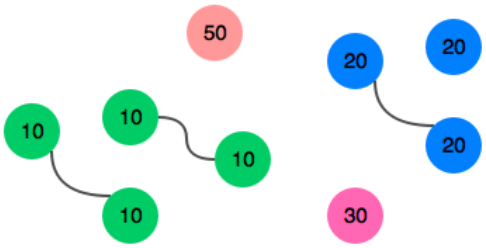
Sample Input

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

Sample Output

```
3
```

Explanation

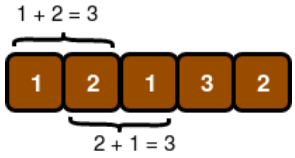


John can match three pairs of socks.

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

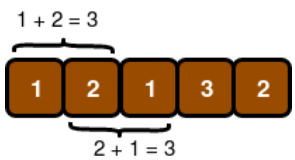
```
5
1 2 1 3 2
3 2
```

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.

The Grid Search

Given a 2D array of digits, try to find the occurrence of a given 2D pattern of digits. For example, consider the following 2D matrix:

```
1234567890
09876543 21
11111111 11
11111111 11
2222222222
```

Assume we need to look for the following 2D pattern:

```
876543
111111
111111
```

If we scan through the original array, we observe that the 2D pattern begins at the second row and the third column of the larger grid (the 8 in the second row and third column of the larger grid is the top-left corner of the pattern we are searching for).

So, a 2D pattern of P digits is said to be present in a larger grid G, if the latter contains a contiguous, rectangular 2D grid of digits matching with the pattern P, similar to the example shown above.

Input Format

The first line contains an integer, T, which is the number of test cases. T test cases follow, each having a structure as described below:
The first line contains two space-separated integers, R and C, indicating the number of rows and columns in the grid G, respectively.
This is followed by R lines, each with a string of C digits, which represent the grid G.
The following line contains two space-separated integers, r and c, indicating the number of rows and columns in the pattern grid P.
This is followed by r lines, each with a string of c digits, which represent the pattern P.

Constraints

$1 \leq T \leq 5$
$1 \leq R, r, C, c \leq 1000$
$1 \leq r \leq R$
$1 \leq c \leq C$

Output Format

Display 'YES' or 'NO', depending on whether (or not) you find that the larger grid G contains the rectangular pattern P. The evaluation will be case sensitive.

Sample Input

```
2
10 10
7283455864
6731158619
8988242643
3830589324
2229505813
5633845374
6473530293
7053106601
0834282956
4607924137
```

```
3 4
9505
3845
3530
15 15
400453592126560
114213133098692
474386082879648
522356951189169
887109450487496
252802633388782
502771484966748
075975207693780
511799789562806
404007454272504
549043809916080
962410809534811
445893523733475
768705303214174
650629270887160
2 2
99
99
```

Sample Output

```
YES
NO
```

Explanation

The first test in the input file is:

```
10 10
7283455864
6731158619
8988242643
3830589324
2229505813
5633845374
6473530293
7053106601
0834282956
4607924137
3 4
9505
3845
3530
```

As one may see, the given 2D grid is indeed present in the larger grid, as marked in bold below.

7283455864
6731158619
8988242643
3830589324
222**9505**813
563**3845**374
647**3530**293
7053106601
0834282956
4607924137

The second test in the input file is:

```
15 15
400453592126560
114213133098692
474386082879648
522356951189169
887109450487496
252802633388782
502771484966748
075975207693780
```

```
511799789562806
404007454272504
549043809916080
962410809534811
445893523733475
768705303214174
650629270887160
2 2
99
99
```

The search pattern is:

```
99
99
```

This cannot be found in the larger grid.

The Hurdle Race

Dan is playing a video game in which his character competes in a hurdle race. Hurdles are of varying heights, and Dan has a maximum height he can jump. There is a magic potion he can take that will increase his maximum height by 1 unit for each dose. How many doses of the potion must he take to be able to jump all of the hurdles.

Given an array of hurdle heights $height$, and an initial maximum height Dan can jump, k, determine the minimum number of doses Dan must take to be able to clear all the hurdles in the race.

For example, if $height = [1, 2, 3, 3, 2]$ and Dan can jump 1 unit high naturally, he must take $3 - 1 = 2$ doses of potion to be able to jump all of the hurdles.

Input Format

Complete the function $hurdleRace$ in the editor below. The code stub reads the input at passes it to the function. Inputs are in the following format:

The first line contains two space-separated integers n and k, the number of hurdles and the maximum height Dan can jump naturally.
The second line contains n space-separated integers $height[i]$ where $0 \leq i < n$.

Constraints

- $1 \leq n, k \leq 100$
- $1 \leq height[i] \leq 100$

Output Format

Print an integer denoting the minimum doses of magic potion Dan must drink to complete the hurdle race.

Sample Input 0

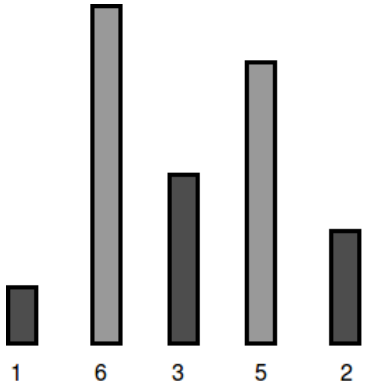
```
5 4
1 6 3 5 2
```

Sample Output 0

```
2
```

Explanation 0

Dan's character can jump a maximum of $k = 4$ units, but the tallest hurdle has a height of $h_1 = 6$:



To be able to jump all the hurdles, Dan must drink $6 - 4 = 2$ doses.

Sample Input 1

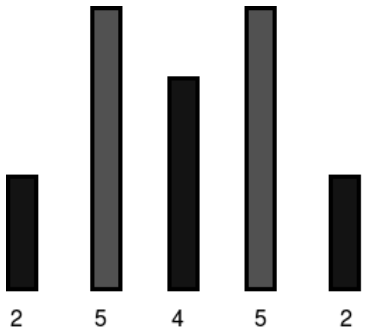
```
5 7
2 5 4 5 2
```

Sample Output 1

```
0
```

Explanation 1

Dan's character can jump a maximum of $k = 7$ units, which is enough to cross all the hurdles:



Because he can already jump all the hurdles, Dan needs to drink 0 doses.

Utopian Tree

The Utopian Tree goes through *2* cycles of growth every year. Each spring, it *doubles* in height. Each summer, its height increases by *1* meter.

Laura plants a Utopian Tree sapling with a height of *1* meter at the onset of spring. How tall will her tree be after N growth cycles?

Input Format

The first line contains an integer, T, the number of test cases.
T subsequent lines each contain an integer, N, denoting the number of cycles for that test case.

Constraints
$1 \leq T \leq 10$
$0 \leq N \leq 60$

Output Format

For each test case, print the height of the Utopian Tree after N cycles. Each height must be printed on a new line.

Sample Input

```
3
0
1
4
```

Sample Output

```
1
2
7
```

Explanation

There are *3* test cases.

In the first case ($N = 0$), the initial height ($H = 1$) of the tree remains unchanged.

In the second case ($N = 1$), the tree doubles in height and is 2 meters tall after the spring cycle.

In the third case ($N = 4$), the tree doubles its height in spring ($H = 2$), then grows a meter in summer ($H = 3$), then doubles after the next spring ($H = 6$), and grows another meter after summer ($H = 7$). Thus, at the end of 4 cycles, its height is 7 meters.

Angry Professor

A Discrete Mathematics professor has a class of N students. Frustrated with their lack of discipline, he decides to cancel class if fewer than K students are present when class starts.

Given the arrival time of each student, determine if the class is canceled.

Input Format

The first line of input contains T, the number of test cases.

Each test case consists of two lines. The first line has two space-separated integers, N (students in the class) and K (the cancelation threshold). The second line contains N space-separated integers ($a_1, a_2, \ldots, a_N$) describing the arrival times for each student.

Note: Non-positive arrival times ($a_i \leq 0$) indicate the student arrived early or on time; positive arrival times ($a_i > 0$) indicate the student arrived a_i minutes late.

Constraints

- $1 \leq T \leq 10$
- $1 \leq N \leq 1000$
- $1 \leq K \leq N$
- $-100 \leq a_i \leq 100, where\ i \in [1, N]$

Output Format

For each test case, print the word `YES` if the class is canceled or `NO` if it is not.

Note
If a student arrives exactly on time $(a_i = 0)$, the student is considered to have entered before the class started.

Sample Input

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

Sample Output

```
YES
NO
```

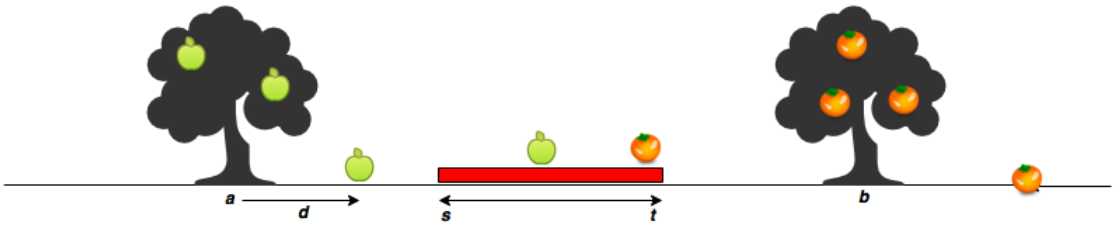
Explanation

For the first test case, $K = 3$. The professor wants at least 3 students in attendance, but only 2 have arrived on time (-3 and -1). Thus, the class is canceled.

For the second test case, $K = 2$. The professor wants at least 2 students in attendance, and there are 2 who have arrived on time (0 and -1). Thus, the class is *not* canceled.

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.

Beautiful Days at the Movies

Lily likes to play games with integers. She has created a new game where she determines the difference between a number and its reverse. For instance, given the number 12, its reverse is 21. Their difference is 9. The number 120 reversed is 21, and their difference is 99.

She decides to apply her game to decision making. She will look at a numbered range of days and will only go to a movie on a *beautiful day*.

Given a range of numbered days, $[i \ldots j]$ and a number k, determine the number of days in the range that are *beautiful*. Beautiful numbers are defined as numbers where $|i - reverse(i)|$ is evenly divisible by k. If a day's value is a beautiful number, it is a beautiful day. Print the number of beautiful days in the range.

Function Description

Complete the *beautifulDays* function in the editor below. It must return the number of beautiful days in the range.

beautifulDays has the following parameter(s):

- *i*: the starting day number
- *j*: the ending day number
- *k*: the divisor

Input Format

A single line of three space-separated integers describing the respective values of i, j, and k.

Constraints

- $1 \leq i \leq j \leq 2 \times 10^6$
- $1 \leq k \leq 2 \times 10^9$

Output Format

Print the number of *beautiful* days in the inclusive range between i and j.

Sample Input

```
20 23 6
```

Sample Output

```
2
```

Explanation

Lily may go to the movies on days 20, 21, 22, and 23. We perform the following calculations to determine which days are *beautiful*:

- Day 20 is *beautiful* because the following evaluates to a whole number: $\frac{|20-02|}{6} = \frac{18}{6} = 3$

- Day 21 is *not beautiful* because the following doesn't evaluate to a whole number: $\frac{|21-12|}{6} = \frac{9}{6} = 1.5$
- Day 22 is *beautiful* because the following evaluates to a whole number: $\frac{|22-22|}{6} = \frac{0}{6} = 0$
- Day 23 is *not beautiful* because the following doesn't evaluate to a whole number: $\frac{|23-32|}{6} = \frac{9}{6} = 1.5$

Only two days, 20 and 22, in this interval are beautiful. Thus, we print 2 as our answer.

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 $0 \leq i < n$.
The third line contains m distinct space-separated integers describing $b[j]$ where $0 \leq j < m$.

Constraints

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

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.

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.

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:

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

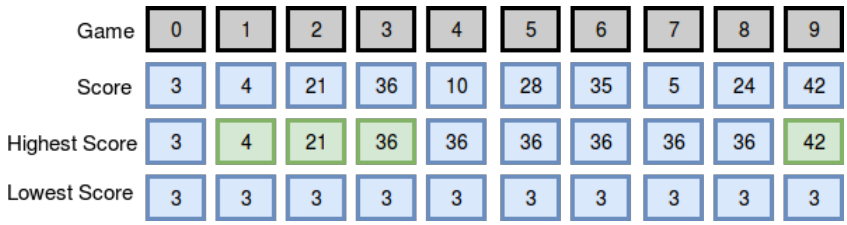
```
10
3 4 21 36 10 28 35 5 24 42
```

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.

Cats and a Mouse

Two cats and a mouse are at various positions on a line. You will be given their starting positions. Your task is to determine which cat will reach the mouse first, assuming the mouse doesn't move and the cats travel at equal speed. If the cats arrive at the same time, the mouse will be allowed to move and it will escape while they fight.

You are given q queries in the form of x, y, and z representing the respective positions for cats A and B, and for mouse C. Complete the function $catAndMouse$ to return the appropriate answer to each query, which will be printed on a new line.

- If cat A catches the mouse first, print `Cat A`.
- If cat B catches the mouse first, print `Cat B`.
- If both cats reach the mouse at the same time, print `Mouse C` as the two cats fight and mouse escapes.

Input Format

The first line contains a single integer, q, denoting the number of queries.
Each of the q subsequent lines contains three space-separated integers describing the respective values of x (cat A's location), y (cat B's location), and z (mouse C's location).

Constraints

- $1 \leq q \leq 100$
- $1 \leq x, y, z \leq 100$

Output Format

For each query, return `Cat A` if cat A catches the mouse first, `Cat B` if cat B catches the mouse first, or `Mouse C` if the mouse escapes.

Sample Input 0

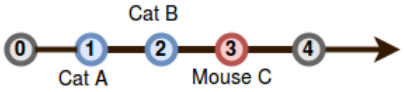
```
2
1 2 3
1 3 2
```

Sample Output 0

```
Cat B
Mouse C
```

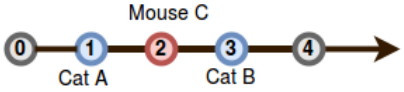
Explanation 0

Query 0: The positions of the cats and mouse are shown below:



Cat B will catch the mouse first, so we print `Cat B` on a new line.

Query 1: In this query, cats A and B reach mouse C at the exact same time:



Because the mouse escapes, we print `Mouse C` on a new line.

Cavity Map

You are given a square map as a matrix of integer strings. Each cell of the map has a value denoting its depth. We will call a cell of the map a *cavity* if and only if this cell is not on the border of the map and each cell adjacent to it has *strictly smaller depth*. Two cells are adjacent if they have a common side, or *edge*.

Find all the cavities on the map and replace their depths with the uppercase character **X**.

For example, given a matrix:

```
989
191
111
```

You should return:

```
989
1X1
111
```

The center cell was deeper than those on its edges: *[8,1,1,1]*. The deep cells in the top two corners don't share an edge with the center cell.

Function Description

Complete the *cavityMap* function in the editor below. It should return an array of strings, each representing a line of the completed map.

cavityMap has the following parameter(s):

- *grid*: an array of strings, each representing a row of the grid

Input Format

The first line contains an integer n, the number of rows and columns in the map.

Each of the following n lines (*rows*) contains n positive digits without spaces (*columns*) representing depth at $map[row, column]$.

Constraints

$1 \leq n \leq 100$

Output Format

Output n lines, denoting the resulting map. Each cavity should be replaced with the character `X`.

Sample Input

```
4
1112
1912
1892
1234
```

Sample Output

```
1112
1X12
18X2
```

```
1234
```

Explanation

The two cells with the depth of 9 are not on the border and are surrounded on all sides by shallower cells. Their values have been replaced by X.

Circular Array Rotation

John Watson knows of an operation called a *right circular rotation* on an array of integers. One rotation operation moves the last array element to the first position and shifts all remaining elements right one. To test Sherlock's abilities, Watson provides Sherlock with an array of integers. Sherlock is to perform the rotation operation a number of times then determine the value of the element at a given position.

For each array, perform a number of right circular rotations and return the value of the element at a given index.

Input Format

The first line contains 3 space-separated integers, n, k, and q.
The second line contains n space-separated integers, where each integer i describes array element a_i (where $0 \leq i < n$).
Each of the q subsequent lines contains a single integer denoting m.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq a_i \leq 10^5$
- $1 \leq k \leq 10^5$
- $1 \leq q \leq 500$
- $0 \leq m < n$

Output Format

For each query, print the value of the element at index m of the rotated array on a new line.

Sample Input 0

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

Sample Output 0

```
2
3
1
```

Explanation 0

After the first rotation, the array becomes $[3, 1, 2]$.
After the second (and final) rotation, the array becomes $[2, 3, 1]$.

Let's refer to the array's final state as array $b = [2, 3, 1]$. For each query, we just have to print the value of b_m on a new line:

1. $m = 0$, $b_0 = 2$.
2. $m = 1$, $b_1 = 3$.

3. $m = 2$, $b_2 = 1$.

Climbing the Leaderboard

Alice is playing an arcade game and wants to climb to the top of the leaderboard and wants to track her ranking. The game uses Dense Ranking, so its leaderboard works like this:

- The player with the highest score is ranked number 1 on the leaderboard.
- Players who have equal scores receive the same ranking number, and the next player(s) receive the immediately following ranking number.

For example, the four players on the leaderboard have high scores of 100, 90, 90, and 80. Those players will have ranks 1, 2, 2, and 3, respectively. If Alice's scores are 70, 80 and 105, her rankings after each game are 4^{th}, 3^{rd} and 1^{st}.

Function Description

Complete the *climbingLeaderboard* function in the editor below. It should return an integer array where each element $res[j]$ represents Alice's rank after the j^{th} game.

climbingLeaderboard has the following parameter(s):

- *scores*: an array of integers that represent leaderboard scores
- *alice*: an array of integers that represent Alice's scores

Input Format

The first line contains an integer n, the number of players on the leaderboard.
The next line contains n space-separated integers $scores[i]$, the leaderboard scores in decreasing order.
The next line contains an integer, m, denoting the number games Alice plays.
The last line contains m space-separated integers $alice[j]$, the game scores.

Constraints

- $1 \leq n \leq 2 \times 10^5$
- $1 \leq m \leq 2 \times 10^5$
- $0 \leq scores[i] \leq 10^9$ for $0 \leq i < n$
- $0 \leq alice[j] \leq 10^9$ for $0 \leq j < m$
- The existing leaderboard, $scores$, is in *descending* order.
- Alice's scores, $alice$, are in *ascending* order.

Subtask

For 60% of the maximum score:

- $1 \leq n \leq 200$
- $1 \leq m \leq 200$

Output Format

Print m integers. The j^{th} integer should indicate Alice's rank after playing the j^{th} game.

Sample Input 0

```
7
100 100 50 40 40 20 10
4
5 25 50 120
```

Sample Output 0

```
6
4
2
1
```

Explanation 0

Alice starts playing with 7 players already on the leaderboard, which looks like this:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
3	Ritika	40
3	Tom	40
4	Heraldo	20
5	Riley	10

After Alice finishes game 0, her score is 5 and her ranking is 6:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
3	Ritika	40
3	Tom	40
4	Heraldo	20
5	Riley	10
6	**Alice**	**5**

After Alice finishes game 1, her score is 25 and her ranking is 4:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
3	Ritika	40
3	Tom	40
4	**Alice**	**25**
5	Heraldo	20
6	Riley	10

After Alice finishes game 2, her score is 50 and her ranking is tied with Caroline at 2:

Rank	Name	Score
1	Emma	100
1	David	100
2	Caroline	50
2	**Alice**	**50**
3	Ritika	40
3	Tom	40
4	Heraldo	20
5	Riley	10

After Alice finishes game 3, her score is 120 and her ranking is 1:

Rank	Name	Score
1	**Alice**	**120**
2	Emma	100
2	David	100
3	Caroline	50
4	Ritika	40
4	Tom	40
5	Heraldo	20
6	Riley	10

Sample Input 1

```
6
100 90 90 80 75 60
5
50 65 77 90 102
```

Sample Output 1

```
6
5
4
2
1
```