Problem A. Sorting: Bubble Sort

OS Linux

Consider the following version of Bubble Sort:

```
for (int i = 0; i < n; i++) {

    for (int j = 0; j < n - 1; j++) {
        // Swap adjacent elements if they are in decreasing order
        if (a[j] > a[j + 1]) {
            swap(a[j], a[j + 1]);
        }
    }

}
```

Given an array of integers, sort the array in ascending order using the *Bubble Sort* algorithm above. Once sorted, print the following three lines:

1. `Array is sorted in numSwaps swaps.`, where $numSwaps$ is the number of swaps that took place.
2. `First Element: firstElement`, where $firstElement$ is the *first* element in the sorted array.
3. `Last Element: lastElement`, where $lastElement$ is the *last* element in the sorted array.

Hint: To complete this challenge, you must add a variable that keeps a running tally of *all* swaps that occur during execution.

Example
$a = [6, 4, 1]$

```
swap    a
0       [6,4,1]
1       [4,6,1]
2       [4,1,6]
3       [1,4,6]
```

The steps of the bubble sort are shown above. It took 3 swaps to sort the array. Output is:

```
Array is sorted in 3 swaps.
First Element: 1
Last Element: 6
```

Function Description

Complete the function *countSwaps* in the editor below.

countSwaps has the following parameter(s):

- *int a[n]:* an array of integers to sort

Prints

Print the three lines required, then return. No return value is expected.

Input Format

The first line contains an integer, n, the size of the array a.
The second line contains n space-separated integers $a[i]$.

Constraints

- $2 \leq n \leq 600$
- $1 \leq a[i] \leq 2 \times 10^6$

Output Format

Input	Output
STDIN Function ----- -------- 3 a[] size n = 3 1 2 3 a = [1, 2, 3]	Array is sorted in 0 swaps. First Element: 1 Last Element: 3

Explanation 0
The array is already sorted, so 0 swaps take place.

Input	Output
3 3 2 1	Array is sorted in 3 swaps. First Element: 1 Last Element: 3

Explanation 1
The array is *not sorted*, and its initial values are: $\{3, 2, 1\}$. The following 3 swaps take place:

1. $\{3, 2, 1\} \rightarrow \{2, 3, 1\}$

2. $\{2, 3, 1\} \rightarrow \{2, 1, 3\}$
3. $\{2, 1, 3\} \rightarrow \{1, 2, 3\}$

At this point the array is sorted and the three lines of output are printed to stdout.

Problem B. Sorting: Comparator

OS Linux

Comparators are used to compare two objects. In this challenge, you'll create a comparator and use it to sort an array. The *Player* class is provided in the editor below. It has two fields:

1. $name$: a string.
2. $score$: an integer.

Given an array of n *Player* objects, write a comparator that sorts them in order of decreasing score. If 2 or more players have the same score, sort those players alphabetically ascending by name. To do this, you must create a *Checker* class that implements the *Comparator* interface, then write an *int compare(Player a, Player b)* method implementing the Comparator.compare(T o1, T o2) method. In short, when sorting in ascending order, a comparator function returns -1 if $a < b$, 0 if $a = b$, and 1 if $a > b$.

Declare a *Checker* class that implements the *comparator* method as described. It should sort first descending by score, then ascending by name. The code stub reads the input, creates a list of Player objects, uses your method to sort the data, and prints it out properly.

Example
$n = 3$ $data = [[Smith, 20], [Jones, 15], [Jones, 20]]$

Sort the list as $data_{sorted} = [[Jones, 20], [Smith, 20], [Jones, 15]]$. Sort first descending by score, then ascending by name.

Input Format

The first line contains an integer, n, the number of players.
Each of the next n lines contains a player's $name$ and $score$, a string and an integer.

Constraints

- $0 \leq score \leq 1000$
- Two or more players can have the same name.
- Player names consist of lowercase English alphabetic letters.

Output Format

You are not responsible for printing any output to stdout. Locked stub code in *Solution* will instantiate a *Checker* object, use it to sort the *Player* array, and print each sorted element.

Input	Output
5 amy 100 david 100 heraldo 50 aakansha 75 aleksa 150	aleksa 150 amy 100 david 100 aakansha 75 heraldo 50

Explanation

The players are first sorted descending by score, then ascending by name.

Problem C. Helpful Maths

Time limit	2000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

Xenia the beginner mathematician is a third year student at elementary school. She is now learning the addition operation.

The teacher has written down the sum of multiple numbers. Pupils should calculate the sum. To make the calculation easier, the sum only contains numbers 1, 2 and 3. Still, that isn't enough for Xenia. She is only beginning to count, so she can calculate a sum only if the summands follow in non-decreasing order. For example, she can't calculate sum 1+3+2+1 but she can calculate sums 1+1+2 and 3+3.

You've got the sum that was written on the board. Rearrange the summans and print the sum in such a way that Xenia can calculate the sum.

Input

The first line contains a non-empty string s — the sum Xenia needs to count. String s contains no spaces. It only contains digits and characters "+". Besides, string s is a correct sum of numbers 1, 2 and 3. String s is at most 100 characters long.

Output

Print the new sum that Xenia can count.

Examples

Input	Output
3+2+1	1+2+3

Input	Output
1+1+3+1+3	1+1+1+3+3

Input	Output
2	2

Problem D. Laptops

Time limit	1000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

One day Dima and Alex had an argument about the price and quality of laptops. Dima thinks that the more expensive a laptop is, the better it is. Alex disagrees. Alex thinks that there are two laptops, such that the price of the first laptop is less (strictly smaller) than the price of the second laptop but the quality of the first laptop is higher (strictly greater) than the quality of the second laptop.

Please, check the guess of Alex. You are given descriptions of n laptops. Determine whether two described above laptops exist.

Input

The first line contains an integer n ($1 \le n \le 10^5$) — the number of laptops.

Next n lines contain two integers each, a_i and b_i ($1 \le a_i, b_i \le n$), where a_i is the price of the i-th laptop, and b_i is the number that represents the quality of the i-th laptop (the larger the number is, the higher is the quality).

All a_i are distinct. All b_i are distinct.

Output

If Alex is correct, print "`Happy Alex`", otherwise print "`Poor Alex`" (without the quotes).

Examples

Input	Output
2 1 2 2 1	Happy Alex

Problem E. Sort the Array

Time limit	1000 ms
Mem limit	262144 kB
Input file	stdin
Output file	stdout

Being a programmer, you like arrays a lot. For your birthday, your friends have given you an array a consisting of n **distinct** integers.

Unfortunately, the size of a is too small. You want a bigger array! Your friends agree to give you a bigger array, but only if you are able to answer the following question correctly: is it possible to sort the array a (in increasing order) by reversing **exactly one** segment of a? See definitions of segment and reversing in the notes.

Input

The first line of the input contains an integer n ($1 \le n \le 10^5$) — the size of array a.

The second line contains n distinct space-separated integers: $a[1], a[2], \ldots, a[n]$ ($1 \le a[i] \le 10^9$).

Output

Print "`yes`" or "`no`" (without quotes), depending on the answer.

If your answer is "`yes`", then also print two space-separated integers denoting start and end (start must not be greater than end) indices of the segment to be reversed. If there are multiple ways of selecting these indices, print any of them.

Examples

Input	Output
3 3 2 1	yes 1 3

Input	Output
4 2 1 3 4	yes 1 2

Input	Output
4 3 1 2 4	no

Input	Output
2 1 2	yes 1 1

Note

Sample 1. You can reverse the entire array to get $[1, 2, 3]$, which is sorted.

Sample 3. No segment can be reversed such that the array will be sorted.

Definitions

A segment $[l, r]$ of array a is the sequence $a[l], a[l + 1], ..., a[r]$.

If you have an array a of size n and you reverse its segment $[l, r]$, the array will become:

$a[1], a[2], ..., a[l - 2], a[l - 1], a[r], a[r - 1], ..., a[l + 1], a[l], a[r + 1], a[r + 2], ..., a[n - 1], a[n]$.

Three of the trouble-makers went to Malaysia this year. A rest house was booked for them. Unlike other rest houses, this rest house was like a normal duplex house. So, it had a kitchen. And the trouble-makers were given all the ingredients to cook, but they had to cook themselves.

None of them had any previous cooking experience, but they became very excited and planned to cook so many delicious foods! Ideas were coming from their minds like rains from clouds. So, they went to the super market and bought a lot of extra ingredients for their great recipes. For example, they bought 20 eggs. The excited trouble-makers returned to the rest house and found that the gas stove was not connected to the gas cylinder. So, they became very sad, because it was not possible for them to connect such complex thing. And so many foods were about to be rotten. But luckily, they found the microwave oven working. So, they tried to boil all the eggs using the microwave oven (may be, first time in history)! And they succeeded to boil the eggs!

Now they have n eggs and a bowl. They put some eggs in the bowl with some water. And after that they put the bowl into the oven to boil the eggs. It's risky to put more than P eggs in the bowl and the bowl can carry at most Q gm of eggs. It takes 12 minutes to boil a bowl of eggs. Now you are given the weight of the eggs in gm, and the trouble-makers have exactly 12 minutes in their hand. You have to find the maximum number of eggs they can boil without taking any risk.

Input

The first line of input will contain T (≤ 100) denoting the number of cases.

Each case starts with 3 integers n ($1 \leq n \leq 30$), P ($1 \leq P \leq 30$) and Q ($1 \leq Q \leq 30$). The next line contains n positive integers (not greater than 10) in non-descending order. These integers denote the weight of the eggs in gm.

Output

For each case, print the case number and the desired result.

Sample Input

```
2
3 2 10
1 2 3
4 5 5
4 4 5 5
```

Sample Output

```
Case 1: 2
Case 2: 1
```

Once upon a time, there lived a chimpanzee called *Luchu Bandor (aka Playboy Chimp)*. Luchu was unhappily married to Bunty Mona, a short but cute little lady chimp. Luchu was tall and handsome — he was feeling uncomfortable taking Bunty to public places along with him. People would stare at them all the while. At one point, Luchu could not stand it anymore and he decided to do some justice to his name. He started looking for *a new hope* in the Lady Chimps' High School. Every day Luchu would climb up a bamboo tree and wait for the morning drill to start. From there he could see each and every lady chimp doing their routine drill. Now, Luchu was looking for the tallest lady chimp that would be shorter than he; he would also like to consider someone a little taller than he. But someone of his same height will never be on his list. Every morning Luchu picks up a line of lady chimps and finds the best two according to his set criterion. His job has been made easy by the fact that the lady chimps in each line are ordered by their height, the shortest one is in the front and the tallest one is at the back. Your task is to help Luchu on one particular day to find two lady chimps: the tallest one shorter than he and the shortest one taller than he.

Input

There will be only one set of input for this problem. The first line of input gives you a number N ($1 \leq N \leq 50000$), the number of lady chimps on the line. In the next line you would have N integers (in the range 1 to $2^{31} - 1$ giving the heights of the N chimps. There would be a single space after every number. You can assume that the chimps are ordered in non-decreasing order of their heights. In the next line you would have an integer Q ($1 \leq Q \leq 25000$) giving the number of queries. Then in the next line Q queries will follow. Then you would have Q numbers giving the height of Luchu! Dont worry, Luchu is from the land where people can have 3 birthdates; Q heights for a chimpanzee will make no difference here. The Q numbers are listed on a line and their range from 1 to $2^{31} - 1$, and as before you would find a single space after every query number. The query numbers are not supposed to come in any particular order.

Output

For each query height, print two numbers in one line. The first one would be the height of the tallest lady chimp that is shorter than Luchu, and the next number would be the height of the shortest lady chimp that is taller than he. These two numbers are to be separated by a single space. Whenever it is impossible to find any of these two heights, replace that height with an uppercase '`X`'.

Sample Input

```
4
1 4 5 7
4
4 6 8 10
```

Sample Output

```
1 5
5 7
7 X
7 X
```

Problem H. Points in Segments

Time limit 2000 ms
Mem limit 65536 kB

Given **n** points (1 dimensional) and **q** segments, you have to find the number of points that lie in each of the segments. A point p_i will lie in a segment **A B** if $A \le p_i \le B$.

For example if the points are **1, 4, 6, 8, 10**. And the segment is **0** to **5**. Then there are **2** points that lie in the segment.

Input

Input starts with an integer **T (≤ 5)**, denoting the number of test cases.

Each case starts with a line containing two integers **n (1 ≤ n ≤ 10^5)** and **q (1 ≤ q ≤ 50000)**. The next line contains **n** space separated integers denoting the points in ascending order. All the integers are distinct and each of them range in **[0, 10^8]**.

Each of the next **q** lines contains two integers A_k B_k **(0 ≤ A_k ≤ B_k ≤ 10^8)** denoting a segment.

Output

For each case, print the case number in a single line. Then for each segment, print the number of points that lie in that segment.

Sample

Input	Output
1 5 3 1 4 6 8 10 0 5 6 10 7 100000	Case 1: 2 3 2

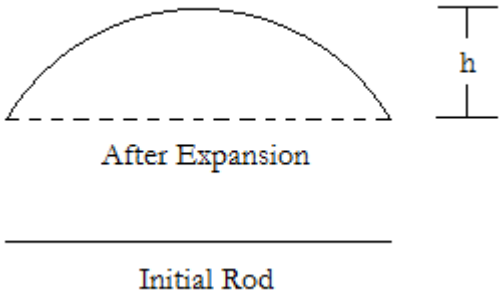
Note

Dataset is huge, use faster I/O methods.

Problem I. Expanding Rods

Time limit 1000 ms
Mem limit 65536 kB

When a thin rod of length **L** is heated **n** degrees, it expands to a new length `L' = (1 + n * C) * L`, where **C** is the coefficient of heat expansion.



When a thin rod is mounted on two solid walls and then heated, it expands and takes the shape of a circular segment, the original rod being the chord of the segment.

Your task is to compute the distance by which the center of the rod is displaced. That means you have to calculate **h** as in the picture.

Input

Input starts with an integer **T (≤ 20)**, denoting the number of test cases.

Each case contains three non-negative real numbers: the initial length of the rod in millimeters **L**, the temperature change in degrees **n** and the coefficient of heat expansion of the material **C**. Input data guarantee that no rod expands by more than one-half of its original length. All the numbers will be between 0 and 1000 and there can be at most 5 digits after the decimal point.

Output

For each case, print the case number and the displacement of the center of the rod in a single line. Errors less than 10^{-6} will be ignored.

Sample

Input	Output
3 1000 100 0.0001 150 10 0.00006 10 0 0.001	Case 1: 61.3289915 Case 2: 2.2502024857 Case 3: 0