NSU INDIVIDUAL CONTEST #1

A. Adding Commas

Pinku is working with large numbers **N (1 <= N <= 2,000,000,000)** like 153920529 and realizes that the numbers would be a lot easier to read with commas inserted every three digits (as is normally done in the USA; some countries prefer to use periods every three digits). Thus, she would like to add commas: 153,920,529 . Please write a program that does this.

INPUT FORMAT

First line of the input will contain the number of testcases **T**.
T lines will follow. Each of them will contain an integer **N** as described in the problem statement.

OUTPUT FORMAT

Output will have **T** lines, i-th line being the answer of i-th input.

SAMPLE INPUT

```
153920529
```

SAMPLE OUTPUT

```
153,920,529
```

B. Alignment of Planets

Actually, this problem is about alignment of **N (1 <= N <= 770)** cows numbered **1..N** who are grazing in their field that is about **15,000x15,000** units. Their grazing locations all fall on integer coordinates in a standard **x, y** scheme (coordinates are in the range **0..15,000**).

Nabila looks up and notices that she is exactly lined up with Sajjad and Osman. She wonders how many groups of three aligned cows exist within the field.

Given the locations of all the cows (no two cows occupy the same location), figure out all sets of three cows are exactly collinear. Keep track of the sets, sorting the cows in each set by their ID number, lowest first. Then sort the sets by the three ID numbers (lowest first), breaking ties by examining the second and third ID numbers.

INPUT FORMAT
First line of the input will contain the number of testcases **T**.
T sets of input will follow. Each set will have the following format:
Line 1: A single integer, **N**
Lines 2..N+1: Line i+1 describes cow i's location with two space-separated integers that are her **x** and **y** coordinates.

OUTPUT FORMAT
T sets of output one after another. Each set will have the following format:

Line 1: A single integer that is the number of sets of three cows that are exactly collinear. A set of four collinear cows would, of course, result in four sets of three collinear cows.

Lines 2..?: Each line contains three space-separated integers that are the cow ID numbers of three collinear cows. The lines are sorted as specified above. This output section is empty if no collinear sets exist.

Please see the sample input/output just to be sure.

SAMPLE INPUT	SAMPLE OUTPUT	HINT
2 8 0 0 0 4 1 2 2 4 4 3 4 5 5 1 6 5 16 39 16 31 6 9 8 28 12 39 17 21 27 46 30 20 26 36 20 29 28 47 28 20 13 21 32 36 3 19 12 33 28	1 1 3 4 4 2 5 11 4 7 9 7 8 16 10 11 16	Be careful of floating point arithmetic. Floating point comparison for equality almost never works as well as one would hope.

Explanation of the sample

Eight cows grazing on a grid whose lower left corner looks like this:

```
. . . . * . *
* . * . . . .
. . . . * . .
. * . . . . .
. . . . . * .
* . . . . . .
```

The digits mark the collinear cow IDs:

```
. . . . * . *
* . 4 . . . .
. . . . * . .
. 3 . . . . .
. . . . . * .
1 . . . . . .
```

C. Max Factor

To improve the organization of his farm, Silent Shaon labels each of his **N (1 <= N <= 5,000)** cows with a distinct serial number in the range **1..20,000.** Unfortunately, he is unaware that the cows interpret some serial numbers as better than others. In particular, a cow whose serial number has the highest prime factor enjoys the highest social standing among all the other cows.

(Recall that a prime number is just a number that has no divisors except for 1 and itself. The number 7 is prime while the number 6, being divisible by 2 and 3, is not).

Given a set of **N (1 <= N <= 5,000)** serial numbers in the range **1..20,000**, determine the one that has the largest prime factor.

INPUT FORMAT

First line of the input will contain the number of testcases **T**.

T sets of input will follow. Each set will have the following format:

* Line 1: A single integer, **N**
* Lines 2..N+1: The serial numbers to be tested, one per line.

OUTPUT FORMAT

T sets of output one after another. Each set will have the following format:

Line 1: The integer with the largest prime factor. If there are more than one, output the one that appears earliest in the input.

Please see the sample input/output just to be sure.

SAMPLE INPUT	**SAMPLE OUTPUT**
2 4 36 38 40 42 1 1	38 1

D. Stump Removal

Always thinking of the cows' grazing experience, Kifayat has found that he must remove **N (1 <= N <= 50,000)** unsightly stumps from the pasture. The stumps are conveniently arranged in a straight line and numbered **1..N** with each stump having some height **H_i (1 <= H_i <= 10,000)**.

Kifayat will use the traditional high explosives to destroy the stumps. These high explosives are formulated to destroy adjacent stumps as long as those adjacent stumps are strictly shorter than the nearest stump being destroyed. The blast can continue past the closest adjacent stump to the next adjacent stump if it is even shorter than the nearest stump just destroyed. As soon as a stump encountered by the blast wave is not shorter, though, no more destruction occurs on that side of the target stump (the other side follows the same rules with whatever stumps might appear there).

Consider a line of nine stumps with these heights:

1 2 5 4 3 3 6 6 2

If Kifayat blows up the **third stump (with height 5)**, then the *second stump will also be destroyed (height 2)* and the *first stump (height 1)* will also be destroyed. Likewise, the *fourth stump (height 4) and fifth stump (height 3) will be destroyed* since they are successively shorter, leaving the line like this:

* * * * * **3 6 6 2**

Two more explosives (at stumps 7 and 8) will destroy the rest.

Help Kifayat determine the minimum number of explosive charges he needs to destroy the stumps.

INPUT FORMAT

First line of the input will contain the number of testcases **T**.
T sets of input will follow. Each set will have the following format:
Line 1: A single integer, **N**
Lines 2..N+1: Line i+1 contains **H_i**

OUTPUT FORMAT

T sets of output one after another. Each set will have the following format:
Lines 1..?: Each line contains one integer which is the index of a stump to blow up. The indices must be listed in increasing order.

Please see the sample input/output just to be sure.

SAMPLE INPUT	**SAMPLE OUTPUT**
1 9 1 2 5 4 3 3 6 6 2	3 7 8

E. Bookshelf

Rakeen recently bought a bookshelf for cow library, but the shelf is getting filled up quite quickly, and now the only available space is at the top.

Each of the **N** cows **(1 <= N <= 20,000)** has some height of **H_i (1 <= H_i <= 10,000)** and a total height summed across all **N** cows of **S**. The bookshelf has a height of **B (1 <= B <= S < 2,000,000,007).**

To reach the top of the bookshelf taller than the tallest cow, one or more of the cows can stand on top of each other in a stack, so that their total height is the sum of each of their individual heights. This total height must be no less than the height of the bookshelf. Since more cows than necessary in the stack can be dangerous, your job is to find the set of cows that produces a stack of the smallest number of cows possible such that the stack can reach the bookshelf.

INPUT FORMAT

First line of the input will contain the number of testcases **T**.

T sets of input will follow. Each set will have the following format:

* Line 1: Two space-separated integers: N and B
* Lines 2..N+1: Line i+1 contains a single integer: H_i

OUTPUT FORMAT

T sets of output one after another. Each set will have the following format:

* Line 1: A single integer representing the size of the smallest set of cows that can reach the bookshelf.

Please see the sample input/output just to be sure.

SAMPLE INPUT	**SAMPLE OUTPUT**
1 6 40 6 18 11 13 19 11	3