Fractions

Mr. Panda's kids recently learnt about mixed fractions in school. Mixed fractions are of the form

$$a\,\frac{b}{c}$$

where a, b, c are non-negative integers such that $b < c$. The value of this mixed fraction is equal to $(a + (b\,/\,c))$. The kids just received a piece of homework from their teacher, which consists of a list of **N** mixed fractions. Their job is to sort the mixed fractions in increasing order, and if two fractions have the same value, then they should be sorted in increasing values of a then tiebreak by increasing values of b and lastly by increasing values of c.

Mr. Panda feels that this is a waste of time for his kids and thinks their time could be better spent learning other things, thus he wants you to help him code a program to help the kids solve their homework.

Input

The first line of input contains an integer **N**. The next **N** lines each contain 3 integers, **a**, **b**, **c** representing a mixed fraction with a value $(a + (b\,/\,c))$.

Output

Output the mixed fractions in increasing order of value, breaking ties by increasing values of a then increasing values of b then increasing values of c.

Limits

- 1 ≤ **N** ≤ 100,000
- 0 ≤ **a** ≤ 1,000,000,000
- 0 ≤ **b** < **c** ≤ 1,000,000,000 (i.e. It is guaranteed that b < c for all provided mixed fractions.)

Sample Input (**fractions1.in**)	Sample Output (**fractions1.out**)

`6`
`1 0 5`
`0 2 4`
`1 1 3`
`0 1 2`
`1 0 3`
`1 2 8`
	`0 1 2`
	`0 2 4`
	`1 0 3`
	`1 0 5`
	`1 2 8`
	`1 1 3`

Hint

When comparing two fractions, using **float** or **double** *may* cause precision issues. You can avoid this by using the following relation which allows you to compare them using only operations involving integers.

$$\frac{x_1}{x_2} < \frac{x_3}{x_4} \Leftrightarrow x_1 * x_4 < x_3 * x_2$$

When doing so, please do remember to use the **long** data type as multiplying 2 large integer values might cause the **int** data type to *overflow*.

Notes:

1. You should develop your program in the subdirectory **ex2** and use the skeleton java file provided. You should not create a new file or rename the file provided.
2. You are free to define your own helper methods and classes (or remove existing ones).
3. Please be reminded that the marking scheme is:
   a. Public Test Cases (1%) - 1% for passing **all** test cases, 0% otherwise
   b. Hidden Test Cases (1%) - Partial scoring depending on test cases passed
   c. Manual Grading (1%)
      i. Overall Correctness (correctness of algorithm, severity of bugs)
      ii. Coding Style (meaningful comments, modularity, proper indentation, meaningful method and variable names)
4. Your program will be tested with a time limit of not less than **2 sec** on Codecrunch.

Skeleton File – Fractions.java

You are given the below skeleton file `Fractions.java`. You should see a non-empty file when you open the skeleton file. Otherwise, you might be in the wrong working directory.

```
/**
 * Name         :
 * Matric. No   :
 * PLab Acct.   :
 */

import java.util.*;

public class Fractions {
    private void run() {
        //implement your "main" method here
    }

    public static void main(String[] args) {
        Fractions newFractions= new Fractions();
        newFractions.run();
    }
}
```