

cout << "hello, world!" << endl

Round 1C 2010

A. Rope Intranet

B. Load Testing

C. Making Chess Boards

Contest Analysis

Questions asked

Submissions Rope Intranet 9pt Not attempted 2989/3075 users correct (97%)Not attempted 13pt 2662/2973 users correct (90%)Load Testing 14pt Not attempted 1060/1468 users correct (72%)Not attempted 829/1020 users correct (81%)Making Chess Boards 18pt Not attempted 640/836 users correct (77%)24pt Not attempted 226/547 users correct (41%)

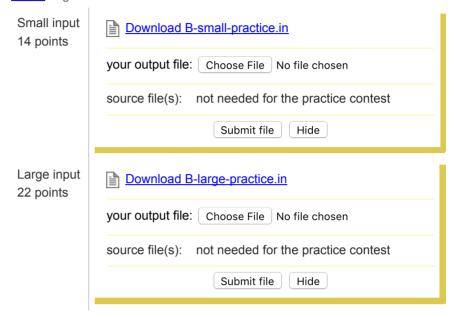
 Top Scores 	
ZhukovDmitry	100
darnley	100
morriship	100
xdliutao	100
Onufry	100
Clann	100
SergeyFedorov	100
kubus	100
K.A.D.R	100
Murphy	100

Practice Mode

Contest scoreboard | Sign in

Problem B. Load Testing

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the <u>Quick-Start</u> Guide to get started.



Problem

Now that you have won Code Jam and been hired by Google as a software engineer, you have been assigned to work on their wildly popular programming contest website.

Google is expecting a lot of participants (**P**) in Code Jam next year, and they want to make sure that the site can support that many people at the same time. During Code Jam 2010 you learned that the site could support at least **L** people at a time without any errors, but you also know that the site can't yet support **P** people.

To determine how many more machines you'll need, you want to know within a factor of $\bf C$ how many people the site can support. This means that there is an integer $\bf a$ such that you know the site can support $\bf a$ people, but you know the site can't support $\bf a$ * $\bf C$ people.

You can run a series of *load tests*, each of which will determine whether the site can support at least **X** people for some integer value of **X** that you choose. If you pick an optimal strategy, choosing what tests to run based on the results of previous tests, how many load tests do you need in the worst case?

Input

The first line of the input gives the number of test cases, **T**. **T** lines follow, each of which contains space-separated integers **L**, **P** and **C** in that order.

Output

For each test case, output one line containing "Case #x: y", where x is the case number (starting from 1) and y is the number of load tests you need to run in the worst case before knowing within a factor of **C** how many people the site can support.

Limits

 $1 \le \mathbf{T} \le 1000.$ $2 \le \mathbf{C} \le 10.$

L, P and C are all integers.

Small dataset

 $1 \le L < P \le 10^3$.

Large dataset

 $1 \le L < P \le 10^9$.

Sample

Input	Output
19 57 3	Case #1: 2 Case #2: 0 Case #3: 4 Case #4: 2

Explanation

In Case #2, we already know that the site can support between 19 and 57 people. Since those are a factor of 3 apart, we don't need to do any testing.

In Case #4, we can test 48; but if the site can support 48 people, we need more testing, because 48*2 < 97. We could test 49; but if the site can't support 49 people, we need more testing, because 24*2 < 49. So we need two tests.

All problem statements, input data and contest analyses are licensed under the Creative Commons Attribution License.

© 2008-2015 Google Google Home - Terms and Conditions - Privacy Policies and Principles

