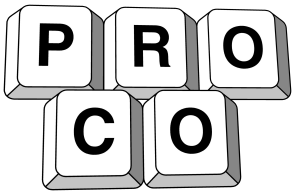
Stanford ProCo 2009: Contest Information



Schedule

Registration	9:00 AM	Gates 104
Sample Problems/ Network Testing	9:30 AM	Clusters
Contest	11:00 AM-1:00 PM	Clusters
Lunch	1:00 PM	Lobby
Awards	2:00 PM	Gates 104
Discussion (Optional)	2:30 PM	Gates 104

Submission Information

• The **contest site** (where you submit solutions) is http://cs.stanford.edu/groups/acm/proco/

• We will be accepting submissions in **C, C++, Java, and Python.** Specifically, we will be using GNU C/C++ 4.3.3, Java 1.6, and Python 2.6.2 on 32-bit Ubuntu 9.04 machines.

• All files **must** be named in the format prXX.<ext>, where <ext> is c for C, cc or cpp for C++, java for Java, and py for Python. Java submissions **must** contain a public class prXX in the file. XX refers to the problem number (pr21 for Problem 2.1, pr94 for Problem 9.4, etc).

• You will be submitting your (**single**) code file to us using a web interface. If you are using Java and want to create multiple classes, put all of the class declarations into **one file**.

• Because we will be recompiling your code on our computers, you should **not** use any **non-standard libraries**.

• Each problem will have a specific **time allocation** (1 second). This refers to the **maximum time** allowed for each test run of the program, not the total time taken. So if we have 20 judge data cases, the submission will be accepted if, for each of the 20 cases, the correct output is produced within the time allocation.

• For time allocation, **Java and Python** submissions will be **scaled** by a factor of 2 (because bytecode runs more slowly). That is, Java and Python submissions that run in 1.54 seconds will be treated as having completed in 0.77 seconds.

Rules

• You are allowed to use only **ONE LAPTOP** for all team members, to use for coding and accessing documentation.

• You will be allowed **access to the Internet** for **documentation purposes only**. Using the internet to search for external code or to communicate is strictly **forbidden** and is grounds for disqualification.

• You will **not** be allowed to use **any programs** for any purposes **other than typing and running code** (IDEs, etc.), **submitting code** (via a web browser), or **looking up language documentation** (via a browser or whatever else is appropriate). For example, you should **not** use any calculators. Violation of this rule is also grounds for **disqualification**.

• During the contest, you should **not** view or use any **pre-written code** (that is, any code written before the start of the sample problem run).

• All input and output will be sent to/received from **console input** and **console output**:

Java	System.in / System.out
C++	cin / cout
C	stdin / stdout
Python	sys.stdin / sys.stdout

• Format all output as closely as possible to the sample output. This includes spacing, capitalization, and punctuation.

• You should **conduct yourselves appropriately** to the contest, the organizers, and the facilities. If you attempt anything **malicious** (e.g. try to break our judging system), you will be **disqualified**.

Scoring

• The contest duration will be 2 hours. There will be five 2-point problems, five 5-point problems, and five 9-point problems, for a total of 15 problems.

• This handout contains two (0-point) sample problems. You are encouraged to use them for testing, but they will not influence contest scoring.

• You will be notified whether your submission passed all judge input within a minute. If the submission is correct, no further points can be awarded for the problem. If the submission is not accepted, you are free to resubmit the problem. There is no penalty for incorrect submissions and no limit on the number of submissions per problem.

• Your submissions do not all need to be in the same language. For example, you could solve pr21 in C, pr52 in Java, and pr93 in C++.

• Placing will be decided as follows:

1. Most total points
2. Most number of solved 9-point problems
3. Most number of solved 5-point problems
4. Earliest time of last correct submission

Problem 0.0 **Matchmaking** (page 1 of 1)

Overview: Print out the difference between two integers, plus one.

Description: Congratulations! You have been hired as the new intracorporate matchmaker at Matchbox.com, the world's largest online dating resource. Your job is to match the despondent single employees at Matchbox using a complicated algorithm which considers, among other things, the distance between the cubicles of two potentially compatible employees.

All cubicles are arranged along a line, and are numbered from 0 to 1,000,000 inclusive (Matchbox has a very wide reach). For ease of calculation, distance is calculated as the target cubicle minus the starting cubicle plus one. Your job for the morning is to write a program that will calculate the distance between any two cubicles, for your employers' benefit. Good luck!

Time Allocation: 1 second

Input: The input consists of two integers *a b*, separated by exactly one space, representing the locations of the target cubicle and the starting cubicle, respectively.

Output: The output should consist of a single integer *c*, representing the value of *b*-*a*+1.

The output is to be formatted exactly like the sample output given below.

Assumptions: *a* will be between 0 and 1,000,000, inclusive.
b will be between *a* and 1,000,000, inclusive.
All input will be valid.

Sample Input #1:
```
3 7
```

Sample Output #1:
```
5
```

Sample Input #2:
```
333 841
```

Sample Output #2:
```
509
```

Problem 0.1 **Hands-On Bacteria Handoff** (page 1 of 2)

Overview: Interactively simulate a bacteria game.

Description: Oh no! The Chief Financial Officers of mAudia and BUJ are at it again. In an epic battle of hand-slapping, they have decided to take turns slapping one of their hands to one of their opponents' hands. During each turn, bacteria on the slapping hand duplicate and transfer over so that at the end of the turn, the slapped hand has the sum of the original number of bacteria on both hands. Whenever there are more than 6 bacteria on a hand, that hand decays and can no longer play.

The CFO of mAudia has already eliminated one of the CFO of BUJ's hands, so her only remaining hand must single-handedly stay the fight. In an act of arrogance, mAudia's CFO has boasted that the CFO of BUJ can claim victory if she eliminates either of his hands. Of course, she loses if her one remaining hand is eliminated. Who will win? Help BUJ's CFO determine the answer to that question.

Time Allocation: 1 second

Input/Output: This is an interactive problem. This means that your program will receive input from the grading environment based on the output your program produces. All input and output will be done through the console (`scanf` for C, `cin` for C++, `System.in` for Java, `sys.stdin` for Python).

Rules of interaction:

1. Your program should begin by reading in three integers *h l r*, each separated by a single space. *h* represents the number of bacteria on your (CFO of BUJ's) hand, while *l* and *r* represent the number of bacteria on your opponent's (mAudia's CEO's) left and right hands, respectively.
2. Then, for each turn during normal gameplay, your program should output a single character `L` or `R`, indicating which of your opponent's two hands you choose to slap.
3. In response, your opponent will give you a single character `L` or `R`, indicating which hand he used to hit your hand.
4. If your hand has more than 6 bacteria after your opponent's move, your program should output `Lose` and quit.
5. If either of your opponent's hands would have more than 6 bacteria after your move, your program follow the output of `L` or `R` with a space and the word `Win` in the same line, and quit.
6. You MUST output a new line character and flush the output stream after each output! To do this:
   In C, use `printf("\n"); fflush(stdout);`
   In C++, use `cout << endl << flush;`
   In Java, use `System.out.println(); System.out.flush();`
   In Python, use `sys.stdout.write("\n"); sys.stdout.flush();`

Problem 0.1 **Hands-On Bacteria Handoff** (page 2 of 2)

Assumptions and Expectations:

h, *l*, and *r* will each be an integer between 1 and 6, inclusive.
Your program does not need to play according to an optimal strategy, nor should you assume that your opponent plays optimally.
If any output is invalid, your program will be deemed incorrect.

Sample Run #1:

Input:	`3 1 1`		
Output:	`R`	Bacteria counts:	`3 1 3`
Input:	`R`	Bacteria counts:	`6 1 3`
Output:	`L Win`	Bacteria counts:	`6 7 3`

Sample Run #2:

Input:	`1 1 2`		
Output:	`R`	Bacteria counts:	`1 1 3`
Input:	`L`	Bacteria counts:	`2 1 3`
Output:	`L`	Bacteria counts:	`2 3 3`
Input:	`L`	Bacteria counts:	`5 3 3`
Output:	`R Win`	Bacteria counts:	`5 3 8`

Sample Run #3:

Input:	`3 1 2`		
Output:	`L`	Bacteria counts:	`3 4 2`
Input:	`L`	Bacteria counts:	`7 4 2`
Output:	`Lose`		