YONSEI UNIVERSITY

CSI3108-01 2015. 11.09

Programming HW#6

(Linear Programming)

Max 40 points

Due on Nov. 20 (Fri), 2015, by 5pm

Write a Java program for solving LP by implementing Simplex algorithm discussed in the lecture. The number n of variables and the number m of constrains are up to 100 each.

Assume that a given LP is always to maximize the objective function. For each iteration of the algorithm, use the following rules:

- Choose the variable with the largest coefficient in the objective function.
- If two or more constraints become tight at the same time, then choose the constraint that appears on top of other constraint(s); that is, if we label the constraints as ①, ②, ③, ... from top to bottom (like in the textbook), choose the constraint with the smallest label as below.

$$\begin{aligned} \max\ & 2x_1 + 5x_2 \\ 2x_1 - x_2 &\leq 4 \quad ① \\ x_1 + 2x_2 &\leq 9 \quad ② \\ -x_1 + x_2 &\leq 3 \quad ③ \\ x_1 &\geq 0 \quad ④ \\ x_2 &\geq 0 \quad ⑤ \end{aligned}$$

Input

The test cases consist of the following format. In the first line, the number of test cases is given. From the next line, each test case is provided in m+1 lines. The first line consists of coefficients of the objective function. From the second line, each line has coefficients of a constraint and then a constant, assuming that each constraint has a form of LHS ≤ RHS. Note that the n non-negativity constraints for n variables are not given in each test case.

Sample Input

20	// the no of test cases.
2 3	// n=2, m=3, test case #1
2 5	// objective function, $\max 2x_1 + 5x_2$
2 -1 4	// constraint ①, $2x_1 - x_2 \leq 4$
1 2 9	// constraint ②, $x_1 + 2x_2 \leq 9$
-1 1 3	// constraint ③,$-x_1 + x_2 \leq 3$
2 2	// n=2, m=2, test case #2
1 1	
2 -3 5	
4 -1 3	
...	

Output

For each test case, print out a sequence of the objective values obtained from the iterations of the simplex algorithm in single line. Each real value should be rounded from the third digit under the decimal point; e.g., for 5/3 = 1.666…, print 1.67.

If the input LP is unbounded, output "unbounded".

Sample Output

0 15 22	// testcase #1
unbounded	// testcase #2
...	