2008-2009 North-America North-East Preliminary Questions

McGill University

October 11, 2008

1 War

Problem Statement
Consider two armies engaged in combat. Army 1 has N_1 soldiers, each with a shooting accuracy p_1 and army 2 has N_2 soldiers, each with a shooting accuracy p_2. Combat occurs in rounds. In each round, all soldiers on both sides fire a single shot at one of the soldiers from the opposite side (but not necessarily at the *same* one). If a round ends and there are still soldiers left standing on both sides, then there is another round. This continues until a round ends with no soldiers left on one of the two sides. You are asked to determine the probability of victory by each side. For the purposes of this question, victory occurs when one side has at least one soldier left and the other side has none.

Assume that each shot either kills the enemy soldier or leaves the enemy soldier unharmed. Assume also that, in the event that there are more soldiers on one side than on the other, the shots are distributed so that the imbalance in the number of shots fired by soldiers on the majority side at the soldiers of the minority side is at most one. For example, if there are six soldiers on the first side and four on the second, two of the soldiers on the second side will have one shot fired at them and the other two will have two shots fired at them. This uniquely determines the number of shots fired against each soldier (up to permutations). Finally, note that since shooting by both sides is simultaneous, it is possible for both sides to lose.

Input
The first line of input will consist of a single integer $T \leq 100$, giving the number of test cases. Each of the next T lines will describe a test case. A

test case consists of two integers and two floating point numbers separated by white space: N_1 N_2 p_1 p_2. No side will have more than one hundred soldiers.

Output

For each test case, print the probabilities of victory by army 1 and army 2, respectively, separated by white space on a single line.

Sample Input

```
1
1 1 0.5 0.5
```

Sample Output

```
0.333333 0.333333
```

2 Brain Pain

Problem Statement

Brain Pain is a math game played with three sets of four cards. Only the cards from one (ace) to ten are used. The goal of the game is to find the highest number that can be made on each of the three sets of cards, using all four cards in each set, and with basic operations only. By basic operations, I mean addition, subtraction, multiplication and division (and parentheses). Note that the maximum need not be an integer.

Input

The first line of input consists of the number of test cases $T \leq 100$ on a line by itself. Each of the next T lines will contain 12 white space separated integers between 1 and 10 (inclusive). The first four are the four cards of the first set, the next four are the four cards of the second set, and the last four are the four cards of the third set.

Output

For each test case, print the value of the highest number that can be formed on all three sets of four cards on a line by itself.

Sample Input

```
1
1 4 7 10 2 4 6 8 3 4 5 6
```

Sample Output

84

3 Robots

Problem Statement

Robots is a game played on an $m \times n$ board in which you must try to survive for as long as possible as a number of robots try to kill you. The robots can move one square horizontally, vertically, or diagonally at a time. Each robot moves one step at a time (and all move simultaneously), taking the shortest path to your position. The robots will always move diagonally first, until they have a straight path (either horizontal or vertical) towards you. If two robots move to the same square at the same time, they crash, leaving wreckage behind at the square which they crashed. If other robots then move onto the square with wreckage, they also crash into it and become wreckage. If a robot moves onto your square, you lose. If there are no robots left and you are still alive, you win. You will be given a board description and asked to determine whether it is a winning or losing position.

Note: In the normal game, your character can move. I have done away with this complication and assumed that you remain in place as the robots come to kill you (yes, it sucks, but it's easier to solve).

Input

The first line of input will contain a single integer $T \leq 10$ giving the number of test cases to follow. Each test case begins with a line consisting of two white space separated integers m and n giving the dimensions of the board. These will be at most 100. The next m lines will describe the m rows of the board. Empty squares on the board will be denoted by a space, robots will be denoted by the letter R and your position will be denoted by a Y. There will be only one Y.

Output

For each test case, you are to output either the word WIN or the word LOSE on a single line.

Sample Input

2
2 2
RR

Y
3 3
RR

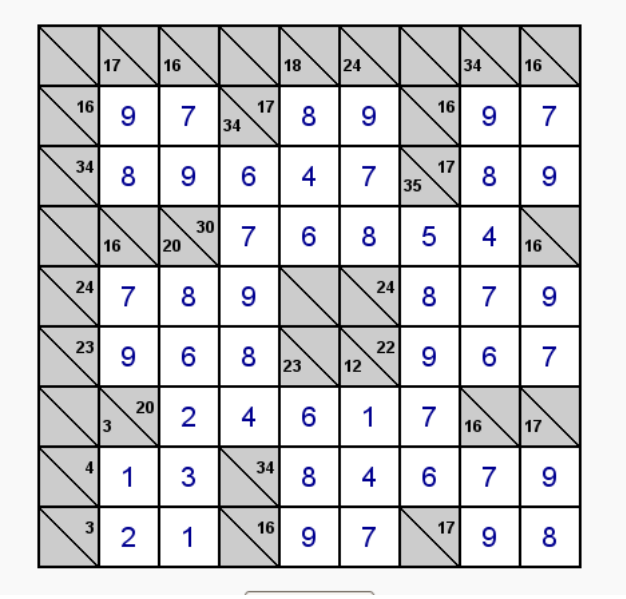
Y

Sample Output
LOSE
WIN

4 Kakuro

Problem Statement
Kakuro puzzles resemble crosswords which use numbers instead of words. The aim of the game is to fill all the blank squares in the grid with only the numbers 1-9 so that the numbers you enter add up to the corresponding clues. Furthermore, you may not use a number more than once in the sum for any clue. For example, if you must make 11 using 4 numbers, there is a unique solution: $1+2+3+5=11$. These numbers need not be placed from smallest to largest, of course.

As an example, look at the following board:



Rather than solving an entire board, your problem is simpler. Given the

number that you must sum to, the maximum value of the numbers that can be used in the sum, and the number of squares to be used to make the sum, find the number of distinct solutions. A solution is distinct from another solution if they cannot be obtained from one another by permutation.

Input

The first line of input consists of the number of test cases $T \leq 100$ on a line by itself. Each of the next T lines will contain 3 white space separated integers C, M and S, where C is the number of squares to be used in the sum, $M \leq 100$ is the maximum number that can be used in any square, and S is the sum to be made. No bounds are given on C or S, except that they can be stored into unsigned integers.

Output

For each test case, print the number of valid solutions on a line by itself. The solution is guaranteed to fit within a 64-bit signed integer.

Sample Input

```
1
5 9 16
```

Sample Output

```
1
```

5 Squares

Problem Statement

You will be given a number of rectangles in the plane and asked to determine the area that they cover. The rectangles may overlap.

Input

The first line of input will contain a single integer $T \leq 10$ giving the number of test cases to follow. Each test case begins with a line with a single integer n giving the number of rectangles that will follow. There will be at most 20. The next n lines will describe the n rectangles. Each of these will consist of four whitespace separated integers $x_{\min}$ $y_{\min}$ $x_{\max}$ $y_{\max}$. These coordinates will will fit within a signed 32-bit integer.

Output

For each test case, you are to output the total area covered by the squares.

Sample Input

```
1
3
0 0 4 10
1 -1 4 1
-2 -2 -1 -1
```

Sample Output

```
44
```

6 Peace

Problem Statement

You will be given a list of nations in the world. There are some pairs of nations which are at peace and some which are at war. Your goal is to help those at war to make peace. You will be given a list of nations, and an array consisting of the confidence level for each pair of nations. You will then be given a list of pairs of nations that you are to aid in making peace. You goal is to find the sequence of intermediate nations (i.e., nations other than than the two trying to make peace) which maximizes the probability of a stable peace agreement. The probability of a peace agreement is simply the product of the confidence levels on the chain between the two nations trying to make peace. For example, if the confidence level between nation A and B is 0.5 and the confidence level between nation B and C is 0.7, then the probability of a stable peace agreement between nations A and C with B acting as intermediary is the product: $0.5 \cdot 0.7 = 0.35$. Two nations are at war precisely when their confidence level is at zero. If there is no way to create a chain of nations to act as intermediaries between two warring nations such that the probability of a stable peace is strictly positive, then nothing can be done, and you should print out a suitable message.

Input

The first line of input will contain a single integer $T \leq 10$ giving the number of test cases to follow. Each test case begins with a line with two whitespace separated integers n and m, on the same line, giving the number of nations in the world, and the number of pairs of nations that you are to help in making peace. There will be at most 100. The next n lines will contain the

nation names (there may be spaces in the names). The next n lines after that will give the the confidence levels for each pair of nations. The i^{th} such line will have n whitespace separated values (between 0 and 1, inclusive). The array will be symmetric, so the off-diagonal entries will be specified twice. The next $2m$ lines will give m pairs of nations that you are to aid in making peace. Each nation will be on a line by itself and will be a nation among those in the list of n nations. You will not be given pairs of nations that are already at peace.

Output

Separate different test cases by a blank line. Also have a blank line separating answers for different attempts at making peace. For a particular peace attempt, print the optimal sequence of intermediary nations, one nation per line. These nations should be ordered as they are in the sequence from the first warring nation to the second. See the sample Input/Output for a clarification of this. *Do not remove the extra blank lines at the end of the file.* There is guaranteed to be a unique solution to each problem.

Sample Input

```
2
4 2
A
B
C
D
  1   0.9   0     0
 0.9   1   0.9    0
  0    0.9   1   0.9
  0     0   0.9   1
A
D
D
A
4 1
A
B
C
D
```

```
  1    0.9    0      0
 0.9    1     0      0
  0     0     1     0.9
  0     0    0.9     1
A
D
```

Sample Output

```
B
C

C
B


NO PEACE
```