THE CODERS CUP

FRESHMEN ROUND

QUESTION SET -C

Competition Rules:

Participation Guidelines:

- The Freshmen Coders Cup round spans for 1 hour. If you have completed the problem set before the allocated time, you may leave the competition room quietly, but inform the coordinator must.
- If you leave the room, you cannot return.
- You can discuss only with your team mates. If you discuss with anyone else, both team will be disqualified from the competition.

Submission Guidelines:

- Find sample inputs from net-storage.
- The problem submission will be through PC^2
- Clarifications to any problem can be obtained using PC^2. No in room managers will be responsible for problems/confusions in problem set given.
- You are allowed to use language
    - C, C++, C#.NET, JAVA.
- IDEs allowed are:
    - Bloodshed Dev C++ for C and C++
    - Visual Studio 2008 or 2010 for C#.NET
    - NetBeans 6.8 for JAVA.
- Make console projects for all afore mentioned IDEs.
- Show output on console; don't write on a text file.
- Do not prompt for input from console in the program.
- Remove system ("pause")/getch()/package inclusion statements from your choice before submitting.

Additional Guidelines:

- The solution will be judged by multiple input files and execution time.
- The decision of judge will stand unchallenged.
- Books, manuals, and any sort of guide materials are not allowed.
- Your team can be dis-qualified, if found hard coding for solutions.
- Your team can be dis-qualified, if found using internet.
- Your team can be dis-qualified, if found unfair in anyway.

Note: Save your work continuously, ACM NUCES is not responsible for any loss of work due to power failure or any other reason.

Question 1:

RIGHT BOX CUBOID

Right Box Cuboid is a new term invented by a few **FASTians** designing questions for this competition. A box/cube is a Right Box Cuboid if and only if its volume is equal to the volume of a cubic box with sides equal to its second largest of side.

INPUT:

First line of input contains an integer 't' which is the number of test cases. Followed by 't' lines.

Each line contains 3 integers separated by a space, which are the sides of the box.

OUTPUT:

For each test case you have output "Tada!!! It's a Right Box Cuboid" if the box is a right box cuboid else you have to output "Nah!!! It's just a Box"

Sample:

Input	Output
3	Tada!!! It's a Right Box Cuboid
6 2 18	Nah!!! It's just a Box
1 8 4	Tada!!! It's a Right Box Cuboid
54 18 6	

Question 2:

SADDLE POINTS

A saddle point is a cell whose value is greater than or equal to any in its row, and less than or equal to any in its column. Your task is to search for the "saddle points" in a 4 * 4 matrix of integers. There may be more than one saddle point in the array.

INPUT:

First line of input contains an integer ‘t’ which is the number of test cases. Followed by ‘t’ test cases i.e. ‘t’ 4 * 4 matrix separated by an empty line.

OUTPUT:

For each test case you have to print the number of saddle points ‘s’ in the matrix. Followed by ‘s’ lines, each containing 2 integer r and c (1 based row and column numbers) i.e. coordinates of each of the ‘s’ saddle points, separated by a space. Write ‘No Saddle Points’, if no saddle point exists.

Each test case output is separated by an empty line.

Sample:

Input	Output
4 3 5 8 9 1 10 0 11 2 4 35 21 1 7 6 9 4 3 2 1 4 3 2 1 4 3 2 1 4 3 2 1 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 2 4 6 8 10 12 14 16 16 14 12 10 8 6 4 2	2 1 4 4 4 4 1 1 2 1 3 1 4 1 1 1 4 No saddle points

Question 3:

THE DARK KNIGHT

Before you read any further- this problem is not about Batman and the Joker. Sorry to have disappointed you. It's about black knight pieces on a chess board. The knight moves in a most unusual manner among chess pieces. When it moves, it can move two squares horizontally and one square vertically, or two squares vertically and one square horizontally. The complete move therefore looks like the letter 'L'. The knight 'captures' an enemy piece by moving into its square. The knight can also jump over other chess pieces.

So, the problem is this. Given a chess board configuration of n x n with only knights on it, you need to figure out whether the configuration is knight-valid. A configuration is knight-valid if none of the knights can capture any of the other knights.

INPUT:

The first line consists of a single positive integer indicating the number of test cases. The first line of each test case consists of a single positive integer n (0 < n < 20) which denotes the chess board size. The following n lines consist of strings of length n denoting the rows of a chess board. The strings are made up of 0s and 1s. A '0' indicates an empty square, and a '1' indicates a square with a black knight in it.

OUTPUT:

For each test case, output a single line containing the word 'VALID' if the board configuration is knight-valid, else 'INVALID'.

Sample

Input	Output
2 4 0000 0010 0000 0100 6 100000 000000 100001 000000 000001 010000	INVALID VALID