Code Jam 2013 - Qualification Round

Analysis: Tic-Tac-Toe-Tomek

In this problem, you had to classify the state of a Tic-Tac-Toe game with a twist. The board is 4x4, and an extra symbol can appear on the board - a "T", which either player can use for victory.

Note that you are guaranteed by the problem description the input will always describe a board that was obtained by a correct sequence of moves. As the game ends when one player wins, this guarantees that only one player can have four symbols (or 3 symbols and a T) in a completed line.

Thus, the simplest way to check whether a player, say "X", won is to check all rows, columns and diagonals whether they contain only "T"s and "X"s. Do not forget to check both diagonals! If there is a row, column or diagonal containing no "."s or "O"s, we know that X won. Similarly, if there is a row, column or diagonal contianing no "X"s or "."s, O won.

If none of the players won, we only have to distinguish between a draw and a game not completed. This is relatively simple - if the board contains even a single ".", the game has not completed yet; otherwise it's a draw.

Note that your solutions are checked automatically, by a program. This means that your output has to be exactly matching the specification. A number of contestants had problems due to returning "O Won" instead of "O won" or "The game has not been completed" instead of "Game has not completed". In a programming competition it is important to follow the specification of the output as exactly as possible.

Here is a complete solution in Python for reference:

```
import sys

def solve(b):
  for c in ['X', 'O']:
    wind1 = True
    wind2 = True
    for x in range(4):
      winh = True
      winv = True
      for y in range(4):
        if b[y][x]!=c and b[y][x]!='T': winv = False
        if b[x][y]!=c and b[x][y]!='T': winh = False
      if winh or winv: return c + ' won'
      if b[x][x]!=c and b[x][x]!='T': wind1 = False
      if b[3-x][x]!=c and b[3-x][x]!='T': wind2 = False
    if wind1 or wind2: return c + ' won'

  for x in range(4):
    for y in range(4):
      if b[y][x]=='.': return 'Game has not completed'

  return 'Draw'


numcases = int(sys.stdin.readline())
```

```
for casenum in range(1,numcases+1):
  board = []
  for i in range(0,5):
    board.append(sys.stdin.readline().strip())
  print 'Case #' + repr(casenum) + ': ' + solve(board)
```