ACM International Collegiate Programming Contest
1998 East Central Regional Practice Contest
University of Waterloo
November 13, 1998

Sponsored by IBM

Rules:

1. All questions require you to read the test data from standard input and write results to standard output. You cannot use files for input or output.
   - No whitespace should appear at the end of a line, and all lines should be terminated with a new-line.
   - Tabs should never be used.
   - Output must correspond *exactly* to the provided sample output, including (mis)spelling and spacing. Multiple spaces will not be used in any of the judges' output, except where expressly stated.
2. All programs will be re-compiled prior to testing with the judges' data.
3. Non-standard libraries cannot be used in your solutions. Neither can some standard libraries, such as the Standard Template Library (STL) and C++ string libraries.
4. Programming style is not considered in this contest. You are free to code in whatever style you prefer. Documentation is not required.
5. All communication with the judges will be handled by the submit command.
6. Each process is given 5 CPU seconds and 16 MB of memory to run.
7. The allowed programming languages are C, C++, and Pascal.
8. Judges' decisions are to be considered final. No cheating will be tolerated.
9. There are **four** questions to be completed in **eighty minutes**.

Problem A: Lost in Space

The USS Enterprise was heavily damaged during its battle with the Klingons, and its computers are offline. Without the sensor information, Captain Kirk cannot find out where his ship is currently located. Luckily, Spock has recorded the path the ship took from the beginning of the battle. Unfortunately, he is injured and is unable to carry out the computation himself. Help Captain Kirk determine if the ship's current location is the same as where it was before the battle begins.

Input

The first line of input is an integer n ($n > 0$) specifying the number of paths. The paths are listed in the next n lines. Each path will be a sequence of directions to travel on a three dimensional grid of lattice points in space. Each letter signifies travel for one unit in the direction indicated: N—north, S—south, E—east, W—west, U—up, D—down. Hence a string of these letters represents a path on a three dimensional lattice. The path will not intersect itself since the Enterprise automatically avoids any debris resulting from the battle. However, it may return to its starting point at the end of the journey.

Output

For each path, print a message indicating whether it returns to its starting point.

Sample Input

```
2
ENENUWDWSDSU
USWNWN
```

Sample Output

```
The path returns to its starting point.
The path does not return to its starting point.
```

Problem B: Triangle Areas

Given any three points (x_1, y_1), (x_2, y_2), and (x_3, y_3), compute the area of the triangle defined by these three points. If the three points are colinear, the area of the triangle is defined to be 0.

Input

The first line of input is an integer n ($n > 0$) specifying the number of test cases to follow. Each of the next n lines contain the integers $x_1, y_1, x_2, y_2, x_3, y_3$. You may assume that $-1000 \leq x_i, y_i \leq 1000$.

Output

For each test case, print the area of the triangle defined by the three given points, accurate to 3 decimal places.

Sample Input

```
3
0 0 10 0 0 10
0 0 10 0 20 0
-10 -20 30 45 20 -10
```

Sample Output

```
Area = 50.000
Area = 0.000
Area = 775.000
```

Problem C: Peg Jump

Peg Jump is a game played on a linear board with several holes. Adjacent holes are equidistant. Some of the holes are filled with pegs. Peg A can "jump" peg B if peg A is next to peg B and there is an open hole on the other side of peg B. When peg A "jumps" peg B, peg A is moved to the open hole on the other side of peg B and peg B is removed from the board. The board is in a winning position if the peg jumping can continue until only one peg is left. Otherwise the board is in a losing position.

Input

The input contains a number of cases. The first line contains an integer n ($n > 0$) containing the number of test cases to follow. Each test case is given in its own line as a string of Xs and dashes representing the board. The Xs represent pegs and the dashes represent open holes. The string will have at most 10 characters. There is at least one peg in each case. The end of input is specified by $n = 0$.

Output

For each case, print "Winning Position" if the board is in a winning position, and print "Losing Position" if the board is in a losing position.

Sample Input

```
2
XX-X-
XXX--
```

Sample Output

```
Winning Position
Losing Position
```

Problem D: Hello, Wolrd!

This question tests your ability to produce correct output.

Input

There is no input for this problem.

Output

Print the line "`Hello, wolrd!`"

Sample Output

```
Hello, wolrd!
```