Student Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Class and Section: COP 3330 0R02

Total Points (100) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Due: Tuesday February 11, 2014 at 11:59 pm**

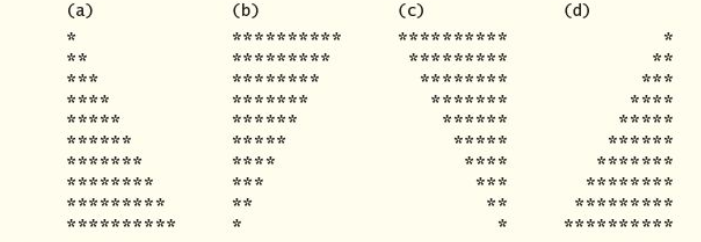
**Programming Assignment 2:**

**Triangle Printing and Pythagorean Triples**

COP 3330 Object Oriented Programming – Spring 2014

**Problem Description(s):**

1. (Triangle Printing Program) Write an application that displays the following patterns separately, one below the other. Use for loops to generate the patterns. All asterisks (\*) should be printed by a single statement of the form System. out. print( '\*' ); which causes the asterisks to print side by side. A statement of the form System. out. println(); can be used to move to the next line. A statement of the form System. out. print( ' ' ); can be used to display a space for the last two patterns. There should be no other output statements in the program. [Hint: The last two patterns require that each line begin with an appropriate number of blank spaces.]



1. (Modified Triangle Printing Program) Modify Part 1 (above) to combine your code from the four separate triangles of asterisks such that all four patterns print side by side. [Hint: Make clever use of nested for loops.]
2. (Pythagorean Triples) A right triangle can have sides whose lengths are all integers. The set of three integer values for the lengths of the sides of a right triangle is called a Pythagorean triple. The lengths of the three sides must satisfy the relationship that the sum of the squares of two of the sides is equal to the square of the hypotenuse. Write an application that displays a table of the Pythagorean triples for side1, side2 and the hypotenuse, all no larger than 500. Use a triple- nested for loop that tries all possibilities. This method is an example of “ brute- force” computing. You’ll learn in more advanced computer science courses that for many interesting problems there’s no known algorithmic approach other than using sheer brute force.

**Analysis: - 10 points (account for all 3 parts)**

(Describe the problems including input and output in your own words.)

Part A:

The program takes no input for the user and prints out 4 triangles one below the other of size 10 consisting of asterisks (\*). E.G. the first line contains one star, the second two, stars up until the tenth line where ten stars are printed out. Triangle b is a vertical reflection of Triangle A. Triangle C is a horizontal reflection of B. And Triangle D is a vertical reflection of Triangle C.

Part B:

This program takes no input from the user and prints the same triangles from part A except they are to be printed side by side instead of one after the other.

Part C:

This program prints a table of values consisting of each integer Pythagorean triple up to the value 500 inclusive. A Pythagorean triple is a set of 3 values that satisfies the equation: a^2 + b^2 = c^2. When going through these values, it is possible a duplicate set will be created, e.g. the set 3 4 5 is the same as 4 3 5. It is at the programmer’s discretion to include or exclude these duplicate sets.

**Design: - 10 points (account for all three parts)**

(Describe the major steps for solving the problem(s). Create a UML diagram to accompany your major steps).

Part A:

|  |
| --- |
| TrianglesPartA |
| numLines: int  numStars: int  numSpaces: int  n: int = 10 |

I began by declaring four variables of type int, numStars, numLines, numSpaces, and n. Using these in for loops would give the number of asterisks and spaces to print out in each line. Variable n was used to determine the size of each triangle. For this assignment, each triangle is 10 lines long but including this variable would make it much easier to modify the code to print out different sized triangles or even a variable-sized triangle. Using for loops, I would cycle through each line of the triangle and print out the required number of spaces and stars in each line, at the end of this for loop, a println function was called to advance to the next line. After printing out each triangle, another println function was called to separate the triangles in increase readability.

Part B:

|  |
| --- |
| TrianglesPartB |
| numLines: int  numStars: int  numSpaces: int  n: int = 10 |

Again, four variables numStars, numLines, numSpaces , and n were created to determine how many of each was to be printed out. Each for loop that prints out the required number of spaces and stars was placed into a single for loop. This top for loop determined how many lines of the triangles would be printed out.

Part C:

|  |
| --- |
| TrianglesPartC |
| sideOne: int  sideTwo: int  hypotenuse: int |

For this part three variables of type int were declared; sideOne, sideTwo, and hypotenuse. The program consists of a triple-nested for loop. At the top, hypotenuse is assigned 0, and increments by 1 up to 500 inclusive. Below this loop, sideOne is assigned 1 and increments by 1 up to the current value of hypotenuse non-inclusive. In the third for loop, sideTwo is assigned 1 and increments by 1 up to the current value of sideOne non-inclusive. In this third loop, and if statement compares the value of the following two equations: sideOne^2 + sideTwo^2 and hypotenuse^2. If the two equations are equal, the values of the three variables are printed out.

**Testing**: **- 10 points** (Describe how you tested, or will test this program)

To test part A and B I ran the application and cross-referenced the output to the examples provided on this sheet. To test part C I cross-referenced the output of my program to a table I found online consisting of every Pythagorean triple up to the integer 10,000. I tried to randomly slect multiple sets from my output to see if any failed. Furthermore, a quick use of a calculator confirmed that my sets satisfied the Pythagorean theorem.

**Submit the following items:**

1. Save this Word file; submit it via Webcourses Assignments on or before the due date and time.

2. Compile, Run, and Submit your .java file(s) to Webcourses. You must submit the program regardless whether it is complete or incomplete, correct or incorrect.