**CSci 1500 - Assignment 4 – 100 pts.**

**Due Date: Oct. 24, 2018**

Write a *detailed* pseudocode solution for each of the following problems. Then, using your pseudocode, convert your pseudocode into C++ programs. Here are the things you need to do for each problem:

* Design your solutions to produce the program output like that given for each of the problems. Note: Your program should work correctly for any valid user input, not just for the example user input values given.
* Convert your pseudocode into C++ code. Follow the coding guidelines specified in the *Coding Guidelines* posted on D2L. Remember to use appropriate data types for all variables. Remember to include each name of your group in a comment at the top of the program. Compile and run the program and verify that it works properly for a variety of input values.

**What you need to turn in:** A printed copy of your pseudocode and your C++ code for each of the problems, arranged in order, and stapled together. Include each name of your group on the front page of what you turn in. Clearly identify which pseudocode and C++ code solves which problem. Use a word processor to write and print your pseudocode. Email all source code (.cpp files) as individual attachments. On the subject line, put CSCI 1500 & the assignment number.

1. Write a program that prompts for and reads in the two side lengths of a right triangle (floating point numbers) and then calls a float-valued function, hypot, that calculates and returns the length of the hypotenuse of the triangle. The program then displays the two side lengths and the hypotenuse length. **Note:** The hypot function ***returns*** the hypotenuse length – ***it does not display it***; the function should not contain any cout nor cin statements. **Math review**: Recall that for any right triangle with side lengths *a* and *b*, and hypotenuse length *c*, . Here is an example of what output should look like from running your program (user input is shown in **bold**).

Enter the side lengths: **1.2 3.4**

First side length = 1.2

Second side length = 3.4

Hypotenuse length = 3.60555

2. A quadratic function is one of the form , where *a*, *b*, and *c* (the *coefficients* of the quadratic function) are constants and . Write a program that will prompt for and read in the coefficients of a quadratic function and a value of *x*, and will then calculate and display the value of *f*(*x*). Your program should make use of a float-valued function, quad, that will accept the values of *a*, *b*, *c*, and *x*, through parameters, and then will calculate and return the value of *f*(*x*). **Note:** Your quad function should not contain any cin or cout statements. Here is an example of what output should look like from running your program (user input is shown in **bold**).

Enter a, b, and c: **3 -5 2**

Enter x: **-1.5**

Quadratic function value = 16.25

3.A leap year is a year that is either evenly divisible by 400 or is both evenly divisible by 4 and not evenly divisible by 100.

Examples:

1. The number 1996 is both evenly divisible by 4 and not evenly divisible by 100, so the year 1996 was a leap year.
2. The number 2000 is both evenly divisible by 4 and evenly divisible by 100, but since it is evenly divisible by 400, the year 2000 was a leap year.
3. The number 1900 is both evenly divisible by 4 and evenly divisible by 100, and since it is not evenly divisible by 400, the year 1900 was not a leap year.

Write a program that will prompt for and read a year number and then determine whether the year is a leap year or not. Your program should make use of a void function, leapYear, which will accept the year number via a parameter and then determine and display the result. Note: Your leapYear function should contain no cin statements. **Hint:** Use the integer modulus operator to determine whether a year is evenly divisible by another number. Here are two examples of what output should look like from running your program (user input is shown in **bold**).

Run 1:

Enter year: **2000**

2000 is a leap year

Run 2:

Enter year: **1900**

1900 is not a leap year

4. Rapid Delivery charges by weight for delivery of packages. The delivery charge for the first pound (16 ounces) is $3.00 and $0.50 is added onto the charge for each additional four ounces. For example, a package weighing more than 16 but at most 20 ounces costs $3.50 to deliver; a package weighing more than 20 but at most 24 ounces costs $4.00 to deliver; etc. Write a program that inputs the weight of a package in ounces (an integer value), then outputs the weight and the charge for delivery. Use an int-valued, data input function, getweight, to prompt for and input the package weight; a float-valued function, deliveryCharge, to calculate and return the delivery charge; and a void function, displayCharge, to display the weight, in pounds and ounces, and the delivery charge. Display the charge with a dollar sign and two places of accuracy.

**Hint:** If the weight of a package in ounces, *w*, is over 16 ounces, then the number of additional weight charges equals [(*w*-13)/ 4], where “/” means C++ integer division.

Here is an example of what output should look like from running your program.

Enter package weight(oz): **23**

Package weight = 1 lb. 7 oz.

Delivery charge = $4.00

5. Write a program that will repeatedly prompt for and read in a positive integer, *n*, and then print out all of its proper divisors (greater than 1 but less than *n*) and then print the number of proper divisors of *n* and their sum. Use a void function **divisors** that will accept the positive integer via an argument and then will determine and display the proper divisor information required. Here is an example of what output should look like from running your program (user input is shown in **bold**):

Enter a number: **28**

Proper divisors: 2 4 7 14

Number of proper divisors = 4

Sum of proper divisors = 27

Continue? (y or n): **y**

Enter a number: **36**

Proper divisors: 2 3 4 6 9 12 18

Number of proper divisors = 7

Sum of proper divisors = 54

Continue? (y or n): **n**