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

**Due Date: Nov 5, 2019**

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.
* Follow the coding guidelines in textbook. Remember to use appropriate data types for all variables. Remember to include each name of your team 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 C++ code for each of the problems, arranged in order, and stapled together. Include each name of your team on the front page of what you turn in. Clearly identify which C++ code solves which problem.

1. Write a function named time that has one integer-valued parameter, seconds, and three integer-valued reference parameters, hours, min, and sec. The function time should convert the passed number of seconds into the equivalent number of hours, minutes, and seconds. (For example, 3789 seconds = 1 hour, 3 minutes, and 9 seconds.) Using the reference parameters, the function should directly alter the respective arguments inside the calling function, “returning” the results through these arguments. Then, write a complete program to use the function time. In main, the program should prompt for and read a number of seconds. Then, a call to the time function should be made to do the conversion. Then, main should display the input number of seconds and the equivalent number of hours, minutes, and seconds. **Hint:** Use integer division and modulus operators. **Note:** The function time should not contain any cout or cin statements. Here is an example of what output should look like from running your program (user input is shown in **bold**):

Enter a time in seconds: **3789**

3789 seconds = 1 hour(s) 3 minute(s) 9 second(s)

2. Write a program that will calculate and display the total surface area and volume of a can (i.e., a right circular cylinder). The program should prompt for and read the can's top radius and height in main and then call a function, calc. This function will receive via arguments the can's top radius and height and will perform the required surface area and volume calculations. The surface area and volume values should be “returned” to mainvia reference parameters. Then, main should display the can’s surface area and volume. Use a global constant for the value of  (use ). **Note:** The function calc should not contain any cout or cin statements.

**Formulas:** ,  where *V* = volume of the can, *A* = surface area of the can, *h* = height of the can and *r* = top radius of the can. Here is an example of what output should look like from running your program (user input is shown in **bold**):

Enter can radius (inches): **3.5**

Enter can height (inches): **6.5**

Can surface area = 219.911 square inches

Can volume = 250.149 cubic inches

3. Write a program that will repeatedly allow the user to determine the change due (the actual number of bills and coins) on a given purchase. First, the program should prompt for and input the purchase amount and the amount of money tendered by the customer (both in dollars). Then, the program should determine the fewest number of bills (ones, fives, tens, and twenties) and coins (pennies, nickels, dimes, and quarters) to return to the customer in change. To do this, the program must use two functions – one function should determine the minimum number of bills of each denomination in the change and “return” these numbers and the other function should determine the minimum number of coins of each denomination in the change and “return” these numbers. Next, the program should display all of these values: The purchase amount, the amount tendered, the change amount, and the minimum number of bills and coins required to make the change. Then, the program should ask if the user wants to determine another change breakdown, and then continue or stop based on the user response.

**Note:** **The functions that determine the change numbers should “return” the numbers to the calling module, *not* *display* them.**

**Hint:** Use integer quantities to do the calculations necessary to determine the minimum number of bills and coins: After calculating the float change due amount, convert the change due into cents, storing the result in an integer variable. Then, use this value, along with the integer division and modulus operators, to determine the number of dollars and cents in the change, storing these values in integer variables. Then, use these values, along with the integer division and modulus operators, to determine the minimum number of bills and coins required to make the change.

Here is an example of what output should look like from running your program (user input is shown in **bold**):

Enter purchase amount: $**100.06**

Enter amount tendered: $**200**

Amount of purchase: $100.06

Amount tendered: $200.00

Change due: $99.94

Bills:

4 - $20's 1 - $10's 1 - $5's 4 - $1's

Coins:

3 - quarter(s) 1 - dime(s) 1 - nickel(s) 4 – penny(ies)

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

Enter purchase amount: $**7.89**

Enter amount tendered: $**10**

Amount of purchase: $7.89

Amount tendered: $10.00

Change due: $2.11

Bills:

0 - $20's 0 - $10's 0 - $5's 2 - $1's

Coins:

0 - quarter(s) 1 - dime(s) 0 - nickel(s) 1 – penny(ies)

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

4. **Overloaded Functions.** Write a C++ program that will allow the user to repeatedly select one of three shapes (circle, rectangle, or triangle), obtain dimensional information from the user for the selected shape, and then determine and display the perimeter of the shape.

**Formulas:**

Triangle perimeter = *a* + *b* + *c* (*a*, *b*, *c* are the triangle side lengths).

Rectangle perimeter = 2(*l*+*w*) (*l* = rectangle length and *w* = rectangle width)

Circle perimeter =  (*d* = circle diameter)

**Requirements:**

* Your program must use three ***overloaded*** functions, each called **input()**, to read in the dimensional information – one for each shape.
* Your program must use three ***overloaded*** functions, each called **perim()**, to determine and return the perimeter – one for each shape.
* Your program must use a **switch** statement.

Here is an example of what output should look like from running your program (user input is shown in **bold**):

Select shape (1=circle, 2=rectangle, 3=triangle, other to exit): **1**

Enter circle diameter: **10**

Perimeter = 31.4159

Select shape (1=circle, 2=rectangle, 3=triangle, other to exit): **2**

Enter rectangle length and width: **2.5** **7**

Perimeter = 19

Select shape (1=circle, 2=rectangle, 3=triangle, other to exit): **3**

Enter triangle side lengths: **3 4 5**

Perimeter = 12

Select shape (1=circle, 2=rectangle, 3=triangle, other to exit): **0**

Bye...

5. **Default Arguments.** Employees of the Acme Widget Co. normally work 40 hours per week. Occasionally, when widget demand is high, employees work overtime. Overtime hours are paid at 50% more than the normal hourly rate. Write a C++ program that uses a function, **wage()**, to determine an employee's weekly wages. The function should have two parameters: The employee's wage rate and the employee's number of overtime hours worked. **The second parameter should have an appropriate default value.** The program should repeatedly do these tasks: Read in an employee's name and wage rate and ask if the employee worked overtime. For an employee who did not work overtime, the program should call the **wage()** function, **using its default parameter value**, to obtain the employee's wages. Otherwise, the program should prompt for and read the number of overtime hours and then call the **wage()** function, **using both parameters**, to obtain the employee's wages. Then, the program should display the employee's name and their wages, and then ask if the user wishes to continue. Here is an example of what output should look like from running your program (user input is shown in **bold**):

Name: **John Doe**

Enter rate: $**15**

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

Enter overtime hours: **20**

Name: John Doe Wages: $1050.00

Enter another? (y or n): **y**

Name: **Mary Smith**

Enter rate: $**20**

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

Name: Mary Smith Wages: $800.00

Enter another? (y or n): **n**