CMPT 130: Lab Work Week 3

1. Write a C++ program that declares three float inputs (call them a, b, c) that represent the coefficients of the quadratic equation $\mathbf{ax^2 + bx + c = 0}$. Read the values for **a**, **b**, and **c** from the user. Then use if- else if - else statements to describe if the quadratic equation has no real number solution, one real number solution or two real numbers solutions. Please note that you are not required to compute and print the actual real number solutions of the quadratic equation. You don't know how to compute the real number solutions yet. What is needed is to print how many real number solutions the quadratic equation has got.

2. Write a C++ program that declares six variables named y1, m1, d1, y2, m2, and d2 all as integer data types. Now read the birth day of a child1 in y1, m1 and d1 variables where y1, m1 and d1 represent the year, month and day of child1. Then read the birthday of child2 in y2, m2, and d2 variables. Finally print the number of years, number of months and numbers of days between the birth dates of the two children. Your program should **NEVER** print any negative years or months or days between the birth dates of the two children. Instead the printed values of years, months and days between the birth dates of the two children MUST be positive or zero values.

3. An employee is paid at a rate of $16.78 per hour for the first 40 hours worked in a week. Any hours over that are paid at the overtime rate of one and-one-half times that. From the worker's gross pay, 6% is withheld for Social Security tax, 14% is withheld for federal income tax, 5% is withheld for state income tax, and $10 per week is withheld for union dues. If the worker has three or more dependents, then an additional $35 is withheld to cover the extra cost of health insurance beyond what the employer pays. Write a program that will read in the number of hours worked in a week and the number of dependents as input and will then output the worker's gross pay, each withholding amount, and the net take-home pay for the week.

4. Write a program that declares ten integer variables, reads in ten integers and then outputs the sum of only positive numbers of the ten input numbers, the average of only the positive numbers, the sum of all the negative numbers, the average of all the negative numbers, the sum of all the numbers, and the average of all the numbers entered.

5. Write a C++ program that calculates and prints a random integer in the range [-1, 1].

6. Write a C++ program that calculates and prints a random float in the range [-1.0, 1.0)

7. Write a C++ program that reads two integers **x** and **y**. Then swaps their values if necessary so that to make sure **y >= x**. Then your program must calculate and print a random integer in the range **[x, y]**.

8. Write a C++ program that reads two floats **x** and **y**. Then swaps their values if necessary so that to make sure **y >= x**. Then your program must calculate and print a random float in the range **[x, y)**.

[1]

9. Write a program that scores a **SIMPLE** blackjack hand. In **SIMPLE** blackjack, a player receives **3** cards. Each card is one of 2 through 10, Jack, Queen, King, and Ace.

   The cards 2 through 10 are scored as 2 through 10 points each. The face cards - Jack, Queen, and King - are scored as 10 points. The goal is to come as close to a score of 21 as possible without going over 21. Hence, any score over 21 is called "busted". The Ace can count as either 1 or 11, whichever is better for the user. For example, an Ace and a 10 can be scored as either 11 or 21. Since 21 is a better score, this hand is scored as 21. An Ace and two 8s can be scored as either 17 or 27. Since 27 is a "busted" score, this hand is scored as 17.

   Assume the **3** cards at hand are represented as characters. The cards 2 through 9 are represented as the characters '2' through '9'; while the cards 10, Jack, Queen, King, and Ace are represented respectively as the characters 'T', 'J', 'Q', 'K', and 'A'.

   Your program should first read the values of three character variables representing the **3** cards and then convert the card values from character values to numeric scores taking special care for Aces. Your program should then calculate the best sum taking care of Aces. The output is either a number between 2 and 21 (inclusive) or the word Busted.

10. **[Challenge]** Write a C++ program that prints **randomly** one of the following three messages: **Yes**, **No** or **Not-Sure**.

11. **[Challenge]** Now using the ideas of Q#11 above, re-write your program of Q#10 so that you **do not cin** the three cards. Instead **assign the three cards random cards** from **'2', '3', '4', ..., '9', 'T', 'J', 'Q', 'K' or 'A'**. Just like in Q#10, you must use char data types for the cards.

12. **Which Pizza to buy!** The large "economy" size of an item is not always a better buy than the smaller size. This is particularly true when buying pizzas. Pizza sizes are given as the diameter of the pizza in inches. However, the quantity of pizza is determined by the area of the pizza, and the area is not proportional to the diameter. Most people cannot easily estimate the difference in area between a 10-inch pizza and a 12-inch pizza and so cannot easily determine which size is the best buy—that is, which size has the lowest price per square inch.

    Write a C++ program that does the following:

[1]

- Declares the four float variables size_smaller, size_larger, price_smaller, and price_larger,
- Reads the size and price of a smaller pizza in the variables size_smaller and price_smaller,
- Reads the size and price of a larger pizza in the variables size_larger and price_larger,
- Declare two float variables area_smaller and area_larger
- Compute the area of each pizza in the variables area_smaller and area_larger. Use the value 3.14 for pi
- Compute the price per square inch for each pizza and determine which pizza is a better buy.

NOTE: The pizza that has lower price per square inch is the better buy.

13.Simple Interest Calculator

Write a program that calculates the accrued total amount given a principal amount and time period in years. The accrued total amount is calculated as

totalAmount = principalAmount * (1.0 + rateOfInterest * timePeriod)

For the rate of interest use the following table:

Principal Amount	Rate of Interest
[0.00, 1000.00)	2.5%
[1000.0, 10,000.00)	2.0%
[10,000.00, 100,000.00)	1.5%
>= 100,000.00	1.0%

The program will proceed as follows:

- Ask the user for the principal amount
- Ask the user for the number of years
- If principal amount is less than 0 or the number of years is less than 0, then print an error message and do nothing
- Else, calculate the accrued total amount and print it
-

14.Multiple if versus if-else if-else Versus statements

What is the output of the following two C++ programs? What is the difference between the following two programs?

Program1.cpp

```
int a = 5, b = 3
if (a == b)
    cout << a << “ is equal to ” << b << endl;
```

```
if (a != b)
    cout << a << “is not equal to ” << b;
if (a < b)
    cout << a << “ is less than ” << b;
if (a > b):
    cout << a << “ is greater than ” << b;
```

Program2.cpp

```
int a = 5, b = 3
if (a == b)
    cout << a << “ is equal to ” << b << endl;
else if (a != b):
    cout << a << “ is not equal to ” << b << endl;
else if (a < b):
    cout << a << “ is less than ” << b << endl;
else:
    cout << a << “ is greater than ” << b << endl;
```

Remark:

- If more than one condition evaluates to True then the multiple if statements is different from if - else if - else statements. If only one condition evaluates to true then they are equivalent; although the later is more efficient.
- Try this by making both a and b to have the same values and you will get the same output from Program1.cpp and Program2.cpp