CISC1600: Computer Science I
How old are you really?

Objective: To *show* our understanding of functions and specifically how arguments are passed to functions (i.e. *pass by value*, *pass by reference*.) To *further* our understanding of *loops*, their logic and the correct use of the ***while()***, ***do .. while()*** and ***for*** control structures in the C++ programming language. Specifically to show an understanding of the logically appropriate use of *nested* loops and embedded *conditional* statements. To *reinforce* our understanding of *data types* and *variables*. Namely, how to create, initialize, display and perform basic arithmetic operations on those variables.

Assignment: Write a program that calculates the elapsed time between two dates in terms of *years*, *months*, and *days*.

Your program should prompt the user to enter the current date in terms of month, day, year. After the date entered has been validated (as specified below), the program should prompt the user to see if they want to calculate how old they are. If they accept, the program should prompt the user to enter the date of their birth also in terms of month, day, and year. This date also needs to be validated (as specified below). Assuming both days have been correctly validated, the program should then calculate the persons age in *years*, *months*, and *days* and display the information accordingly. If the current date happens to be the person's birthday, your program should output a special congratulations message.

The logic that controls the execution of your program should be in the main() body, however, the program should be designed to accomplish its' objective around *at least* the following logical functions:

1. bool enterDate(int month, int day, int year)
   This function prompts the user to enter a date (*in some specified format as: m d y* ***or*** *m/d/y*) and stores the data entered into the three arguments passed. *Note you must decide if the arguments to this function should be passed by value or passed by reference*. This function should invoke the validDate() function and return **true** or **false** false depending on whether the date entered is a valid date.

2. bool enterBirthDate(int cmonth, int cdate, int cyear, int bmonth, int bday, int byear)
   This function should invoke the enterDate() function to get the date of the person's birthday. Assuming a valid date was entered, this function should make sure that it is a valid *birth* date (i.e. date enter occurs before the current date) by invoking the dateBefore() function. This function should return **true** or **false** false depending on whether the date entered is a valid birth date. *Think carefully which arguments to this function should be passed by reference and which arguments should be passed by value.*

3. bool validDate(int month, int day, int year)
   This function checks that the month, day and year passed to the function represent a valid date. The function should return the boolean value **true** or **false** accordingly. For a date to be valid:
   - the month must be between 1 and 12.
   - if the month is in {4,6,9,11}, the day must be in {1 .. 30}
   - if the month is in {1,3,5,7,8,10,12}, the the day must be in {1 .. 31}
   - if the month is in {2}, the day must be in {1 .. 28}. *Note, we will ignore leap years for the moment*.

4. bool dateBefore(int month1, int day1, int year1, int month2, int day2, int year2)
   This function checks that the date corresponding to the parameters month1, day1, year1 is before the date corresponding to the parameters month2, day2, year2 (i.e. cannot have a birthday if you have not been born yet). The function should return the boolean value **true** or **false** accordingly.

5. void calculateAge()
   This function calculates your age in years, months, and days. *Note: You decide what arguments this function needs to take and how they should be passed. This requires some thought because, as the name of this function indicates, the purpose of this function is to* ***calculate*** *the age in years, months, and days. The display of this information, therefore, should take place after this function returns to wherever it was invoked from. In other words, there should be no cout statements in this function*.

   To keep the logic simple, assume the following:
   a year = 365 days (ignore leap years)
   a month = 30.5 days
   All dates are in this century (2000 onwards).

Following is one possible sample run, ***but feel free to be creative***:

Welcome to Age Calculator
Please enter today's date as mm/dd/yyyy:

Response from program is one of:

1. Date entered is: mm/dd/yyyy
2. The entered date is invalid, please re-enter, Enter Date as mm/dd/yyyy:

Enter your date of birth, as mm/dd/yyyy:

Assuming a valid birth day has been entered, calculate how old they are and display the results as...

You are Y years, N months and D days old.

Note: Allow the user ***three*** *attempts only to enter a valid date for each date entered (i.e. current and birth date.) After the 3rd invalid date entered in a row, display a message indicating that three tries have been exhausted, and the program should terminate.*

Important

1. Think through the logic carefully. **This is not a typing exercise**. If you find yourself repeating blocks of code, stop and re-think the logic of the loops and functions.
2. The program flow should be controlled by use of conditional logic, there should not be any statement to return or exit the program anywhere in your code.
3. Each function should use logic to control execution. If a function is expected to return a value (i.e. it is not a void function), there should only be ONE return statement at the end of the function. The body of the function should set the return value accordingly. There should NEVER BE MORE THAN ONE RETURN STATEMENT WITHIN THE BODY OF A FUNCTION!
4. Follow an incremental approach in writing your program. In other words, write a few lines of code, make sure it compiles and executes as expected, then continue with more code.
5. If the program compiles but does not work as expected, consider using cout statements to display the value of variables in order to identify the errors.

For full credit be sure to:

- Include a descriptive Comment Block.
- Use correct indentation and allignment.
- Use descriptive identifiers for variable names as well as appropriate data types.
- Use blank lines to separate the code into appropriate blocks.
- Include comments to help others understand your program, and help yourself think!