## COP2220
## Project 3 – Simple Operations
## 100 Points

### Submission Requirements

- Submit your project folder via the Submit tool provided on the website
    - Follow the project submission guidelines for the class

### Design Documentation Requirements

- Structure chart of the project
- Flowchart of the **isLeapYear**() function
- Pseudocode of the **isPrime**() function

### Design Specification Requirements

Note:  Refer to the sample output in the Example Output section below.

```
int countPrimes();          bool isPrime();
void displayReport();       void performOperations();
int getFactorial();         bool validateInput();
bool isLeapYear();          bool withinRange();
```

1. The program must use the above 8 functions, as well as the main function
    A. **No additional functions are allowed**
    B. Choose appropriate parameters for the above functions
    Note:  Each function will have at least one parameter.
        1. Use appropriate variable types (int, bool, double, etc.)
        2. Use descriptive variable names
        3. Designate variables as pointers (*pValue), as necessary
2. The program must accept 3 command line arguments (integer) separated by spaces
    A. The first integer must be between 1 and 10,000
    B. The second integer must be between 1 and 12
    C. The third integer must be between 1 and 4,000
3. Display your name and the project title on separate lines, followed by a blank line
4. Perform the following operations with the provided integers
    A. First Integer:  Count the prime numbers between 0 and the integer, inclusively
        1. Include the integer in the count if it is prime
        2. See the **List of Prime Numbers** file for a list of prime number less than 10,000
        3. See the **Counting Prime Numbers Algorithm** in the Additional Notes section below
    B. Second Integer:  Calculate the factorial value of the integer (n!)
        1. Example:  5! = 5 * 4 * 3 * 2 * 1 = 120
        2. 0! and 1! are both equal to 1
    C. Third Integer:  Determine whether the integer represents a leap year
        1. See the **Determine If a Year Is a Leap Year Algorithm** in the Additional Notes section below
5. Display the results of the above operations in a report
    A. See Example Output section for the contents of the report

**Additional Notes**

- Use one variable for each entered or calculated value in the main function (6 variables)
    - Use bool for the leapYear result variable
        - The variable that stores the output of the IsLeapYear function
    - Use int for the other variables
- Use “flag” variables, as necessary, in all functions except main
    - Use bool variables whenever a variable would normally store either a 0 or 1
        - Add the stdbool.h file to the project (will be discussed in class)
- Ensure your source code conforms to the commenting standards for the class

**Counting Prime Numbers Algorithm [C(n)]**

Note: Neither 0 nor 1 is prime. If n is equal to 0 or 1, the prime number count must be 0.

1. For every positive integer (i), where i > 1 and i <= n
    A. i is prime, if and only if, it cannot be evenly divided by any positive number between 2 and the square root of i (√i), inclusively
    Note: Since all even numbers can be evenly divided by 2, there is no need to test even numbers greater than 2.
    B. If i is prime, increase the prime counter by 1

**Determine If a Year Is a Leap Year Algorithm**

1. If the year is evenly divisible by 4, it is a leap year
    A. Unless the year is also evenly divisible by 100, which means it is not a leap year
        1. Unless the year is also evenly divisible by 400, which means it is a leap year
2. Otherwise, it is not a leap year
    A. Examples:
        1. 2000 and 2012 are leap years
        2. 1900 and 1981 are not leap years

**Example Output**

```
C:\>Project3 1000 5 1900

Ima C Student
Project 3 - Simple Operations

Operation Results:
------------------
Exactly 168 prime numbers exist between 0 and 1000
The value of 5! is 120
The year 1900 is not a leap year

----------------------------------------------------------------------

Project3 0 5 1900

Ima C Student
Project 3 - Simple Operations

The prime value is out of range.
```

```
----------------------------------------------------------------------

Project3 abc 5 1900

Ima C Student
Project 3 - Simple Operations

The prime value is invalid.

----------------------------------------------------------------------
```

**Optional Improvements (extra credit)**

Note:  Do not attempt the extra credit until the program is working correctly.

1. Perform the factorial calculation using a recursive function rather than a simple loop (**10 points**)