Programming Assignment 1
Out: September 19, 2019, Wednesday -- DUE: October 7, 2019, Monday, 11:59pm
EC327 Introduction to Software Engineering – Fall 2019

Total: 100 points

- *You may use any development environment you wish, as long as it is ANSI C++ compatible. Please make sure your code compiles and runs properly under the Linux environment on the PHO307/305 (or eng-grid) machines before submitting.*
- *Labs may be submitted up to a week late at the cost of a **30% fixed penalty** (e.g., submitting a day late and a week late is equivalent). It is in your best interest to complete as many lab questions as possible before the deadline. If you have missing questions in your original submission, you may complete and submit the missing solutions during the following week. Any submissions after the deadline will be subject to the 30% penalty. No credit will be given to solutions submitted after the 1-week late submission period following the deadline.*
- *Please make sure to follow the assignment submission guidelines for **GitHub**. These will be posted in a separate document.*

Please write C++ code for solving the following problems related to data types, operations, if statements, simple loops, and basic functions (Chapters 2-3-4-5 of the textbook by Liang).

Q1. *GPA Calculator* [20 points]
Write a program that will calculate your semester GPA based on 5 classes you are taking. Assume the following: A (4.0), A- (3.7), B+ (3.3), B (3.0), B- (2.7), C+ (2.3), C (2.0), C- (1.7), D (1.0), F(0). Perform error checking to ensure only grades of these type are entered.

Sample runs:
Class 1 *<A><enter>*
Class 2 *<B><enter>*
Class 3 *<C+><enter>*
Class 4 *<A-><enter>*
Class 5 *<C-><enter>*
GPA: 2.94.* [Program Exits]

Class 1 *<B><enter>*
Class 2 *<6><enter>*
Incorrect input! [Program Exits]

**You should display at least two fractional digits. You can display more of the fractional part if you like.*

Q2: *Intersecting Squares* [20 points]

Write a program that reads in two squares from the console (STDIN). The squares are indicated by their (x, y) **upper left coordinate** and their area. The program should then tell if the squares intersect or not. You are able to use the <math.h> library in this assignment. For more see: http://www.cplusplus.com/reference/cmath/

Text in < > in the examples demonstrates the user inputs entered via the keyboard.

Sample run:

Enter the information for the first square
x-coordinate: <2> <enter>
y-coordinate: <0> <enter>
area: <4> <enter>
Enter the information for the second square
x-coordinate: <5> <enter>
y-coordinate: <1> <enter>
area: <9> <enter>
THE SQUARES DO NOT INTERSECT [Program Exits]

Enter the information for the first square
x-coordinate: <0> <enter>
y-coordinate: <3> <enter>
area: <16> <enter>
Enter the information for the second square
x-coordinate: <1> <enter>
y-coordinate: <4> <enter>
area: <25> <enter>
THE SQUARES INTERSECT [Program Exits]

Q3. *Hamming Distance* [25 points]

In information theory, the **Hamming distance** between two sequences of equal length is the number of positions at which the corresponding symbols are different. Put another way, it measures the minimum number of *substitutions* required to change one string into the other, or the number of *errors* that transform one string into the other.

Examples:

The Hamming distance between:

- "**t**o**n**e**d**" and "**r**o**s**e**s**" is 3.
- **1011101** and **1001001** is 2.
- **2173896** and **2233796** is 3.

(Description above is from Wikipedia: http://en.wikipedia.org/wiki/Hamming_distance)

Write a program that prompts the user to enter two numbers between 0-100 (decimal) and computes the Hamming distance between the two numbers when the numbers are represented in base-3 (ternary format). The program then displays the Hamming distance on the screen. **Hint: You don't need to convert the numbers to 1's, 2's, and 0's to figure this out. Think about how to figure this out with arithmetic. Start out thinking how you would do this with base-10.**

Sample runs:

Enter two numbers between 0-100: <8> <enter>
<23> <enter>
Hamming distance between 8 and 23 when numbers are in ternary format is: 2 [Program Exits]*

Enter two numbers between 0-100: <27> <enter>
<30> <enter>
Hamming distance between 27 and 30 when numbers are in ternary format is 1. [Program Exits]**

** 8 and 23 are represented as 022 and 212, respectively, in ternary form.*
***27 and 30 are represented as 1000 and 1010 in ternary form.*

Q4. *Palindromic Number Strings* [10 points]

A **palindromic number** (also known as a **numeral palindrome** or a **numeric palindrome**) is a number that remains the same when its digits are reversed. Like 16461 or 123321.

Write a program that prompts the user to enter a number string and determines whether or not the number is a palindrome. The number entered by the user should only contain digits 0-9 and should return false if any invalid characters are entered.

Sample runs:

Enter a number: <81818> <enter>
The number is 81818 is a palindrome. [Program Exits]

Enter a number: <1121> <enter>
The number 1121 is not a palindrome. [Program Exits]

Q5. *Simple Ciphering with ASCII* [25 points]
Write code that asks the user to enter a single letter. Then choose one of three operations: (1) change case; (2) reverse alphabet; (3) encrypt.

1. Change case should convert the letter to the opposite case (upper to lower; lower to upper).
2. Reverse alphabet should find the corresponding letter starting from the end of the alphabet AND change the case. For example, 'a' equals 'Z' and 'Y' equals 'b'.
3. Encrypt should take a letter and change it six letters "ahead" (assume the alphabet "wraps around" and all of the same case letters are together). For example 'C' is 'I' and 'z' is 'F'

If the input is not a letter (e.g., user entered a number instead of a letter), your code should print an error message. If the input is more than one letter, an error message should also be produced.

Sample runs:
Enter char, operation:*<b> <1><enter>*
Result: B [Program Exits]

Enter char, operation:*<E> <2><enter>*
Result: v [Program Exits]

Enter char, operation:*<K> <3><enter>*
Result: Q [Program Exits]

Enter char, operation:*<*> <1><enter>*
Result: Illegal char [Program Exits]

Enter char, operation:*<g> <5><enter>*
Result: Illegal operation [Program Exits]

Submission Details
Each program should be a single file. Please use the file names **Q1.cpp, Q2.cpp, etc.** for questions 1-5, respectively. Make sure to **comment** your code. **We will provide more information on how to submit this to GitHub** (where all PAs for this semester will be submitted). Do **NOT** submit your executable files (a.out or others). We only need the source code.