Lab 3

Each lab will begin with a recap of last lab and a brief demonstration by the TAs for the core concepts examined in this lab. As such, this document will not serve to tell you everything the TAs will in the demo. It is highly encouraged that you ask questions and take notes. In order to get credit for the lab, you need to be checked off by the end of lab. For non-zero labs, you can earn a maximum of 3 points for lab work completed outside of lab time, but you must finish the lab before the next lab. For extenuating circumstance, contact your lab TAs and Instructor.

(2 pts) Demo: Loops

Following along with your TA as they walk you through for loops, while loops and do while loops. You will also learn how to kill a runaway loop (infinite loop).

(2 pts) Design

You should independently develop your design but feel free to work in pairs to implement the design.

Design is very important when developing programs and there are a variety of ways to approach it. You may draw pictures, write it out in prose or structured text, use pseudo code, and more! The point of design is to give you a blueprint to follow while you are coding. This saves time debugging your program as you can catch mistakes early. It is better to spend one hour of designing than it is to spend five hours debugging.

For this lab you must design a solution to the following problem statement. You will implement your solution.

Problem Statement

A good password has many requirements. Humans have a hard time meeting these requirements left to their own devices. You are tasked with creating a program that will generate a password based on what the user wants in the password.

The user should be able to choose if they want a password with:

*letters

*upper case

*lower case

*numbers

The user should also be able to indicate the number of characters of each type (upper case, lower case, and numbers). You can generate in order or out of order (see next page).

For example:

```
Welcome to Password Creator!

This program will ask about various password requirements and will then generate a potential
password.

Do you want letters (0-no, 1-yes)? 1

Do you want uppercase letters (0-no, 1-yes)? 1

How many characters of the password should be uppercase? 2

Do you want lowercase letters (0-no, 1-yes)? 1

How many characters of the password should be lowercase? 4

Do you want numbers (0-no, 1-yes)? 1

How many characters of the password should be numbers? 4

Your random password is: ACbzdf1254 or Ab1Cz2d5f4 (either would be acceptable passwords)

Would you like to create another password (0-no, 1-yes)? 0
```

You need to generate characters and numbers **for** the password length specified by the user. Therefore, you need a way to repeat generating random numbers and characters. You could use a **while** loop, but it would be more appropriate to use a **for** loop. For example:

```
for (int i=0; i < pass_length; i++) {

   // write code you want repeated inside here

}
```

(6 pts) Implementation

Your design must be checked off by a TA before proceeding to code.

In your implementation, you must use the ASCII chart, and you must use rand and srand to generate random values. Think about how we used rand to get a range of numbers in class.

Hint 1: Notice characters and numbers are in ranges in the ASCII chart: http://www.asciitable.com/

Hint 2: You can add a character to a C++ string by using the **push_back** function and specifying the decimal equivalent of the ASCII character.
For example: **my_str.push_back(74)** would append the character 'J' to the end of my_str (since 74 is the corresponding value for a 'J' character).

(1 point each) Extra credit

- Add the ability for a user to incorporate symbolic printable characters such as: $(@!<
- Implement your program so that the order of the numbers and characters is randomized.