COP3331 Lab 7

Submission Instructions:

1. Create a folder named Lab7_LastName_FirstInitial (e.g. Lab7_Neal_T).
2. In your folder, place a **PDF** file containing your answers to questions with a ♢.
3. Copy your directories containing your programs for questions with a ♠ into the folder; **these directories should only contain files needed to run your program, which may include one or more of the following file types: .cpp, .h., and .txt**. Do NOT include the full project (e.g., solution file). Test your program on CIRCE before submitting by compiling and running with g++. Your file containing main() should **always** be named main.cpp.
4. Ensure that all programs have block comments at the very beginning (starting at the first line) in the file containing main() with your name and the program's description. **The block comment's format should be identical to what's provided in Figure 2-1.**
5. Use single-line comments to describe your code's functionality as needed.
6. Do not submit anything for questions with a ♣.
7. Zip the folder and submit it via Canvas.

♢ = 5 points each, ♠ = 15 points each

---

1. ♣ Read *Chapter 15: How to work with inheritance.*
2. ♢ What keyword do you use in a function declaration to indicate that the function can be overridden?
   a. override
   b. virtual
   c. abstract
   d. base
3. ♢ If you want to create objects from a subclass that inherits an abstract class, the subclass
   a. must override all of the virtual functions in the abstract class
   b. must override all of the pure virtual functions in the abstract class
   c. must override all of the final functions in the abstract class
   d. can't override any of the functions in the abstract class
4. ♢ In which of the following cases does it not make sense to use inheritance?
   a. When many of the member functions in the superclass need to be overridden
   b. When the superclass and subclass are part of the same domain
   c. When the subclass object is a type of the superclass object
   d. When the subclass primarily adds features to the superclass

5. **♠ Random Integer Vector**

Create an object-oriented program that uses a class that inherits the vector class to automatically generate and work with a sequence of random integers. Save in folder lab7_q5.

Console

```
Random Integer List

How many random integers should the list contain?: 12

Random Integers
===============
Integers: 56, 51, 5, 67, 48, 97, 57, 88, 23, 31, 53, 65
Count:    12
Total:    641
Average:  53.4167

Continue? (y/n): y

Random Integers
===============
Integers: 62, 80, 65, 26, 9, 51, 14, 34, 78, 35, 57, 53
Count:    12
Total:    564
Average:  47

Continue? (y/n): n

Bye!
```

Specifications

- Create a RandomIntVector class that inherits the vector class. This class should allow a programmer to create a vector of random integers from 1 to 100 by writing one line of code. For example, a programmer should be able to create a vector named rand_ints that stores 12 random integers with this code: RandomIntVector rand_ints(count);
- The RandomIntVector class should add get_total() and get_avg() member functions that get the total and average of the numbers in the vector. In addition, it should provide a get_str() function that displays a comma-separated list of integers as shown above.
- The code for the program should use the RandomIntVector class to create the vector of random integers, display the integers, and get the summary data (count, total, and average).

6. ♠ **Customer/Employee Data Entry**

Create an object-oriented program that allows you to enter data for customers and employees. Save in folder lab7_q6.

Console

```
Customer/Employee Data Entry

Customer or employee? (c/e): c

DATA ENTRY
First name: Frank
Last name: Wilson
Email: frank44@gmail.com
Number: M10293

CUSTOMER
Name: Frank Wilson
Email: frank44@gmail.com
Number: M10293

Continue? (y/n): y

Customer or employee? (c/e): e

DATA ENTRY
First name: Joel
Last name: Murach
Email: joel@murach.com
SSN: 123-45-6789

EMPLOYEE
Name: Joel Murach
Email: joel@murach.com
SSN: 123-45-6789

Continue? (y/n): n

Bye!
```

Specifications

- Create a Person class that provides data members for first name, last name, and email address. This class should provide a member function that returns the person's full name.
- Create a Customer class that inherits the Person class. This class should add a data member for a customer number.
- Create an Employee class that inherits the Person class. This class should add a data member for a social security number (SSN).
- The program should create a Customer or Employee object from the data entered by the user, and it should use this object to display the data to the user.