COP3331 Lab 6

Submission Instructions:

1. Create a folder named Lab6_lastNamefirstInitial (e.g. Lab6_NealT).
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 14: How to define classes.*
2. ♢ Encapsulation allows you to control access to
   a. the member functions of an object
   b. the data members of an object
   c. the helper functions of an object
   d. the private members of an object
3. ♢ A helper function is a member function
   a. whose definition is coded in the header file for the class
   b. whose function calls are replaced by the code in the function itself

c. that overloads another member function
d. that can only be used by other member functions

4. ♢ You can code a default constructor that
a. accepts no parameters
b. accepts parameters with default values
c. doesn't initialize one or more data members
d. all of the above
e. a and c only

5. ♢ Which of the following is an advantage of using an inline function?
a. You don't have to recompile the header file that contains the function if you change the implementation code for the function.
b. The compiler always replaces each call to the function with the code for the function, resulting in reduced overhead.
c. The code for the function is more concise because it's declared and defined in one place.
d. all of the above
e. a and c only

6. ♢ You can use UML to
a. create diagrams for the classes of an object-oriented program
b. create diagrams of the function hierarchy of an object-oriented program
c. convert diagrams into source code
d. all of the above
e. a and b only

7. ♢ Which members of the following class are private?

```
class Employee {
    string name;
    double salary;

    double to_hourly(double);

    void set_name(string);
    string get_name();
    void set_salary(double);
    double get_salary();

    double get_weekly_salary();
    double get_monthly_salary();
}
```

a. all of them
b. all of them except for the getter and setter functions
c. all of the data members, but none of the member functions
d. none of them

8. ♠ **Project Lab6_Q8: Rectangle Calculator**
   Create an object-oriented program that performs calculations on a rectangle. Save in folder `lab6-q8`.

   Console

```
Rectangle Calculator

Height:    10
Width:     20
Perimeter: 60
Area:      200
* * * * * * * * * * * * * * * * * * * *
*                                     *
*                                     *
*                                     *
*                                     *
*                                     *
*                                     *
*                                     *
*                                     *
* * * * * * * * * * * * * * * * * * * *

Continue? (y/n): y

Height:    5
Width:     10
Perimeter: 30
Area:      50
* * * * * * * * * *
*                 *
*                 *
*                 *
* * * * * * * * * *

Continue? (y/n): n

Bye!
```

   Specifications

   - Use a Rectangle class that provides data members and corresponding getter and setter functions to store the height and width of a rectangle. This class should also provide member functions that calculate the perimeter and area of the rectangle as well as a member function that gets a string representation of the rectangle.

- Store the Rectangle class in a header file and a corresponding implementation file.
- When the program starts, it should prompt the user for height and width. Then, it should create a Rectangle object from the height and width and use the member functions of that object to get the perimeter, area, and string representation of the object.

9. ♠ **Project Lab6_Q9: Card Dealer**
Create an object-oriented program that creates a deck of cards, shuffles them, and deals the specified number of cards to the player. Save in folder `lab6-q9`.

Console

```
Card Dealer

I have shuffled a deck of 52 cards.

How many cards would you like?: 7

Here are your cards:
Jack of Hearts
Jack of Diamonds
2 of Diamonds
6 of Spades
Jack of Spades
6 of Hearts
King of Diamonds

There are 45 cards left in the deck.

Good luck!
```

Specifications

- Use a Card class to store the rank and suit for each card. In addition, use a member function to get a string representation for each card such as “Ace of Spades”, “2 of Spades”, etc.

- Use a Deck class to store the 52 cards in a standard playing deck (one card for each rank and suit):
  Ranks: 2, 3, 4, 5, 6, 7, 8, 9, 10, Jack, Queen, King, Ace
  Suits: Clubs, Diamonds, Hearts, Spades
  This class should include a member function that shuffles the deck, a member function that counts the number of cards in the deck, and a member function that deals a card from the deck, which should reduce the count of the cards in the deck by 1.
- Store the Card and Deck classes in separate header and implementation files.
- When the program starts, it should get a new deck of cards, shuffle them, and display a message that indicates the total number of cards in the deck. To shuffle the cards, you can use the shuffle function of the random module described in chapter 6.
- The program should prompt the user for the desired number of cards. Then, it should deal the user the desired number of cards and display a message that indicates the number of cards left in the deck.

10. ♠ Create UML diagrams for the Card and Deck classes in Lab6_Q9.

11. ♢ Participate in the discussion "Lab6: Introduction to OOP" on Canvas. This should be an active discussion, meaning that you can write in response to one of your peers as well. Feel free to (among others):

    - interact with your classmates on topics from Chapter 14.
    - provide links to other OOP resources.
    - discuss the advantages of object-oriented programming.
    - compare structures and classes.
    - ask your classmates to play a game that you wrote using OOP.