Self-Check Problems

Section 8.1: Object-Oriented Programming

1. Describe the difference between object-oriented programming and procedural programming. Object Oriented programming has objects, while procedural has actions, known as procedures

2. What is an object? How is an object different from a class?
A class is a blueprint for an object, while an object is

3. What is the state of a String object? What is its behavior? a thing
A String object has a set of chars for its state, and its behavior allows you to search/compare these chars

4. What is the output of the following program?

```
public class ReferenceMystery3 {
    public static void main(String[] args) {
        int a = 7;
        int b = 9;
        Point p1 = new Point(2, 2);
        Point p2 = new Point(2, 2);
        addToXTwice(a, p1);
        System.out.println(a + " " + b + " " + p1.x + " " + p2.x);
        addToXTwice(b, p2);
        System.out.println(a + " " + b + " " + p1.x + " " + p2.x);
    }

    public static void addToXTwice(int a, Point p1) {
        a = a + a;
        p1.x = a;
        System.out.println(a + " " + p1.x);
    }
}
```

14 14
7 9 14 2
18 18
7 9 14 18

The state would probably include the current sum, the history, and where on the screen the cursor is. The behavior would allow for input, output, and performing mathematical operations

5. Imagine that you are creating a class called Calculator. A Calculator object could be used to program a simple mathematical calculator device like the ones you have used in math classes in school. What state might a Calculator object have? What might its behavior be?

Section 8.2: Object State and Behavior

6. Explain the differences between a field and a parameter. What is the difference in their syntax? What is the difference in their scope and the ways in which they may be used?
A field is a parameter of an instance, while parameters are static. Parameters use the static keyword & ~~ust~~ are available only accessed in static instance ways

7. Create a class called Name that represents a person's name. The class should have fields representing the person's first name, last name, and middle initial. (Your class should contain only fields for now.)

```
public class Name {
private String first;
private String last;
private char middle;
}
```

8. What is the difference between an accessor and a mutator? What naming conventions are used with accessors and mutators? Accesors get information, mutators change it, mutators usually start with set, accessors begin with get or, if they're returning a boolean, they return is

9. Suppose we have written a class called BankAccount with a method inside it, defined as:

```
public double computeInterest(int rate)
```

If the client code has declared a BankAccount variable named acct, which of the following would be a valid call to the above method?

a. `double result = computeInterest(acct, 42);`
b. `acct.computeInterest(42.0, 15);`
c. `int result = BankAccount.computeInterest(42);`

[✓ d.] `double result = acct.computeInterest(42);`

e. `new BankAccount(42).computeInterest();`

```
public double distance (Point other)
    return Math.pow(other.x - x, 2) +
        Math.pow(other.y - y, 2);
}
```

10. Add a new method to the `Point` class we developed in this chapter:

```
public double distance(Point other)
```

Returns the distance between the current `Point` object and the given other `Point` object. The distance between two points is equal to the square root of the sum of the squares of the differences of their x- and y-coordinates. In other words, the distance between two points (x_1, y_1) and (x_2, y_2) can be expressed as the square root of $(x_2 - x_1)^2 + (y_2 - y_1)^2$. Two points with the same (x, y) coordinates should return a distance of `0.0`.

```
public String getNormalOrder () {
    return first + " " + middle + ". " + last;
}
public String getReverseOrder () {
    return last + ", " + first + " " + middle + ".";
}
```

11. (You must complete Self-Check Problem 7 before answering this question.)

Add two new methods to the `Name` class:

```
public String getNormalOrder()
```

Returns the person's name in normal order, with the first name followed by the middle initial and last name. For example, if the first name is `"John"`, the middle initial is `"Q"`, and the last name is `"Public"`, returns `"John Q. Public"`.

```
public String getReverseOrder()
```

Returns the person's name in reverse order, with the last name preceding the first name and middle initial. For example, if the first name is `"John"`, the middle initial is `"Q"`, and the last name is `"Public"`, returns `"Public, John Q."`.

12. How do you write a class whose objects can easily be printed on the console?

have a toString method

13. The following `println` statement (the entire line) is equivalent to what?

```
Point p1 = new Point();
...
System.out.println(p1);
```

```
public String toString () {
    return "liam.chapter8.selfChecks.Point[x=" + x + ",y=" + y + "]";
}
```

a. `System.out.println(toString(p1));`

b. `p1.toString();`

[✓ c.] `System.out.println(p1.toString());`

d. `System.out.println(p1.string());`

e. `System.out.println(Point.toString());`

```
public ~~static~~ String toString () {
    return normalOrder();
}
```

14. The `Point` class in the `java.awt` package has a `toString` method that returns a `String` in the following format:

```
java.awt.Point[x=7,y=2]
```

Write a modified version of the `toString` method on our `Point` class that returns a result in this format.

15. (You must complete Self-Check Problem 7 before answering this question.)

Write a `toString` method for the `Name` class that returns a `String` such as `"John Q. Public"`.

16. Finish the following client code so that it constructs two `Point` objects, translates each, and then prints their coordinates.

```
// construct two Point objects, one at (8, 2) and one at (4, 3)

System.out.println("p1 is " ...);    // display the objects' state
System.out.println("p2 is " ...);

System.out.println("p1's distance from origin is " ...);

// translate p1 to (9, 4) and p2 to (3, 13)

System.out.println("p1 is now " ...);    // display state again
System.out.println("p2 is now " ...);
```

```
p1.setLocation(9, 4);
p2.setLocation(3, 13);
```

Section 8.3: Object Initialization: Constructors

17. What is a constructor? How is a constructor different from other methods?
A consorvctor concsmcts an object

18. What are two major problems with the following constructor?
return type means its a method, not a constructor

```
public void Point(int initialX, int initialY) {
    int x = initialX;
    int y = initialY;
}
```

These are declaring new variables

19. (You must complete Self-Check Problem 7 before answering this question.)

Add a constructor to the Name class that accepts a first name, middle initial, and last name as parameters and initializes the Name object's state with those values.

```
public Name (String first, char middle, String last) {
    this.first = first;
    this.middle = middle;
    this.last = last;
}
```

20. What is the meaning of the keyword this? Describe three ways that the keyword can be used.
It means this. It can be used to acess properties/methods of an object from within, and as a constructor for itself.

21. Add a constructor to the Point class that accepts another Point as a parameter and initializes this new Point to have the same (x, y) values. Use the keyword this in your solution.

```
public Point (Point p)
    this(p.x, p.y);
}
```

Section 8.4: Encapsulation

22. What is abstraction? How do objects provide abstraction?
Abstraction is focusing only on the essential pieces. Objects use private feilds

23. What is the difference between the public and private keywords? What items should be declared private?
Public is accessable by client code, private is not. Private should be used when changing the value could create an illegal state but NOT ALL FEILDS NEED TO BE PRIVATE!!!

24. When fields are made private, client programs cannot see them directly. How do you allow classes access to read these fields' values, without letting the client break the object's encapsulation?
Accessors

25. Add methods named setX and setY to the Point class that allow clients to change a Point object's x- and y-coordinates, respectively.

```
public void setX (int x) {
    this.x = x;
}
public void setY(int y) {
    this.y = y;
}
```

26. (You must complete Self-Check Problem 7 before answering this question.)

Encapsulate the Name class. Make its fields private and add appropriate accessor methods to the class.

27. (You must complete Self-Check Problem 26 before answering this question.)

Add methods called setFirstName, setMiddleInitial, and setLastName to your Name class. Give the parameters the same names as your fields, and use the this keyword in your solution.

NO. You should only make feilds private if changing them has the potential to put the object into an illegal state. Encapsulating it for any other reason adds unnesesary code and (slightly) decreses performance. Therefore, I see no reson to encapsulate Name, so I won't

28. How does encapsulation allow you to change the internal implementation of a class?
You can get values differently from how their stored

Section 8.5: Case Study: Designing a Stock Class

29. What is cohesion? How can you tell whether a class is cohesive?
Making each class do only one thing

30. Why didn't we choose to put the console I/O code into the Stock class?
These are two seperate abstractions

31. Add accessor methods to the Stock class to return the stock's symbol, total shares, and total cost.
public String getSymbol() { return symbol;} public int getTotalShares() { return totalShares;} public double getTotalCost() { return totalCost; }

Exercises

1. Add the following accessor method to the Point class:

```
public int quadrant()
```

Returns which quadrant of the x/y plane the current Point object falls in. Quadrant 1 contains all points whose x and y values are both positive. Quadrant 2 contains all points with negative x but positive y. Quadrant 3 contains all