CIS 36A :: LAB 04 - Introducing Classes, Objects and Methods

Student Name:

Task 1: Definitions & Concepts

Instructions: Briefly answer the questions below.

1. Keywords: To you best knowledge, describe below words:
    a. class => A class is an object that has its own properties, variables, and methods to make it function as intended. It also acts as a scope.
    b. method => A method is a function. When called, it runs XYZ code.
    c. return => When a method is called it returns at the end with a value. Either being null to signify nothing, or a value of some type.
    d. void => Void means that the method it's in doesn't have a return value.
    e. object => A object is a instance of a class, same like a instance of a server.
    f. constructor => An initialization of an object.
    g. new => Im not sure myself but google says it initalizes the memory application needed to run the class, which makes sense to me.
2. What is the difference between a class and an object, or how are they related?
   => An object is a instance of a class. The class defines what the object is.
3. What name does a constructor have?
   => It has the same name as the class?

Task 2: Understanding Programming

Instructions: Answer each question below. Try to understand and explain the code. **Do not put an IDE code screenshot.**

1. **Exercise 15:** Suppose you have a class **MyClass** with one instance variable **x**. What will be printed by the following code segment?

```
MyClass c1 = new MyClass();
c1.x = 3;
MyClass c2 = c1;
c2.x = 4;
System.out.println(c1.x);
```

   Explain your answer.
   => 3 because class 2 is a child of class 1 and doesn't redefine the class 1 variables.

2. **Exercise 17:** The following method has a flaw (in fact, due to this flaw it will not compile). What is the flaw?

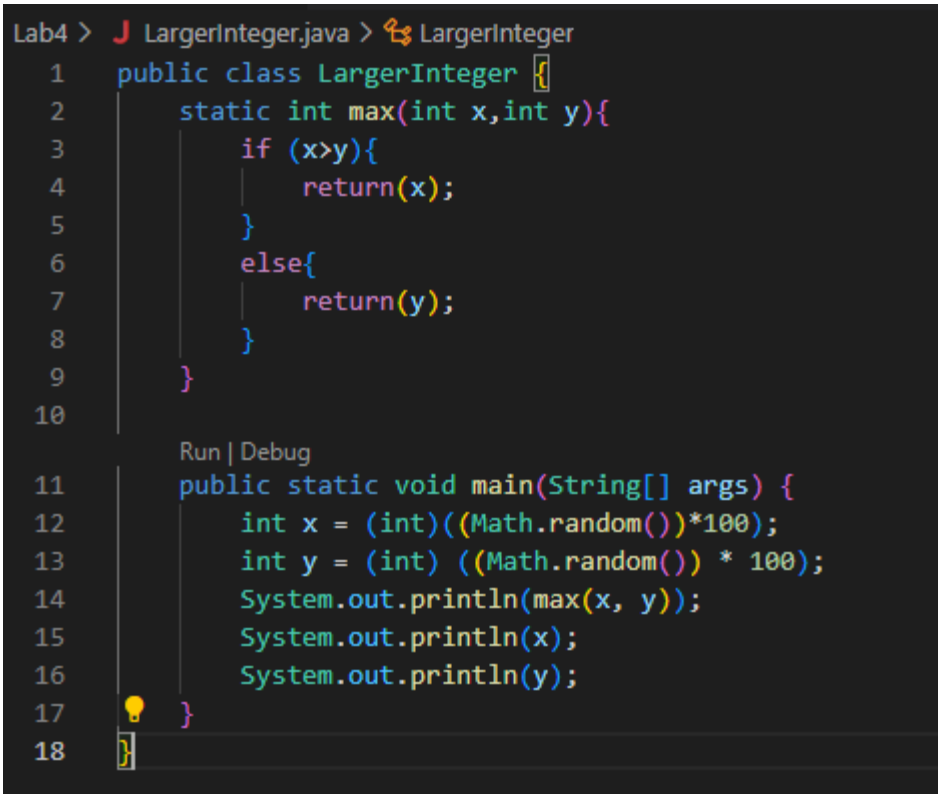
```
void displayAbsX(int x) {
    if (x > 0) {
        System.out.println(x);
    return;
    }
```

```
        else {
              System.out.println(-x);
              return;
        }
       System.out.println("Done");
    }
```

=> A void specifies that the method does not return a value but the code has a return. Therefore it causes an error.

3. **Exercise 18:** Create a method **max( )** that has two integer parameters **x** and **y** and returns the larger of **x** and **y**.

=>



4. **Exercise 19:** Create a method **max( )** that has three integer parameters x, y, and z, and it returns the largest of the three. Do it two ways: once using an if-else-if ladder and once using nested if Statements.

Help

gerInteger.java 1 ×

LargerInteger.java > LargerInteger

```java
public class LargerInteger {
    static int max(int x,int y, int z){
        if ((x>y)&&(x>z)){
            return(x);
        }
        else{
            if((y>z)&&(y>x)){
                return(y);
            }
            else{
                return(z);
            }
        }
    }
```

=>

LargerInteger.java > LargerInteger

```java
public class LargerInteger {
    static int max(int x,int y, int z){
        int biggest = 0;
        if(x>y){
            if(x>z){
                biggest = x;
            }
        }
        if(y>x){
            if(y>z){
                biggest = y;
            }
        }
        if(z>x){
            if(z>y){
                biggest = z;
            }
        }
        return biggest;
    }
```

5. **Exercise 21:** Find all the errors (if any) in the following class declaration:

```
Class MyCla$$ {
    integer x = 3.0;
    boolean b == false

    // constructor
    MyClass(boolean b) { b = b; }
    int doIt() {}
    int don'tDoIt () { return this; }
```

```
}
```

=> You cant have special characters in a class name. Integer x won't work, needs to be int. 3.0 is not an integer, it's a double. == is comparison, not an assignment. Needs a semicolon after false. Cannot assign a variable to itself. doIt() method needs to return a value. Can't have special characters in a method name.

Task 3: Programming Exercises

Instructions: Use any IDE to write and execute below exercises from the book chapter 4. Attach Snipping photos of your source code and execution of the code in the console. Make sure to create separate files for each exercise.

Chapter Examples: Follow the lectures to do the programs with the instructor. Share the screenshots below.

1. **BankAccount (improved version):**
2. **Glass (cup): liquid type, volume, fill, hot/cold**

Chapter Exercises: Do the following chapter exercises.

- **Exercise 13**: Die, DieDemo

```
4 > Die.java > ...
3   public class Die {
4       int sideUp;
5       public Die(){
6           Random rand = new Random();
7           sideUp = rand.nextInt(bound: 6);
8       }
9       int getSideUp(){
0           return sideUp;
1       }
2       void roll(){
3           Random rand = new Random();
4           sideUp = rand.nextInt(bound: 6);
5       }
6   }
7
```

```
1   public class DieDemo {
        Run | Debug
2       public static void main(String[] args){
3           Die Die1 = new Die();
4           Die Die2 = new Die();
5           Die1.roll();
6           System.out.println(Die1.getSideUp());
7           Die2.roll();
8           System.out.println(Die2.getSideUp());
9
10      }
11  }
12
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS G:\Java 1\CIS36A\Lab4>  & 'C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe' '-XX:
59982c6379751721bb868ffcd\redhat.java\jdt_ws\Lab4_6ebf39c5\bin' 'DieDemo'
5
0
PS G:\Java 1\CIS36A\Lab4>
```

```java
public class Card {
    public Card(){
        rank = 3;
        suit = 'R';
    }
    int rank;
    char suit;
    char getSuit(){
        return suit;
    }
    int getRank(){
        return rank;
    }
}
```

- **Exercise 14:** Car, CarTester

Lab4 > CardTester.java > CardTester > main(String[])

```java
public class CardTester {
    Run | Debug
    public static void main(String[] args){
        Card Card1 = new Card();
        Card Card2 = new Card();
        Card Card3 = new Card();
        System.out.println(Card1.getRank() + " " + Card1.getSuit());
        System.out.println(Card2.getRank() + " " + Card2.getSuit());
        System.out.println(Card3.getRank() + " " + Card3.getSuit());
    }
}
```

```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS G:\Java 1\CIS36A\Lab4>  & 'C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe
59982c6379751721bb868ffcd\redhat.java\jdt_ws\Lab4_6ebf39c5\bin' 'CardTester'
3 R
3 R
3 R
PS G:\Java 1\CIS36A\Lab4>
```

- **Exercise 22:** Swapper

```
ab4 > J Swapper.java > ...
1   public class Swapper {
2       int x;
3       int y;
4       public Swapper(){
5           x = 5;
6           y = 3;
7       }
8       int getX(){
9           return x;
10      }
11      int getY(){
12          return y;
13      }
14      void swap(){
15          int storage = x;
16          x = y;
17          y = storage;
18      }
19  }
20
```

```
4 > J SwapperDemo.java > ...
1   public class SwapperDemo {
        Run | Debug
2       public static void main(String[] args){
3           Swapper Swap1 = new Swapper();
4           System.out.println(Swap1.getX());
5           System.out.println(Swap1.getY());
6           Swap1.swap();
7           System.out.println(Swap1.getX());
8           System.out.println(Swap1.getY());
9       }
0   }
```

```
PS G:\Java 1\CIS36A\Lab4>  & 'C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe'
59982c6379751721bb868ffcd\redhat.java\jdt_ws\Lab4_6ebf39c5\bin' 'SwapperDemo'
5
3
3
5
PS G:\Java 1\CIS36A\Lab4>
```

- **Exercise 24:** USMoney
    - I dont understand the instructions for this one. Specifically, this “. It creates and returns a new USMoney object representing the sum of the object whose plus( ) method is being invoked and the parameter.”

- **Exercise 25:** Date

```java
public class Date {
    int day, month, year;
    public Date(){
        day = 1; month = 2; year = 2022;
    }
    int daysSinceJan1(){
        int janDay = 1;
        int janMonth = 1;
        int diffMonth = month-janMonth;
        int diffDay = day-janDay;
        return(((diffMonth*2)*30)+(diffDay+1));
    }
}
```

```java
public class DateDemo {
    Run | Debug
    public static void main(String[] args) {
        Date date = new Date();
        System.out.println(date.daysSinceJan1());
    }
}
```

```
61
PS G:\Java 1\CIS36A\Lab4>  g:; cd 'g:\Java 1\CIS36A\Lab4'; & 'C:\Program F
de\User\workspaceStorage\17253e159982c6379751721bb868ffcd\redhat.java\jdt_
61
PS G:\Java 1\CIS36A\Lab4>
```

Task 4 - Programming Application

A rational number is a number that can be represented as the ratio of two integers. For example, 2/3 is a rational number, and you can think of 7 as a rational number with an implicit 1 in the denominator or 7/1.

1. Define a class called ***Rational***. A Rational object should have two integer instance variables that store the **numerator** and **denominator**.

2. Write a **constructor** that takes two arguments and that sets the numerator and denominator with these two parameters.
3. Write an instance method called ***printRational*** that displays a Rational in some reasonable format.
4. Write a main method that creates a new object with type **Rational**, sets its instance variables to some values, and displays the object.
5. At this stage, you have a minimal testable program. Test it and, if necessary, debug it.
6. Write an instance method called ***negate*** that reverses the sign of a rational number. This method should be a modifier, so it should be *void*. Add lines to **main** to test the new method.
7. Write an instance method called **invert** that inverts the number by swapping the numerator and denominator. It should be a modifier. Add lines to main to test the new method.
8. Write an instance method called ***toDouble*** that converts the rational number to a double (floating-point number) and returns the result. This method is a pure method; it does not modify the object. As always, test the new method.
9. Write an instance method named reduce that reduces a rational number to its lowest terms by finding the greatest common divisor (GCD) of the numerator and denominator and dividing through. This method should be a pure method; it should not modify the instance variables of the object on which it is invoked but it should return a new rational number or it should print a reduced format. Hint: Finding the GCD only takes a few lines of code. Search the web for “Euclidean algorithm”. **Do not use recursion.**
10. Write an instance method called ***add*** that takes a Rational number as an argument, adds it to ***this***, and returns a new Rational object.

    There are several ways to add fractions. You can use any one you want, but you should make sure that the result of the operation is reduced so that the numerator and denominator have no common divisor (other than 1).

Rational.java > Rational > gcd(int, int)

```java
public class Rational {
    int numerator;
    int denominator;
    public Rational(){
        numerator = 8;
        denominator = 1;
    }
    public void printRational(){
        System.out.println(numerator + "/" + denominator);
    }
    public void negate(){
        if ((numerator>0)&&(denominator>0)){
            denominator = 0-denominator;
        }
        if ((numerator<0)&&(denominator<0)){
            denominator = denominator-0;
        }
        else{
            if (denominator>0){
                denominator = 0-denominator;
            }
            else{
                denominator = denominator-0;
            }
        }
    }
    public void invert(){
        int storage = denominator;
        denominator = numerator;
        numerator = storage;
    }
    public double toDouble(){
        return(numerator/denominator);
    }
    public int reduce(){
        int i;
        int gcd = 1;
        for (i = 2; i <= Math.min(numerator, denominator); i++) {
            if (numerator % i == 0 && denominator % i == 0) {
                gcd = i;
            }
        }
        return gcd;

    }

    public int gcd(int a, int b) {
        if (b == 0)
            return a;
        return gcd(b, a % b);
    }
```

```java
        return(numerator/denominator);
    }
    public int reduce(){
        int i;
        int gcd = 1;
        for (i = 2; i <= Math.min(numerator, denominator); i++) {
            if (numerator % i == 0 && denominator % i == 0) {
                gcd = i;
            }
        }
        return gcd;

    }

    public int gcd(int a, int b) {
        if (b == 0)
            return a;
        return gcd(b, a % b);
    }
    public int[] add(int num, int den){
        int comd = denominator * den;
        int newNomAnswer = numerator * den + num * denominator;

        int d;
        d = gcd(comd, newNomAnswer);

        comd = comd / d;
        newNomAnswer = newNomAnswer / d;

        int[] newFraction = new int[2];
        newFraction[0] = newNomAnswer;
        newFraction[1] = comd;
        return newFraction;
    }
}
```

Lab4 > RationalDemo.java > ...

```java
public class RationalDemo {
    Run | Debug
    public static void main(String[] args){
        Rational rational = new Rational();
        rational.denominator = 8;
        rational.numerator = 4;
        rational.printRational();
        rational.invert();
        rational.printRational();
        System.out.println(rational.toDouble());
        System.out.println(rational.reduce());
        System.out.println(Arrays.toString(rational.add(num: 3, den: 7)));
    }
}

```

```
[17, 7]                > g:; cd 'g:\Java 1\CIS36A\Lab4'; & 'C:\Program Files\Java\jdk-18.0.2.1\bin\java.exe' '-XX:+
de\User\workspaceStorage\17253e159982c6379751721bb868ffcd\redhat.java\jdt_ws\Lab4_6ebf39c5\bin' 'RationalDemo' '-XX:+
4/8
8/4
2.0
4
[17, 7]
PS G:\Java 1\CIS36A\Lab4>
```