ECE 558 Winter 2021
Homework #1

THIS ASSIGNMENT SHOULD BE SUBMITTED TO D2L BY 10:00 PM ON SAT, 16-JAN-2021. THE ASSIGNMENT WILL BE GRADED AND IS WORTH 100 POINTS. IT IS EASIEST TO GRADE, AND I BELIEVE EASIEST FOR YOU SUBMIT, IF YOU SUBMIT YOUR SOURCE CODE AS TEXT FILES INSTEAD OF TRYING TO CUT/PASTE YOUR CODE INTO A `.doc` OR `.docx` FILE. CLEARLY NAME THE FILES SO THAT WE KNOW WHICH PROBLEM THE CODE REFERS TO AND SUBMIT THE PACKAGE AS A SINGLE .ZIP OR .RAR FILE (EX: `rkravitz_ece558w21_hw1.zip`) TO YOUR D2L HOMEWORK #1 DROPBOX.

Question 1: Short Answer (25 pts)

a. [5] Circle T if the answer is true or F for false

1.	When using the keyword `extends`, the subclass inherits the `private` members of its superclass.	T	F
2.	In Java a `static` method can access static class variables but a non-static method cannot.	T	F
3.	An interface can contain one or more abstract methods	T	F
4.	We can instantiate an array by assigning values when the array is declared.	T	F
5.	The automatic conversion of a Java primitive numeric type to its wrapper class is called autoboxing	T	F

b. [15] Three key tenets of OO programming are encapsulation, inheritance, and overloading. Provide a short definition and describe how they are realized in the Java programming language.

- Encapsulation
- Inheritance
- Overloading

c. [5] (Select the best answer) Interfaces are a special Java concept. Which of the following statements is true about interfaces?

- In a class that implements two interfaces, the methods of both interfaces must be implemented.
- If one of the interface methods in an interface class (defined by 'interface CLASSNAME') has a body, all of them must have one.
- The constructor of an interface allocates the memory for its attributes.
- In class that implements two interfaces, only the methods of one interface must be implemented.
- Interfaces can only extend one superinterface because Java does not support multiple inheritance.

Question 2: Java Basics (25 pts)

a. [10] Write a method that takes an integer input from the user, then prompts for additional integers and prints all of the integers that are greater than or equal to the original input until the user enters a negative number, which is not printed

b. [15] Correct the following code snippets:

- The following code sequence is intended to print `Hello` three times; however, it only prints `Hello` once. Where is the problem in this code sequence?

```
for (int i = 0; i < 3; i++);
     System.out.println("Hello");
```

- You coded the following in class Hw1.java:

```
int a = 32, b = 10;
double c = a / b;
System.out.println("The value of c is " + c);
```

You expected the value of c to be 3.2, but instead c was displayed as 3. Explain what the problem is and write the code to fix it.

- You coded the following:

```
int[][] a = {{9,8,7,6},{10,20,30,40}};
for (int j = 0, j <= a[1].length; j++) {
    if (a[1][j] == 20) {
        System.out.println("Found 20 at column index " + j +
            + " of second row");
    }
}
```

The code compiles properly, but when you run the program you get the following output:

```
Found 20 at column index 1 of second row

Exception in thread "main"
 java.lang.ArrayIndexOutOfBoundsException: 4 at
 Test.main(Test.java:14)
```

Explain what the problem is and how to fix it.

Question 3: Object Oriented Programming (35 pts)

Consider a class encapsulating the concept of a rational number. According to Wikipedia (https://en.wikipedia.org/wiki/Rational_number) a rational number is any number that can be expressed as the quotient or fraction *p/q* of two integers, a numerator *p* and a non-zero denominator *q*. A rational number has the following instance variables:

- An integer representing the numerator of the rational number
- An integer representing the denominator of the rational number

a. [5] Define the class and write two constructors for the class, the first constructor being a default (no parameters) constructor and the second constructor being a fully qualified constructor (parameters for numerator and denominator). Make use of the setters you are going to write in part b.

b. [5] Write getters and setters for your instance variables. You should not allow the value of the denominator to be 0; instead, give the denominator a default value of 1.

c. [5] Write `toString()` and `equals()` methods for your class. The `equals()` method should check if the difference between the fractional results (numerator/denominator) is < 0.001.

d. [10] Write two additional methods for your class. The first method should perform a multiplication of two rational numbers and the second method should perform an addition of two rational numbers. Both methods should return a rational number. Keep in mind, that these two methods are the API for your rational number class so, in effect you are performing math on the rational number encapsulated in the object and a second rational number. That is, the signature for the multiplication method is:

```
public Rational multiply( Rational r ){};x
```

The Wikipedia article lists the equations for performing these operations.

e. [10] Write a client class to test all the methods in your class.

Question 4: Object Oriented Programming (15 pts)

Consider the following class definition:

```
public class Game {
   private String mDescription;

   public Game(String description) {
      setDescription(description);
   }

   public String getDescription() {
      return mDescription;
   }

   public void setDescription(String description) {
      mDescription = description;
   }

   public String toString() {
      return("description: " + mDescription);
   }
}
```

a. [5] Write a class encapsulating a board game, which inherits from Game. A board game has the following additional attributes:

- The number of players
- Whether the game can end in a tie

Code the constructor(s) and the `toString()` method of the new class.

b. [10] Write a class encapsulating a board game, which inherits from Game and the board game class in part a. This type of board game has the following additional attributes:

- The minimum number of players
- The maximum number of players
- Whether there is a time limit to finish the game

Code the getters and setters and the `toString()` method of the new class. You also need to provide a client class to test your code