**Lab 7: Implementing Interface**

1. Consider the following class RationalNumbers (from chapter 5):

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// RationalNumber.java Java Foundations

// Represents one rational number with a numerator and denominator.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class RationalNumber

{

private int numerator, denominator;

//-----------------------------------------------------------------

// Constructor: Sets up the rational number by ensuring a nonzero

// denominator and making only the numerator signed.

//-----------------------------------------------------------------

public RationalNumber (int numer, int denom)

{

if (denom == 0)

denom = 1;

if (denom < 0)

{

numer = numer \* (-1);

denom = denom \* (-1);

}

numerator = numer;

denominator = denom;

}

//-----------------------------------------------------------------

// Returns the numerator of this rational number.

//-----------------------------------------------------------------

public int getNumerator ()

{

return numerator;

}

//-----------------------------------------------------------------

// Returns the denominator of this rational number.

//-----------------------------------------------------------------

public int getDenominator ()

{

return denominator;

}

//-----------------------------------------------------------------

// Returns this rational number as a string.

//-----------------------------------------------------------------

public String toString ()

{

String result;

if (numerator == 0)

result = "0";

else if (denominator == 1)

result = numerator + "";

else

result = numerator + "/" + denominator;

return result;

}

}

1. Design another interface Combine<T> with method **public T combine(T t1).** This method combines two objects in some way (needs to be defined by the class that implements this interface) and returns the new object of the same type.
2. Modify RationalNumber class so that it implements Combine<T> interface in a way that adds two rational numbers. The algorithm for adding two rational numbers are given below:

* r1 = p/q;
* r2 = x/y;
* r1+r2 = (p\*y+q\*x)/q\*y;

1. A tester class is given. Modify this tester class so that you can test the combine() method.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// RationalTester.java Java Foundations

//

// Driver to exercise the use of multiple Rational objects.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

public class RationalTester

{

//-----------------------------------------------------------------

// Creates some rational number objects and performs various

// operations on them.

//-----------------------------------------------------------------

public static void main (String[] args)

{

RationalNumber r1 = new RationalNumber (6, 8);

RationalNumber r2 = new RationalNumber (1, 3);

System.out.println ("First rational number: " + r1);

System.out.println ("Second rational number: " + r2);

}

}

1. Modify RationalNumber class so that it implements Comparable Interface. The goal is to compare two rational numbers.
2. Modify the tester class so that it invokes the compareTo() method and displays which rational number is greater than the other.