Sheet 3 (Static)

1. Define a class named Money whose objects represent amounts of U.S. money. The class should have two instance variables of type int for the dollars and cents in the amount of money. Include a constructor with two parameters of type int for the dollars and cents, one with one constructor of type int for an amount of dollars with zero cents, and a no-argument constructor. Include the methods add for addition and minus for subtraction of amounts of money. These methods should be static methods and should each have two parameters of type Money and return a value of type Money. Include a reasonable set of accessor and mutator methods as well as the methods equals and toString. Write a test program for your class.

**Part Two:** Add a second version of the methods for addition and subtraction. These methods should have the same names as the static version but should use a calling object and a single argument. For example, this version of the add method (for addition) has a calling object and one argument. So m1.add(m2) returns the result of adding the Money objects m1 and m2. Note that your class should have all these methods; for example, there should be two methods named add.

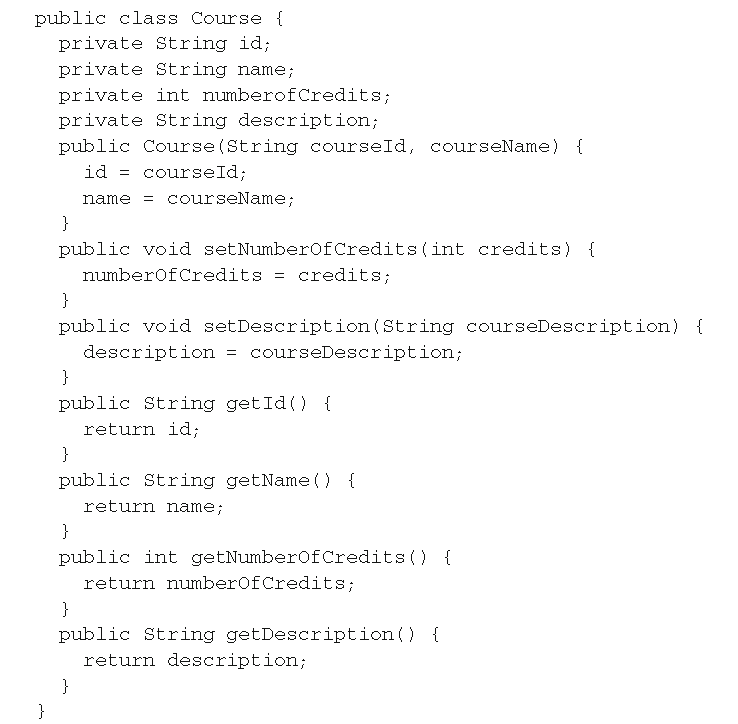
**Alternate Part Two** (If you want to do both Part Two and Alternate Part Two, they must be two classes. You cannot include the methods from both Part Two and Alternate Part Two in a single class. Do you know why?): Add a second version of the methods for addition and subtraction. These methods should have the same names as the static version but should use a calling object and a single argument. The methods should be void methods. The result should be given as the changed value of the calling object. For example, this version of the add method (for addition) has a calling object and one argument. Therefore, m1.add(m2); changes the values of the instance variables of m1 so they represent the result of adding m2 to the original version of m1. Note that your class should have all these methods; for example, there should be two methods named add.

1. We had a class called Course, which had a method that creates Section objects. Modify the two classes so that

(a) Course class maintains the list of all sections.

(b) Section stores the capacity and the number of students enrolled in the class.

(c) Course has a search facility that returns a list of sections that are not full.



Assignment

1. Define a class for rational numbers. A rational number is a number that can be represented as the quotient of two integers. For example, 1/2, 3/4, 64/2, and so forth are all rational numbers. (By 1/2 and so forth, we mean the everyday meaning of the fraction, not the integer division this expression would produce in a Java program.) Represent rational numbers as two values of type int, one for the numerator and one for the denominator. Your class should have two instance variables of type int. Call the class Rational. Include a constructor with two arguments that can be used to set the instance variables of an object to any values.

Also include a constructor that has only a single parameter of type int; call this single parameter wholeNumber and define the constructor so that the object will be initialized to the rational number wholeNumber/1. Also include a no-argument constructor that initializes an object to 0 (that is, to 0/1). Note that the numerator, the denominator, or both may contain a minus sign. Define methods for addition, subtraction, multiplication, and division of objects of your class Rational. These methods should be static methods that each have two parameters of type Rational and return a value of type Rational. For example, Rational.add(r1, r2) will return the result of adding the two rational numbers (two objects of the class Rational, r1 and r2). Define accessor and mutator methods as well as the methods equals and toString. You should include a method to normalize the sign of the rational number so that the denominator is positive and the numerator is either positive or negative. For example, after normalization, 4/–8 would be represented the same as –4/8. Also write a test program to test your class.

Hints: Two rational numbers a/b and c/d are equal if a\*d equals c\*b.

**Part Two:** Add a second version of the methods for addition, subtraction, multiplication, and division. These methods should have the same names as the static version but should use a calling object and a single argument. For example, this version of the add method (for addition) has a calling object and one argument. So r1.add(r2) returns the result of adding the rationals r1 and r2. Note that your class should have all these methods; for example, there should be two methods named add.

**Alternate Part Two** (If you want to do both Part Two and Alternate Part Two, they must be two classes. You cannot include the methods from both Part Two and Alternate Part Two in a single class. Do you know why?): Add a second version of the methods for addition, subtraction, multiplication, and division. These methods should have the same names as the static version but should use a calling object and a single argument. The methods should be void methods. The result is given as the changed value of the calling object. For example, this version of the add method (for addition) has a calling object and one argument. Therefore, r1.add(r2); changes the values of the instance variables of r1 so they represent the result of adding r2 to the original version of r1. Note that your class should have all these methods; for example, there should be two methods named add.