(Enhancing Class Time2) Modify class Time2 of Fig. 8.5 to include a tick method that increments the time stored in a Time2 object by one second. Provide method incrementMinute to increment the minute and method incrementHour to increment the hour. The Time2 object should always remain in a consistent state. Write a program that tests the tick method, the increment-Minute method and the incrementHour method to ensure that they work correctly. Be sure to test the following cases:

1. incrementing into the next minute,
2. incrementing into the next hour and
3. incrementing into the next day (i.e., 11:59:59 PM to 12:00:00 AM).

```plaintext
E:\THUWorks\java程式設計\hw4\Time2\jrun -r -a
TimeTest
Written by Stevennick.
All rights reserved.
t1: hour, minute and second specified
Original:
   11:25:42
   11:25:42 AM
t1: after tick once:
   11:25:43 AM
t1: after 62 seconds:
   11:26:45 AM
t1: after one minute:
   11:27:45 AM
t1: after 35 minutes:
   12:02:45 PM
t1: after one hour:
   1:02:45 PM
t1: after half day:
   1:02:45 AM
t1: Reset to 23:59:59:
   11:59:59 PM
t1: after one second:
   12:00:00 AM
```
// StdNo.: s942864    邱逸夫
// input: none
// output: see code.
// Date: 05/23/07

// Fig. 8.5: Time2.java
// Time2 class declaration with overloaded constructors.

public class Time2
{
    private int hour;  // 0 - 23
    private int minute; // 0 - 59
    private int second; // 0 - 59

    // Time2 no-argument constructor: initializes each instance variable
to zero; ensures that Time2 objects start in a consistent state
    public Time2()
    {
        this(0, 0, 0);  // invoke Time2 constructor with three arguments
    } // end Time2 no-argument constructor

    // Time2 constructor: hour supplied, minute and second defaulted to 0
    public Time2(int h)
    {
        this(h, 0, 0);  // invoke Time2 constructor with three arguments
    } // end Time2 one-argument constructor

    // Time2 constructor: hour and minute supplied, second defaulted to 0
    public Time2(int h, int m)
    {
        this(h, m, 0);  // invoke Time2 constructor with three arguments
    } // end Time2 two-argument constructor

    // Time2 constructor: hour, minute and second supplied
    public Time2(int h, int m, int s)
    {
        setTime(h, m, s);  // invoke setTime to validate time
    } // end Time2 three-argument constructor

    // Time2 constructor: another Time2 object supplied
    public Time2(Time2 time)
    {
        // invoke Time2 three-argument constructor
        this(time.getWorldTime());  // invoke Time2 constructor with a Time2 object argument
    } // end Time2 constructor with a Time2 object argument

    // Set Methods
    // set a new time value using universal time; ensure that
    // the data remains consistent by setting invalid values to zero
    public void setTime(int h, int m, int s)
    {
        setHour(h);  // set the hour
        setMinute(m);  // set the minute
        setSecond(s);  // set the second
    } // end method setTime
// validate and set hour
public void setHour(int h)
{
    hour = ( ( h >= 0 && h < 24 ) ? h : 0 );
} // end method setHour

// validate and set minute
public void setMinute(int m)
{
    minute = ( ( m >= 0 && m < 60 ) ? m : 0 );
} // end method setMinute

// validate and set second
public void setSecond(int s)
{
    second = ( ( s >= 0 && s < 60 ) ? s : 0 );
} // end method setSecond

// Get Methods
// get hour value
public int getHour()
{
    return hour;
} // end method getHour

// get minute value
public int getMinute()
{
    return minute;
} // end method getMinute

// get second value
public int getSecond()
{
    return second;
} // end method getSecond

// convert to String in universal-time format (HH:MM:SS)
public String toUniversalString()
{
    return String.format("%02d:%02d:%02d", getHour(), getMinute(), getSecond());
} // end method toUniversalString

// convert to String in standard-time format (H:MM:SS AM or PM)
public String toString()
{
    return String.format("%d:%02d:%02d %s", 
                        ( (getHour() == 0 || getHour() == 12) ? 12 : getHour() % 12 ),
                        getMinute(), getSecond(), ( getHour() < 12 ? "AM" : "PM" ) );
} // end method toString

/*
 * Homework added here.
 */
private void NormalizeTime()
{
    if (this.second >= 60){
        this.minute += this.second / 60;
        this.second = this.second % 60;
    }
    if (this.minute >= 60){
        this.hour += this.minute / 60;
        this.minute = this.minute % 60;
    }
    if (this.hour >= 24){
        // No DAY const.
        this.hour = this.hour % 24;
    }
}

public void tick()
{
    this.second++;
    this.NormalizeTime();
}

public void incrementMinute()
{
    this.minute++;
    this.NormalizeTime();
}

public void incrementHour()
{
    this.hour++;
    this.NormalizeTime();
}

} // end class Time2
/**
 * TimeTest class
 *
 * Used for test Time2 class working correctly.
 */

public class TimeTest {

    /**
     * @param args
     */
    public static void main(String[] args) {
        System.out.println("TimeTest
Written by Stevennick.
All rights reserved.");
        Time2 t1 = new Time2(11, 25, 42); // 12:25:42
        System.out.println("t1: hour, minute and second specified
Original:");
        System.out.printf(" %s
", t1.toUniversalString());
        System.out.printf(" %s
", t1.toString());
        t1.tick();
        System.out.println("t1: after tick once:");
        System.out.printf(" %s
", t1.toString());
        for(int i = 1; i <= 62; i++){
            t1.tick();
        }
        System.out.println("t1: after 62 seconds:");
        System.out.printf(" %s
", t1.toString());
        t1.incrementMinute();
        System.out.println("t1: after one minute:");
        System.out.printf(" %s
", t1.toString());
        for(int i = 1; i <= 35; i++){
            t1.incrementMinute();
        }
        System.out.println("t1: after 35 minutes:");
        System.out.printf(" %s
", t1.toString());
        t1.incrementHour();
        System.out.println("t1: after one hour:");
        System.out.printf(" %s
", t1.toString());
        for(int i = 1; i <= 12; i++){
            t1.incrementHour();
        }
    }
}
56   }
57   System.out.println("t1: after half day:");
58   System.out.printf(" %s\n", t1.toString());
59
60   t1.setSecond(59);
61   t1.setMinute(59);
62   t1.setHour(23);
63   System.out.println("t1: Reset to 23:59:59:");
64   System.out.printf(" %s\n", t1.toString());
65   t1.tick();
66   System.out.println("t1: after one second:");
67   System.out.printf(" %s\n", t1.toString());
68
69   }
70
71   }
72
Write an enum type TRafficLight, whose constants (RED, GREEN, YELLOW) take one parameter: the duration of the light. Write a program to test the trafficLight enum so that it displays the enum constants and their durations.

```
E:\THUWorks\java\程式設計\hw4\TRafficLight>TrafficLightTest.class
TrafficLightTest
Written by Stevennick.
All rights reserved.
TrafficLight is GREEN
YELLOW light is displayed, hurry up!
TrafficLight now is RED, please stop your step and wait...
TrafficLight now is GREEN again.
```
```java
// StdNo.:s942864 邱逸夫
// input: none
// output: see code.
// Date: 05/23/07
/**
 * TrafficLight
 */

public class TrafficLight {

   private enum trafficLight{RED, GREEN, YELLOW};
   private trafficLight status;

   public TrafficLight(){
      this.status = trafficLight.GREEN;
   }

   public String toString(){
      return String.format(this.status.toString());
   }

   public void changeStatus(){
      switch(this.status){
         case GREEN:
            this.status = trafficLight.YELLOW;
            break;
         case YELLOW:
            this.status = trafficLight.RED;
            break;
         case RED:
            this.status = trafficLight.GREEN;
            break;
      }
   }
}
```
// StdNo: s942864
// input: none
// output: see code.
// Date: 05/23/07

/**
 * TrafficLightTest
 */

public class TrafficLightTest {

    /**
     * @param args
     */
    public static void main(String[] args) {
        System.out.println("TrafficLightTest\nWritten by Stevennick.\nAll rights reserved.");
        TrafficLight t = new TrafficLight();
        System.out.println("TrafficLight is " + t);
        t.changeStatus();
        System.out.println(t + " light is displayed, hurry up!");
        t.changeStatus();
        System.out.println("TrafficLight now is " + t + ", please stop your step and wait...");
        t.changeStatus();
        System.out.println("TrafficLight now is " + t + " again.");
    }
}

SteveHome Computing Workstation, Inc.
(Complex Numbers) Create a class called Complex for performing arithmetic with complex numbers. Complex numbers have the form

\[ a + bi \]

where \( i \) is \( \sqrt{-1} \).

Write a program to test your class. Use floating-point variables to represent the private data of the class. Provide a constructor that enables an object of this class to be initialized when it is declared. Provide a no-argument constructor with default values in case no initializers are provided. Provide public methods that perform the following operations:

a. Add two Complex numbers: The real parts are added together and the imaginary parts are added together.

b. Subtract two Complex numbers: The real part of the right operand is subtracted from the real part of the left operand, and the imaginary part of the right operand is subtracted from the imaginary part of the left operand.

c. Print Complex numbers in the form \((a, b)\), where \(a\) is the real part and \(b\) is the imaginary part.

```
E:\THUWorks\java 程式設計\hw4\Complex>ComplexTest.class
Input frist complex number: real number:5
Input frist complex number: imaginaty number:7
The frist complex number is: 5.0+7.0i
Input second complex number: real number:3.2
Input second complex number: imaginaty number:9.66
The second complex number is: 3.2+9.66i
Frist + Second is:8.2+16.66i
Second - Frist is:-1.799999999999999+2.66i
```
/**
 * @author Stevennick
 * *
 */
 public class Complex {
 private double realPart;
 private double imaginatyPart;
 private double result;

 public Complex(){
 this.realPart = 0.0;
 this.imaginatyPart = 0.0;
 this.result = 0.0;
 }

 public Complex(double realPart, double imaginatyPart){
 this.realPart = realPart;
 this.imaginatyPart = imaginatyPart;
 this.result = this.realPart + this.imaginatyPart * java.lang.Math.sqrt(-1);
 }

 /*
 * This method is absolute.
 * Do not use it.
 */
 public double result(){
 System.out.print(this.result);
 return this.result;
 }

 // Do java have operator overloading? The answer is no.
 public void Add(Complex e){
 this.realPart += e.realPart;
 this.imaginatyPart += e.imaginatyPart;
 }

 public void Subtract(Complex e){
 this.realPart -= e.realPart;
 this.imaginatyPart -= e.imaginatyPart;
 }

 public String toString(){
 return String.format(this.realPart + "+" + this.imaginatyPart + "i");
 }

 public void print(){

}
System.out.print(this.toString());
// StdNo.:s942864 邱逸夫
// input: none
// output: see code.
// Date: 05/23/07

/**
 * @author Stevennick
 */

import java.util.Scanner;

public class ComplexTest {

    /**
     * @param args
     */
    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);
        double realpart = 0;
        double imaginatypart = 0;

        System.out.print("Input frist complex number: real number:");
        realpart = input.nextDouble();

        System.out.print("Input frist complex number: imaginaty number:");
        imaginatypart = input.nextDouble();

        Complex c = new Complex(realpart, imaginatypart);
        System.out.println("The frist complex number is: " + c);

        System.out.print("Input second complex number: real number:");
        realpart = input.nextDouble();

        System.out.print("Input second complex number: imaginaty number:");
        imaginatypart = input.nextDouble();

        Complex d = new Complex(realpart, imaginatypart);
        System.out.println("The second complex number is: " + d);

        c.Add(d);
        System.out.println("Frist + Second is:" + c);

        c.Subtract(d);
        d.Subtract(c);

        System.out.println("Second - Frist is:" + d);
    }
}
(Rational Numbers) Create a class called Rational for performing arithmetic with fractions. Write a program to test your class. Use integer variables to represent the private instance variables of the class—the numerator and the denominator. Provide a constructor that enables an object of this class to be initialized when it is declared. The constructor should store the fraction in reduced form. The fraction

\[ \frac{2}{4} \]

is equivalent to \( \frac{1}{2} \) and would be stored in the object as 1 in the numerator and 2 in the denominator. Provide a no-argument constructor with default values in case no initializers are provided. Provide public methods that perform each of the following operations:

a. Add two Rational numbers: The result of the addition should be stored in reduced form.

b. Subtract two Rational numbers: The result of the subtraction should be stored in reduced form.

c. Multiply two Rational numbers: The result of the multiplication should be stored in reduced form.

d. Divide two Rational numbers: The result of the division should be stored in reduced form.

e. Print Rational numbers in the form \( a/b \), where \( a \) is the numerator and \( b \) is the denominator.

f. Print Rational numbers in floating-point format. (Consider providing formatting capabilities that enable the user of the class to specify the number of digits of precision to the right of the decimal point.)

E:\THUWorks\java 程式設計\hw4\Rational>RationalTest.class
RationalTest
Written by Stevennick.
All rights reserved.
Input numerator number:56
Input denominator number:497
The Numerator is 8 , and the Denominator is 71
Add:79
Subtract:63
Multiply:568
Divide:0
You just get 8/71
And it's floating-point format is 0.112676054
// StdNo.: s942864 邱逸夫
// input: none
// output: see code.
// Date: 05/23/07

public class Rational {

    private int numerator;
    private int denominator;

    public Rational() {
        this.numerator = 0;
        this.denominator = 0;
    }

    public Rational(int numerator, int denominator) {
        this.numerator = numerator;
        this.denominator = denominator;
        this.CalcReducedNumber();
    }

    /*
     * 顯示目前數字
     */
    public void print() {
        System.out.print(this.numerator + "/", + this.denominator);
    }

    public void printfloat() {
        System.out.print(this.CalcRationalNumber());
    }

    public int AddNumber() {
        return this.numerator + this.denominator;
    }

    public int SubtractNumber() {
        return this.denominator - this.numerator;
    }

    public int MultiplyNumber() {
        return this.numerator * this.denominator;
    }

    public int DivideNumber() {
        int DividedNumber = 0;
        try {
            DividedNumber = this.numerator / this.denominator;
        } catch (Exception e) {
            throw new Exception("Rational.DivideNumber: Devided by zero.");
            e.printStackTrace();
        }
        return DividedNumber;
    }

    /*
private float CalcRationalNumber(){
    float RationalNumber = 0;
    try {
        RationalNumber = (float)this.numerator / (float)this.denominator;
    } catch (Exception e) {
        //throw new Exception("Rational.DivideNumber: Devided by zero.");
        e.printStackTrace();
    }
    return RationalNumber;
}

/*
* CalcRationalNumber
* 計算最簡化分數
*/
private Boolean CalcReducedNumber(){
    if(this.denominator > 0)
    {
        int gcdNumber = this.GCDNumber(this.numerator, this.denominator);
        this.numerator = this.numerator / gcdNumber;
        this.denominator = this.denominator / gcdNumber;
        return true;
    }
    return false;
}

/*
* CalcRationalNumber
* 回傳分母
*/
public int getNumerator(){
    return this.numerator;
}

/*
* getDenominator
*/
public int getDenominator(){
    return this.denominator;
}

/*
* setNumerator
*/
public void setNumerator(int numerator){
    this.numerator = numerator;
}

/*
* setDenominator
*/
public void setDenominator(int denominator){

this.denominator = denominator;

/*
  * GCDNumber
  * 計算最大公因數
  */
private int GCDNumber(int m, int n){
    int r;
    while(n != 0) {
        r = m % n;
        m = n;
        n = r;
    }
    return m;
}
import java.util.Scanner;

public class RationalTest {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int Numerator = 0;
        int Denominator = 0;
        Scanner inputStream = new Scanner(System.in);
        System.out.println("RationalTest\nWritten by Stevennick.\nAll rights reserved.\n");
        System.out.println("Input numerator number:");
        Numerator = inputStream.nextInt();
        System.out.println("Input denominator number:");
        Denominator = inputStream.nextInt();
        inputStream.close();
        Rational myRational = new Rational(Numerator, Denominator);
        System.out.println("The Numerator is " + myRational.getNumerator() + ", and the Denominator is " + myRational.getDenominator());
        System.out.println("Add:" + myRational.AddNumber());
        System.out.println("Subtract:" + myRational.SubtractNumber());
        System.out.println("Multiply:" + myRational.MultiplyNumber());
        System.out.println("Divide:" + myRational.DivideNumber());
        System.out.println("You just get ");
        myRational.print();
        System.out.println();
        System.out.println("And it's floating-point format is ");
        myRational.printf();
        System.out.println();
    }
}

Complete following code:

```java
E:\THUWorks\java程式設計\hw4\college>collegeTest.class
collegeTest
Written by Stevennick.
All rights reserved.
teacher Data:
ssn: 111
name: John
sex: M
course: JAVA
salary: 666666

student Data:
ssn: 222
name: Mary
sex: F
course: JAVA
student no: 222222
```
public class college{
    private int ssn;
    private String name;
    private char sex;

    public college(int s, String n, char x){
        ssn = s;
        name = n;
        sex = x;
    }

    /*
    *(non-Javadoc)
    *@see java.lang.Object#toString()
    */
    public String toString(){
        return String.format("ssn: " + ssn + "name: " + name + "sex: " + sex);
    }
}
public class teacher extends college{
    private String course;
    private int salary;
    
    public teacher(int s, String n, char x, String c, int s1){
        super(s,n,x);
        this.course = c;
        this.salary = s1;
    }
    
    public String toString(){
        return String.format(super.toString() + "course: " + this.course + "salary: " + this.salary);
    }
}
```java
public class student extends college{

    private String course;
    private int student_no;

    // ssn,name,sex,course,number
    public student(int ssn, String name, char sex, String course, int student_no)
    {
        super(ssn, name, sex);
        this.course = course;
        this.student_no = student_no;
    }

    public String toString()
    {
        return String.format(super.toString() + "\ncourse: " + this.course + "\nstudent no: " + this.student_no);
    }
}
```
public class collegeTest{
    public static void main(String args[]){
        System.out.println("collegeTest
Written by Stevennick\nAll rights reserved.");
        teacher tt = new teacher(111,"John", 'M', "JAVA", 666666);
        System.out.println("%s\n%s\n", "teacher Data:", tt.toString());
        System.out.println("teacher Data:
" + tt);
        System.out.println();
        student ss = new student(222,"Mary", 'F', "JAVA", 222222);
        System.out.println("%s\n%s\n", "student Data:", ss.toString());
        System.out.println("student Data:\n" + ss);
    }
}