Tutorial 10

Objects & Classes: Methods | Constructors | Access Control

Exercise 1:

A. Suppose that that the class **`F`** is defined in (a). Let **`f`** be an instance of **`F`**. Which of the statements in (b) are correct?

```
public class F{
  int i;
  static String s;
  void imethod(){
  }
  static void smethod() {
  }
}
```

(a)

```
01 System.out.println(f.i);
02 System.out.println(f.s);
03 f.imethod();
04 f.smethod();
05 System.out.println(F.i);
06 System.out.println(F.s);
07 F.imethod();
08 F.smethod();
```

(b)

B. Add static keyword in place of ? if appropriate

```
public class Test {
  int count;
  public ? void main(String[] args) {
    ...
  }
  public ? int getCount() {
    return count;
  }
  public ? int factorial(int n) {
    int result = 1;
    for (int i = 1; i <= n; i++)
      result *= i;
    return result;
  }
}
```

C. In each place where there is a ?, list all properties of class C1 that are accessible and the ones that are not accessible. Also list all methods that can be invoked and the ones that cannot be invoked.

```
package p1;
public class C1 {
  public int x;
  int y;
  private int z;
  public void m1(){
  }
  void m2(){
  }
  private void m3(){
  }
}
```

```
package p1;
public class C2 {
  void aMethod(){
    C1 o = new C1();
  }
}
```

```
package p2;
public class C3 {
  void aMethod(){
    C1 o = new C1();
  }
}
```

D. Put a line under the errors in the following program (Notice: the program consists of two files):

```
01 class FullOfErrors {
02   private int prop1;
03   private doule prop2;
04   public FullOfErrors(int p1) {
05     prop1 = p1;
06   }
07   public void setProp1(double p) {
08     prop1 = p;
09   }
10   public int getProp2() {
11     return prop2;
12   }
13   public int getProp1() {
14     System.out.println("prop1= "+prop1);
15   }
16   public void setProp1Prop2(double a, int b) {
17     prob1 = b; prob2 = a;
18   }
19 }
20
21 class TestFullOfErrors {
22   public static void main(String[] args) {
23     FullOfErrors m = new FullOfErrors();
24     FullOfErrors m2 = new FullOfErrors(5);
25     int x = 1; int y;
26     y = m.setProp1(x + 3);
27     m.setProp1prop2(1, 1.0);
28     m.prop2 = 2.0;
29   }
30 }
```

E. What is the output of the following program?

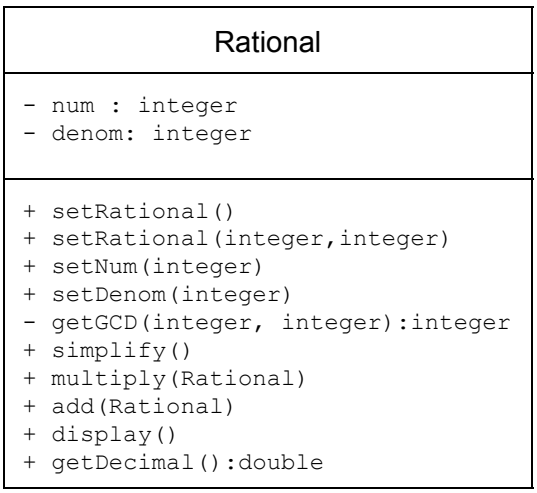
```
class Magic {
  int i;
  double j;
}
public class TestMagic {
  public static void main(String[] args) {
    Magic m = new Magic();
    m.i = 11;
    m.j = 5.5;
    Magic m2 = new Magic();
    m2 = m;
    m2.i = m2.i + 2;
    m2.j = 1 + m2.i / ((m.i - 9) / 2);
    System.out.println(m.i + ", " + m.j + ", " + m2.i + ", " +
m2.j);
  }
}
```

Exercise 2:

A rational number is any number that can be expressed as the quotient or fraction num/denom of two integers: numerator num and denominator denom, with the denominator not equal to zero.

The class offersthe following services:

- `setNum` to change the numerator
- `setDenom` to change the denominator
- `setFraction` to change both components
- `getGCD` returns the greatest common divisor of two integers
- `simplify` converts the fraction to its simplest form
- `multiply` multiplies the current fraction with another and saves the result in the current fraction
- `add` adds the current fraction to another and saves the result in the current fraction
- `display` displays the fraction in the form: num/denom
- `getDecimal` returns the fraction's value in the decimal form



A. Implement the class Rational described above.

B. Write a test program that:

1. creates the rational number r1 = 12/8 using the constructor that takes no parameters
2. creates the rational number r2 = -7/3 using the constructor that takes two parameters
3. display r1, simplify it, then display it again
4. multiply r1 with r2 and store the result in r1, then display r1
5. display r2, add r1 to it, then display it again
6. display the rational number of greatest value.

Tutorial 10 Solutions

Exercise 1:

A. Correct: 01, 02, 03, 04, 06, 08

B.
```
public class Test {
    int count;
    public static void main(String[] args) {
      ...
    }
    public int getCount() {
      return count;
    }
    public static int factorial(int n) {
      int result = 1;
      for (int i = 1; i <= n; i++)
        result *= i;
      return result;
    }
}
```

C. `aMethod` of `C2` can access `o.x` and `o.y` but not `o.z` and invoke `o.m1` and `o.m2` but not `o.m3`

D. Errors: 08, 11, 13, 23, 24, 26, 27, 28

E.

```
13, 7.0, 13, 7.0
```

Exercise 2:

A.
```
class Rational {
    private int num;
    private int denom;

    public void setNum(int x) {
      num = x;
    }
    public void setDenom(int x) {
      denom = x!=0? x:1;
    }
    public void setFraction(int x, int y){
      setNum(x);
      setDenom(y);
    }
    public int getGCD(int x, int y) {
      while (x != y)
        if (x>y) x-=y;
        else y-=x;
      return x;
    }
```

```
  public void simplify() {
    int gcd = getGCD(num, denom);
    num /= gcd;
    denom /= gcd;
  }
 public void multiply(Rational other) {
    num *= other.num;
    denom *= other.denom;
  }
  public void add(Rational other) {
    num=num*other.denom+denom*other.num;
    denom*=other.denom;
  }
  /* or
  public void add(Rational other) {
    int lcm=num*denom/getGCD(num,denom);
    int f1 = denom/lcm;
    int f2 = other.denom/lcm;
    num = num*f1 + other.num*f2;
    denom = denom*f1 + other.denom*f2;
  } */
  public void display() {
    System.out.println(num+"\"+denom);
  }
  public double getDecimal() {
    return num/(double)denom;
  }
}
```

B.

```
class TestRational{
  public static void main(String[] args) {
    // 1
    Rational r1 = new Rational();
    r1.setNum(12);
    r1.setDenom(8);
    // 2
    Rational r2 = new Rational(-7, 3);
    // 3
    r1.display();
    r1.simplify();
    r1.display();
    // 4
    r1.multiply(r2);
    r1.display();
    // 5
    r2.display();
    r2.add(r1);
    r2.display();
```

```
    // 6
    System.out.print("The greatest rational number is: ");
    if (r1.getDecimal() > r2.getDecimal())
      r1.display();
    else
      r2.display();
  }
}
```