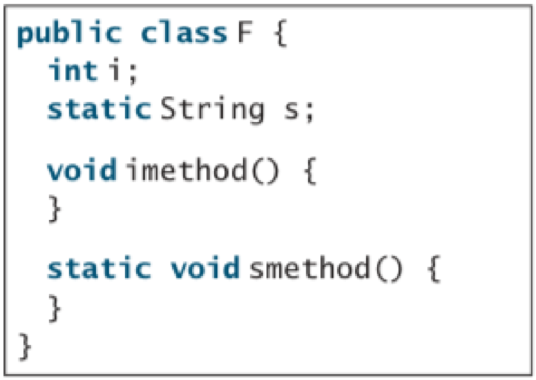
King Saud University
College of Computer & Information Science
CSC111 – Tutorial10
Object Oriented Programming – III –
--

Exercise 1

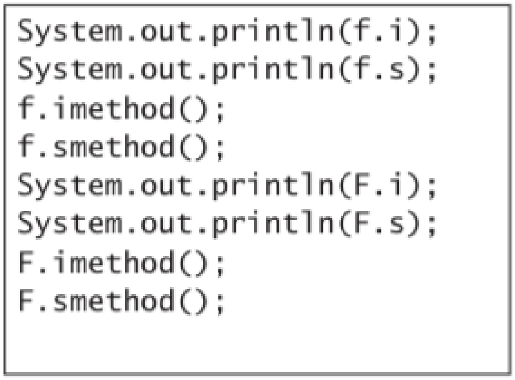
1. Suppose that the class F is defined in (a). Let f be an instance of F.

   Which of the statements in (b) are correct?



```
public class F {
  int i;
  static String s;

  void imethod() {
  }

  static void smethod() {
  }
}
```

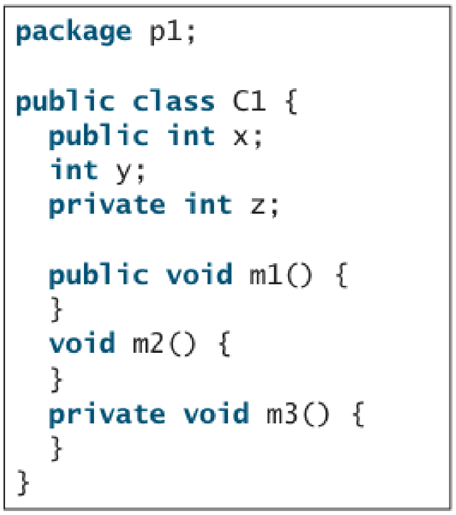
(a)



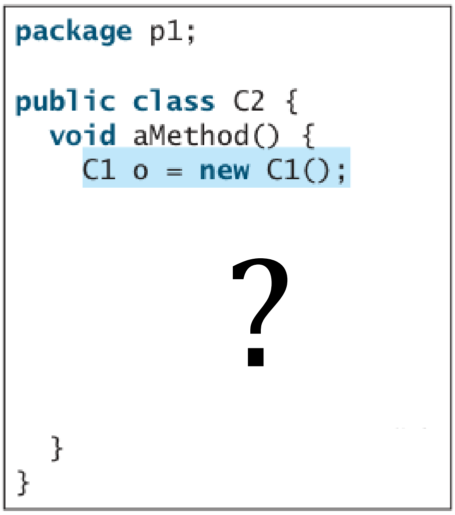
```
System.out.println(f.i);
System.out.println(f.s);
f.imethod();
f.smethod();
System.out.println(F.i);
System.out.println(F.s);
F.imethod();
F.smethod();
```

(b)

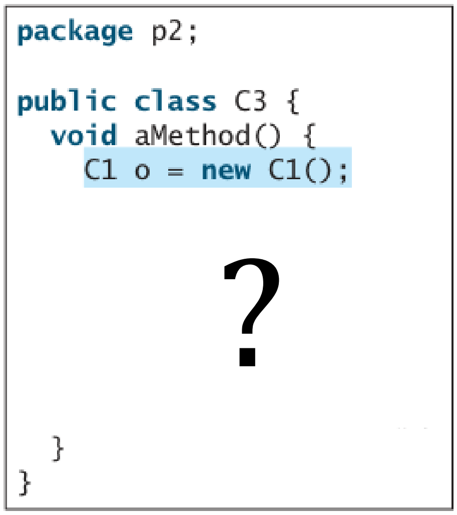
2. In each place where there is a **?**, list all properties of class C1 that are accessible and the ones that are not accessible. Also list all methods that can be invoked and the ones that cannot be invoked.



```
package p1;

public class C1 {
  public int x;
  int y;
  private int z;

  public void m1() {
  }
  void m2() {
  }
  private void m3() {
  }
}
```

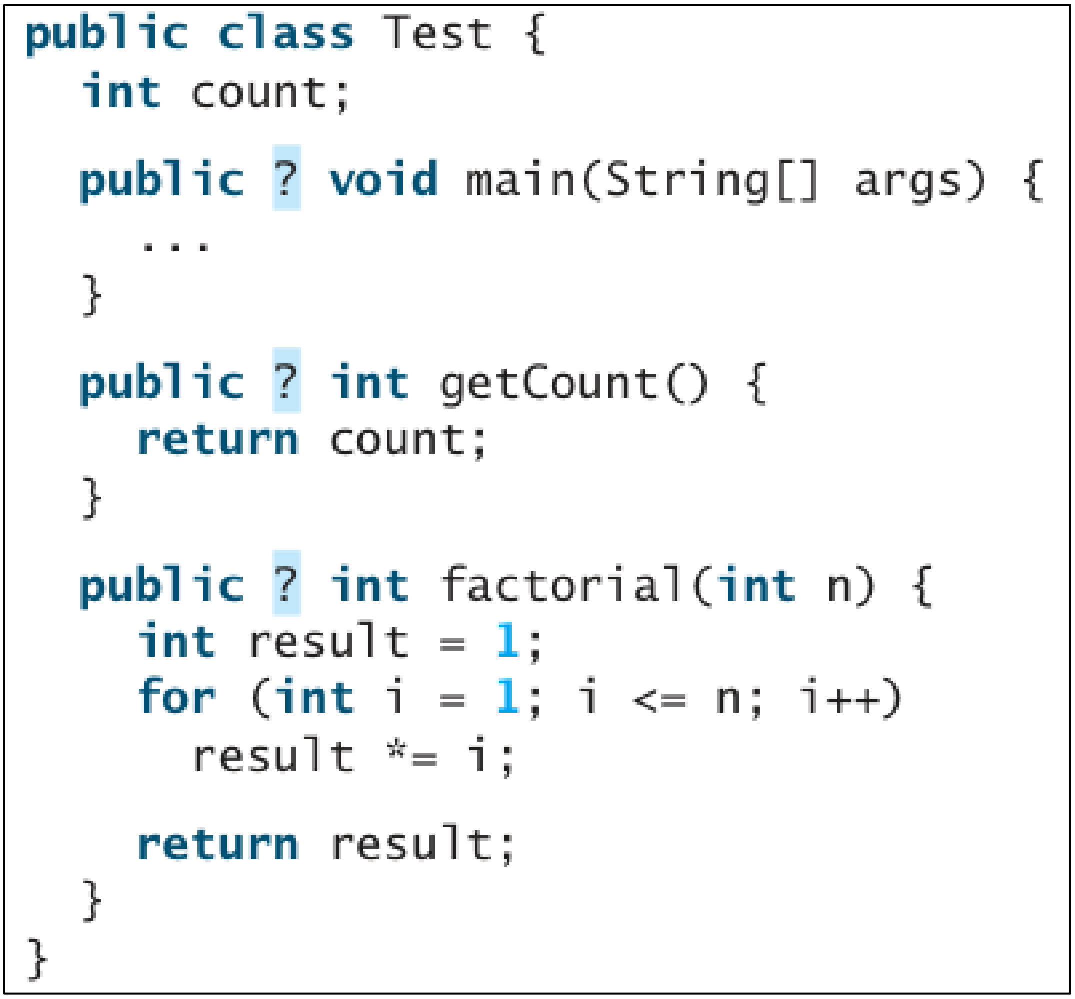


```
package p1;

public class C2 {
  void aMethod() {
    C1 o = new C1();

            ?

  }
}
```



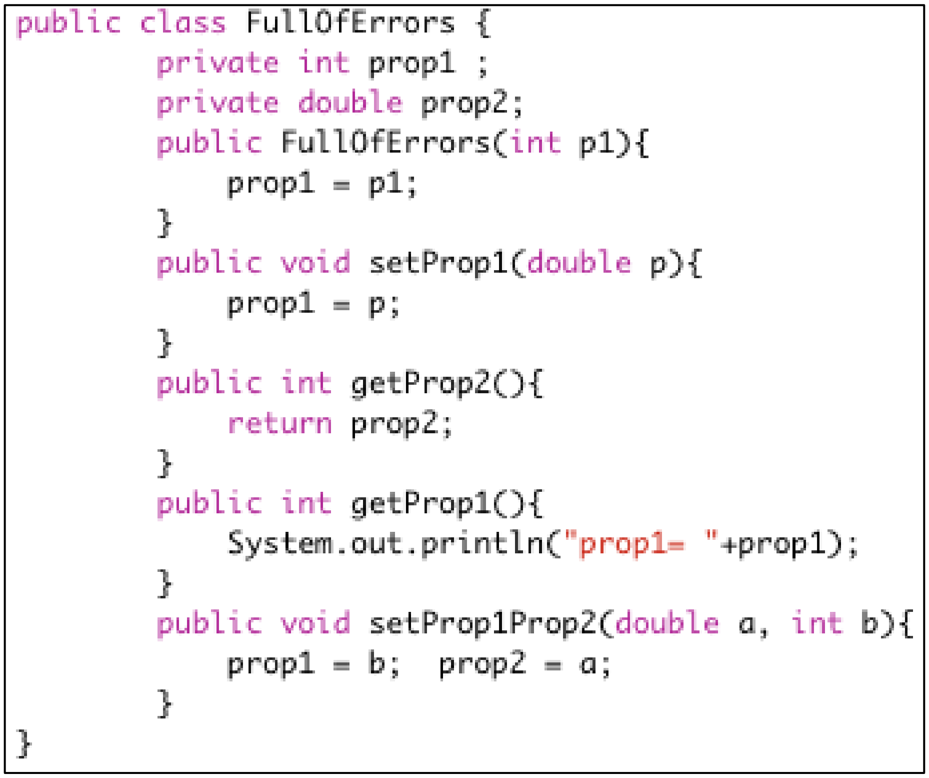
```
package p2;

public class C3 {
  void aMethod() {
    C1 o = new C1();

            ?

  }
}
```

3. Add static keyword in place of ? if appropriate

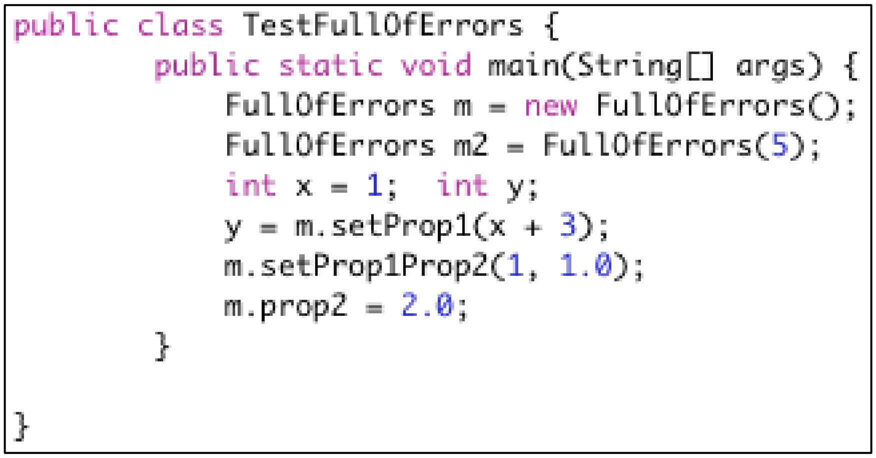


```
public class Test {
  int count;

  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;
  }
}
```

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

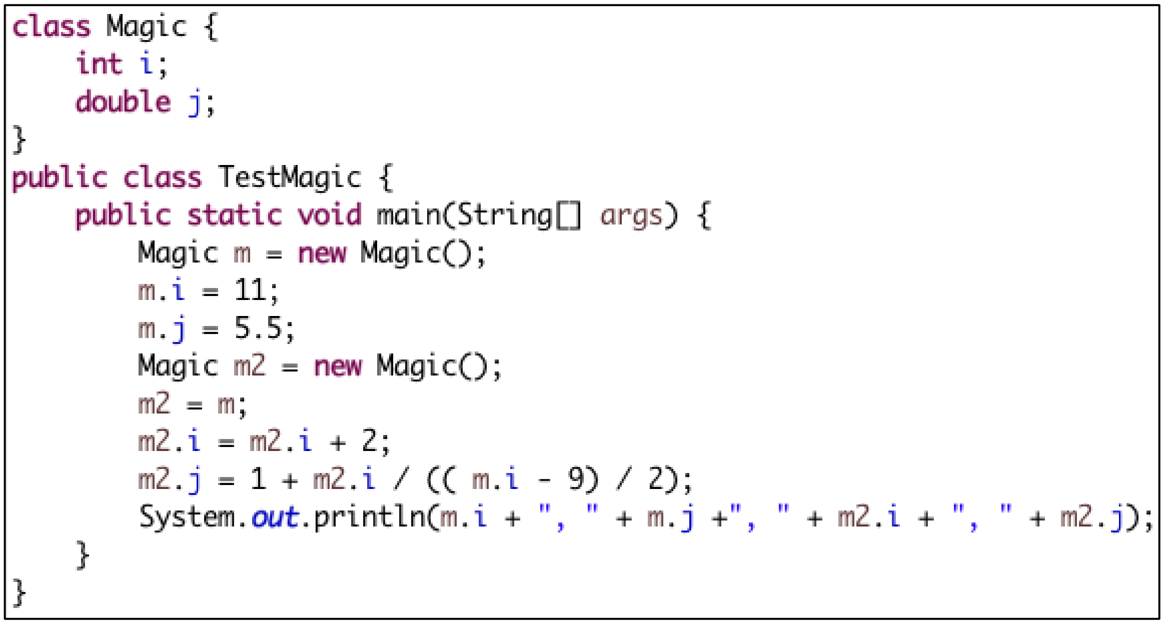


```
public class FullOfErrors {
        private int prop1 ;
        private double prop2;
        public FullOfErrors(int p1){
            prop1 = p1;
        }
        public void setProp1(double p){
            prop1 = p;
        }
        public int getProp2(){
            return prop2;
        }
        public int getProp1(){
            System.out.println("prop1= "+prop1);
        }
        public void setProp1Prop2(double a, int b){
            prop1 = b;  prop2 = a;
        }
}
```



```
public class TestFullOfErrors {
        public static void main(String[] args) {
            FullOfErrors m = new FullOfErrors();
            FullOfErrors m2 = FullOfErrors(5);
            int x = 1;  int y;
            y = m.setProp1(x + 3);
            m.setProp1Prop2(1, 1.0);
            m.prop2 = 2.0;
        }

}
```

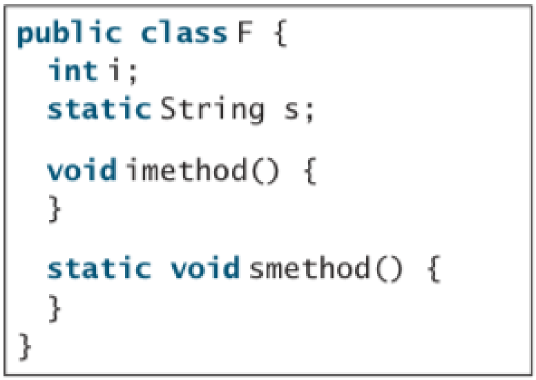
5. What is the output of the following program?



```java
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);
    }
}
```

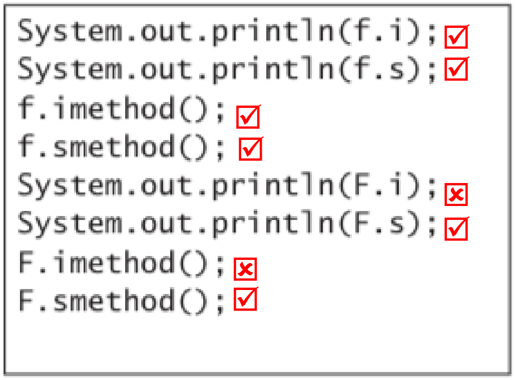
Solution

1)



```
public class F {
  int i;
  static String s;

  void imethod() {
  }

  static void smethod() {
  }
}
```

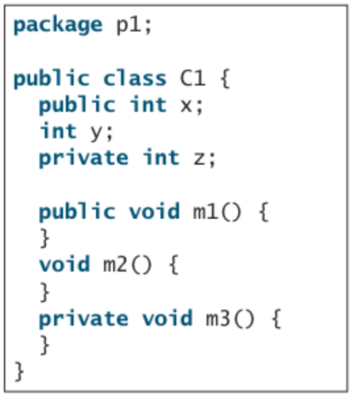
(a)



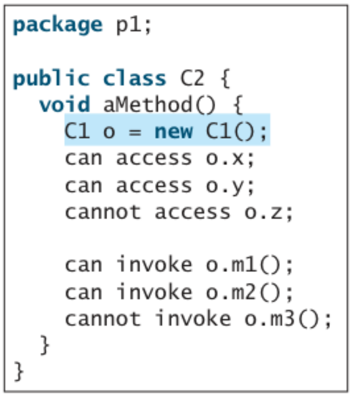
```
System.out.println(f.i); ☑
System.out.println(f.s); ☑
f.imethod(); ☑
f.smethod(); ☑
System.out.println(F.i); ☒
System.out.println(F.s); ☑
F.imethod(); ☒
F.smethod(); ☑
```

(b)

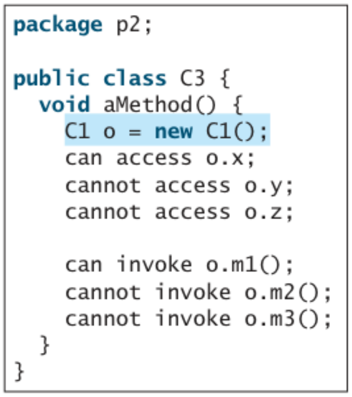
2)



```
package p1;

public class C1 {
  public int x;
  int y;
  private int z;

  public void m1() {
  }
  void m2() {
  }
  private void m3() {
  }
}
```

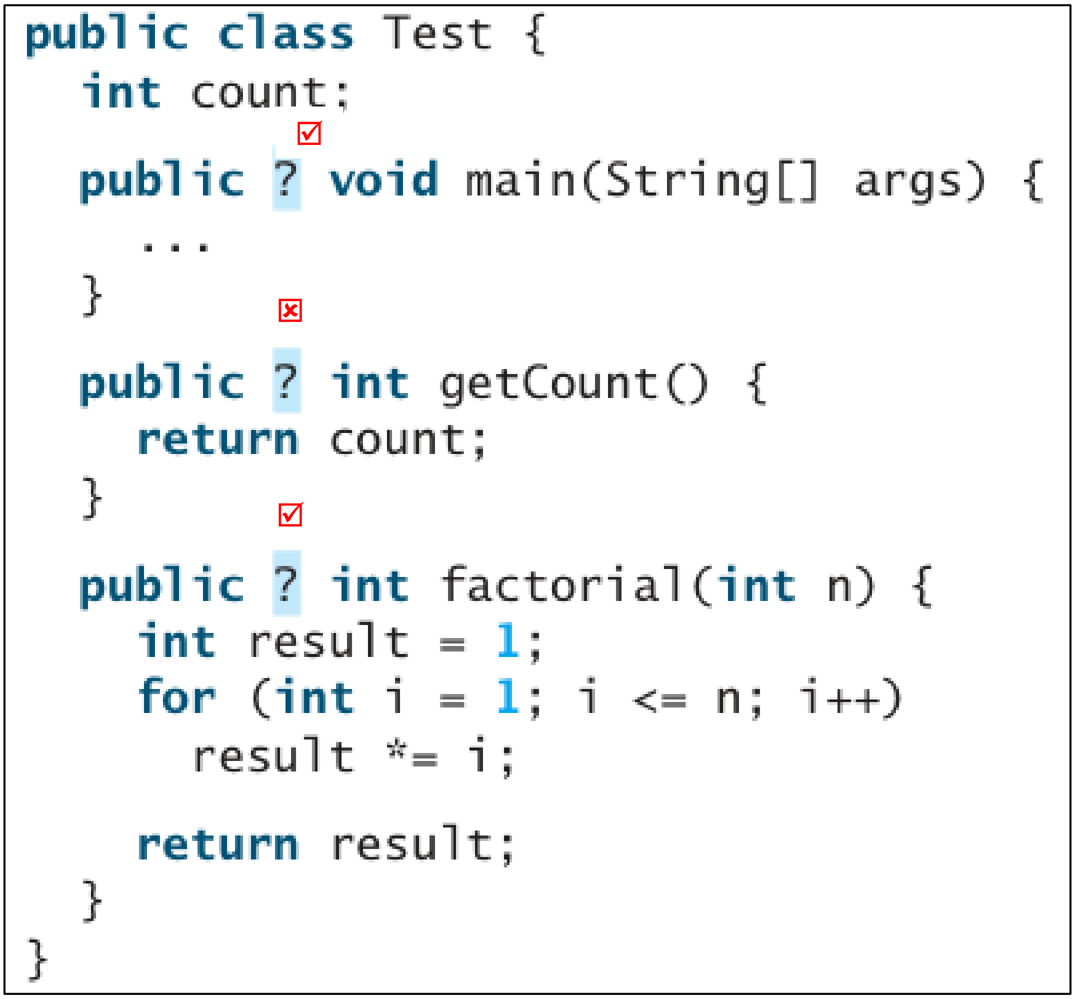


```
package p1;

public class C2 {
  void aMethod() {
    C1 o = new C1();
    can access o.x;
    can access o.y;
    cannot access o.z;

    can invoke o.m1();
    can invoke o.m2();
    cannot invoke o.m3();
  }
}
```



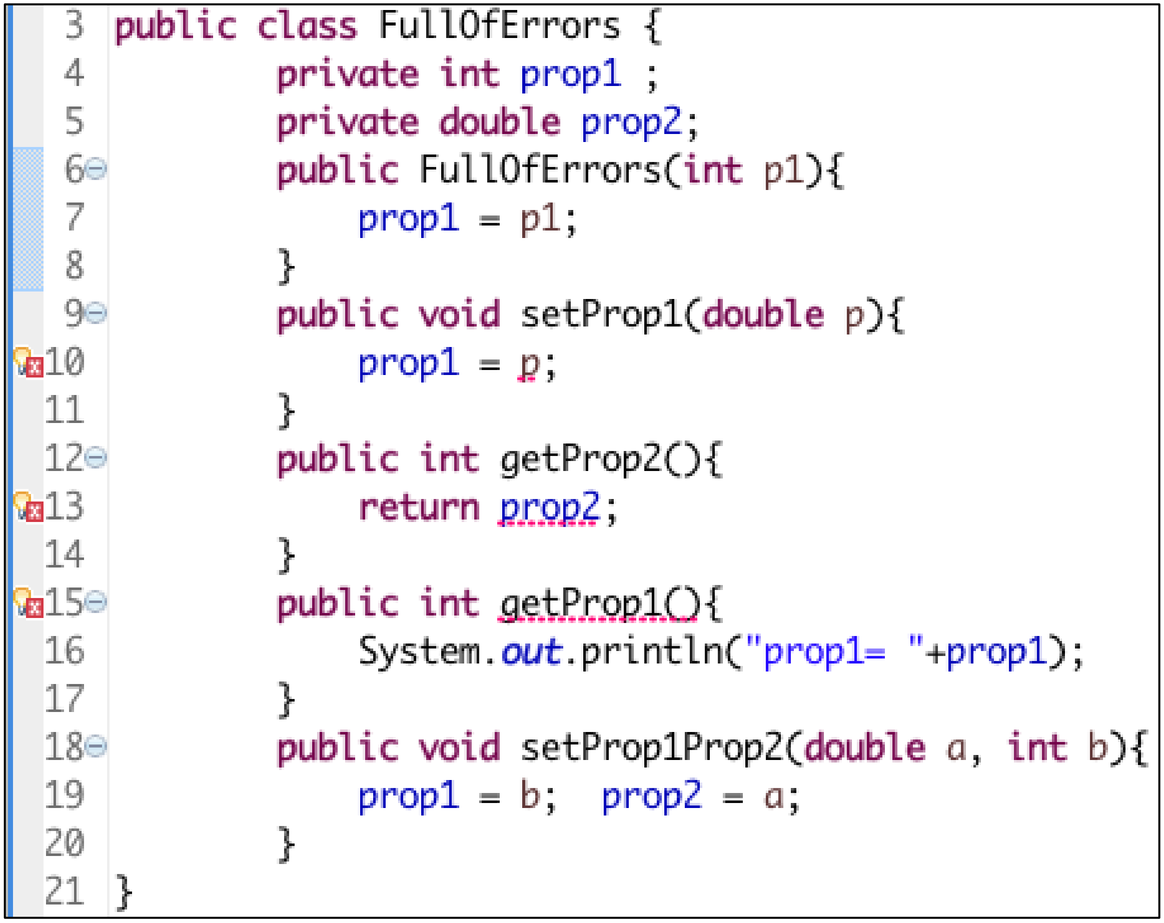
```
package p2;

public class C3 {
  void aMethod() {
    C1 o = new C1();
    can access o.x;
    cannot access o.y;
    cannot access o.z;

    can invoke o.m1();
    cannot invoke o.m2();
    cannot invoke o.m3();
  }
}
```

3)

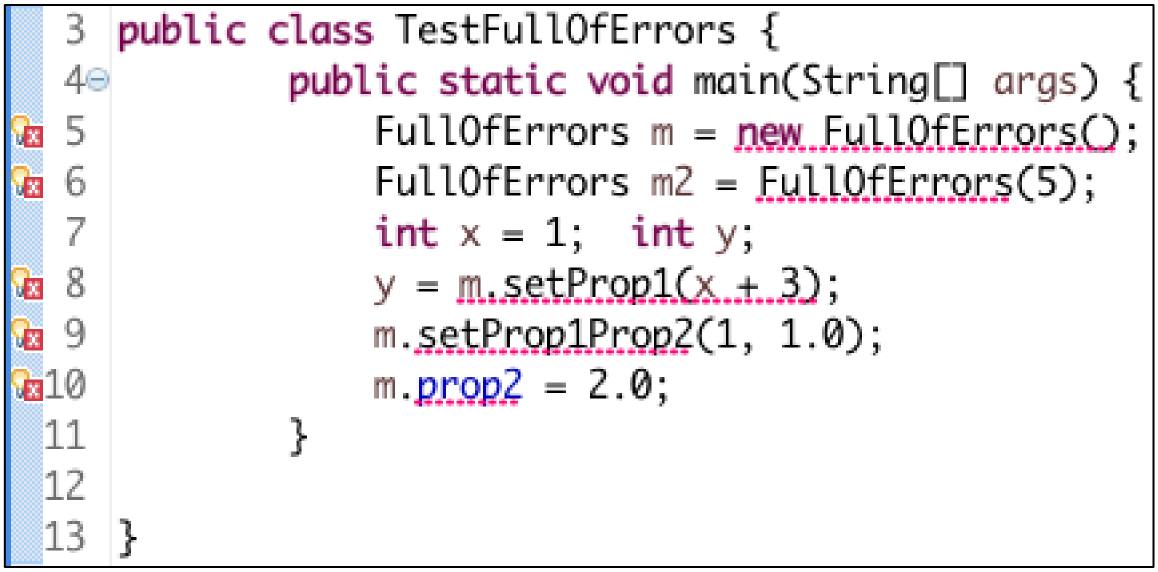


```
public class Test {
  int count;
            ☑
  public ? void main(String[] args) {
    ...
  }
           ☒
  public ? int getCount() {
    return count;
  }
           ☑
  public ? int factorial(int n) {
    int result = 1;
    for (int i = 1; i <= n; i++)
      result *= i;

    return result;
  }
}
```

4)



```
public class FullOfErrors {
        private int prop1 ;
        private double prop2;
        public FullOfErrors(int p1){
            prop1 = p1;
        }
        public void setProp1(double p){
            prop1 = p;
        }
        public int getProp2(){
            return prop2;
        }
        public int getProp1(){
            System.out.println("prop1= "+prop1);
        }
        public void setProp1Prop2(double a, int b){
            prop1 = b;  prop2 = a;
        }
}
```



```
public class TestFullOfErrors {
        public static void main(String[] args) {
            FullOfErrors m = new FullOfErrors();
            FullOfErrors m2 = FullOfErrors(5);
            int x = 1;  int y;
            y = m.setProp1(x + 3);
            m.setProp1Prop2(1, 1.0);
            m.prop2 = 2.0;
        }

}
```

5) OUTPUT

13, 7.0, 13, 7.0