# Exercises on OOP, Inheritance, and Polymorphism

1. The following Java applications contain errors. Point out the statement(s) that contain errors. Explain what each of the errors is, and how it can be fixed.

## Exercises 1.1

```
public class OOPExercises {
   public static void main(String[] args) {
      A objA = new A();
      System.out.println("in main(): ");
      System.out.println("objA.a = "+objA.a);
      objA.a = 222;
   }
}
```

```
public class A {
    private int a = 100;
    public void setA( int value) {
       a = value;
   }
    public int getA() {
       return a;
    }
} //class A
```

**Point out the error(s) and how they can be fixed.**

## Exercises 1.2

```
public class OOPExercises {
   public static void main(String[] args) {
      System.out.println("in main(): ");
      System.out.println("objA.a = "+getA() );
      setA(123);
   }
}
```

```
public class A {
    private int a = 100;
    public void setA( int value) {
       a = value;
   }
    public int getA() {
       return a;
    }
} //class A
```

**Point out the error(s) and how they can be fixed.**

## Exercises 1.3

```
public class OOPExercises {
   public static void main(String[] args) {
      A objA = new A( );
      double result;
       result = objA.getA( );
       System.out.println("objA.a = "+ result);
   }
}
```

```
public class A {
     private int a = 100;
     public void setA( int value) {
        a = value;
    }
     public int getA() {
        return a;
     }
} //class A
```

**Point out the error(s) and how they can be fixed.**

## Exercises 1.4

```
public class B extends A {
   private int a = 222;

    public static void main(String[] args) {
       System.out.println("in main(): ");
       System.out.println("a = "+a );
       a = 123;
   }
}
```

```
public class A {
     private int a = 100;
     public void setA( int value) {
        a = value;
    }
     public int getA() {
        return a;
     }
} //class A
```

**Point out the error(s) and how they can be fixed.**

2. Show the output of the following applications.

## Exercises 2.1

```
public class OOPExercises {
    public static void main(String[] args) {
       A objA = new A();
       B objB = new B();
       System.out.println("in main(): ");
```

**Output:**

```java
      System.out.println("objA.a = "+objA.getA());
      System.out.println("objB.b = "+objB.getB());
      objA.setA (222);
      objB.setB (333.33);
      System.out.println("objA.a = "+objA.getA());
      System.out.println("objB.b = "+objB.getB());
   }
}
```

```java
public class A {
    int a = 100;
    public A() {
       System.out.println("in the constructor of class A: ");
       System.out.println("a = "+a);
       a = 333;
       System.out.println("a = "+a);
    }
    public void setA( int value) {
       a = value;
    }
    public int getA() {
       return a;
    }
 } //class A
```

```java
 public class B {
    double b = 123.45;
    public B() {
       System.out.println("-----in the constructor of class B: ");
       System.out.println("b = "+b);
       b = 3.14159;
       System.out.println("b = "+b);
    }
    public void setB( double value) {
       b = value;
    }
    public double getB() {
       return b;
    }
 } //class B
```

## Exercises 2.2

```java
public class OOPExercises {
   public static void main(String[] args) {
      //A objA = new A();
      B objB = new B();
      System.out.println("in main(): ");
      //System.out.println("objA.a = "+objA.getA());
      System.out.println("objB.b = "+objB.getB());
      //objA.setA (222);
      objB.setB (333.33);
      //System.out.println("objA.a = "+objA.getA());
      System.out.println("objB.b = "+objB.getB());
```

**Output:**

```
   }
}
```

```
public class A {
     int a = 100;
     public A() {
        System.out.println("in the constructor of class A: ");
        System.out.println("a = "+a);
        a = 333;
        System.out.println("a = "+a);
     }
     public void setA( int value) {
        a = value;
     }
     public int getA() {
        return a;
     }
 } //class A
```

```
 public class B extends A {
    double b = 123.45;
    public B() {
       System.out.println("-----in the constructor of class B: ");
       System.out.println("b = "+b);
       b = 3.14159;
       System.out.println("b = "+b);
    }
    public void setB( double value) {
       b = value;
    }
    public double getB() {
       return b;
    }
 } //class B
```

## Exercises 2.3

```
public class OOPExercises {
   static int a = 555;
   public static void main(String[] args) {
      A objA = new A();
      B objB = new B();
      System.out.println("in main(): ");
      System.out.println("a = "+a);
      a = 444;
      System.out.println("objB.a = "+objB.getA());
      objA.setA (77777);
      objB.rollBackA();
      System.out.println("After roll back -----");
      System.out.println("a = "+a);
      System.out.println("objA.a = "+objA.getA());
      System.out.println("objB.a = "+objB.getA());
   }
}
```

**Output:**

```
public class A {
   int a = 100;
   public A() {
      //System.out.println("in the constructor of class A: ");
      //System.out.println("a = "+a);
      a = 333;
      //System.out.println("a = "+a);
   }
   public void setA( int value) {
      a = value;
   }
   public int getA() {
      return a;
   }
} //class A
```

```
public class B extends A {
   private int a = 123;
   public B() {
      a = 2222;
   }
   public void rollBackA () {
      a = super.getA();
   }
   public void setA( int value) {
      a = value;
   }
   public int getA() {
      return a;
   }
} //class B
```

## Exercises 2.4

```
public class OOPExercises {
   static int a = 555;
   public static void main(String[] args) {
      A objA = new A();
      B objB1 = new B();
      A objB2 = new B();
      C objC1 = new C();
      B objC2 = new C();
      A objC3 = new C();
      objA.display();
      objB1.display();
      objB2.display();
      objC1.display();
      objC2.display();
      objC3.display();
   }
}
```

**Output:**

```
public class A {
   int a = 100;
```

```
    public void display() {
        System.out.printf("a in A = %d\n", a);
    }
} //class A
```

```
public class B extends A {
    private int a = 123;
    public void display() {
        System.out.printf("a in B = %d\n", a);
    }
} //class B
```

```
public class C extends B {
    private int a = 543;
    public void display() {
        System.out.printf("a in C = %d\n", a);
    }
} //class C
```

3. Tracing programs: The above is the program demonstrated in class. Now, what gets printed to the screen when we execute the following classes on the left? Why?

**Exercises 3.1**

| | |
|---|---|
| <pre>public class A {<br>        public int x = 1;<br>        public void setX(int a){<br>                x=a;<br>        }<br>}<br>public class B extends A {<br>        public int getB(){<br>                setX(2);<br>                return x;<br>        }<br>}<br>public class C {<br>        public static void main(String [] args){<br>                A a = new A();<br>                B b = new B();<br>                System.out.println(a.x);<br>                System.out.println(b.getB());<br>        }<br>}</pre> | Result: |

| | |
|---|---|
| <pre>public class A {<br>        private int x = 1;<br>        protected void setX(int a){<br>                x=a;<br>        }<br>        protected int getX(){<br>                return x;}<br>        }<br>public class B extends A {<br>        public int getB(){<br>                setX(2);<br>                return getX();<br>        }<br>}<br>public class C {<br>        public static void main(String [] args){<br>                A a = new A();<br>                B b = new B();<br>                System.out.println(a.getX());<br>                System.out.println(b.getB());<br>        }<br>}</pre> | Result:<br>, |
| <pre>public class A {<br>        protected int x = 1;<br>        protected void setX(int a){x=a;}<br>        protected int getX(){return x;}<br>}<br>public class B extends A {<br>        public int getB(){<br>                setX(2);<br>                return x;<br>        }<br>}<br>public class C {<br>        public static void main(String [] args){<br>                A a = new A();<br>                B b = new B();<br>                System.out.println(a.getX());<br>                System.out.println(b.x);<br>                System.out.println(b.getB());<br>        }<br>}</pre> | Result |
| <pre>public class A {<br>        protected int x = 1;<br>        protected void setX(int a){<br>                x=a;<br>        }<br>        protected int getX(){<br>                return x;}<br>}<br>public class B extends A {<br>        protected int x = 3;</pre> | Resutls: |

```java
    public int getX(){
        return x; }
    public int getB(){
        return x;
    }
}
public class C {
    public static void main(String [] args){
        A a = new A();
        B b = new B();
        System.out.println(a.getX());
        System.out.println(b.getB());
        System.out.println(b.getX());
        System.out.println(a.x);
        System.out.println(b.x);
    }
}
```

```java
public class A {
    protected int x = 1;
    protected void setX(int a){
        x=a;
    }
protected int getX(){
        return x;}
}
public class B extends A {
    protected int x = 3;
    public int getX(){
        return x; }
    public int getB(){
        return x;
    }
}
public class C {
    public static void main(String [] args){
        A a = new A();
        A b = new B();
        System.out.println(a.getX());
        System.out.println(b.getX());
        System.out.println(a.x);
        System.out.println(b.x);
    }
}
```

Results:

```java
public class A {
    protected int x = 1;
    protected void setX(int a){ x=a;
    }
    protected int getX(){ return x;}
}
public class B extends A {
    protected int x = 3;
    public int getX(){
```

Results:

```
                setX(2);
                return x; }
        public int getB(){ return x;
        }
}
public class C {
        public static void main(String [] args){ A a = new A();
                A b = new B();
                System.out.println(a.getX());
                System.out.println(b.getX());
                System.out.println(a.x);
                System.out.println(b.x);
        }
}
```

**Exercises 3.2**

```
public class A
{
        private String x = "Ax";
        protected String y = "Ay";
        public String z = "Az";
        public String toString() {
                return x + y + z;
        }

        public static void main(String [] args){
                A a = new A();
                System.out.println(a);
        }
}
public class B extends A
{
        private String x = "Bx";
        public String z = "Bz";
        public String toString() {
                return x + y + z;
        }

        public static void main(String [] args){
                B b = new B();
                System.out.println(b);
        }
}
```

```
public class C extends A
{
        private String x = "Cx";
        public static void main(String [] args){
                C c = new C();
                System.out.println(c.x);
                System.out.println(c);
        }
}
public class D extends C
{
        private String x = "Dx";
        public String z = "Dz";
        public static void main(String [] args){
                D d = new D();
                System.out.println(d.x);
                System.out.println(d.y);
                System.out.println(d.z);
                System.out.println(d);

                C c = new D();
                System.out.println(c.x);
                System.out.println(c.y);
                System.out.println(c.z);
                System.out.println(c);
        }
}
```

4. Declare the following classes:

**Exercises 4.1**

- The **Rectangle** class, that extends Shape, describes a simple rectangle.

- The **Circle** class, that extends Shape, describes a simple circle.

- The **Shape** class is given as follows:

```
public abstract class Shape
{
    public abstract double area();
    public String toString()
    {
        return "The area is " + area();
    }
}
```

Check your classes using the following application:

```
public class Test
{
    public static void main(String args[])
    {
        Shape vec[] = {new Circle(3), new Rectangle(4,5),
new Circle(4), new Circle(8)};
        for(int index = 0; index < vec.length; index ++)
        {
            System.out.println(vec[index]);
        }
    }
}
```

**Exercises 4.2**

The given application should sort 50 objects that represent 50 Rectangles according to their area. In order to make this application run the Rectangle class should extends the Shape abstract class (as given below) and implement the java.lang.Comparable interface (more details can be found in the SDK 1.3 API). Declare the class Rectangle.

```
public class RectangleSort {
    public static void main(String args[]) {
        Rectangle[] vec;
        vec = new Rectangle[50];
        double randomWidth = 0, randomHeight = 0;
        for (int index = 0; index < vec.length; index++) {
            randomWidth = 1000 * Math.random();
            randomHeight = 1000 * Math.random();
            vec[index] = new Rectangle(randomWidth,
randomHeight);
        }

        Arrays.sort(vec);

        for (int index = 0; index < vec.length; index++) {
            System.out.println(vec[index]);
        }
    }
}
```