```
1  public abstract class Thing implements Comparable<Thing>
2  {
3     private String color;
4
5     public Thing(String color)
6     {
7        setColor(color);
8     }
9
10    public Thing()
11    {
12       this("");
13    }
14
15    public void setColor(String color)
16    {
17       this.color = color;
18    }
19
20    public String getColor()
21    {
22       return color;
23    }
24
25
26    public String toString()
27    {
28       return "I am a "+getColor()+" thing";
29    }
30
31    public int compareTo(Thing other)
32    {
33       if (this.getArea() == other.getArea())
34          return 0;
35       if (this.getArea() > other.getArea())
36          return 1;
37       return -1;
38    }
39
40 /* Example compareTo method using the compareTo of the String class
41    public int compareTo(Thing other)
42    {
```

```
43        return this.getColor().compareTo(other.getColor());
44    }
45 */
46
47    public abstract String tellAboutYourself();
48
49    public abstract double getArea();
50
51    public static void main(String[] args)
52    {}
53 }
```

```
1 public class RectangularThing extends Thing
2 {
3    private int length;
4    private int width;
5
6    public RectangularThing(String color, int length, int width)
7    {
8       super(color);
9       setSides(length, width);
10   }
11
12   public RectangularThing()
13   {
14      this("",2,1);
15   }
16
17   public void setSides(int length, int width)
18   {
19      if (length<=0)
20         throw new InvalidThingException("Length of rectangle is invalid, must be greater
   than 0: length = "+length);
21      if (width<=0)
22         throw new InvalidThingException("Width of rectangle is invalid, must be greater
   than 0: width = "+width);
23      if (length <= width)
24         throw new InvalidThingException("Values for length and width of rectangle are
   invalid, length must be greate than width: length = "+length+" and width = "+width);
25      this.width=width;
26      this.length=length;
27   }
28
29   public int getLength()
30   {
31      return length;
32   }
33
34   public int getWidth()
35   {
36      return width;
37   }
38
39   public double getArea()
```

```
40    {
41       return getLength() * getWidth();
42    }
43
44    public String toString()
45    {
46       return "I am a "+getColor()+" rectangular thing with length "+getLength()+", width
   "+getWidth()+" and area "+getArea();
47    }
48
49    public String tellAboutYourself()
50    {
51       return "I am a "+getColor()+" rectangular thing. I just lie around and do nothing";
52    }
53
54    public static void main(String[] args)
55    {
56       RectangularThing block = new RectangularThing("Green",20,10);
57       System.out.println(block);
58    }
59 }
```

```
1  public class RoundThing extends Thing
2  {
3     private int radius;
4
5     public RoundThing(String color, int radius)
6     {
7        super(color);
8        setRadius(radius);
9     }
10
11    public RoundThing()
12    {
13       this("",1);
14    }
15
16    public int getRadius()
17    {
18       return radius;
19    }
20
21    public void setRadius(int r)
22    {
23       if (r<=0)
24       {
25          throw new InvalidThingException("Invalid RoundThing: Radius must be greater than
   0: Radius = "+r);
26       }
27       else
28       {
29          radius=r;
30       }
31    }
32
33    public double getArea()
34    {
35       return Math.PI*Math.pow(getRadius(),2);
36    }
37
38    public String toString()
39    {
40       return "I am a "+getColor()+" round thing with radius "+getRadius()+" and area
   "+getArea();
```

```
41    }
42
43    public String tellAboutYourself()
44    {
45       return "I am a "+getColor()+" round thing with radius "+getRadius()+" and I can
   rol. WHEEE!";
46    }
47
48    public static void main(String[] args)
49    {
50       RoundThing rt1 = new RoundThing();
51       try
52       {
53           rt1 = new RoundThing("Blue",10);
54          System.out.println("It works!!!");
55       }
56       catch (InvalidThingException error)
57       {
58          System.out.println(error);
59
60       }
61       System.out.println("And my program continues!!!!");
62       System.out.println(rt1);
63
64      try
65       {
66           rt1 = new RoundThing("Red",-1);
67          System.out.println("It works???!!!");
68       }
69       catch (InvalidThingException error)
70       {
71          System.out.println(error);
72
73       }
74       System.out.println("And my program continues again???!!!!");
75       System.out.println(rt1);
76       System.out.println("\nASSIGNMENT: Explain what just happened!");
77
78
79    }
80 }
```

```
1 public class SquareThing extends Thing
2 {
3    private int length;
4
5    public SquareThing(String color, int length)
6    {
7       super(color);
8       setLength(length);
9    }
10
11   public SquareThing()
12   {
13      this("",1);
14   }
15
16   public void setLength(int length)
17   {
18      if (length<=0)
19         throw new InvalidThingException("Length of square is invalid, must be greater
   than 0: length = "+length);
20      this.length=length;
21   }
22
23   public double getArea()
24   {
25      return Math.pow(getLength(),2);
26   }
27
28   public String tellAboutYourself()
29   {
30      return "I am a square thing. I do not do cool...";
31   }
32
33
34   public int getLength()
35   {
36      return length;
37   }
38
39   public String toString()
40   {
41      return "I am a "+getColor()+" Square thing with sides "+getLength()+" units long
```

```
  and area "+getArea();
42    }
43
44    public static void main(String[] args)
45    {
46       SquareThing block = new SquareThing("Green",10);
47       System.out.println(block);
48    }
49 }
```

```
1 import java.util.*;
2
3 public class StuffTest
4 {
5     public static final int MAX_NO_OF_THINGS = 5;
6
7     public static void main(String[] args)
8     {
9         Thing[] stuff = new Thing[MAX_NO_OF_THINGS];
10
11        stuff[0] = new RoundThing("Blue",5);
12        stuff[1] = new SquareThing("Red",2);
13        stuff[2] = new RoundThing("Green",1);
14        stuff[3] = new RectangularThing("Purple",20,10);
15        stuff[4] = new SquareThing("Cyan",20);
16
17        for (int i = 0; i < MAX_NO_OF_THINGS; i++)
18            System.out.println("stuff["+i+"] = "+stuff[i]);
19
20        Arrays.sort(stuff);
21
22        System.out.println("After sorting");
23
24        for (int i = 0; i < MAX_NO_OF_THINGS; i++)
25            System.out.println("stuff["+i+"] = "+stuff[i]);
26
27
28    }
29
30 }
```