JAVA PROGRAMMING LANGUAGE
Assignment 10

1. Write an inheritance hierarchy for classes of simple shapes.

   Create a class *Shape*. Derive the four classes *ZeroDimensionalShape*, *OneDimensionalShape*, *TwoDimensionalShape*, and *ThreeDimensionalShape* from the *Shape* class. Derive the classes *Point*, *Line*, *Circle*, and *Sphere* from these four classes.

   Use abstract classes to define *Shape*, *OneDimensionalShape*, *TwoDimensionalShape*, and T*hreeDimensionalShape* classes, and then implement them in the *Line*, *Circle,* and *Sphere* classes.

   - Include a *shapeID* variable in the *Shape* class and a *getID*() method to get the ID for each shape created. Create a unique ID for each instance of a Shape. Include an abstract move() method to move the shape in the x, y, and z directions.

   - Derive the *Point* class from *ZeroDimensionalShape*. Give the point class an X, Y, and Z coordinate, and provide methods to get and set the coordinates.

   - Derive the *Line* class from *OneDimensionalShape*. One dimensional shapes have an abstract *getLength()* method. The line class is constructed using two points, example: `new Line(new Point(0,0,0),new Point(4,4,4))` creates a line from location 0,0,0 to location 4,4,4.

   - Derive the *Circle* class from *TwoDimensionalShape*. Two dimensional shapes have a abstract *getArea()* method. The circle class is constructed using a point for the center, and a radius, example: `new Circle(new Point(2,2,2),2)` creates a circle centered at 2,2,2 with a radius of 2. Assume the circle has the same Z value for all points in the circle.

   - Derive the *Sphere* class from *ThreeDimensionalShape*. Three dimensional shapes have both a abstract *getArea()* and abstract *getVolume()* methods. The sphere class is constructed using a point for the center, and a radius, example: `new Sphere(new Point(2,2,2), 2)` creates a sphere centered at 2,2,2 with a radius of 2.

   - Include a *toString()* that returns a String description of the key properties of each object, including its ID.

   - Use a *ShapeTester* program to test the *Shape* hierarchy. Illustrate the use of polymorphism in your tester. For example, first create an array of Shapes. Using loops, print the current location, move the shapes, and print the new location (See Sample Tester Code below).

```
public static void main(String[] args) {
   Shape[] s = new Shape[4];
   s[0] = new Point(1,1,1);
   s[1] = new Line(new Point(1,2,3), new Point(3,4,5));
   s[2] = new Circle(new Point(1,2,3), 2);
   s[3] = new Sphere(new Point(2,2,2), 3);
   System.out.println("Shapes:");
   for (int i = 0; i <  s.length; i++) {
      System.out.println(s[i]);
   }
   System.out.println("\nMove 2,2,2\nShapes:");
   for (int i = 0; i <  s.length; i++) {
      s[i].move(2,2,2);
      System.out.println(s[i]);
   }
   System.out.println("\nDimensions:");
   for (int i = 0; i <  s.length; i++) {
      if (s[i] instanceof OneDimensionalShape) {
         OneDimensionalShape ods = (OneDimensionalShape)s[i];
         System.out.printf("%s length is %f\n",ods,ods.getLength());
      }
      if (s[i] instanceof TwoDimensionalShape) {
         TwoDimensionalShape tds = (TwoDimensionalShape)s[i];
         System.out.printf("%s area is %f\n", tds, tds.getArea());
      }
      if (s[i] instanceof ThreeDimensionalShape) {
         ThreeDimensionalShape tds = (ThreeDimensionalShape)s[i];
         System.out.printf("%s area is %f\n", tds, tds.getArea());
         System.out.printf("%s volume is %f\n", tds, tds.getArea());
      }
   }
}
```

Use the *ShapeTester* program to test the *Shape* hierarchy.