411 Week 13 –Polymorphism Coding Example

\*Finish coding the TestArea class below where comments in **bold** so designate.

public abstract class Shape

{

protected String shapeName;

// abstract getArea method must be implemented by concrete subclasses

public abstract double getArea();

public String getName()

{

return shapeName;

}

} // end class Shape

|  |  |
| --- | --- |
| public class Square extends Shape {  private double side;  // constructor  public Square( double s )  {  side = ( s < 0 ? 0 : s );  shapeName = "Square";  }  // return the area of a Square  public double getArea()  {  return side \* side;  }  } // end class Square | public class Rectangle extends Shape {  private double length, width;  // constructor  public Rectangle( double s1, double s2 )  {  length = ( s1 < 0 ? 0 : s1 );  width = ( s2 < 0 ? 0 : s2 );  shapeName = "Rectangle";  }  // return the area of a Rectangle  public double getArea()  {  return length \* width;  }  } // end class Rectangle |

public class TestArea

{

public static void main( String args[] ) {

double side = 5.0 ; double length = 10.0; double width = 12.0;

Shape arrayOfShapes[] = new Shape[ 2 ];

**/\*fill in the assignments below for each index of the array of shapes to instantiate**

**a Square object and a Rectangle Object \*/**

arrayOfShapes[ 0 ] =

arrayOfShapes[ 1 ] =

**/\* Create a for – each loop to iterate over each arrayOfShapes object \*/**

for ( )

**/\*Within for loop, display each shape by name along with its corresponding area\*/**

System.out.printf( );

} // end main

} // end class TestArea