**package** topic3\_2;

**public** **class** implemants {

}

**package** topic3\_2;

**public** **interface** ShapeInterface

{

/\*\*

\*

\* **@return**

\*/

**int** calculateArea();

**double** calculateArea1();

}

**package** topic3\_2;

**public** **class** ShapeBase **implements** ShapeInterface {

**protected** String name;

**protected** **int** width, height;

**private** **double** area; //Shape of the Square

**private** **double** sideLength; //Shape of the Square

**private** String name1; //Shape of the Square

/\*\*

\* This is name of shapes Rectangle and Triangle

\* **@param** name

\* **@param** width

\* **@param** height

\*/

**public** ShapeBase(String name, **int** width, **int** height)

{

**this**.name = name;

**this**.width = width;

**this**.height = height;

}

/\*\*

\*

\* **@return** Name

\*/

**public** String getName()

{

**return** **this**.name;

}

@Override

**public** **int** calculateArea() {

**return** -1;

}

/\*\*

\* This is the output of the shape Square

\* **@param** name

\* **@param** area

\* **@param** sideLength

\*/

**public** ShapeBase(String name, **double** area, **double** sideLength)

{

**this**.name = name;

**this**.area = area;

**this**.sideLength = sideLength;

}

**public** String getName1()

{

**return** **this**.name;

}

@Override

**public** **double** calculateArea1() {

// **TODO** Auto-generated method stub

**return** area ;

}

}

**package** topic3\_2;

**public** **class** Rectangle **extends** ShapeBase {

/\*\*

\* This is the Rectangle shape

\* **@param** name

\* **@param** width

\* **@param** height

\*/

**public** Rectangle(String name, **int** width, **int** height)

{

**super**(name, width, height);

}

@Override

**public** **int** calculateArea()

{

**return** width \* height;

}

}

**package** topic3\_2;

**public** **class** Triangle **extends** ShapeBase {

/\*\*

\* This is the shape of the Triangle

\* **@param** name

\* **@param** width

\* **@param** height

\*/

**public** Triangle (String name, **int** width, **int** height)

{

**super**(name, width, height);

}

@Override

**public** **int** calculateArea()

{

**return** width \* height/2;

}

}

**package** topic3\_2;

**public** **class** Square **extends** ShapeBase{

**protected** String name1;

**public** **double** area;

**public** **double** sideLength = 4.9;

/\*\*

\* This is the shape of the Square

\* **@param** name

\* **@param** area

\* **@param** sideLength

\*/

**public** Square(String name, **double** area, **double** sideLength)

{

**super**(name, area, sideLength);

}

@Override

**public** **double** calculateArea1()

{

**return** area = sideLength \* sideLength;

}

}

**package** topic3\_2;

**public** **class** Test {

/\*\*

\*

\* **@param** shape

\*/

**private** **static** **void** displayArea(ShapeBase shape)

{

System.***out***.println("This is a shape named " + shape.getName() + " with an area of " + shape.calculateArea());

System.***out***.println("This is a shape named " + shape.getName() + " with an area of " + shape.calculateArea1());

}

/\*\*

\*

\* **@param** args

\*/

**public** **static** **void** main(String[] args) {

// Create an array of Base Shapes and initialize to specific Shapes

ShapeBase[] shapes = **new** ShapeBase[3];

shapes[0] = **new** Rectangle("Rectangle" , 10 , 200);

shapes[1] = **new** Triangle("Triangle" , 10 , 50);

shapes[2] = **new** Square("Square", 10, 16);

// For all Shapes display its area

**for**(**int** x = 0; x < shapes.length; ++x)

{

*displayArea*(shapes[x]);

}

}

}

/\*\*

\* Notes

\* Polymorphism is demonstrated through the usage of the ShapeBase class and its subclasses (Rectangle, Triangle, and Square), as well as the ShapeInterface.

\*

\* Polymorphism is evident in my ShapeInterface interface, which declares two methods: calculateArea() and calculateArea1()

\*

\* polymorphism promotes code reuse, flexibility, and extensibility, leading to more modular, maintainable, and scalable software systems.

\* It enables the creation of generic and adaptable code that can work with various types of objects, making object-oriented programming more powerful and expressive.

\*

\*/