LAB #6: POLYMORPHISM

1. Using an inheritance hierarchy, design a Java program to model 3-dimensional shapes (square pyramid, sphere, rectangular prism, cube, cylinder, circular cone). Have a top level shape interface with methods for getting the area and the volume (+ methods `toString` and `equals`). Next, build classes and subclasses for the above 3-dimensional shapes. Make sure that you place common behavior in superclasses whenever possible. Also, use abstract classes as appropriate. Add methods to subclasses to represent unique behavior particular to each 3-dimensional shape.
Write the definitions of these classes and do the testing with the client program provided.

SOLUTION:

```
// Interface Shape3D: for three-dimensional shapes.
public interface Shape3D {
    public double getArea();
    public double getVolume();
    public String toString();
    public boolean equals(Object obj);
}

// Class SquarePyramid. Implements Shape3D
// Represents a pyramid with a square as its base.
public class SquarePyramid implements Shape3D {
    private double length;
    private double height;

    public SquarePyramid() {
        length = 0;
        height = 0;
    }
    public SquarePyramid(double l, double h) {
        ...
    }

    public double getLength() {
        ...
    }

    public double getHeight() {
       ...
    }

    public double getArea() {
       return length * (length + Math.sqrt(length * length + 4 * height * height));
    }

    public double getVolume() {
       return length * length * height / 3.0;
    }

    public String toString() {
       ...
    }

    public boolean equals(Object obj) {
       ...
    }
}

// Class Sphere. Implements Shape3D
// Represents a perfect sphere.

public class Sphere implements Shape3D {
    private double radius;

    public Sphere() {
       ...
    }

    public Sphere(double r) {
       ...
    }

    public double getRadius() {
       ...
    }

    public double getArea() {
       return 4 * Math.PI * Math.pow(radius, 2);
    }


    public double getVolume() {
       return 4.0 * Math.PI * Math.pow(radius, 3) / 3.0;
    }

    public String toString() {
       ...
    }

    public boolean equals(Object obj) {
       ...
    }
}

// Class RectangularPrism. Implements Shape3D
// Represents a three-dimensional rectangular shape.

public class RectangularPrism implements Shape3D {
    private double length;
    private double width;
    private double height;

    public RectangularPrism() {
       ...
    }

    public RectangularPrism(double l, double w, double h) {
       ...
    }

    public double getLength() {
       ...
    }

    public double getWidth() {
       ...
    }

    public double getHeight() {
       ...
    }

    public double getArea() {
       return 2 * (length * width + width * height + length * height);
    }

    public double getVolume() {
       return length * width * height;
    }

    public String toString() {
       ...
    }

    public boolean equals(Object obj) {
       ...
    }
}

// Class Cube, subclass of RectangularPrism
// Represents a perfect cube.

public class Cube extends RectangularPrism {
    public Cube() {
       ...
    }

    public Cube(double size) {
       ...
    }

    public String toString() {
       ...
    }
}

// Class CircularShape. Implements Shape3D.
// ABSTRACT CLASS --> no objects of this type!
// An abstract superclass for shapes with a circular cross-section.

public abstract class CircularShape implements Shape3D {
    private double radius;

    public CircularShape() {
       ...
    }

    public CircularShape(double r) {
       ...
    }

    public double getDiameter() {
       ...
    }

    public double getRadius() {
       ...
    }

    public double getCrossSectionArea() {
       return Math.PI * Math.pow(radius, 2);
    }

    public double getCrossSectionPerimeter() {
       return 2 * Math.PI * radius;
    }
}

// Class CircularShapeWithHeight. Subclass of CircularShape
// ABSTRACT CLASS --> no objects of this type!
// An abstract superclass for shapes with a circular cross-section that extends over
some height.

public abstract class CircularShapeWithHeight extends CircularShape {
    private double height;

    public CircularShapeWithHeight() {
       ...
    }
    public CircularShapeWithHeight(double radius, double height) {
       ...
    }

    public double getHeight() {
       ...
    }
}

// Class Cylinder, subclass of CircularShapeWithHeight
// Represents a cylinder shape.
```

```
public class Cylinder extends CircularShapeWithHeight {
    public Cylinder() {
       ...
    }

    public Cylinder(double radius, double height) {
       ...
    }

    public double getArea() {
       return getCrossSectionPerimeter() * getHeight() + 2 * getCrossSectionArea();
    }

    public double getVolume() {
       return getCrossSectionArea() * getHeight();
    }

    public String toString() {
       ...
    }

    public boolean equals(Object obj) {
       ...
    }
}

// Class CircularCone, subclass of CircularShapeWithHeight
// Represents cones with a circular base.

public class CircularCone extends CircularShapeWithHeight {
    public CircularCone() {
       ...
    }

    public CircularCone(double radius, double height) {
       ...
    }

    public double getArea() {
       double r = getRadius();
       double h = getHeight();
       return Math.PI * r * Math.sqrt(r * r + h * h);
    }

    public double getVolume() {
       return getCrossSectionArea() * getHeight() / 3.0;
    }

    public String toString() {
       ...
    }

    public boolean equals(Object obj) {
        ...
    }
}

//USE this client to test them all! Analyze the client.
public class Shape3D_Client {
    public static final int MAX = 6;
    public static void main(String[] args) {
        Shape3D[] shapes = new Shape3D[MAX];
        shapes[0] = new SquarePyramid(37, 20);
        shapes[1] = new Sphere(20);
        shapes[2] = new RectangularPrism(10, 20, 37);
        shapes[3] = new Cube(10);
        shapes[4] = new Cylinder(10, 20);
        shapes[5] = new CircularCone(10, 20);

        for (int i = 0; i < shapes.length; i++) {
            System.out.print("\nThis is a ");
            switch(i) {
                case 0:
                    System.out.print("square pyramid. ");
                    break;
                case 1:
                    System.out.print("sphere. ");
                    break;
                case 2:
                    System.out.print("rectangular prism. ");
                    break;
                case 3:
                    System.out.print("cube. ");
                    break;
                case 4:
                    System.out.print("cylinder. ");
                    break;
                case 5:
                    System.out.print("circular cone. ");
            }
            System.out.printf("Area = %.2f", shapes[i].getArea());
            System.out.printf(". Volume = %.2f\n", shapes[i].getVolume());
            System.out.println("Output calling the method printInfo - polymorphism
at work!");
            printInfo(shapes[i]);
            System.out.println("----------------------------------------------------
-------");
        }
    }

    public static void printInfo(Shape3D s) {
        System.out.println(s);
        System.out.printf("Area = %.2f", s.getArea());
        System.out.printf(". Volume = %.2f\n", s.getVolume());
    }
}
```

EXPECTED OUTPUT:

```
This is a square pyramid. Area = 3385.08. Volume = 9126.67
Output calling the method printInfo - polymorphism at work!
For this square pyramid the base has the length = 37.0 and the height = 20.0
Area = 3385.08. Volume = 9126.67
-----------------------------------------------------------

This is a sphere. Area = 5026.55. Volume = 33510.32
Output calling the method printInfo - polymorphism at work!
The radius of this sphere = 20.0
Area = 5026.55. Volume = 33510.32
-----------------------------------------------------------

This is a rectangular prism. Area = 2620.00. Volume = 7400.00
Output calling the method printInfo - polymorphism at work!
For this rectangular prism the base has the length = 10.0 and the width = 20.0
The height of the prism = 37.0
Area = 2620.00. Volume = 7400.00
-----------------------------------------------------------

This is a cube. Area = 600.00. Volume = 1000.00
Output calling the method printInfo - polymorphism at work!
For this cube all sides = 10.0
Area = 600.00. Volume = 1000.00
-----------------------------------------------------------

This is a cylinder. Area = 1884.96. Volume = 6283.19
Output calling the method printInfo - polymorphism at work!
For this cylinder the radius = 10.0 and the height = 20.0
Area = 1884.96. Volume = 6283.19
-----------------------------------------------------------

This is a circular cone. Area = 702.48. Volume = 2094.40
Output calling the method printInfo - polymorphism at work!
For this circular cone the radius = 10.0 and the height = 20.0
Area = 702.48. Volume = 2094.40
-----------------------------------------------------------
```