Maddison Kiefer

Dr. Schwartz

Advanced Java Programming

11/13/2023

**Project 4-1 Creating an Abstract Class**

**UML Diagram:**



**Source Code:**

// @author Maddison Kiefer

import java.util.Locale;

// Abstract base class for the geometric shapes

abstract class Shape {

// Abstract methods to be implemented by subclasses

public abstract double surface\_area();

public abstract double volume();

}

// Subclass representing a sphere

class Sphere extends Shape {

// Radius of the sphere

private double radius;

// Constructor to initialize the sphere with a given radius

public Sphere(double radius) {

this.radius = radius;

}

// Override toString method to provide a formatted string

@Override

public String toString() {

// Format and return a string with surface area and volume

return String.format(Locale.US, "Surface area of sphere: %.3f Volume of sphere: %.3f",surface\_area(), volume());

}

// Calculate and return the surface area of the sphere

@Override

public double surface\_area() {

return 4.0 \* Math.PI \* radius \* radius;

}

// Calculate and return the volume of the sphere

@Override

public double volume() {

return 4.0 \* Math.PI \* radius \* radius \* radius / 3.0;

}

}

// Subclass representing a cylinder

class Cylinder extends Shape {

// Radius of the base and height of the cylinder

private double radius;

private double height;

// Constructor to initialize the cylinder with given radius and height

public Cylinder(double radius, double height) {

this.radius = radius;

this.height = height;

}

// Override toString method to provide a formatted string

@Override

public String toString() {

// Format and return a string with surface area and volume

return String.format(Locale.US, "Surface area of cylinder: %.3f Volume of cylinder: %.3f", surface\_area(), volume());

}

// Calculate and return the surface area of the cylinder

@Override

public double surface\_area() {

return 2.0 \* Math.PI \* radius \* (radius + height);

}

// Calculate and return the volume of the cylinder

@Override

public double volume() {

return Math.PI \* radius \* radius \* height;

}

}

// Subclass representing a cone

class Cone extends Shape {

// Radius of the base and height of the cone

private double radius;

private double height;

// Constructor to initialize the cone with given radius and height

public Cone(double radius, double height) {

this.radius = radius;

this.height = height;

}

// Override toString method to provide a formatted string

@Override

public String toString() {

// Format and return a string with surface area and volume

return String.format(Locale.US, "Surface area of cone: %.3f Volume of cone: %.3f", surface\_area(), volume());

}

// Calculate and return the surface area of the cone

@Override

public double surface\_area() {

return Math.PI \* radius \* (radius + Math.sqrt(height \* height + radius \* radius));

}

// Calculate and return the volume of the cone

@Override

public double volume() {

return Math.PI \* radius \* radius \* height / 3.0;

}

}

// Class with the main method to demonstrate the use of the shapes

class ShapeArray {

public static void main (String[] args) {

// Array to hold different shapes

Shape[] shapeArray = new Shape[3];

// Initialize array with instances of Sphere, Cylinder, and Cone

shapeArray[0] = new Sphere(3.5);

shapeArray[1] = new Cylinder(3.0, 5.5);

shapeArray[2] = new Cone(4.5, 6.5);

// Iterate through the array and print information about each shape

for(int i = 0; i < 3; i++) {

System.out.println(shapeArray[i].toString());

}

}

}

**Executing the Application:**