**ACTIVIDAD 1**

**Pregunta 1:**

public class Círculo {

private double radio = 12.5;

private String color = "azul";

public Círculo() {}

public Círculo(double radio) {

this.radio = radio;

}

public double getRadio() {

return radio;

}

public double getArea() {

return radio \* radio \* Math.PI;

}

public void setRadio(double radio) {

this.radio = radio;

}

public void setColor(String color) {

this.color = color;

}

@Override

public String toString() {

return "Círculo [radio=" + radio + ", color=" + color + ", área=" + getArea() + "]";

}

}

**Pregunta 2:**

No, porque no tiene un método main()

**Pregunta 3:**

public class CírculoTest {

public static void main(String[] args) {

Círculo c1 = new Círculo();

System.out.println("Radio (default): " + c1.getRadio());

System.out.println("Área (default): " + c1.getArea());

Círculo c2 = new Círculo(6.55);

System.out.println("Radio (sobrecargado): " + c2.getRadio());

System.out.println("Área (sobrecargado): " + c2.getArea());

}

}

**Pregunta 4:**

**a) ¿Qué es el constructor y para qué sirve?**

Es un método especial que se ejecuta al crear un objeto e inicializa los atributos

**b) ¿Con qué método recupero el radio del círculo?**

Con getRadio()

**c) ¿Por qué los atributos son privados y los métodos públicos?**

Para el encapsulamiento donde los atributos se protegen y solo se modifican mediante métodos controlados (setters/getters)

**d) ¿Qué pasa si accedes directamente a los atributos desde main()?**

Hay error de compilación

**e) ¿Qué pasa si modificas directamente el atributo área desde main()?**

No se puede porque área no es un atributo almacenado sino calculado en getArea()

**f) En la clase “CirculoTest.java”, ejecuta la siguiente instrucción: System.out.println(c1.radio), donde c1 es una instancia del objeto Círculo. ¿Qué ocurrió? Explique el mensaje de error.**Hay error porque radio es privado

**g) En la clase “CirculoTest.java”, ejecuta la siguiente instrucción: c1.radio=42.0, donde c1 es una instancia del objeto Círculo. ¿Qué ocurrió? Explique el mensaje de error.**

Tiene error porque el acceso es privado

**h) ¿Con qué método modificas el valor de un atributo de la clase “Circulo.java”?**

setRadio(double radio)

**i) Crear un objeto de tipo Círculo y modifica los valores por defecto.**

Círculo c3 = new Círculo();

c3.setRadio(10.0);

c3.setColor("rojo");

**j) ¿Para qué sirve la palabra reservada “this”? Modifique el constructor con**

**argumentos (radio y color) y utiliza “this” y métodos setters en la clase**

**“Circulo.java”.**

public Círculo(double radio, String color) {

this.radio = radio;

this.color = color;

}

**k) ¿Para qué sirve el método toString()?**

Devuelve una representación en String del objeto

**Interfaz de usuario gráfica, Texto, Tabla, Excel

El contenido generado por IA puede ser incorrecto.Pregunta 5:**

**ACTIVIDAD 2**

**Pregunta 1:**

public class Punto {

protected double x, y;

public Punto() {}

public Punto(double x, double y) {

this.x = x;

this.y = y;

}

public double getX() { return x; }

public void setX(double x) { this.x = x; }

public double getY() { return y; }

public void setY(double y) { this.y = y; }

@Override

public String toString() {

return "El punto tiene las siguientes coordenadas: " + x + "/" + y;

}

public double calcularDistancia() {

return calcularDistancia(0, 0);

}

public double calcularDistancia(Punto otro) {

return Math.sqrt(Math.pow(this.x - otro.x, 2) + Math.pow(this.y - otro.y, 2));

}

}

**Pregunta 2:**

public class Triángulo extends FiguraGeométrica {

private Punto punto1, punto2, punto3;

public Triángulo() {}

public Triángulo(String nombre, Punto p1, Punto p2, Punto p3) {

super(nombre);

this.punto1 = p1;

this.punto2 = p2;

this.punto3 = p3;

}

public Punto getPunto1() { return punto1; }

public void setPunto1(Punto p1) { this.punto1 = p1; }

public Punto getPunto2() { return punto2; }

public void setPunto2(Punto p2) { this.punto2 = p2; }

public Punto getPunto3() { return punto3; }

public void setPunto3(Punto p3) { this.punto3 = p3; }

@Override

public double calcularArea() {

// Fórmula de Herón o otro método

return 0.0; // Implementar lógica real

}

@Override

public boolean esRegular() {

// Comparar lados/ángulos

return false; // Implementar lógica real

}

@Override

public String toString() {

return "Triángulo: " + getNombre() + " tiene 3 Puntos: " +

punto1.toString() + "/" + punto2.toString() + "/" + punto3.toString();

}

}

**Pregunta 3:**

public abstract class FiguraGeométrica {

protected String nombre;

public FiguraGeométrica(String nombre) {

this.nombre = nombre;

}

public String getNombre() { return nombre; }

public void setNombre(String nombre) { this.nombre = nombre; }

public abstract double calcularArea();

public abstract boolean esRegular();

}

**Pregunta 4:**

**Identificar las clases que heredan, ¿qué atributos y métodos heredan?**

Triángulo hereda de FiguraGeométrica

Atributos: nombre (protected)

Métodos: Debe implementar calcularArea() y esRegular()

**Pregunta 5:**

**Identificar las clases que conforman una composición.**

Triángulo tiene 3 Puntos

**Pregunta 6:**

**¿Qué es una superclase y una subclase?**

La superclase es Clase padre (FiguraGeométrica) y la subclase es Clase hija (Triángulo).

**Pregunta 7:**

**¿Por qué usamos abstract? ¿Se puede dejar de heredar un método de una clase abstracta?**

Para obligar a las subclases a implementar ciertos métodos, no deben implementarse obligatoriamente

**Pregunta 8:**

**¿Qué anotación utilizo para sobrescribir métodos?**

@Override

**Pregunta 9:**

**Los atributos de la clase Figura Geométrica conviértalas en protected. ¿En qué condición convierte a los atributos? ¿Es posible acceder a los atributos protegidos sin utilizar una invocación a super() o sin método get?.**

Son accesibles desde la misma clase, subclases y paquete; se puede acceder sin super o get directamente en subclases.

**Pregunta 10:**

**¿Cómo aplicarías polimorfismo?**

Lo aplicaría en la clase abstracta FiguraGeométrica

Como FiguraGeométrica tiene métodos abstractos (calcularArea(), esRegular()), las subclases deben implementarlos.

FiguraGeométrica figura1 = new Triángulo("Triángulo 1", p1, p2, p3);

FiguraGeométrica figura2 = new Círculo(5.0);

// Suponiendo que Círculo hereda de FiguraGeométrica