|  |
| --- |
| **CURSO: Análise e Desenvolvimento de Sistemas** |
| COMPONENTE CURRICULAR/TEMA: PROG. DE SISTEMAS II |
| **INTEGRANTES: Allan Pradella Frushio** |

**Implementação da classe Circulo:**

package TestaCirculo; // Declaração do pacote TestaCirculo

import TestaPonto.Ponto; // Importação da classe Ponto do pacote TestaPonto

// Classe Circulo que estende FiguraGeometrica e implementa o Calcula.jav

class Circulo extends FiguraGeometrica implements Calcula { // Declaração da classe Circulo que estende FiguraGeometrica e implementa Calcula

// Atributos da classe Circulo

    private Ponto centro; // Declaração do atributo centro do tipo Ponto

    private double raio; // Declaração do atributo raio do tipo double

// Construtor da classe Circulo

    public Circulo(int x, int y, double raio, String cor) { // Declaração do construtor Circulo com parâmetros x, y, raio e cor – como na aula.

        super(cor); // Chamada do construtor da classe FiguraGeometrica

        this.centro = new Ponto(x, y); // Cria um novo objeto Ponto para representar o centro

        this.raio = raio; // Atributo raio

    }

// Override do método compare da classe FiguraGeometrica

    @Override

    public boolean compare(FiguraGeometrica figura) { // Override do método compare da classe FiguraGeometrica

        if (figura instanceof Circulo) { // Verifica se a figura é um círculo, mas é desnec

            Circulo outroCirculo = (Circulo) figura; // Referência da figura para Circulo

// Verifica se os atributos raio, cor e centro são iguais

            return this.raio == outroCirculo.raio && this.cor.equals(outroCirculo.cor)

                    && this.centro.equals(outroCirculo.centro);

        }

        return false; // Retorna false se a figura não for um circulo

    }

// Implementação do método calculaArea da interface Calcula

    @Override

    public double calculaArea() { // Implementação do método calculaArea

        return Math.PI \* Math.pow(raio, 2); // Retorna a área do círculo usando a fórmula pi \* raio^2

    }

// Implementação do método calculaPerimetro da interface Calcula

    @Override

    public double calculaPerimetro() { // Implementação do método calculaPerimetro

        return 2 \* Math.PI \* raio; // Retorna o perímetro do círculo usando a fórmula 2 \* pi \* raio

    }

// Método para verificar interseção dos círculos

    public boolean interseccao(Circulo c) { // Método para verificar interseção deles

        double distanciaCentros = this.centro.distancia(c.centro); // Calcula a distância do ½ raio

        return distanciaCentros < (this.raio + c.raio); // Verifica se a distância entre os centros é menor que a soma dos raios

    }

}