Trabalho Prático 3 de PM

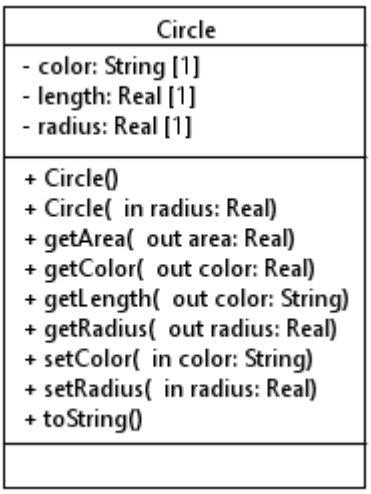
Classes e Objetos

Douglas Rodrigues de Almeida
douglasralmeida@live.com

Introdução

O objetivo desse trabalho foi demonstrar os recursos da programação orientada a objetos, projetando e implementando classes e instanciando objetos.

Na primeira parte do trabalho foi implementada a classe Circle que representa uma circunferência, conforme o diagrama de classe abaixo:



Resultados do desafio 1

Programa de Teste:

```
O circulo tem o raio de 1.0 e area de 3.141592653589793
O circulo tem raio de 2.0 e area de 12.566370614359172
```

Avaliação: Instâncias diferentes de uma classe possuem valores diferentes para seus atributos.

Teste 1:

```
O circulo tem o raio de 1.0 e area de 3.141592653589793
O circulo tem raio de 2.0 e area de 12.566370614359172
O circulo tem raio de 5.0 e area de 78.53981633974483
```

Teste 2:

```
O circulo tem o raio de 1.0 e area de 3.141592653589793
O circulo tem raio de 2.0 e area de 12.566370614359172
O circulo tem raio de 5.0 e area de 78.53981633974483 e cor green
```

Teste 3:
`Mensagem recebida:` The field Circle.radius is not visible.
`Avaliação:` O atributo radius é privado. Esse encapsulamento isola o atributo não permitindo seu acesso externo.

Teste 4:

```
O circulo tem o raio de 1.0 e area de 3.141592653589793
O circulo tem raio de 2.0 e area de 12.566370614359172
O circulo tem raio de 4.0 e area de 50.26548245743669 e cor blue
```

Teste a:

```
Circulo: raio = 5.0 cor = red
```

Teste b:

```
Circulo: raio = 5.0 cor = red
Circulo: raio = 1.2 cor = red
Circulo: raio = 1.2 cor = red
Operator '+' invokes toString() too: Circulo: raio = 1.2 cor = red
```

Código fonte do Desafio 1

Arquivo circle.java:

```java
public class Circle {
  private double radius;
  private String color;
  private double length;

  public Circle() {
    this.radius = 1.0;
    this.color = "red";
    this.length = this.radius * 2 * Math.PI;
  }

  public Circle(double radius) {
    this();
    this.radius = radius;
  }

  public Circle (double radius, String color) {
    this(radius);
    this.color = color;
  }

  public String getColor() {
    return this.color;
  }

  public double getLength() {
    return this.length;
  }

  public double getRadius() {
    return this.radius;
  }
```

```
  public double getArea() {
    return this.radius*this.radius*Math.PI;
  }

  public void setRadius(double radius) {
    this.radius = radius;
    this.length = radius * 2 * Math.PI;
  }

  public void setColor(String color) {
    this.color = color;
  }

  public String toString() {
    return "Circulo: raio = " + this.radius + " cor = " + this.color;
  }
}
```

Arquivo testcircle.java:

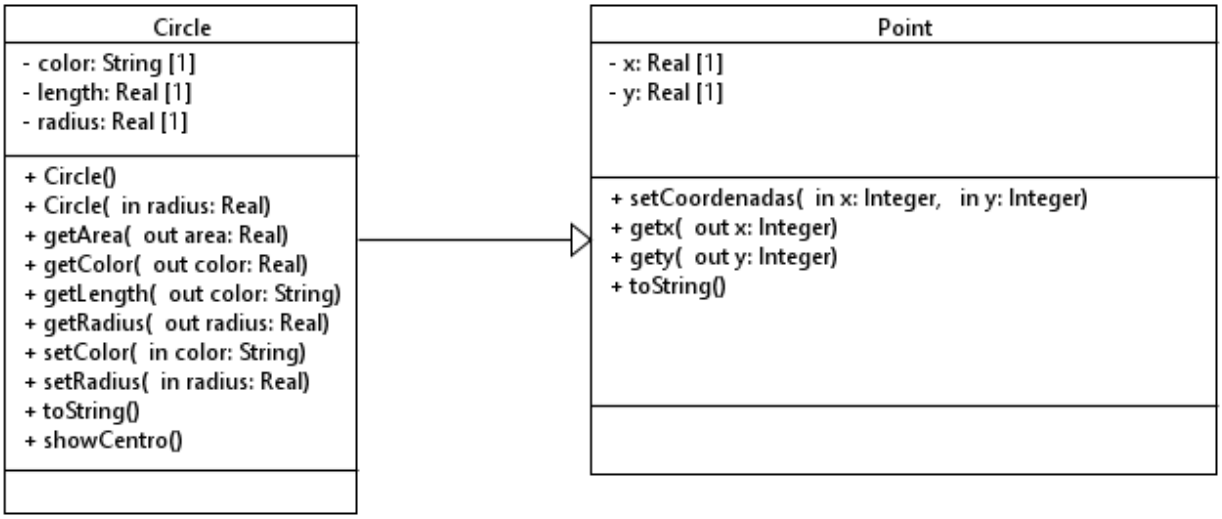
```
public class TestCircle {

  public static void main(String[] args) {
    Circle c1;
    c1 = new Circle(5.0);
    System.out.println(c1.toString());

    Circle c2 = new Circle(1.2);
    System.out.println(c2.toString());
    System.out.println(c2);
    System.out.println("Operator '+' invokes toString() too: " + c2);
  }
}
```

Diagrama de Classe do Desafio 2

No desafio 2 foi criada a classe Point que representa uma coordenada com a classe Circle herdando seus métodos e atributos, conforme diagrama abaixo.



Teste: Resultado do teste realizado na classe Circle:

```
Circulo: raio=2.0 cor=red comprimento=6.283185307179586 centro=(Coordenada:
x=1.0 y=3.0)
```

Código fonte do Desafio 2

Arquivo point.java:

```
public class Point {
  private double x;
  private double y;

  public void setCoordenadas(int x, int y) {
    this.x = x;
    this.y = y;
  }

  public int getx() {
    return (int) x;
  }

  public int gety() {
    return (int) y;
  }

  public String toString() {
    return "Coordenada: x=" + this.x + " y=" + this.y;
  }
}
```

Arquivo circle.java:

```
public class Circle extends Point{
  private double radius;
  private String color;
  private double length;

  public Circle() {
    super();
    super.setCoordenadas(0, 0);
    this.radius = 1.0;
    this.color = "red";
    this.length = this.radius * 2 * Math.PI;
  }

  public Circle(double radius) {
    this();
    this.radius = radius;
  }

  public Circle (double radius, String color) {
    this(radius);
    this.color = color;
  }

  public String getColor() {
    return this.color;
  }

  public double getLength() {
    return this.length;
  }

  public double getRadius() {
    return this.radius;
  }

  public double getArea() {
    return this.radius*this.radius*Math.PI;
  }

  public void setRadius(double radius) {
    this.radius = radius;
    this.length = radius * 2 * Math.PI;
  }

  public void setColor(String color) {
    this.color = color;
  }

  public void showCentro() {
```

```
    System.out.println("x=" + getx() + " y=" + gety());
  }

  public String toString() {
    return "Circulo: raio=" + this.radius + " cor=" + this.color + " comprimento="
+ this.length + " centro=(" + super.toString() + ")";
  }
}
```

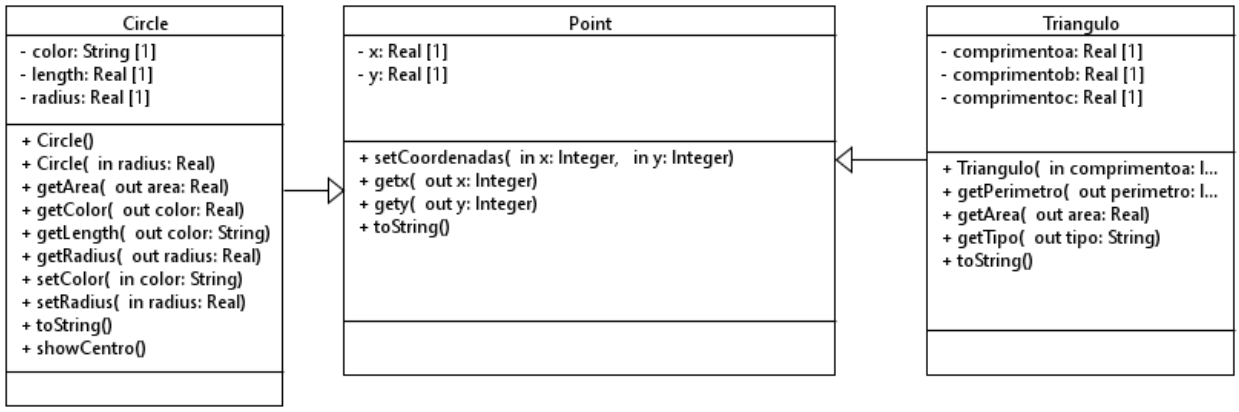
Arquivo testcircle.java:

```
public class TestCircle {

  public static void main(String[] args) {
    Circle c1;
    c1 = new Circle(2.0);
    c1.setCoordenadas(1, 3);
    System.out.println(c1);
  }
}
```

Diagrama de Classe do Desafio 3

No desafio 3 foi implementado uma classe Triangulo para representar triângulos no plano, conforme diagrama de classe abaixo.



Teste: Resultado do teste realizado na classe Triangulo:

```
Triangulo: lados=3.0;4.0;5.0 area=6.0 perimetro=12 tipo= triangulo escaleno
centro=(Coordenada: x=0.0 y=0.0)
Triangulo: lados=3.0;4.0;3.0 area=4.47213595499958 perimetro=10 tipo= triangulo
isosceles centro=(Coordenada: x=0.0 y=0.0)
Triangulo: lados=3.0;3.0;3.0 area=2.0 perimetro=9 tipo= triangulo equilatero
centro=(Coordenada: x=0.0 y=0.0)
java.lang.Exception: Informado lado com comprimento negativo ou igual a zero.
      at Triangulo.<init>(Triangulo.java:9)
      at TestCircle.main(Test.java:16)
```

A exceção foi gerada porque foi informado um triangulo com lado igual a 0.

Código fonte do Desafio 3

Arquivo point.java:

```
public class Point {
  private double x;
  private double y;

  public void setCoordenadas(int x, int y) {
    this.x = x;
    this.y = y;
  }

  public int getx() {
    return (int) x;
  }

  public int gety() {
    return (int) y;
  }

  public String toString() {
    return "Coordenada: x=" + this.x + " y=" + this.y;
  }
}
```

Arquivo triangulo.java:

```
public class Triangulo extends Point{
  private double comprimentoa;
  private double comprimentob;
  private double comprimentoc;
  public Triangulo(int comprimentoa, int comprimentob, int comprimentoc) throws
Exception {
    super();

    if (comprimentoa <= 0 || comprimentob <= 0 || comprimentoc <= 0)
      throw new Exception("Informado lado com comprimento negativo ou igual a
zero.");
    if (Math.abs(comprimentob - comprimentoc) < comprimentoa && comprimentob +
comprimentoc > comprimentoa) {
      this.comprimentoa = comprimentoa;
      this.comprimentob = comprimentob;
      this.comprimentoc = comprimentoc;
    } else
      throw new Exception("Triangulo invalido.");
  }
```

```
  public int getPerimetro() {
    return (int) (this.comprimentoa + this.comprimentob + this.comprimentoc);
  }

  public double getArea() {
    double sp = getPerimetro()/2;

    return Math.sqrt(sp*(sp-comprimentoa)*(sp-comprimentob)*(sp-comprimentoc));
  }

  public String getTipo() {
    if (this.comprimentoa == this.comprimentob && this.comprimentob ==
this.comprimentoc)
      return "triangulo equilatero";
    else if (this.comprimentoa == this.comprimentob || this.comprimentob ==
this.comprimentoc || this.comprimentoa == this.comprimentoc)
      return "triangulo isosceles";
    else return "triangulo escaleno";
  }

  public String toString() {
    return "Triangulo: lados="+comprimentoa+";"+comprimentob+";"+comprimentoc+"
area="+getArea()+" perimetro="+getPerimetro()+" tipo= "+getTipo() + " centro=(" +
super.toString() + ")";
  }
}
```

Arquivo test.java:

```
public class Test {

  public static void main(String[] args) {
    Triangulo t1, t2, t3, t4;

    try {
      t1 = new Triangulo(3, 4, 5);
      System.out.println(t1);

      t2 = new Triangulo(3, 4, 3);
      System.out.println(t2);

      t3 = new Triangulo(3, 3, 3);
      System.out.println(t3);

      t4 = new Triangulo(0, 3, 4);

    } catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```