Programação Orientada a Objetos *

Métodos

PROFA CRISTINA VERÇOSA PÉREZ BARRIOS DE SOUZA

PUCPR

* ADAPTADO DO MATERIAL DO PROF. ALCIDES CALSAVARA

Verificação de exemplos de classes com atributos de tipos básicos e métodos com parâmetros codificados por meio de variáveis locais, comandos imperativos e uso de referência a objetos para invocação de métodos.

AULA 3

Conceitos

1. Objeto (instância de uma classe)
2. Instanciação (criação de um objeto)
3. Método construtor
4. Membros (atributos e métodos) de objetos
5. Estado de um objeto
6. Referência a objeto
7. Chamada de método de um objeto
8. Parâmetro (e retorno) cujo tipo é uma classe

Exemplo: Classes Circulo, Arquiteto e Escritorio

1. Crie um projeto contendo as três classes no pacote default.
2. Execute o projeto a partir da classe Escritorio.

Classe Circulo

1. Observe que o atributo **raio** e os métodos da classe Circulo não são qualificados como static. Qual é o efeito disso?
2. Por que o atributo PI da classe Circulo é static?
3. Observe que a classe Circulo possui um método com o mesmo nome da classe. Esse é o **método construtor** da classe.

```
class Circulo {

    private static double PI = 3.141516;
    private double raio;

    public Circulo(double r) {
        raio = r;
    }

    public double area( ) {
        return PI * raio * raio;
    }

    public double perimetro( ) {
        return 2 * PI * raio;
    }
}
```

Classe Arquiteto

1. Observe que nenhum atributo e nenhum método da classe Arquiteto é qualificado como static. Qual é o efeito disso?
2. Observe o método construtor da classe Arquiteto. Por que foi necessário utilizar o símbolo **this** ?
3. No método **trabalhe,** observe a criação de objetos da classe Circulo através do comando **new**.
4. No método **trabalhe,** observe que as variáveis locais **a** e **b** são referências do tipo Circulo.
5. No método **trabalhe,** observe as chamadas de métodos de objetos da classe Circulo através de referências a esses. Por que tais métodos precisam ser públicos?

```
class Arquiteto {

    private String nome;
    private int idade;

    public Arquiteto(String nome, int idade) {
        this.nome = nome;
        this.idade = idade;
    }

    public void exiba_dados_pessoais() {
        System.out.println( nome );
        System.out.println( idade + " anos");
        System.out.println();
    }

    …
```

```
…
    public void trabalhe(double r1, double r2, double r3) {
        Circulo a = new Circulo(r1);
        double x = a.area();
        double y = a.perimetro();
        imprima(r1,x,y);

        Circulo b = new Circulo(r2);
        x = b.area();
        y = b.perimetro();
        imprima(r2,x,y);

        b = new Circulo(r3);
        x = b.area();
        y = b.perimetro();
        imprima(r3,x,y);
    }

    private void imprima(double raio, double area, double perimetro) {
        System.out.println("raio      :" + raio);
        System.out.println("area      :" + area);
        System.out.println("perimetro :" + perimetro);
        System.out.println();
    }
}
```

Classe Escritorio

1. Observe que não há um construtor para a classe Escritorio.
2. No método **main,** observe a criação de objetos da classe Arquiteto através do comando **new**.
3. No método **main,** observe que as variáveis locais **oscar** e **kengo** são referências do tipo Arquiteto.
4. No método **desenhar**, observe que o tipo do parâmetro **arq** é a classe Arquiteto. Qual o tipo de passagem de parâmetro nesse caso?

```
public class Escritorio {

  public static void main(String[] args) {
     Arquiteto oscar = new Arquiteto("Oscar Niemeyer", 104);
     Arquiteto kengo = new Arquiteto("Kengo Kuma", 64);

     desenhar(oscar, 2.0, 5.2, 3.7);
     desenhar(kengo, 7.5, 4.0, 9.6);
  }

  private void desenhar(Arquiteto arq,
                        float a, float b, float c) {
     arq.exiba_dados_pessoais();
     arq.trabalhe(a,b,c);
  }
}
```