

Programação Orientada a Objetos

Sobrecarga de Métodos

profº Mauricio Conceição Mario

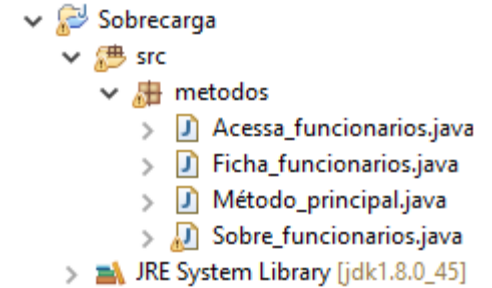
Sobrecarga de métodos: conceitos

- A linguagem Java permite que vários métodos com o mesmo nome sejam definidos, contanto que esses métodos tenham conjuntos diferentes de parâmetros (com base na quantidade, nos tipos e na ordem de parâmetros). Quando se chama um método sobrecarregado, o compilador Java seleciona o método adequado examinando a quantidade, os tipos e a ordem dos argumentos na chamada. A sobrecarga de métodos é comumente utilizada para criar vários métodos com o mesmo nome que realizam tarefas semelhantes, mas sobre tipos de dados diferentes.
- Os métodos sobrecarregados são distinguidos por sua *assinatura*, uma combinação do nome do método e dos tipos dos seus parâmetros.
- Os métodos não podem ser distinguidos pelo tipo de valor devolvido. Os métodos sobrecarregados podem ter tipos de valores devolvidos diferentes, mas devem ter listas de parâmetros diferentes.

```

1 package metodos;
2
3 public class Ficha_funcionarios {
4
5     private String nome;
6     private int idade;
7     private String profissao;
8     private double salario;
9
10    Ficha_funcionarios e;
11    Ficha_funcionarios(String nome, String profissao, int idade, double salario){
12
13        this.nome = nome;
14        this.profissao = profissao;
15        this.idade = idade;
16        this.salario = salario;
17
18    }
19
20    Ficha_funcionarios(int idade, double salario, String nome, String profissao ){
21        this.nome = nome;
22        this.profissao = profissao;
23        this.idade = idade;
24        this.salario = salario;
25    }
26
27    Ficha_funcionarios(Ficha_funcionarios e){
28        this.e = e;
29        mostra_funcionarios(e);
30    }
31
32    public void mostra_funcionarios(Ficha_funcionarios f)
33    {
34        System.out.println("Nome: " + f.nome + "\nIdade " + f.idade + "\nProfissao "
35            + f.profissao + "\nSalario " + f.salario);
36    }
37
38    public String mostra_funcionarios( ){
39    return ("Nome: " + nome + "\nIdade " + idade + "\nProfissao "
40        + profissao + "\nSalario " + salario);
41    }
42 }

```



```
1 package metodos;
2
3 public class Acesso_funcionarios extends Ficha_funcionarios{
4
5     static Ficha_funcionarios d = new Ficha_funcionarios("Sueli", "Professora", 32, 3876.00);
6     Acesso_funcionarios(){
7         super(d);
8     }
9
10 }
11
```

```
1 package metodos;
2
3 public class Método_principal extends Acesso_funcionarios {
4
5
6     Método_principal(){
7         super();
8     }
9
10
11
12 }
13
```

```

1 package metodos;
2
3 import java.awt.Container;
4 import javax.swing.*;
5
6
7 public class Sobre_funcionarios extends JApplet {
8     Ficha_funcionarios funcionarios = new Ficha_funcionarios(22, 2888.09, "Gisele", "Pedagoga");
9     Método_principal a = new Método_principal();
10
11
12     public void init(){
13         JTextArea saida = new JTextArea();
14         Container container = getContentPane();
15         container.add(saida);
16         saida.setText(funcionarios.mostra_funcionarios());
17     }
18 }
19
20

```

Visu... — □ ×

Applet

Nome: Gisele
Idade 22
Profissão Pedagoga
Salário 2888.09

Applet iniciado.

Sobre_funcionarios [Java Applet] C:\Arquivos de Progra

Nome: Sueli
Idade 32
Profissão Professora
Salário 3876.0

Exercício 1:

- a) Fazer o diagrama de classes da aplicação que fornece dados dos funcionários;
- b) Identificar as ocorrências de sobrecarga nos métodos da aplicação;
- c) Fazer um fluxograma identificando quando os métodos são invocados em função das suas respectivas assinaturas, inclusive os construtores invocados através do método *super()*.

```

public class Figuras_geometricas {
    private double lado_a;
    private static double lado_b;
    private double lado_c;
    private double area;

    Figuras_geometricas ( ){
        lado_a = 2.0;
        lado_b = 3.0;
        lado_c = 1.0;}

    public void setLado_a(double lado_a)
    {this.lado_a = lado_a;}

    public void setLado_b(double lado_b)
    {this.lado_b = lado_b;}

    public double getLado_a() {
        return lado_a;
    }

    public double getLado_b() {
        return lado_b;
    }
}

```

Exercício 2:

Verificar o cálculo das áreas de retângulo e quadrado através do uso da sobrecarga de métodos, compilando e executando as classes Figuras_geometricas e Calcula_areas:

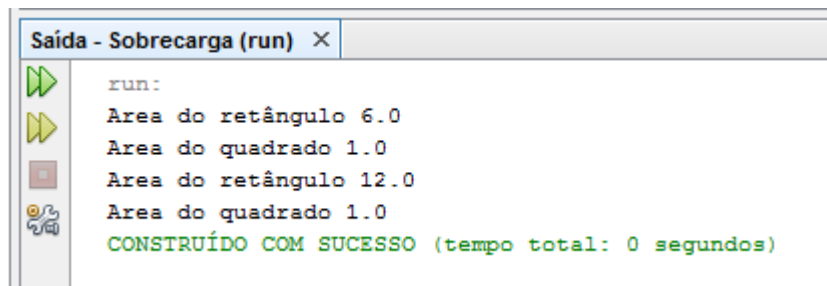
```

public double getArea (double lado_a, double lado_b )
{this.lado_a = lado_a; this.lado_b = lado_b;
area =lado_a*lado_b; return area; }

public double getArea ( )
{ area = lado_c*lado_c; return area;}

```

```
public class Calcula_areas {  
    public static void main (String args[]){  
        Figuras_geometricas u = new Figuras_geometricas ( );  
        System.out.println("Area do retângulo " + u.getArea(u.getLado_a(),u.getLado_b()));  
        System.out.println("Area do quadrado " + u.getArea());  
        u.setLado_a(4.0);  
        System.out.println("Area do retângulo " + u.getArea(u.getLado_a(),u.getLado_b()));  
        System.out.println("Area do quadrado " + u.getArea());  
    }  
}
```

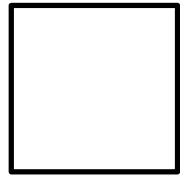


Saída - Sobrecarga (run) X

run:
Area do retângulo 6.0
Area do quadrado 1.0
Area do retângulo 12.0
Area do quadrado 1.0
CONSTRUÍDO COM SUCESSO (tempo total: 0 segundos)

Exercício 3:

As áreas de um quadrado, retângulo e um cubo podem ser calculadas, respectivamente, conforme abaixo:



a

$$\text{área} = a * a$$

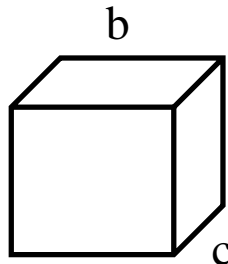
a



a

$$\text{área} = a * b$$

b



b

a

$$\text{área} = a * b * c$$

c

Utilizando o conceito de sobrecarga de métodos, construir uma classe que contenha os métodos `area()` que possam calcular as áreas do quadrado, retângulo e cubo.

Referências Bibliográficas

Java 7 - Ensino Didático

Sérgio Furgeri - Editora Érica

Java - como Programar - 4edição

H. M. Deitel & P. J. Deitel / Ed. Bookman – 2003.