

```
int add(int a, int b) {  
    return a + b;  
}  
  
int add(int a, int b, int c) {  
    return a + b + c;  
}  
  
double add(double a, double b) {  
    return a + b;  
}  
  
}
```

Aula 8 Sobrecarga de Métodos

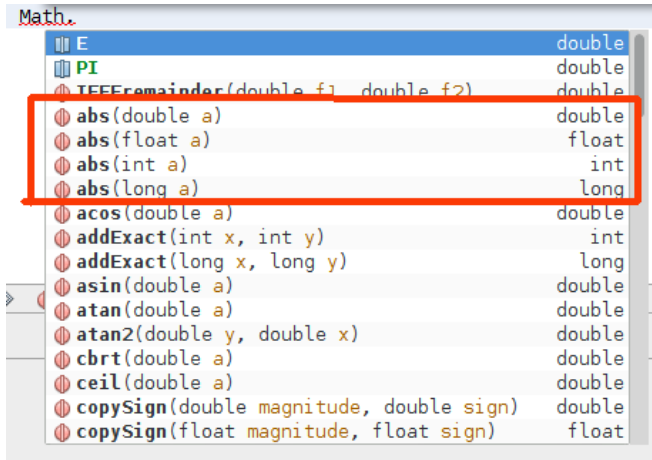
Sobrecarga de Métodos

- O conceito de sobrecarga é muito importante e utilizado na programação orientada a objetos
- **A sobrecarga permite que métodos com o MESMO NOME possam ser declarados na mesma classe**, contanto que tenham diferente **conjunto de parâmetros**

Sobrecarga de Métodos

- Quando um método sobrecarregado é chamado, o **compilador seleciona o método adequado examinando o número, os tipos e a ordem dos argumento na chamada**
- **A sobrecarga de métodos é comumente utilizada para criar vários métodos com o mesmo nome que realizam as mesmas tarefas ou tarefas semelhantes, mas sobre tipos diferentes ou números diferentes de argumentos**

Sobrecarga de Métodos



Sobrecarga de Métodos

Math |

max(double a, double b)	double
max(float a, float b)	float
max(int a, int b)	int
max(long a, long b)	long
min(double a, double b)	double
min(float a, float b)	float
min(int a, int b)	int
min(long a, long b)	long
multiplyExact(int x, int y)	int
multiplyExact(long x, long y)	long
negateExact(int a)	int
negateExact(long a)	long
nextAfter(double start, double direction)	double
nextAfter(float start, double direction)	float
nextDown(double d)	double
nextDown(float f)	float
nextUp(double d)	double

Sobrecarga de Métodos

```
12 public class Matematica {  
13  
14     public static int quadrado(int numero){  
15         System.out.println("Chamando função quadrado para um número inteiro");  
16         return numero * numero;  
17     }  
18  
19     public static double quadrado(double numero){  
20         System.out.println("Chamando função quadrado para um número double");  
21         return numero * numero;  
22     }  
23  
24 }
```

Sobrecarga de Métodos

```
14 public class Teste {  
15  
16     public static void main(String[] args) {  
17  
18         int num1 = 2;  
19         double num2 = 4.0;  
20  
21         int quadradoNum1 = Matematica.quadrado(num1);  
22         System.out.println("O quadrado no num1 é: " + quadradoNum1);  
23  
24         double quadradoNum2 = Matematica.quadrado(num2);  
25         System.out.println("O quadrado no num2 é: " + quadradoNum2);  
26  
27     }  
28  
29 }  
30
```

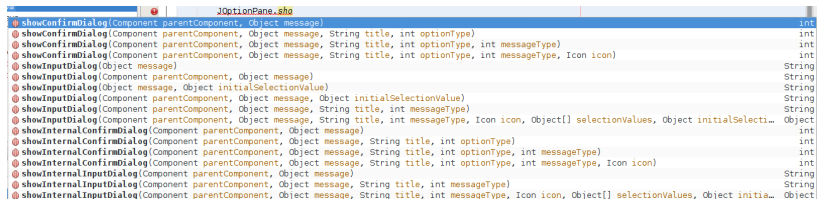
teste.Teste > main >

Saída - Teste (run) x

```
run:  
Chamando função quadrado para um número inteiro  
O quadrado no num1 é: 4  
Chamando função quadrado para um número double  
O quadrado no num2 é: 16.0
```

Sobrecarga de Métodos

Lembrando que também **pode-se utilizar a sobrecarga para criar métodos que realizam a mesma função mas com número de parâmetros diferentes**



```
1 JOptionPane.showMessageDialog(Component parentComponent, Object message) int
2 JOptionPane.showMessageDialog(Component parentComponent, Object message, String title, int optionType) int
3 JOptionPane.showMessageDialog(Component parentComponent, Object message, String title, int optionType, int messageType) int
4 JOptionPane.showMessageDialog(Component parentComponent, Object message, String title, int optionType, int messageType, Icon icon) int
5 JOptionPane.showInputDialog(Object message) String
6 JOptionPane.showInputDialog(Component parentComponent, Object message) String
7 JOptionPane.showInputDialog(Component parentComponent, Object message, Object initialSelectionValue) String
8 JOptionPane.showInputDialog(Component parentComponent, Object message, String title, int messageType) String
9 JOptionPane.showInputDialog(Component parentComponent, Object message, String title, int messageType, Icon icon, Object[] selectionValues, Object initialSelectionValue) Object
10 JOptionPane.showInternalConfirmDialog(Component parentComponent, Object message) int
11 JOptionPane.showInternalConfirmDialog(Component parentComponent, Object message, String title, int optionType) int
12 JOptionPane.showInternalConfirmDialog(Component parentComponent, Object message, String title, int optionType, int messageType) int
13 JOptionPane.showInternalConfirmDialog(Component parentComponent, Object message, String title, int optionType, int messageType, Icon icon) int
14 JOptionPane.showInternalInputDialog(Component parentComponent, Object message) String
15 JOptionPane.showInternalInputDialog(Component parentComponent, Object message, String title, int messageType) String
16 JOptionPane.showInternalInputDialog(Component parentComponent, Object message, String title, int messageType, Icon icon, Object[] selectionValues, Object initialSelectionValue) Object
```


Sobrecarga de Métodos

```
12 public class Matematica {  
13  
14     public static int max(int num1, int num2){  
15         if(num1 > num2){  
16             return num1;  
17         }else{  
18             return num2;  
19         }  
20     }  
21  
22     public static int max(int num1, int num2, int num3){  
23         if(num1 > num2){  
24             if(num1 > num3){  
25                 return num1;  
26             }else{  
27                 return num3;  
28             }  
29         }else{  
30             if(num2 > num3){  
31                 return num2;  
32             }else{  
33                 return num3;  
34             }  
35         }  
36     }  
37 }  
38
```

Sobrecarga de Métodos

```
14 public class Teste {  
15  
16     public static void main(String[] args) {  
17  
18         int num1 = 10;  
19         int num2 = 30;  
20         int num3 = 50;  
21  
22         System.out.println("Maior número é: " + Matematica.max(num1, num2));  
23         System.out.println("Menor número é: " + Matematica.max(num1, num2, num3));  
24     }  
25 }  
26  
27 }  
28
```

teste.Teste > main >

Saída - Teste (run) x

```
run:  
Maior número é: 30  
Menor número é: 50
```

Sobrecarga de Métodos

- Bem... construtores também são métodos, não??
- Bem... então podemos utilizar a conceito de **sobrecarga também nos construtores**

Sobrecarga de Métodos

```
12 public class Pessoa {  
13  
14     public String nome;  
15     public int idade;  
16  
17     public Pessoa(){  
18         this.nome = "";  
19         this.idade = 0;  
20     }  
21  
22     public Pessoa(String nome, int idade){  
23         this.nome = nome;  
24         this.idade = idade;  
25     }  
26  
27     public String getNome() {  
28         return nome;  
29     }  
30 }
```

Sobrecarga de Métodos

```
14 public class Teste {  
15  
16     public static void main(String[] args) {  
17  
18         Pessoa pessoal = new Pessoa();  
19         pessoal.setNome("Rafael Rossi");  
20         pessoal.setIdade(30);  
21  
22         Pessoa pessoa2 = new Pessoa("Ricardo Marcacini", 35);  
23  
24         System.out.println("Informações Pessoal =====\nNome " + pessoal.getNome() + ", " + pessoal.getIdade() + " anos.");  
25         System.out.println("Informações Pessoa2 =====\nNome " + pessoa2.getNome() + ", " + pessoa2.getIdade() + " anos.");  
26     }  
27  
28 }  
29
```

teste.Teste > main >

Saida - Teste (run) x

```
run:  
Informações Pessoal =====  
Nome Rafael Rossi, 30 anos.  
Informações Pessoa2 =====  
Nome Ricardo Marcacini, 35 anos.
```

Distinguindo Métodos Sobrecarregados

- O compilador distingue os métodos sobrecarregados pelas suas **assinaturas** → combinação do **nome, número, tipos e ordem dos parâmetros**
- **OBSERVAÇÃO 1:** as chamadas de método não podem ser distinguidas por tipo de retorno

Distinguindo Métodos Sobrecarregados

```
public double getQts_pessoas() {  
    return 0.0;  
}  
  
public int getQts_pessoas() {  
    return qtd_pessoas;  
}
```

method getQts_pessoas() is already defined in class Pessoa

(Alt-Enter mostra dicas)

Material Complementar

- Curso de Java #15 - Métodos

https:

[//youtu.be/EuxmbXZCFVQ?list=PLHz_AreHm4dkI2ZdjTwZA4mPMxWTfNSpR](https://youtu.be/EuxmbXZCFVQ?list=PLHz_AreHm4dkI2ZdjTwZA4mPMxWTfNSpR)

- Orientação a Objetos: Sobrecarga de métodos e construtores (overload)

https://www.youtube.com/watch?v=ZpssJov_5_A

- Introdução: Sobrecarga de Métodos e Tipos Genéricos em Java

<http://www.devmedia.com.br/>

[introducao-sobrecarga-de-metodos-e-tipos-genericos-em-java/22853](http://www.devmedia.com.br/introducao-sobrecarga-de-metodos-e-tipos-genericos-em-java/22853)

Imagem do dia



Programação Orientada a Objetos

<http://lives.ufms.br/moodle/>

Rafael Geraldeli Rossi
rafael.g.rossi@ufms.br

Slides baseados em [Deitel and Deitel, 2010]

Referências Bibliográficas I



Deitel, P. and Deitel, H. (2010).

Java: How to Program.

How to program series. Pearson Prentice Hall, 8th edition.