# Programação de Sistemas Computacionais

Sobrecarga de métodos

Construtores

emerson@paduan.pro.br

# Sobrecarga / Overloading

A sobrecarga ou overloading ocorre quando temos mais de um método com o mesmo nome.

A diferenciação entre os métodos é feita por meio dos parâmetros. Ou seja, dois métodos sobrecarregados devem necessariamente diferir em <u>pelo menos</u> um parâmetro.

Não é possível realizar sobrecarga modificando APENAS o tipo de retorno do método.

# Exemplo

```
Considere a seguinte classe.
```

```
public class Calculadora {
    public int soma (int a, int b) {
        return a + b;
    }
```

Faça a classe AppCalculadora (main) para testar.

emerson@paduan.pro.b

# Exemplo

E se quisermos ter um método que soma números do tipo double?

```
public class Calculadora {
    public double soma (double a, double b) ;
    return a + b;
}
```

### Exemplo

E se quisermos ter um método que recebe duas Strings?

```
public class Calculadora {
    public double soma (String a, String b) {
        return Double.parseDouble(a) + Double.parseDouble(a);
    }
}
```

emerson@paduan.pro.br

### Exercício

Escreva uma classe chamada Impressora. Essa classe tem como finalidade exibir conteúdos diversos na tela. Ela deve possuir diversos métodos chamados exibir(). Eles poderão coexistir graças ao mecanismo conhecido como sobrecarga de métodos.

Escreva versões do método exibir que tenham as seguintes listas de parâmetros:

- 1. um valor double.
- 2. dois valores double.
- 3. um double e uma String, nessa ordem.
- 4. uma String e um double, nessa ordem.
- 5. dois int e uma String, nessa ordem.

Em todos os casos exiba todos os valores usando System.out.println, separando cada um informação com vírgulas.

Escreva uma classe de teste que chama cada um dos métodos que você escreveu.

### Construtor

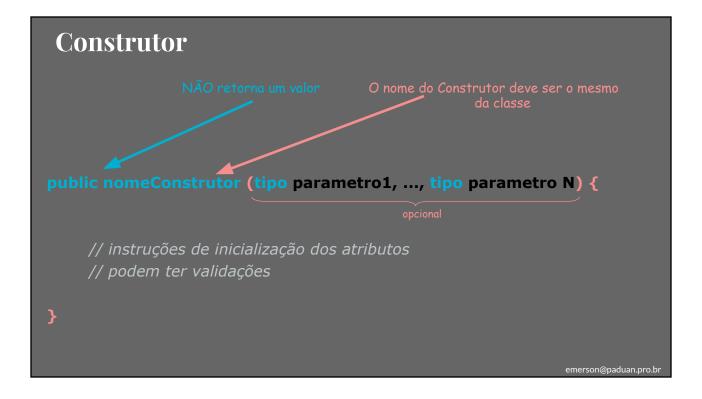
São *métodos* especiais utilizados para inicialização dos *atributos* de um objeto no "momento" da criação do objeto.

#### Detalhes:

Os construtores podem ter ou não parâmetros

O nome do construtor deve ser o mesmo da classe

Construtores NÃO possuem valor de retorno



```
Exemplo

public class Pessoa {
    String nome;
    float salario;

public Pessoa(String nomePessoa, float sal){
    nome = nomePessoa;
    salario = sal;
    }

public void exibir(){
    System.out.println("Pessoa: "+nome + ": R$ "+salario);
    }
}
```

### this

O operador *this* é utilizado para se referir aos membros (atributos/métodos) do próprio objeto.

```
public class Pessoa {
   String nome;
   float salario;

public Pessoa(String nome, float salario){
     this.nome = nome;
     this.salario = salario;
   }
}
```

# Overloading em Construtores

emerson@paduan.pro.bi

## Exemplo

# Let's code!



emerson@paduan.pro.br

### Exercício

Crie uma classe Placar, que representa o placar de um jogo de futebol. Discuta quais os atributos devem ser adicionados à classe (e os tipos de dados), e faça 3 construtores:

- 1 construtor padrão (default, sem parâmetros);
- 1 construtor que defina apenas quais são os times do jogo (mantendo o placar em 0 a 0);
- 1 construtor que defina os times e o placar do jogo;

Crie no *main* três objetos do tipo placar, um testar para cada tipo de construtor, e exiba os dados do placar no formato: Time1 Gols X Gols Time2.

Exemplo: São Paulo 3 X 0 Corinthians

