INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
RIO GRANDE DO NORTE

AULA:

Introdução a Orientação a Objetos (Classes e Objetos)

Programação Orientada a Objetos

Alba Lopes, Profa.

http://docentes.ifrn.edu.br/albalopes
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Na computação, temos um problema a ser resolvido
  - Devemos analisar o que deve ser feito
  - Definir como deve ser feito
  - Escrever um algoritmo/programa que implemente a solução
- Um programa realiza operações sobre dados

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Modelagem Orientada a Objetos (OO)
  - O ser humano conhece o mundo e gerência sua complexidade através de objetos
  - É como desenvolvemos nossa cognição (conhecimento).
- Desenvolvemos o conceito de OBJETO
  - Exemplos de objetos: bola, carro, camisa, luz, casa, calça,música, conta bancária, poema, etc

Conceitos Fundamentas

- O que é um objeto?

> "É a representação de uma coisa do mundo real." (BARNES,2009)

- Exemplos:
  - O carro vermelho que está ali no estacionamento.
  - Este lápis na minha mão.
  - A peruca do Silvio Santos.

OBJETO = DADOS + OPERAÇÕES

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Objetos Possuem:
  - Estado:
    - Representado pelos valores dos atributos de um objeto
  - Comportamento:
    - Definido pelo conjunto de operações (métodos) do objeto;
    - Estado representa o resultado cumulativo de seu comportamento;
  - Identidade:
    - Dois objetos são distintos mesmo que os valores de seus atributos sejam exatamente iguais.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- O que é uma classe?

> "É um projeto de um objeto. Ela informa como cria um objeto de um tipo específico."
> (SIERRA & BATES, 2007)

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Analogia

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Classe
  - É onde conceituamos o objeto
  - É a essência do objeto
  - Define os atributos e métodos
- Objeto
  - É a instância de uma classe
  - Objetos semelhantes pertencem a uma mesma classe

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Atributos
  - Atributos são as propriedades de um objeto
  - Exemplo:
    - Um objeto carro pode ter que propriedades?
      - Cor
      - Modelo
      - Marca
      - Potência do motor
      - Quantidade de portas
      - Velocidade atual
      - Etc..

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Métodos
  - Métodos são as ações que um objeto pode realizar
  - Exemplo:
    - Um objeto carro pode realizar que ações?
      - Acelerar
      - Frear
      - Buzinar
      - Etc..

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Como representamos uma Classe?
  - Através da UML (Unified Modeling Language)
  - Retângulo com três divisões:
    - Nome da Classe
    - Atributos
    - Métodos (ações).

Automóvel
Atributos
Métodos (ações)

Automóvel
+ modelo : String + cor : String + velocidade : int
+ buzinar ( ) : void + acelerar( ) : void + reduzir( ) : void

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

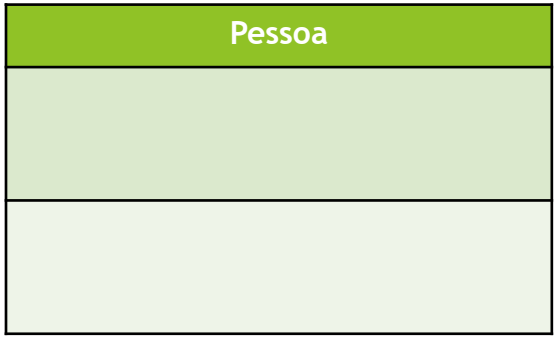
- O que são Instâncias?
  - Chamamos de instância, cada objeto criado a partir de uma classe.

Automóvel
+ modelo : String + cor : String + velocidade : int
+ buzinar ( ) : void + acelerar( ) : void + reduzir( ) : void

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Se definirmos o objeto Pessoa, que atributos ela pode ter?



Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- Se definirmos o objeto Pessoa, que atributos ela pode ter?

Pessoa
+ nome : String + idade : int + profissao :String + rg : int

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- E em relação aos métodos?

Pessoa
+ nome : String + idade : int + profissao :String + rg : int

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Conceitos Fundamentas

- E em relação aos métodos?

Pessoa
+ nome : String + idade : int + profissao :String + rg : int
+ andar( ) + falar ( ) + trabalhar ( )

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Exercícios

- Definir os atributos e métodos para as classes abaixo:
  - Conta Corrente
  - Lâmpada
  - Aluno
  - Calculadora
  - Data

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- Até o momento, nossos códigos em Java eram construídos da seguinte maneira

```
<DECLARAÇÃO_DA_CLASSE>

      <MÉTODO_MAIN>
            <VARIÁVEIS>
            <OPERAÇÕES>
```

```
public class Calculadora
      public static void main String [] args
            int numero1 = 10;
            int numero2 = 20;
            int soma;
            soma = numero1 + numero2;
            System.out.println("A soma é: " + soma);
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- A partir de agora, iremos abranger o nosso modo de construir as nossas classes

```
<DECLARAÇÃO_DA_CLASSE>
      <ATRIBUTOS>
      <MÉTODOS>
```

Nossa classe passará a ter mais métodos além do método main.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- Definindo Atributos
  - Atributo é o termo como se denomina uma variável de classe em Java
    - Para definirmos um atributo em Java, devemos identificar o tipo de dado que o representa (int, float, string, boolean, etc)
  - Que tipo de dados representam os atributos abaixo da classe Automóvel?
    - Cor
    - Modelo
    - Marca
    - Potência do motor
    - Quantidade de portas
    - Velocidade atual

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- Definindo Atributos
  - Que tipo de dados representam os atributos abaixo da classe Automóvel?
    - Cor **String**
    - Modelo **String**
    - Marca **String**
    - Potência do motor **float**
    - Quantidade de portas **int**
    - Velocidade atual **int**

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- Definindo atributos
  - Os atributos são definidos seguindo o padrão abaixo:

```
<modificador_de_acesso> <tipo_de_dado> <nome_do_atributo> = <valor_inicial> ;
```

opcional opcional

  - Exemplo (omitindo o modificador de acesso):

```
String marca;
```

  - Com modificador de acesso:

```
private String cor;
```

  - Com valor inicial

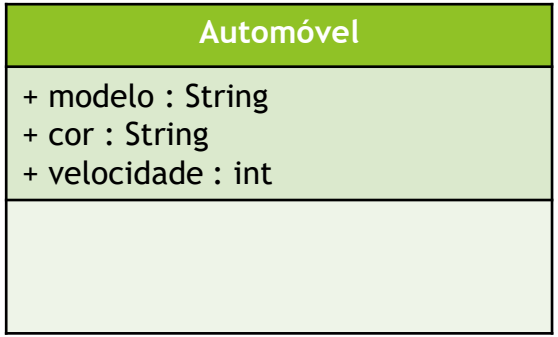
```
private int velocidade = 0;
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- Definindo atributos
  - Exemplo: classe Automovel

```
public class Automovel

      String modelo;
      String cor;
      int velocidade = 0;
```



Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- Definindo métodos
  - Métodos são similares às funções em linguagens de programação
  - Um método é representado por uma operação que realiza ações e pode modificar os valores dos atributos do objeto
  - Pode possuir ou não uma lista parâmetros.
  - Pode retornar algum valor ou não.
    - Caso retorne algum valor, deverá informar que tipo de valor o método retorna (int, float, String, boolean, etc)
    - Caso não retorne nenhum valor, o tipo informado deverá ser void

Criando Classes

- Definindo métodos
  - A sintaxe de definição de métodos em Java é a seguinte:

```
<modificador_de_acesso> <tipo_de_retorno> <nome_do_método> (<lista_de_parâmetros>){
        <operações>
}
```

  - A declaração mais simples que podemos fazer é a seguinte:

```
void <nome_do_metodo> ( ){
        <operacões>
}
```

  - Exemplo:

```
void acelerar ( ) {
        velocidade = velocidade + 1;
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

- Exemplo simples de uma classe (Numero) com um atributo e um método:

```
public class Numero{

      int valor = 1;

      void escrever () {
            System.out.println("O valor é: "+ valor);
      }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Criando Classes

Automóvel
+ modelo : String + cor : String + velocidade : int
+ buzinar ( ) : void + acelerar( ) : void + reduzir( ) : void

- Exemplo de código para a classe Automóvel

```
public class Automovel{

    //Atributos
    String marca;
    String cor;
    int velocidade = 0;

    //Métodos
    void buzinar(){
         System.out.println("BEEEEPPPP....");
    }

    void acelerar(){
         velocidade = velocidade + 1;
    }

    void reduzir(){
         velocidade = velocidade - 1;
    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Instanciando Objetos

- Vamos criar uma nova classe para testar a classe construída anteriormente
- Voltamos a usar o nosso já conhecido método main

```
public class TestarAutomovel{

    public static void main(String [] args){




    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Instanciando Objetos

- Precisamos declarar uma variável qualquer como sendo do tipo da classe criada:

```
public class TestarAutomovel{

    public static void main(String [] args){
        Automovel meuCarro;




    }

}
```

Instanciando Objetos

- Depois atribuímos à variável criada o resultado obtido pelo instanciador new mais o método construtor, que é o nome da nossa classe seguido de parênteses (mais detalhes sobre método construtor nas próximas aulas)

```
public class TestarAutomovel{

    public static void main(String [] args){
        Automovel meuCarro;
        meuCarro = new Automovel();



    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Acessando elementos dos Objetos

- Como mencionado anteriormente, cada objeto possui seus próprios valores dos atributos e possui seu próprios métodos
- Podemos acessar os elementos (atributos ou métodos) de um objeto através da seguinte sintaxe:

```
<nome_do_objeto>.<elemento>
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Acessando elementos dos Objetos

- Para executar o método buzinar, por exemplo, utilizamos a seguinte sintaxe:

```
public class TestarAutomovel{

    public static void main(String [] args){
        Automovel meuCarro;
        meuCarro = new Automovel();
        meuCarro.buzinar();


    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Acessando elementos dos Objetos

- Para acessar um atributo, e, por exemplo, exibir a velocidade do objeto meuCarro:

```
public class TestarAutomovel{

    public static void main(String [] args){
        Automovel meuCarro;
        meuCarro = new Automovel();
        meuCarro.buzinar();
        System.out.println("A velocidade é: "+ meuCarro.velocidade );

    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Acessando elementos dos Objetos

- Para alterar valores dos atributos, utiliza-se a sintaxe padrão de atribuição:

```
public class TestarAutomovel{

    public static void main(String [] args){
        Automovel meuCarro;
        meuCarro = new Automovel();
        meuCarro.buzinar();
        System.out.println("A velocidade é: "+ meuCarro.velocidade);
        meuCarro.marca = "Ford";
        meuCarro.velocidade = 10;

    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Acessando elementos dos Objetos

- Os métodos acelerar e reduzir da classe Automovel, alteram o valor da variável velocidade.

```
public class TestarAutomovel{

    public static void main(String [] args){
        Automovel meuCarro;
        meuCarro = new Automovel();
        meuCarro.buzinar();
        System.out.println("A velocidade é: "+ meuCarro.velocidade);
        meuCarro.marca = "Ford";
        meuCarro.velocidade = 10;
        meuCarro.acelerar();

    }

}
```

Após a execução dessa linha, qual o novo valor da variável velocidade?

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Acessando elementos dos Objetos

- Descubra o valor escrevendo o valor do atributo

```
public class TestarAutomovel{

    public static void main(String [] args){
        Automovel meuCarro;
        meuCarro = new Automovel();
        meuCarro.buzinar();
        System.out.println("A velocidade é: "+ meuCarro.velocidade);
        meuCarro.marca = "Ford";
        meuCarro.velocidade = 10;
        meuCarro.acelerar();
        System.out.println("A velocidade é: " + meuCarro.velocidade)
    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Exercício

1. Crie uma classe Produto que obedeça a descrição apresentada na representação abaixo.
   - Possua os atributos **nome** e **preco**. E os métodos: **informarNome**, **informarPreco** e **fazerPromocao**.
     - O método **informarNome** deve escrever na tela a mensagem: “O nome do produto é ”, e o valor do atributo **nome.**
     - O método **informarPreco** deve escrever na tela a mensagem: “O preço do produto é”, e o valor do atributo **preco.**
     - O método **fazerPromocao** deve reduzir 1 real do preço do produto.
     - Crie também uma classe TestarProduto que possua um método **main** que realize as seguintes operações:
       - Instancie um objeto do tipo Produto.
       - Atribua o um valor ao atributo **nome (ex:** Notebook**).**
       - Atribua o preço ao atributo **preco (ex:** 999).
       - Chame o método **informarNome**
       - Chame o método **informarPreco**
       - Chame o método **fazerPromocao**
       - Chame novamente o método **informarPreco**

Produto
+ nome : String + preco: double
+ informarNome() : void + informarPreco() : void + fazerPromocao() : void

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Exercício

2. Crie uma classe Interruptor que obedeça a descrição apresentada na representação abaixo.

- Possua o atributo **ligado**, iniciado com o valor *false*. E os método: **pressionar** e **verificarLampada**.
  - O método **pressionar** deve inverter o valor do atributo **ligado** (colocar como *true*, caso esteja *false* ou colocar para *false* caso esteja *true*.
  - O método **verificarLampada** deve informar a mensagem "A luz está acesa" caso o atributo **ligado** seja igual a *true* ou a mensagem "A luz está apagada" caso o atributo **ligado** seja igual a *false*.
  - Crie também uma classe TestarInterruptor que possua um método **main** que realize as seguintes operações:
    - Instancie um objeto do tipo Interruptor.
    - Chame o método **pressionar.**
    - Chame o método **verificarLampada**.
    - Chame o método **pressionar** novamente.
    - Chame o método **verificarLampada** novamente.

Interruptor
+ ligado: boolean
+ pressionar() : void + verificarLampada() : void

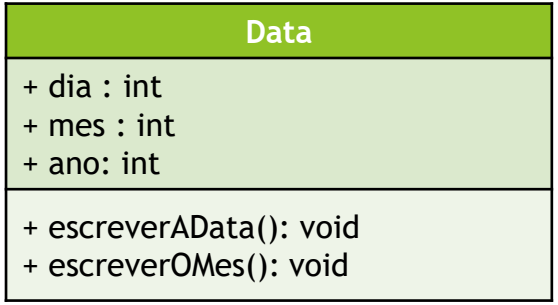
Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Exercício

3. Crie uma classe Data que obedeça a descrição apresentada abaixo.
   - Possua os atributos **dia**, **mes** e **ano**. E os métodos: **escreverAData**, **escreverOMes.**
     - O método escreverAData deve mostrar a data na tela no formato DD/MM/AAA
     - O método **escreverOMes** deve mostrar o nome do mês. Por exemplo , para mês 2, escrever “Fevereiro”

Crie também uma classe TestarData que possua o método **main** que realize as seguintes operações:

- Instancie um objeto do tipo **Data**.
- Atribua um valor para o atributo **dia**
- Atribua um valor para o atributo **mes**
- Atribua um valor para o atributo **ano**
- Chame o método **escreverAData**
- Chame o método **escreverOMes**
- Altere o valor do atributo **mes**
- Chame o método **escreverAData**
- Chame o método **escreverOMes**



Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Métodos

- Parâmetros
  - Em Java, os métodos podem possuir ou não parâmetros
  - Na exemplo anterior, construímos classes que possuem métodos sem parâmetros
  - Agora, vamos estudar como construir métodos que utilizam parâmetros
- Os parâmetros são utilizados para alterar os valores dos atributos ou o comportamento dos métodos durante sua execução

Métodos

- No exemplo da classe Automovel abaixo:

```
public class Automovel{

      //Atributos
      String marca;
      String cor;
      int velocidade = 0;

      //Métodos
      void buzinar(){
            System.out.println("BEEEEPPPP....");
      }

      void acelerar(){
            velocidade = velocidade + 1;
      }

      void reduzir(){
            velocidade = velocidade - 1;
      }

}
```

Se nos métodos acelerar e reduzir quiséssemos variar o valor do incremento de acordo com a execução e não deixar amarrado em incrementar apenas em 1 unidade?

Podemos utilizar parâmetros nos métodos!

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Parâmetros

- Inserindo parâmetros nos métodos
  - Um método pode possuir um ou mais parâmetros
    - Se o método possuir mais de um parâmetro, deve-se separá-los por vírgulas

```
<tipo_do_param1> <nome_do_param1>, <tipo_do_param2> <nome_do_param2>,...
```

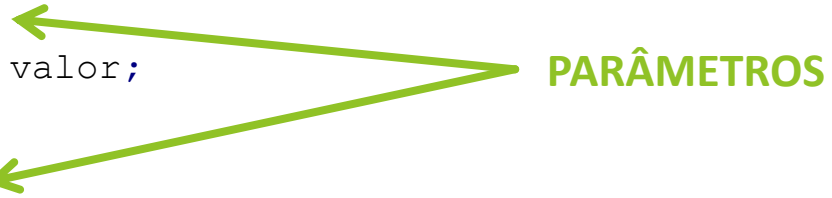
  - Exemplo: se quiséssemos definir em quanto a velocidade aumentou ao acelerar, poderíamos definir o método acelerar da seguinte forma:

```
void acelerar ( int valor ){
        velocidade = velocidade + valor;
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Parâmetros

```
public class Automovel{

    //Atributos
    String marca;
    String cor;
    int velocidade = 0;

    //Métodos
    void buzinar(){
        System.out.println("BEEEEPPPP....");
    }

    void acelerar( int valor ){
        velocidade = velocidade + valor;
    }

    void reduzir( int valor ){
        velocidade = velocidade - valor;
    }

}
```



Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Parâmetros

- Ao instanciar um objeto da classe e chamar um método que possui parâmetro, a chamada do método deve incluir o valor do parâmetro passado:

```
public class TestarAutomovel{

     public static void main(String [] args){
          Automovel meuCarro = new Automovel();
          meuCarro.marca = "Fiat";
          meuCarro.cor = "branco";

          System.out.println("Acelerando...");
          meuCarro.acelerar(10);
          System.out.println("Acelerando...");
          meuCarro.acelerar(20);
          System.out.println("A velocidade é: " + meuCarro.velocidade);
          System.out.println("Reduzindo...");
          meuCarro.reduzir(5);
          System.out.println("A velocidade é: " + meuCarro.velocidade);
     }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Parâmetros

- Um método pode possuir mais de um parâmetro:

```
public class Calculadora(){
     int operador1;
     int operador2;
     int resultado;

     void somar(int a, int b){
               operador1 = a;
               operador2 = b;
               resultado = a + b;
     }

     void subtrair(int a, int b){
          operador1 = a;
          operador2 = b;
          resultado = a - b;
     }
     void mostrarResultado(){
          System.out.println("O resultado da operação é: " + resultado);
     }
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Parâmetros

- Um método pode possuir mais de um parâmetro:

```
public class Calculadora(){
      int operador1;
      int operador2;
      int resultado;

      void somar(int a, int b){
                operador1 = a;
                operador2 = b;
                resultado = a + b;
      }

      void subtrair(int a, int b){
            operador1 = a;
            operador2 = b;
            resultado = a - b;
      }
      void mostrarResultado(){
            System.out.println("O resultado da operação é: " + resultado);
      }
}
```

Lista de Parâmetros (separados por vírgula)

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Parâmetros

- Chamada de método com mais de um parâmetro:

```
public class TestarCalculadora(){
    public static void main(String [] args){
        Calculadora calc = new Calculadora();
        calc.somar(10,5);
        calc.mostrarResultado();
    }

}
```


Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Retorno de Métodos

- Até o momento, só trabalhamos com métodos do tipo void, que não retornam nenhum valor
- Mas os métodos podem retornar algum valor para o trecho que o chamou
  - Nesse caso, deve-se informar o tipo do retorno (int, String, double, etc)
- Um método que retorna algum valor tem a seguinte sintaxe:

```
<tipo_de_retorno> <nome_do_método> (<lista_de_parâmetros>){
        <operações>
        return <valor_de_retorno>;
}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Retorno de Métodos

- Na classe Calculadora, do exemplo anterior, os métodos podem ser construídos retornando um valor:

```
public class Calculadora(){

     int operador1;
     int operador2;

     int somar(int a, int b){
          operador1 = a;
          operador2 = b;
          return a + b;
     }

     void subtrair(int a, int b){
          operador1 = a;
          operador2 = b;
          return a - b;
     }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Retorno de Métodos

- Na classe Calculadora, do exemplo anterior, os métodos podem ser construídos retornando um valor:

```
public class Calculadora(){

    int operador1;
    int operador2;

    int somar(int a, int b){
        operador1 = a;
        operador2 = b;
        return a + b;
    }

    void subtrair(int a, int b){
        operador1 = a;
        operador2 = b;
        return a - b;
    }

}
```

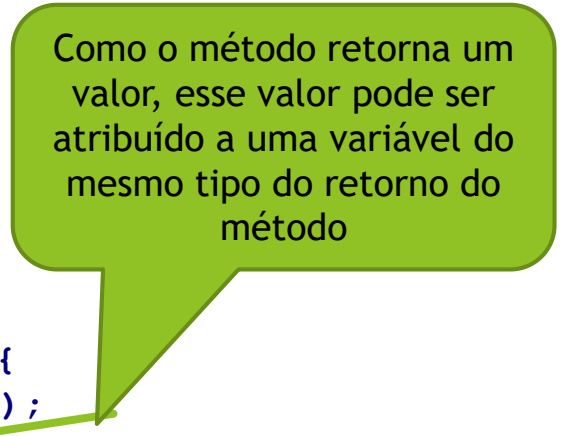
Tipo de retorno do método

Palavra **return** e valor a ser retornado

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Retorno de Métodos

- Utilização de um método que retorna um valor:




```
public class TestarCalculadora(){
    public static void main(String [] args){
        Calculadora calc = new Calculadora();
        int valor = calc.somar(10,5);
        System.out.println("O resultado é: " + valor);
    }

}
```

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

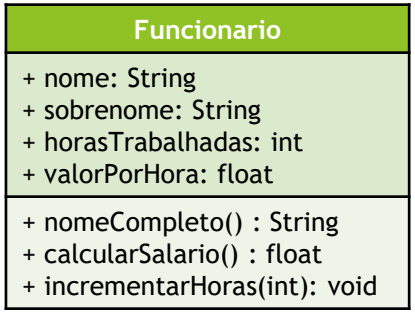
Exercícios

ContaCorrente
+ saldo: double
+ definirSaldoInicial(double) : void + depositar(double) : void + sacar(double): void

1. Crie uma classe ContaCorrente que obedeça à descrição abaixo:
   - A classe possui o atributo saldo do tipo double e os métodos definirSaldoInicial, depositar e sacar.
   - O método definirSaldoInicial deve atribuir o valor passado por parâmetro ao atribuito saldo
   - O método depositar, deve adicionar o valor passado por parâmetro ao atributo saldo
   - O método sacar deve reduzir o valor passado por parâmetro do salto já existente
- Crie uma classe TestarContaCorrente que possua o método main. Realize as seguintes operações:
  - Crie um objeto novaConta do tipo ContaCorrente.
  - Chame o método definirSaldoInicial passando o valor 1000 como parâmetro.
  - Escreva o valor do atributo saldo
  - Realize um saque de 500 reais (utilize o método sacar).
  - Faça um depósito de 50 reais (utilize o método depositar)
  - Escreva o valor do atributo saldo na tela.
  - Realize um saque de 600 reais.
  - Escreva o valor do atributo saldo na tela.

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Exercícios



2. Crie uma classe Funcionário:
   - A classe possui os atributos nome, sobrenome, horasTrabalhadas e valorPorHora.
   - O método nomeCompleto deve retornar o atributo nome concatenado ao atributo sobrenome
   - O método calcularSalario faz o cálculo de quanto o funcionário irá receber no mês, multiplicando o atributo horasTrabalhadas pelo atributo valorPorHora e retornar esse valor.
   - O método de incrementarHoras adiciona um valor passado por parâmetro ao valor já existente no atributo valorPorHora.

- Crie uma classe TestarFuncionario, que possua o método main e instancie a classe Funcionário criada, criando um objeto novoFuncionario do tipo Funcionario
  - Atribua o valor “Luis” ao atributo nome
  - Atribua o valor “Silva” ao atributo sobrenome
  - Atribua o valor 10 ao atributo horasTrabalhadas
  - Atribua o valor 25.50 ao atributo valorPorHora
  - Escreva o nome completo do funcionário na tela, utilizando o retorno do método nomeCompleto
  - Escreva o valor do salario que o funcionario ira receber, utilizando o retorno do método calcularSalario
  - Adicione 8 ao atributo horasTrabalhadas utilizando o método incrementarHoras
  - Escreva o salario do funcionário (retorno do método calcularSalario)

Alba Lopes, Profa.
alba.lopes@ifrn.edu.br

Referências

- http://www.hardware.com.br/artigos/programacao-orientada-objetos/
- http://www.fontes.pro.br/educacional/materialpaginas/java/arquivos/jdbc/jdbc.php
- http://www.dm.ufscar.br/~waldeck/curso/java