INSTITUTO SUPERIOR POLITECNICO DE TECNOLOGIAS E CIENCIAS

1º Semestre

2025/26

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS

MISSÃO DO ISPTEC:

Formar profissionais qualificados e comprometidos com o desenvolvimento sustentável de Angola, por meio da geração e disseminação do conhecimento.

OBJECTIVO DO CURSO:

Forma profissionais com uma base sólida e de largo espectro em Engenharia Informática que permite aos licenciados dar resposta às necessidades e aos problemas observados no domínio da Informática.

OBJECTIVOS DA DISCIPLINA:

- Aprender a programar usando o paradigma orientado a objetos
- Conhecer a sintaxe e as classes mais importantes da linguagem Java e aplica-las adequadamente tendo como referencia os exemplos dados na aula

Martins, F. Mário – Java 6 e Programação Orientado pelos Objectos

Objectivos

- Aprender a programar usando o paradigma orientado a objectos

- Conhecer a sintaxe e as classes mais importantes da linguagem Java e aplica-las adequadamente tendo como referencia os exemplos dados na aula

- **Na computação, quando 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**

- **Modelagem Orientada a Objectos (OO)**
  - O ser humano conhece o mundo e gere a sua complexidade através de objectos
    - Tudo à nossa volta pode ser visto como um **objecto**.

- **Desenvolvemos o conceito de Objecto**
  - **Exemplos de objetos:** bola, carro, camisa, luz, casa, calça, música, conta bancária, poema, etc

O que é um objeto ?

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

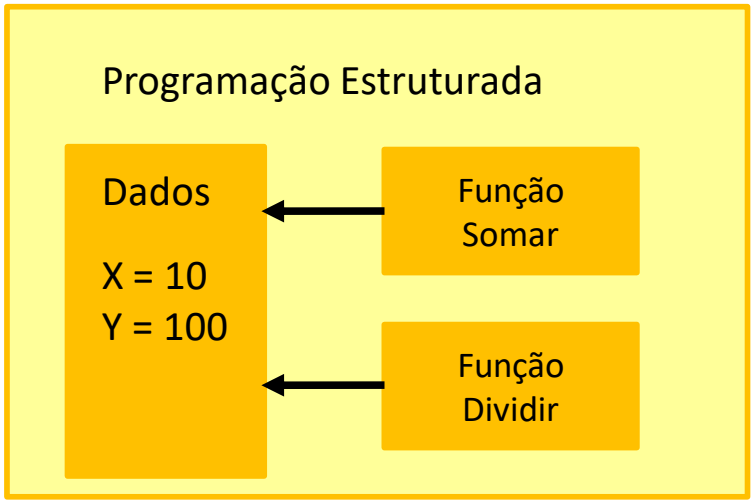
- **Exemplos**
  - O carro vermelho que está ali no estacionamento
  - Este lápis na minha mão
  - A peruca da Liliana Barbosa

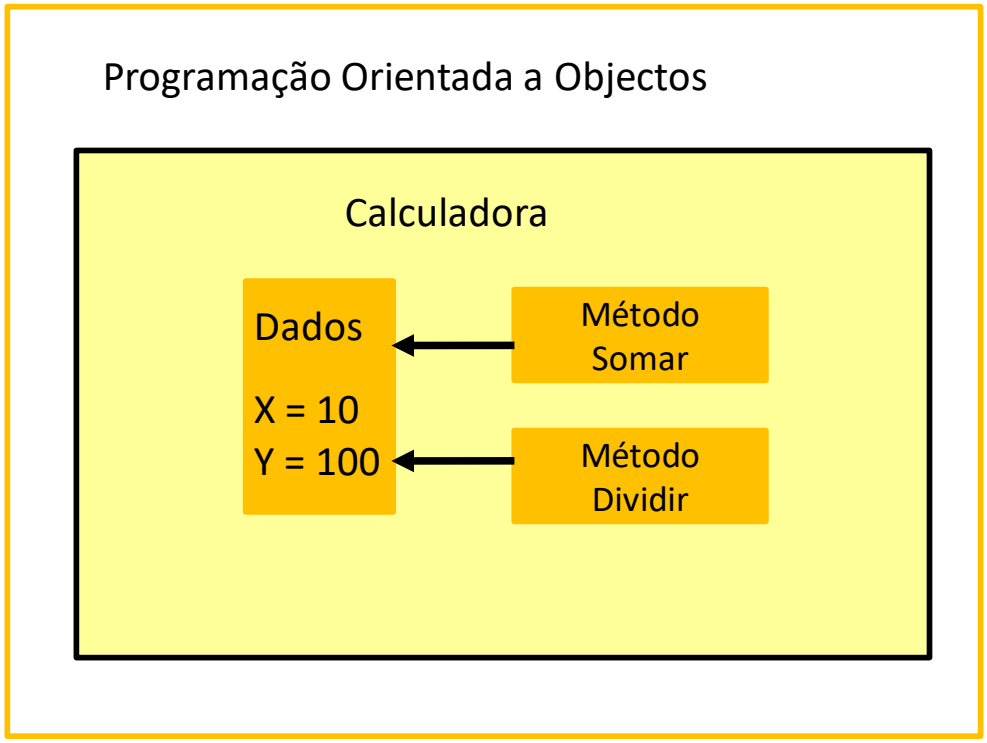
OBJETO = DADOS + OPERAÇÕES

O que é um objecto ?

- **Objetos possuem**
  - **Estado:**
    - Representados pelos valores dos **atributos/propriedades** de um objeto
  - **Comportamento:**
    - Definido pelo conjunto de **operações** (métodos) do objeto
  - **Identidade**
    - Dois objetos são distintos mesmo que os valores dos seus atributos sejam exatamente iguais

- **Calculadora**





Agrupamento

O que é uma classe?

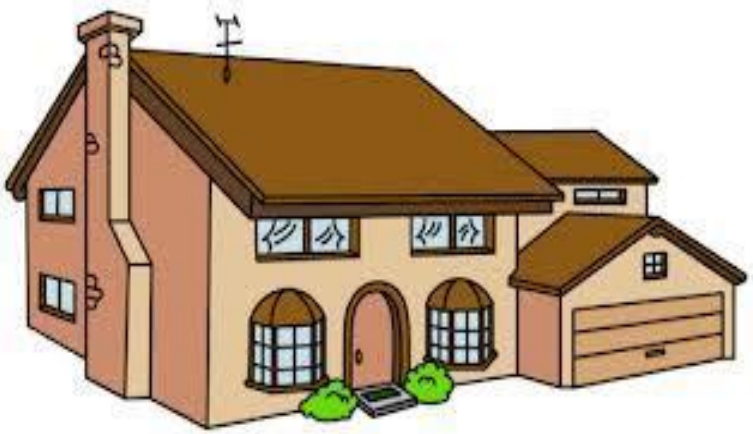
"É um projeto de um objeto. Ela informa como criar um objeto de um tipo específico."

(SIERRA & BATES, 2007)

O que é uma classe?

- **Analogia**

Classe



Objeto

Classe **VS** Objeto

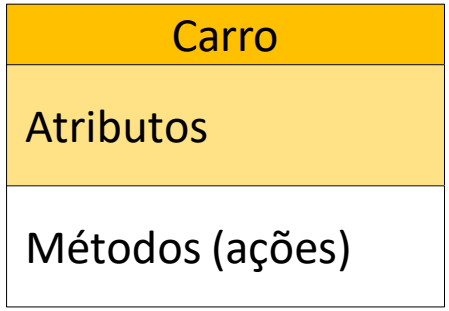
- **Classe**
  - É onde conceituamos o objeto.
  - É a essência do objeto.
  - Define os atributos e métodos.

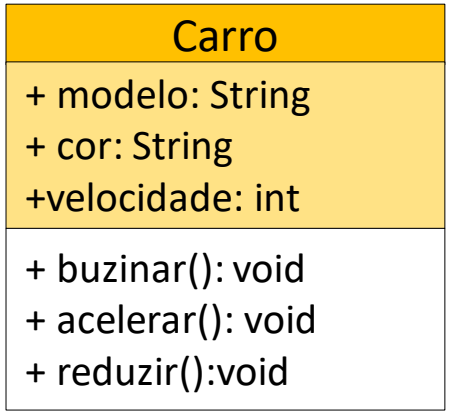
- **Objeto**
  - É a instância de uma classe.
  - Objetos semelhantes pertencem a uma mesma classe.

- **Atributos**
  - São as propriedades de um objeto
  - **Exemplo**
    - Um objeto **carro** pode ter quais propriedades?
      - Cor
      - Modelo
      - Marca
      - Potência do motor
      - Quantidade de portas
      - Velocidade actual
      - Etc..

- **Métodos**
  - São as ações que um objeto pode realizar
  - **Exemplo**
    - Um objeto **carro** pode ter que ações?
      - Acelerar
      - Travar
      - Buzinar
      - Etc..

- **Como representamos uma Classe**
  - Através da UML (Unified Modeling Language)
  - Rectângulo com três divisões:
    - Nome da Classe
    - Atributos
    - Métodos (ações)
  - **Exemplo**





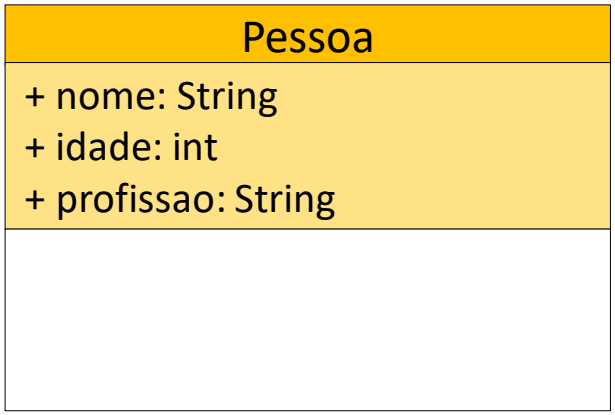
O que são instâncias ?

Chamamos de instância, cada objeto criado a partir de uma classe

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

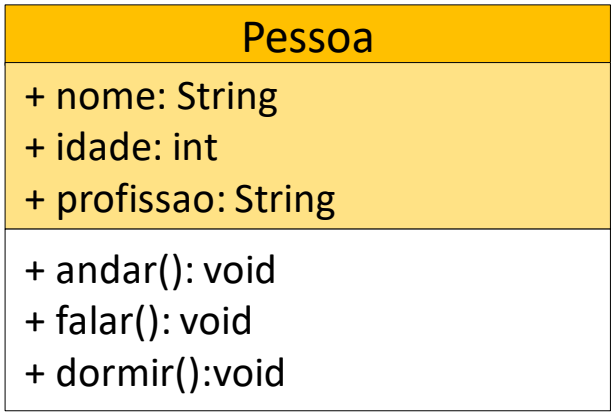
Exercício de Fixação ?

- Se definirmos a classe Pessoa, que **atributos** ela pode ter?



Exercício de Fixação ?

- E em relação aos **métodos**?



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

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

```
<DECLARAÇÃO_DA_CLASSE>{

        <MÉTODO_MAIN>{
                <VARIÁVEIS>
                <OPERAÇÕES>
        }
}
```

```
8   public class Calculadora {
9
10      public static void main(String[] args) {
11          int numero1 = 10;
12          int numero2 = 20;
13          int soma;
14          soma = numero1 + numero2;
15          System.out.println("A soma é: " + soma);
16      }
17  }
```

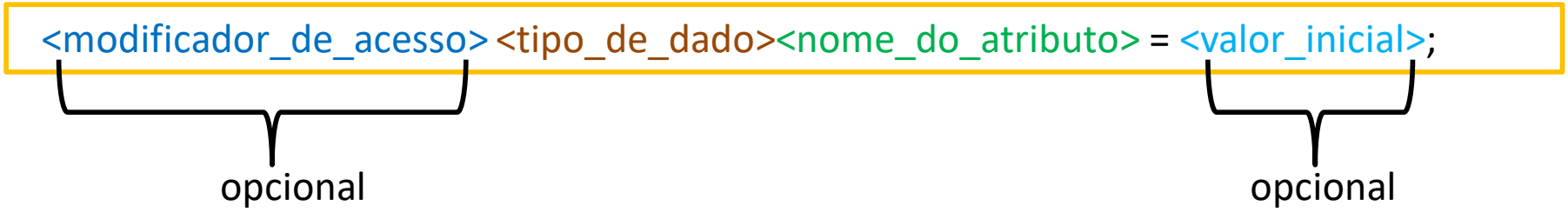
- A partir de agora, iremos abranger o nosso modo de construir as nossas classes

```
<DECLARAÇÃO_DA_CLASSE>{
                <ATRIBUTOS>
                 <MÉTODOS>
}
```

A nossa classe passará a ter mais métodos além do método main

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

- **Definição dos Atributos**
  - Os **atributos** são definidos seguindo a sintaxe abaixo



- **Exemplo**
  - String marca; (omitindo o modificador de acesso)
  - private String cor; (com modificador de acesso)
  - private int velocidade = 0; (com valor inicial)

- **Definição dos Atributos**
  - Exemplo: classe Carro

Carro
+ modelo: String + cor: String +velocidade: int

```
8  /**
9   *
10  * @author Daniel Nunes
11  */
12 public class Carro {
13
14     String modelo;
15     String cor;
16     int velocidade = 0;
17 }
18
```

- **Definição dos métodos**
  - Métodos são similares as 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 ter ou não uma lista de parâmetros.
  - Pode **retornar** algum valor ou não.
    - Caso retorne algum valor, têm de se indicar que **tipo de valor** o método retorna (int, float, String, boolean, etc)
    - Caso não retorne nenhum valor, o tipo de retorno têm de ser **void**

- **Definição dos 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>

}
```

- **Exemplo**

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

- Exemplo de código para a classe **Carro**

```
4   public class Carro {
5
6       // ATRIBUTOS
7       String marca;
8       String cor;
9       int velocidade = 0;
10
11      // MÉTODOS
12      void buzinar() {
13          System.out.println("BEEEEEEPPP...");
14      }
15
16      void acelerar() {
17          velocidade = velocidade + 1;
18      }
19
20      void reduzir() {
21          velocidade = velocidade - 1;
22      }
23  }
24
```

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

Instanciação de Objetos

- Para criar ou instanciar um objecto usamos a seguinte sintaxe

Classe nome_da_instancia = **new** **Classe**();

- **Exemplo**
  - Carro meuCarro = **new** Carro();

Instância ou objeto

Classe

O comando **new** devolve a **referencia** do objeto que foi criado

- Podemos **alterar** ou **acessar** os valores guardados nos atributos de um objeto se tivermos a **referência** a esse objeto

- Os **atributos/propriedades** são acessados pelo nome

- Para acessar um **atributo** utilizamos o operador "**.**"

```
1
2   Aluno aluno1 = new Aluno(); // Criação ou instanciação
3   Aluno aluno2; // Declaração de aluno2
4
5   aluno1.codigo ="20242324";
6   aluno1.nome = "Satoru Gojo";
7
8   //Instanciação de aluno2
9   aluno2 = new Aluno();
10
11  //Impressão dos dados do aluno1
12  System.out.println(aluno1.codigo);
13  System.out.println(aluno1.nome);
14  // ---- Fim aluno1 ---------------
15
16  //Impressão dos dados do aluno2
17  System.out.println(aluno2.codigo);
18  System.out.println(aluno2.nome);
19  // ---- Fim aluno2 ---------------
```

Construtores

- Quando utilizamos a key word **new**, estamos a construir um objeto

```
Carro meuCarro = new Carro();
```

- Quando se cria um **objecto**, normalmente queremos inicializar as suas variáveis.
- O construtor é um **método especial** que pode ser implementado em todas as **classes**, permitindo inicializar variáveis e desempenhar quaisquer outras operações quando um **objecto** é criado de outra classe.
- Ao **construtor**, é sempre dado o **mesmo nome da classe**.
- Este método especial **é usado para criar novos objetos**, opcionalmente aceitando parâmetros que este usa para definir os atributos do objeto.
- Por regra, definem-se os construtores necessários de modo a permitir inicializar um objeto da **Classe** num estado válido.
- Em Java, um construtor deve ter o mesmo da **classe** na qual ele foi definido.

Construtores

- É um método especial que permite que um trecho de código seja executado sempre que um **objeto** é criado ou seja toda vez que o operador **new** é chamado.
- Tal como os métodos, os construtores podem receber parâmetros
- Contudo, diferente dos métodos, os construtores não devolvem resposta
- Numa **classe** podemos ter mais do que um **construtor**
- Em Java, um construtor deve ter o mesmo da **classe** na qual ele foi definido

Construtores (Cont...)


```
1  public class Aluno {
2      int numero;
3      String nome;
4
5      public Aluno() {
6          this.numero = 20203056;
7          this.nome = "Lionel Messi";
8      }
9
10     public Aluno(int numero, String nome) {
11         this.numero = numero;
12         this.nome = nome;
13     }
14 }
15
```

Construtor sem parâmetros

Construtor com parâmetros

Construtores Padrão

- Sempre que criamos um **objeto**, um construtor da **classe** correspondente deve ser chamado**.**
- Mesmo quando nenhum **construtor** for definido explicitamente, há um construtor padrão que será inserido pelo próprio compilador
- O **construtor padrão** não recebe parâmetros e será chamado sempre que o programador não definir pelo menos um construtor explicitamente

```
1  class Conta {
2
3  }
4
5
6  // Invocação do construtor padrão
7  Conta c = new Conta();
```

Dúvidas