RECODE pro

POO – Programação Orientada a Objetos

A orientação a objetos, também conhecida como Programação Orientada a Objetos (POO) ou Object-Oriented Programming (OOP) **é um paradigma de análise, projeto e programação de sistemas de software** baseado na composição e interação entre diversas unidades de software chamadas de objetos.

Na programação orientada a objetos, implementa-se um conjunto de classes que definem os objetos presentes no sistema de software. Cada classe determina o comportamento (definidos nos métodos) e estados possíveis (atributos) de seus objetos, assim como o relacionamento com outros objetos.

POO – Programação Orientada a Objetos

Abstração é a habilidade de concentrar nos aspectos essenciais de um contexto qualquer, ignorando características menos importantes ou acidentais. Em modelagem orientada a objetos, uma classe é uma abstração de entidades existentes no domínio do sistema de software

POO – Programação Orientada a Objetos

Classe representa um conjunto de objetos com características afins. Uma classe define o comportamento dos objetos, através de métodos, e quais estados ele é capaz de manter, através de atributos. Exemplo de classe:

Pessoa:
Métodos: Andar(), Comer(), Estudar()
Atributos / Propriedades: Nome, RG, CPF, Telefone

POO – Programação Orientada a Objetos


```ts
class Pessoa{
    nome: string = "";
    rg: string = "";
    cpf: string = "";
    telefone: string = "";
}
```


```js
class Pessoa {
    constructor() {
        this.nome = "";
        this.rg = "";
        this.cpf = "";
        this.telefone = "";
    }
}
```

Exemplo de classes

POO – Programação Orientada a Objetos

Atributos são características de um objeto. Basicamente a estrutura de dados que vai representar a classe. Exemplos: Funcionário: nome, endereço, telefone, CPF, ...; Carro: marca, ano, cor, ...; Livro: autor, editora, ano. Por sua vez, os atributos possuem valores. Por exemplo, o atributo cor pode conter o valor azul. O conjunto de valores dos atributos de um determinado objeto é chamado de estado.

POO – Programação Orientada a Objetos


```ts
class Pessoa{
    nome: string = "";
    rg: string = "";
    cpf: string = "";
    telefone: string = "";
}
```


```js
class Pessoa {
    constructor() {
        this.nome = "";
        this.rg = "";
        this.cpf = "";
        this.telefone = "";
    }
}
```

Exemplo de atributos

POO – Programação Orientada a Objetos

Métodos definem as habilidades dos objetos. Bidu é uma instância da classe Cachorro, portanto tem habilidade para latir, implementada através do método Latir(). Um método em uma classe é apenas uma definição. A ação só ocorre quando o método é invocado através do objeto. Normalmente, uma classe possui diversos métodos, que no caso da classe Cachorro poderiam ser sentar(), comer() e morder().

POO – Programação Orientada a Objetos

```ts
class Pessoa{
    nome: string = "";
    rg: string = "";
    cpf: string = "";
    telefone: string = "";

    falar(): void {
        alert("Olá")
    }
    andar(): void{
        console.log("Andooooou!!!")
    }
}
```

TS

```js
class Pessoa {
    constructor() {
        this.nome = "";
        this.rg = "";
        this.cpf = "";
        this.telefone = "";
    }
    falar() {
        alert("Olá");
    }
    andar() {
        console.log("Andooooou!!!");
    }
}
```

JS

Exemplo de métodos

POO – Programação Orientada a Objetos

Objeto é uma instância (Cópia) de uma classe. Um objeto é capaz de armazenar estados através de seus atributos e reagir a mensagens enviadas a ele, assim como se relacionar e enviar mensagens a outros objetos. Exemplo de objetos da classe Pessoa: João, José, Maria, Flávio.

RECODE pro

POO – Programação Orientada a Objetos

TS

```ts
class Pessoa{
    nome: string = "";
    rg: string = "";
    cpf: string = "";
    telefone: string = "";

    falar(): void {
        alert("Olá")
    }
    andar(): void{
        console.log("Andooooou!!!")
    }
}

let x: Pessoa = new Pessoa();
x.falar();
x.andar();
```

JS

```js
class Pessoa {
    constructor() {
        this.nome = "";
        this.rg = "";
        this.cpf = "";
        this.telefone = "";
    }
    falar() {
        alert("Olá");
    }
    andar() {
        console.log("Andooooou!!!");
    }
}
let x = new Pessoa();
x.falar();
x.andar();
```

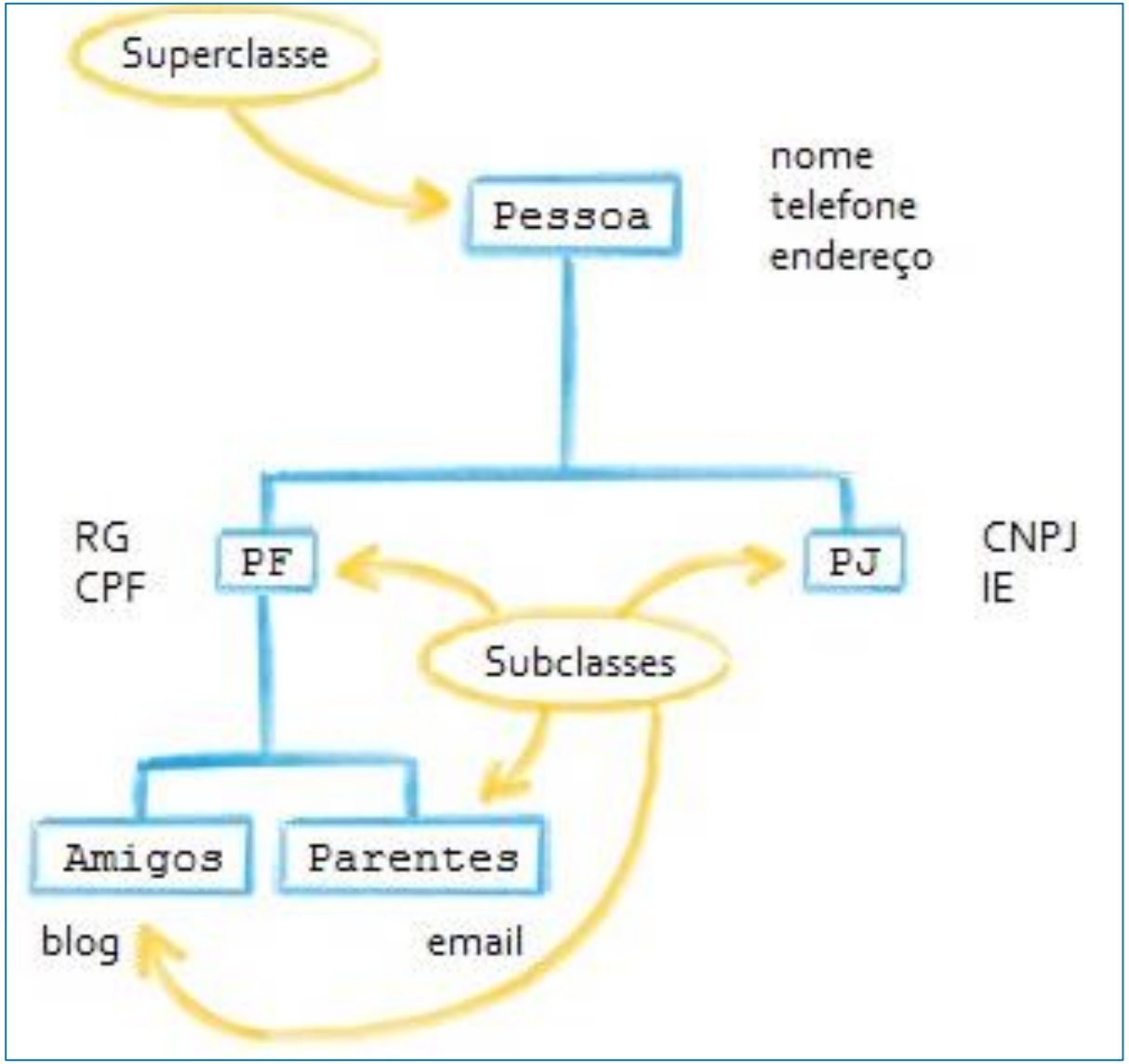
Exemplo de objetos

POO – Programação Orientada a Objetos

Herança (ou generalização) é o mecanismo pelo qual uma classe (subclasse) pode estender outra classe (superclasse), aproveitando seus comportamentos (métodos) e variáveis possíveis (atributos). Um exemplo de herança: Mamífero é superclasse de Humano. Ou seja, um Humano é um mamífero

POO – Programação Orientada a Objetos

Exemplo Herança



RECODE pro

POO – Programação Orientada a Objetos

TS

```ts
class Pessoa{
    nome: string = "";
    telefone: string = "";

    falar(): void {
        alert("Olá")
    }
}
class PF extends Pessoa {
    rg: string = "";
    cpf: string = "";
}
class PJ extends Pessoa {
    cnpj: string = "";
    ie: string = "";
}
let x: PF = new PF();
x.nome = "Flavio Mota"

let y: PJ = new PJ();
y.telefone = "36587874"
```

JS

```js
class Pessoa {
    constructor() {
        this.nome = "";
        this.telefone = "";
    }
    falar() {
        alert("Olá");
    }
}
class PF extends Pessoa {
    constructor() {
        super(...arguments);
        this.rg = "";
        this.cpf = "";
    }
}
class PJ extends Pessoa {
    constructor() {
        super(...arguments);
        this.cnpj = "";
        this.ie = "";
    }
}
let x = new PF();
x.nome = "Flavio Mota";
let y = new PJ();
y.telefone = "36587874";
```

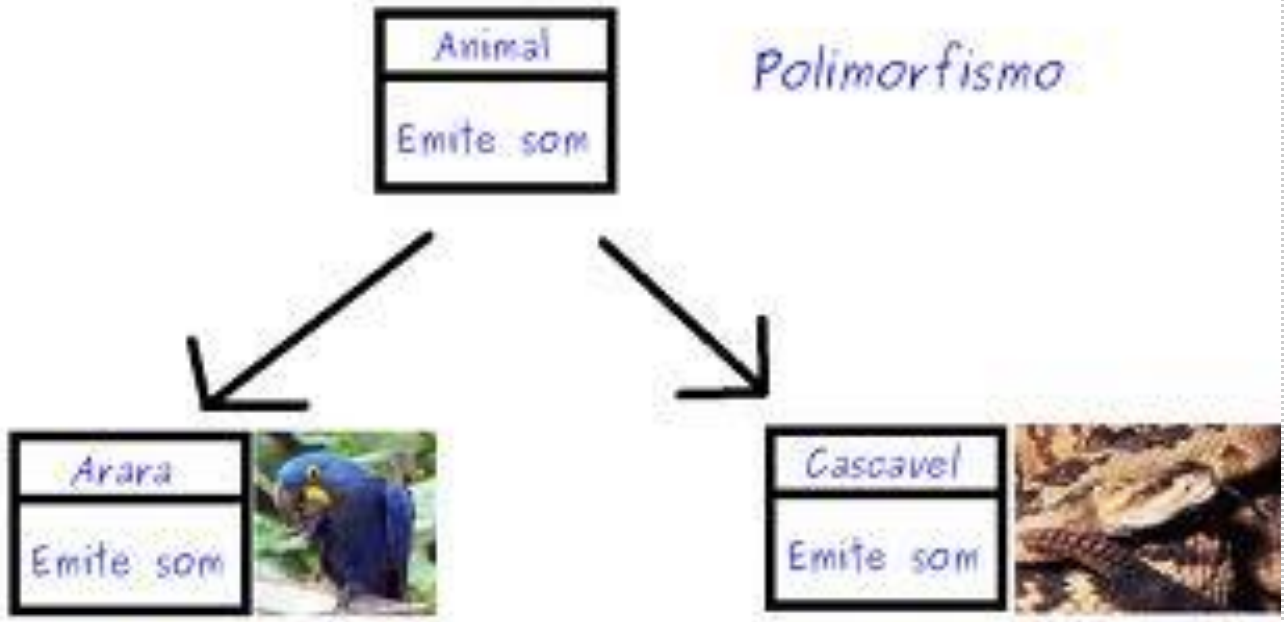
POO – Programação Orientada a Objetos

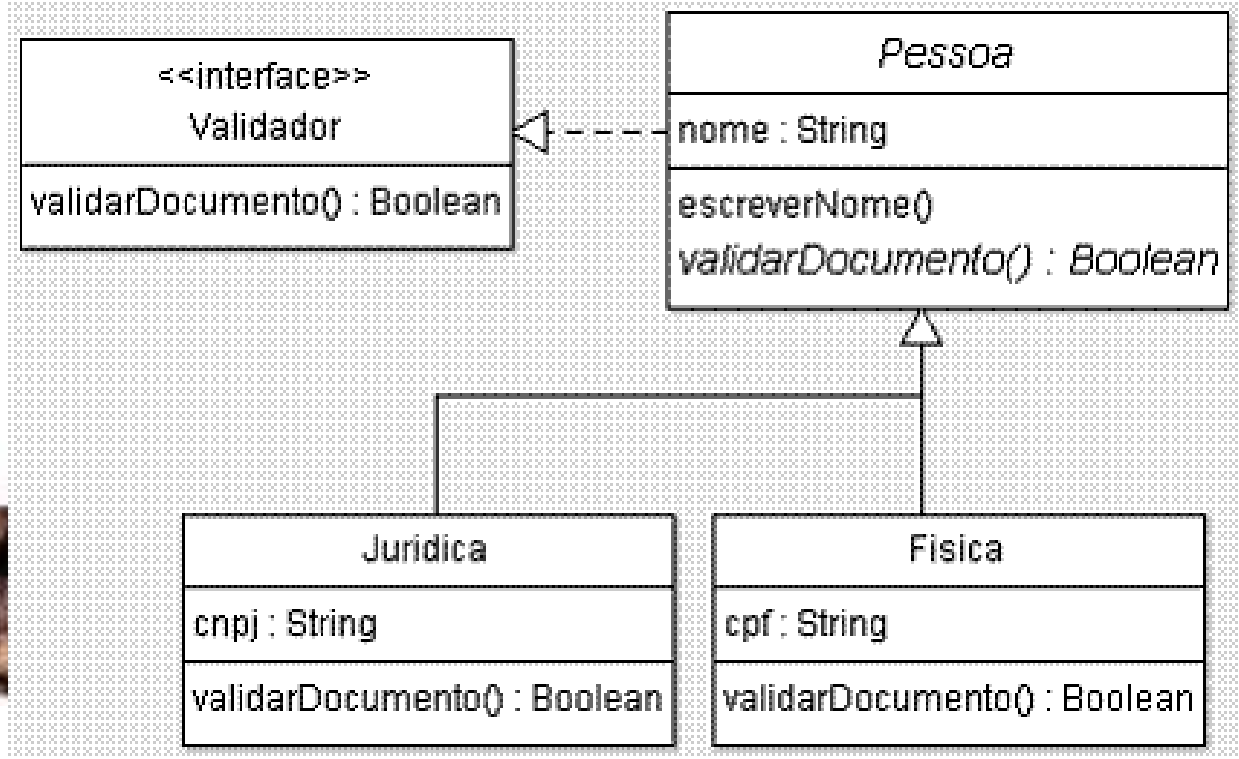
Polimorfismo é a capacidade de um método poder ser implementado de diferentes formas, ou mesmo de realizar coisas diferentes.

Sobrescrita e sobrecarga são dois tipos de **polimorfismo.**
Polimorfismo Sobrescrita - Ocorre quando uma classe filha redefine um método herdado. Os métodos têm o mesmo nome e a mesma assinatura, mas na classe filha ele está implementado de forma diferente, ou seja, ele sobrescreve o método já existente da classe pai.

Polimorfismo Sobrecarga - Ocorre quando dois ou mais métodos possuindo o mesmo nome são implementados com assinaturas diferentes, ou seja, recebem parâmetros de diferentes tipos ou em diferentes quantidades. A escolha de qual método utilizar baseia-se nos parâmetros recebidos na chamada do mesmo.

POO – Programação Orientada a Objetos





Exemplo Polimorfismo