Orientação a Objetos 1

Herança

Prof. Dr. Vinícius Camargo Andrade

vcandrade@utfpr.edu.br

Departamento Acadêmico de Informática
Universidade Tecnológica Federal do Paraná

Pilares da Orientação a Objetos

- *Abstração;*
- *Encapsulamento;*
- *Herança;*
- *Polimorfismo.*

Pilares da Orientação a Objetos

- *Abstração;*
- *Encapsulamento;*
- *Herança;*
- *Polimorfismo.*

Herança

Herança

Herança é um princípio de orientação a objetos, que permite que classes compartilhem atributos e métodos.

Herança

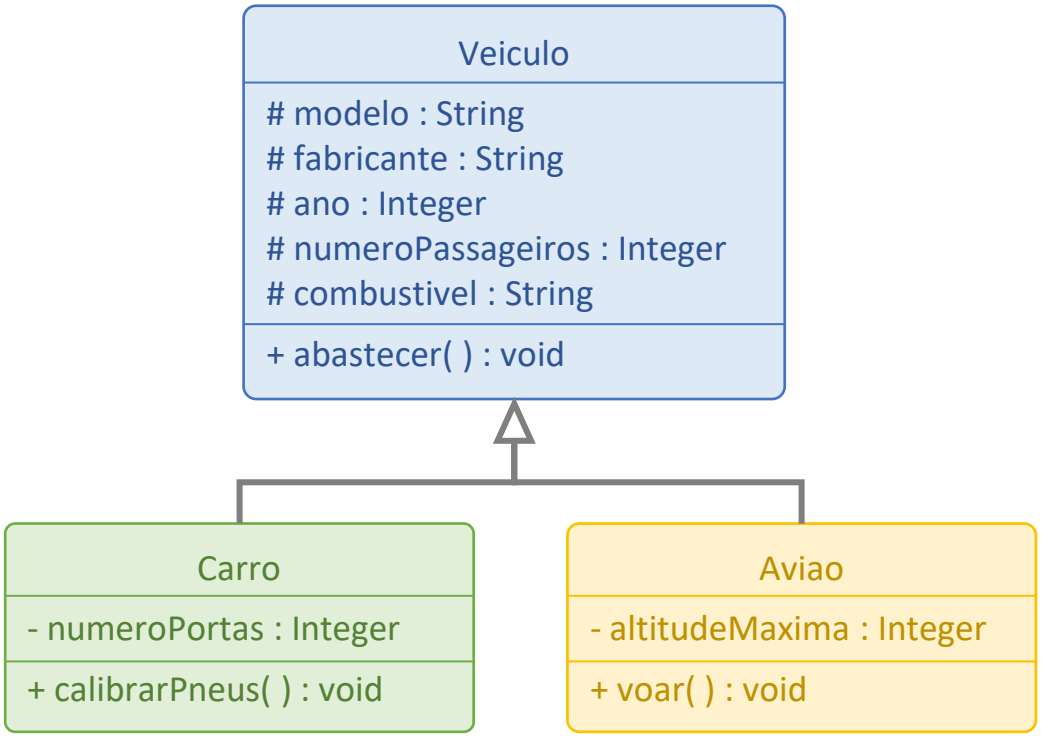
Uma subclasse pode:

- *Utilizar métodos da classe pai;*
- *Executar construtores da classe pai;*
- *Sobrepor (anular) métodos da classe (superclasse) pai de forma que objetos da classe derivada (subclasse) o utilizem de forma diferente.*
- *Implementar novos códigos nos métodos da subclasse aproveitando o código escrito na classe pai.*

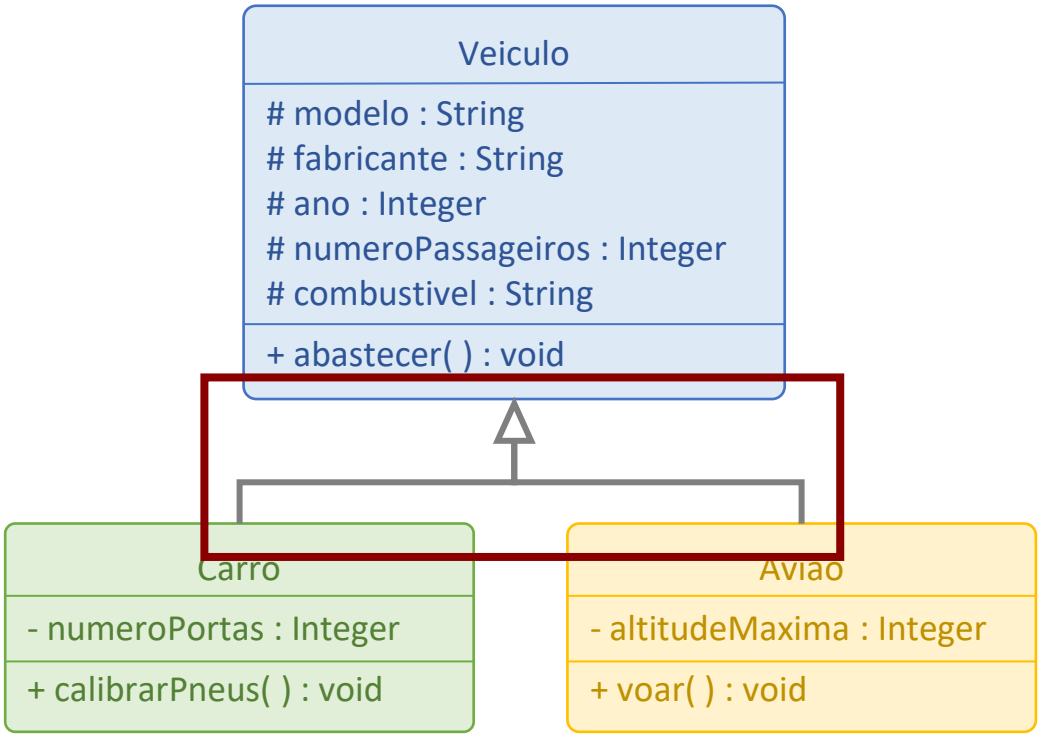
Exemplo 1

Herança

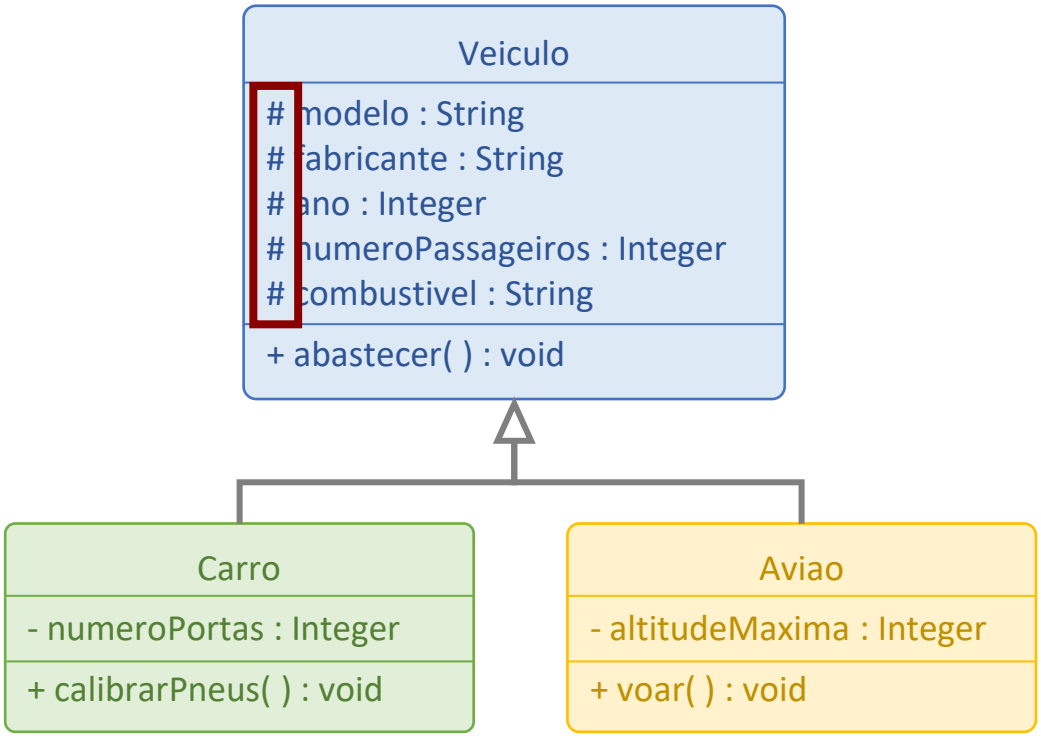
Herança


Veiculo
modelo : String
fabricante : String
ano : Integer
numeroPassageiros : Integer
combustivel : String
+ abastecer( ) : void
Carro
- numeroPortas : Integer
+ calibrarPneus( ) : void
Aviao
- altitudeMaxima : Integer
+ voar( ) : void

Herança


Veiculo
modelo : String
fabricante : String
ano : Integer
numeroPassageiros : Integer
combustivel : String
+ abastecer( ) : void
Carro
- numeroPortas : Integer
+ calibrarPneus( ) : void
Avião
- altitudeMaxima : Integer
+ voar( ) : void

Herança


Veiculo
modelo : String
fabricante : String
ano : Integer
numeroPassageiros : Integer
combustivel : String
+ abastecer( ) : void
Carro
- numeroPortas : Integer
+ calibrarPneus( ) : void
Aviao
- altitudeMaxima : Integer
+ voar( ) : void

Implementação

Exemplo 1

Implementação

Na linguagem Java, para que uma classe herde as características da outra, usa-se a palavra-chave extends na assinatura da classe.

```java
public class Veiculo {

    protected String modelo;
    protected String fabricante;
    protected Integer ano;
    protected Integer numeroPassageiros;
    protected String combustivel;

    public void abastecer() {

    }

    public String getModelo() {
        return modelo;
    }

    public void setModelo(String modelo) {
        this.modelo = modelo;
    }
```

```java
public class Carro extends Veiculo {

    private Integer numeroPortas;

    public void calibrarPneus() {

    }

    public Integer getNumeroPortas() {
        return numeroPortas;
    }

    public void setNumeroPortas(Integer numeroPortas) {
        this.numeroPortas = numeroPortas;
    }
}
```

```java
public class Aviao extends Veiculo {

    private Integer altitudeMaxima;

    public void voar() {

    }

    public Integer getAltitudeMaxima() {
        return altitudeMaxima;
    }

    public void setAltitudeMaxima(Integer altitudeMaxima) {
        this.altitudeMaxima = altitudeMaxima;
    }
}
```

```java
public class Veiculo {

    protected String modelo;
    protected String fabricante;
    protected Integer ano;
    protected Integer numeroPassageiros;
    protected String combustivel;

    public void abastecer() {

    }

    public String getModelo() {
        return modelo;
    }

    public void setModelo(String modelo) {
        this.modelo = modelo;
    }
```

```java
public class Carro extends Veiculo {

    private Integer numeroPortas;

    public void calibrarPneus() {

    }

    public Integer getNumeroPortas() {
        return numeroPortas;
    }

    public void setNumeroPortas(Integer numeroPortas) {
        this.numeroPortas = numeroPortas;
    }
}
```

```java
public class Aviao extends Veiculo {

    private Integer altitudeMaxima;

    public void voar() {

    }

    public Integer getAltitudeMaxima() {
        return altitudeMaxima;
    }

    public void setAltitudeMaxima(Integer altitudeMaxima) {
        this.altitudeMaxima = altitudeMaxima;
    }
}
```

```java
public class Veiculo {

    protected String modelo;
    protected String fabricante;
    protected Integer ano;
    protected Integer numeroPassageiros;
    protected String combustivel;

    public void abastecer() {

    }

    public String getModelo() {
        return modelo;
    }

    public void setModelo(String modelo) {
        this.modelo = modelo;
    }
```

```java
public class Carro extends Veiculo {

    private Integer numeroPortas;

    public void calibrarPneus() {

    }

    public Integer getNumeroPortas() {
        return numeroPortas;
    }

    public void setNumeroPortas(Integer numeroPortas) {
        this.numeroPortas = numeroPortas;
    }
}
```

```java
public class Aviao extends Veiculo {

    private Integer altitudeMaxima;

    public void voar() {

    }

    public Integer getAltitudeMaxima() {
        return altitudeMaxima;
    }

    public void setAltitudeMaxima(Integer altitudeMaxima) {
        this.altitudeMaxima = altitudeMaxima;
    }
}
```

Métodos Construtores

Herança

Métodos Construtores

Os métodos construtores não são herdados. Para executar o método construtor da classe pai, deve-se invocá-lo utilizando a palavra reservada super( ) na primeira linha do método construtor da classe filha.

Métodos Construtores

```java
public class Veiculo {

    protected String modelo;
    protected String fabricante;
    protected Integer ano;
    protected Integer numeroPassageiros;
    protected String combustivel;

    public Veiculo(String modelo, String fabricante, Integer ano, Integer numeroPassageiros, String combustivel) {

        this.modelo = modelo;
        this.fabricante = fabricante;
        this.ano = ano;
        this.numeroPassageiros = numeroPassageiros;
        this.combustivel = combustivel;
    }

    public void abastecer() {

        System.out.println(this.getModelo() + " está abastecendo.");
    }

    public String getModelo() {
        return modelo;
    }
```

```java
public class Carro extends Veiculo {

    private Integer numeroPortas;

    public Carro(String modelo, String fabricante, Integer ano, Integer numeroPassageiros, String combustivel, Integer numeroPortas) {

        super(modelo, fabricante, ano, numeroPassageiros, combustivel);
        this.numeroPortas = numeroPortas;
    }

    public void calibrarPneus() {

        System.out.println(this.getModelo() + " está calibrando os pneus.");
    }

    public Integer getNumeroPortas() {
        return numeroPortas;
    }
```

```java
public class Aviao extends Veiculo {

    private Integer altitudeMaxima;

    public Aviao(String modelo, String fabricante, Integer ano, Integer numeroPassageiros, String combustivel, Integer altitudeMaxima) {

        super(modelo, fabricante, ano, numeroPassageiros, combustivel);
        this.altitudeMaxima = altitudeMaxima;
    }

    public void voar() {

        System.out.println(this.getModelo() + " está voando.");
    }

    public Integer getAltitudeMaxima() {
        return altitudeMaxima;
    }
```

```java
public class Carro extends Veiculo {

    private Integer numeroPortas;

    public Carro(String modelo, String fabricante, Integer ano, Integer numeroPassageiros, String combustivel, Integer numeroPortas) {

        super(modelo, fabricante, ano, numeroPassageiros, combustivel);
        this.numeroPortas = numeroPortas;
    }

    public void calibrarPneus() {

        System.out.println(this.getModelo() + " está calibrando os pneus.");
    }

    public Integer getNumeroPortas() {
        return numeroPortas;
    }
```

```java
public class Aviao extends Veiculo {

    private Integer altitudeMaxima;

    public Aviao(String modelo, String fabricante, Integer ano, Integer numeroPassageiros, String combustivel, Integer altitudeMaxima) {

        super(modelo, fabricante, ano, numeroPassageiros, combustivel);
        this.altitudeMaxima = altitudeMaxima;
    }

    public void voar() {

        System.out.println(this.getModelo() + " está voando.");
    }

    public Integer getAltitudeMaxima() {
        return altitudeMaxima;
    }
```

```java
public class Carro extends Veiculo {

    private Integer numeroPortas;

    public Carro(String modelo, String fabricante, Integer ano, Integer numeroPassageiros, String combustivel, Integer numeroPortas) {

        super(modelo, fabricante, ano, numeroPassageiros, combustivel);
        this.numeroPortas = numeroPortas;
    }

    public void calibrarPneus() {

        System.out.println(this.getModelo() + " está calibrando os pneus.");
    }

    public Integer getNumeroPortas() {
        return numeroPortas;
    }
```

```java
public class Aviao extends Veiculo {

    private Integer altitudeMaxima;

    public Aviao(String modelo, String fabricante, Integer ano, Integer numeroPassageiros, String combustivel, Integer altitudeMaxima) {

        super(modelo, fabricante, ano, numeroPassageiros, combustivel);
        this.altitudeMaxima = altitudeMaxima;
    }

    public void voar() {

        System.out.println(this.getModelo() + " está voando.");
    }

    public Integer getAltitudeMaxima() {
        return altitudeMaxima;
    }
```

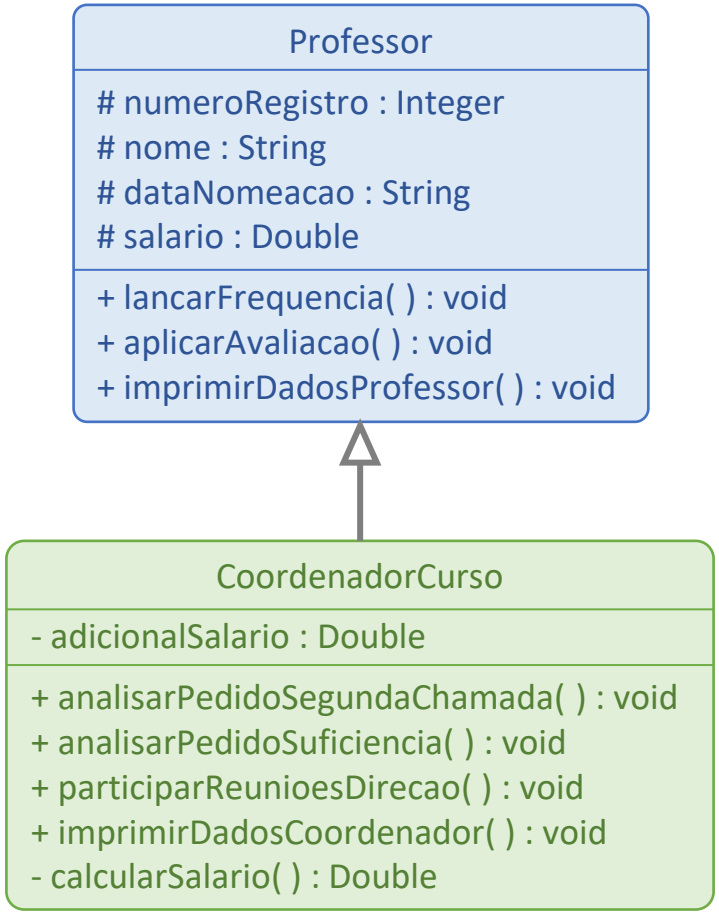
```java
public class VeiculoTeste {

    public static void main(String[] args) {

        Carro carro1 = new Carro("Gol", "Volks", 2015, 5, "Flex", 5);
        carro1.abastecer();
        carro1.calibrarPneus();

        Aviao aviao1 = new Aviao("Airbus A380-800", "EADS Airbus", 853, 2004, "Combustível de Aviação", 43100);
        aviao1.abastecer();
        aviao1.voar();
    }
}
```

```
run:
Gol está abastecendo.
Gol está calibrando os pneus.
Airbus A380-800 está abastecendo.
Airbus A380-800 está voando.
BUILD SUCCESSFUL (total time: 0 seconds)
```

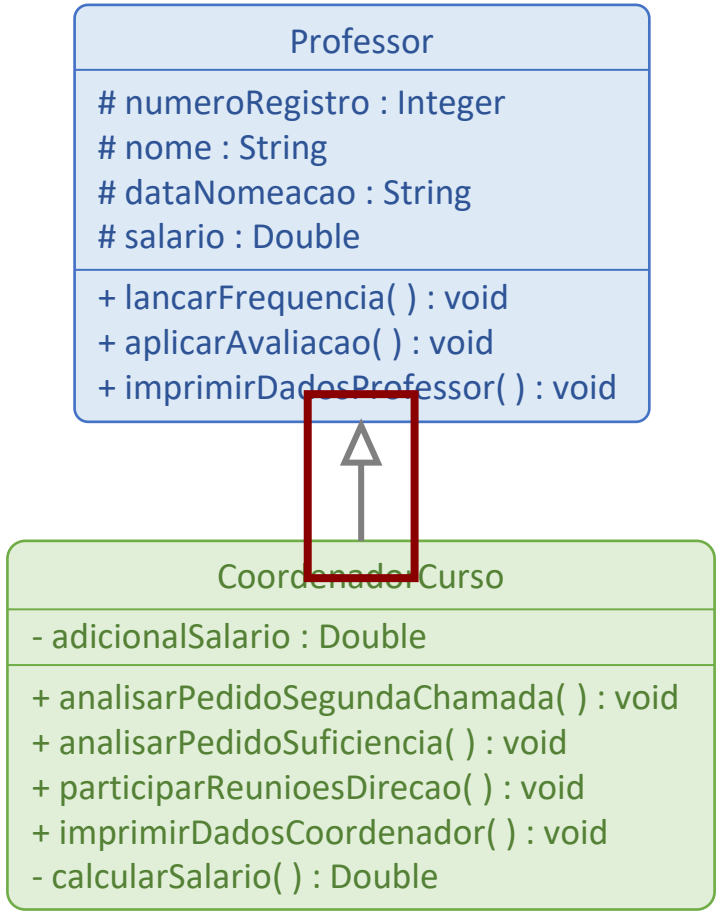
Exemplo 2

Herança

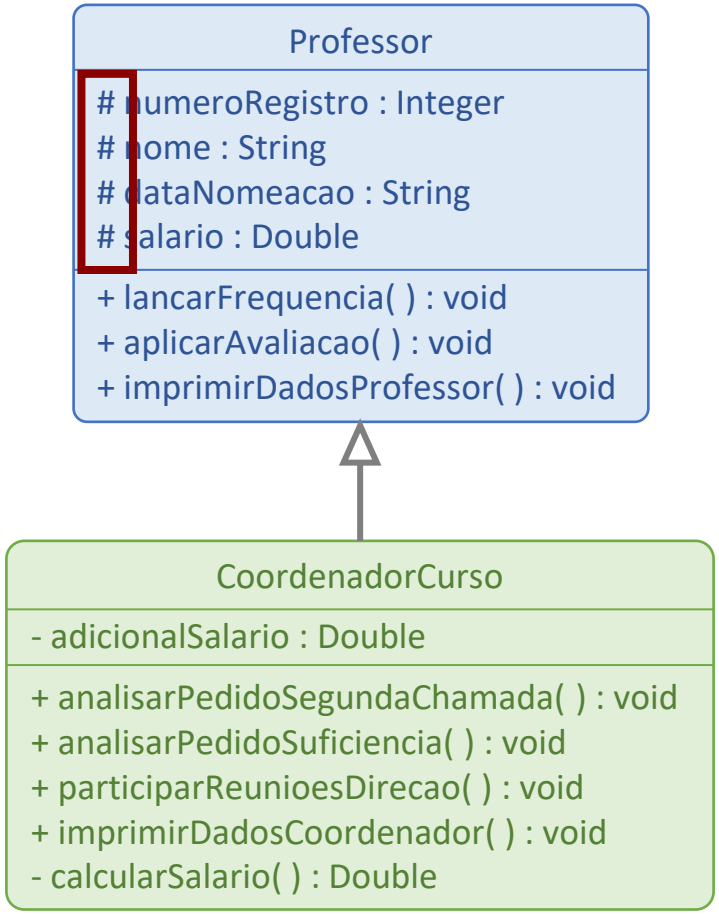
Herança


Professor
numeroRegistro : Integer
nome : String
dataNomeacao : String
salario : Double
+ lancarFrequencia( ) : void
+ aplicarAvaliacao( ) : void
+ imprimirDadosProfessor( ) : void
CoordenadorCurso
- adicionalSalario : Double
+ analisarPedidoSegundaChamada( ) : void
+ analisarPedidoSuficiencia( ) : void
+ participarReunioesDirecao( ) : void
+ imprimirDadosCoordenador( ) : void
- calcularSalario( ) : Double

Herança


Professor
numeroRegistro : Integer
nome : String
dataNomeacao : String
salario : Double
+ lancarFrequencia( ) : void
+ aplicarAvaliacao( ) : void
+ imprimirDadosProfessor( ) : void
CoordenadorCurso
- adicionalSalario : Double
+ analisarPedidoSegundaChamada( ) : void
+ analisarPedidoSuficiencia( ) : void
+ participarReunioesDirecao( ) : void
+ imprimirDadosCoordenador( ) : void
- calcularSalario( ) : Double

Herança


Professor
numeroRegistro : Integer
nome : String
dataNomeacao : String
salario : Double
+ lancarFrequencia( ) : void
+ aplicarAvaliacao( ) : void
+ imprimirDadosProfessor( ) : void
CoordenadorCurso
- adicionalSalario : Double
+ analisarPedidoSegundaChamada( ) : void
+ analisarPedidoSuficiencia( ) : void
+ participarReunioesDirecao( ) : void
+ imprimirDadosCoordenador( ) : void
- calcularSalario( ) : Double

Implementação

Exemplo 2

Herança

```java
public class Professor {

    protected int numeroRegistro;
    protected String nome;
    protected String dataNomeacao;
    protected double salario;

    public Professor(int numeroRegistro, String nome, String dataNomeacao, double salario) {

        this.numeroRegistro = numeroRegistro;
        this.nome = nome;
        this.dataNomeacao = dataNomeacao;
        this.salario = salario;
    }

    public void lancarFrequencia() {

        System.out.println(this.nome + " está lançando frequência.");
    }

    public void aplicarAvaliacao() {

        System.out.println(this.nome + " está aplicando avaliação.");
    }

    public void imprimirDadosProfessor() {

        System.out.println("Número Registro: " + this.numeroRegistro);
        System.out.println("Nome: " + this.nome);
        System.out.println("Data da Nomeação: " + this.dataNomeacao);
        System.out.println("Salário: R$" + this.salario);
    }
}
```

Herança

```java
public class Professor {

    protected int numeroRegistro;
    protected String nome;
    protected String dataNomeacao;
    protected double salario;

    public Professor(int numeroRegistro, String nome, String dataNomeacao, double salario) {

        this.numeroRegistro = numeroRegistro;
        this.nome = nome;
        this.dataNomeacao = dataNomeacao;
        this.salario = salario;
    }

    public void lancarFrequencia() {

        System.out.println(this.nome + " está lançando frequência.");
    }

    public void aplicarAvaliacao() {

        System.out.println(this.nome + " está aplicando avaliação.");
    }

    public void imprimirDadosProfessor() {

        System.out.println("Número Registro: " + this.numeroRegistro);
        System.out.println("Nome: " + this.nome);
        System.out.println("Data da Nomeação: " + this.dataNomeacao);
        System.out.println("Salário: R$" + this.salario);
    }
}
```

Herança

```java
public class CoordenadorCurso extends Professor {

    private double adicionalSalario;

    public CoordenadorCurso(int numeroRegistro, String nome, String dataNomeacao, double salario, double adicionalSalario) {

        super(numeroRegistro, nome, dataNomeacao, salario);
        this.adicionalSalario = adicionalSalario;
    }

    public void analisarPedidoSegundaChamada() {
        System.out.println(this.nome + " está analisando pedidos de avaliações de 2a chamada.");
    }

    public void analisarPedidoSuficiencia() {
        System.out.println(this.nome + " está deferindo pedidos de exames de suficiência.");
    }

    public void participarReunioesDirecao() {
        System.out.println(this.nome + " está participando das reuniões da direção do campus.");
    }

    public void imprimirDadosCoordenador() {

        super.imprimirDadosProfessor();

        System.out.println("Adicional: R$" + this.adicionalSalario);
        System.out.println("Salário Total: R$" + this.calcularSalario());
    }

    private double calcularSalario() {
        return this.salario + this.adicionalSalario;
    }
}
```

Herança

```java
public class CoordenadorCurso extends Professor {

    private double adicionalSalario;

    public CoordenadorCurso(int numeroRegistro, String nome, String dataNomeacao, double salario, double adicionalSalario) {

        super(numeroRegistro, nome, dataNomeacao, salario);
        this.adicionalSalario = adicionalSalario;
    }

    public void analisarPedidoSegundaChamada() {
        System.out.println(this.nome + " está analisando pedidos de avaliações de 2a chamada.");
    }

    public void analisarPedidoSuficiencia() {
        System.out.println(this.nome + " está deferindo pedidos de exames de suficiência.");
    }

    public void participarReunioesDirecao() {
        System.out.println(this.nome + " está participando das reuniões da direção do campus.");
    }

    public void imprimirDadosCoordenador() {

        super.imprimirDadosProfessor();

        System.out.println("Adicional: R$" + this.adicionalSalario);
        System.out.println("Salário Total: R$" + this.calcularSalario());
    }

    private double calcularSalario() {
        return this.salario + this.adicionalSalario;
    }
}
```

Herança

```java
public class CoordenadorCurso extends Professor {

    private double adicionalSalario;

    public CoordenadorCurso(int numeroRegistro, String nome, String dataNomeacao, double salario, double adicionalSalario) {

        super(numeroRegistro, nome, dataNomeacao, salario);
        this.adicionalSalario = adicionalSalario;
    }

    public void analisarPedidoSegundaChamada() {
        System.out.println(this.nome + " está analisando pedidos de avaliações de 2a chamada.");
    }

    public void analisarPedidoSuficiencia() {
        System.out.println(this.nome + " está deferindo pedidos de exames de suficiência.");
    }

    public void participarReunioesDirecao() {
        System.out.println(this.nome + " está participando das reuniões da direção do campus.");
    }

    public void imprimirDadosCoordenador() {

        super.imprimirDadosProfessor();

        System.out.println("Adicional: R$" + this.adicionalSalario);
        System.out.println("Salário Total: R$" + this.calcularSalario());
    }

    private double calcularSalario() {
        return this.salario + this.adicionalSalario;
    }
}
```

Herança

```java
public class ProfessorTeste {

    public static void main(String[] args) {

        Professor professor1 = new Professor(111, "João da Silva", "01/01/2010", 2000.00);
        professor1.imprimirDadosProfessor();
        professor1.aplicarAvaliacao();
        professor1.lancarFrequencia();

        System.out.println("====================================");

        CoordenadorCurso coordenadorCurso1 = new CoordenadorCurso(222, "Maria de Oliveira", "02/02/2020", 2000.00, 1000.00);
        coordenadorCurso1.imprimirDadosCoordenador();
        coordenadorCurso1.aplicarAvaliacao();
        coordenadorCurso1.lancarFrequencia();
        coordenadorCurso1.analisarPedidoSegundaChamada();
        coordenadorCurso1.analisarPedidoSuficiencia();
        coordenadorCurso1.participarReunioesDirecao();
    }
}
```

Herança

```
public class ProfessorTeste {

    public static void main(String[] args) {

        Professor professor1 = new Professor(111, "João da Silva", "01/01/2010", 2000.00);
        professor1.imprimirDadosProfessor();
        professor1.aplicarAvaliacao();
        professor1.lancarFrequencia();

        System.out.println("====================================");

        CoordenadorCurso coordenadorCurso1 = new CoordenadorCurso(222, "Maria de Oliveira", "02/02/2020", 2000.00, 1000.00);
        coordenadorCurso1.imprimirDadosCoordenador();
        coordenadorCurso1.aplicarAvaliacao();
        coordenadorCurso1.lancarFrequencia();
        coordenadorCurso1.analisarPedidoSegundaChamada();
        coordenadorCurso1.analisarPedidoSuficiencia();
        coordenadorCurso1.participarReunioesDirecao();
    }
}
```

Herança

```java
public class ProfessorTeste {

    public static void main(String[] args) {

        Professor professor1 = new Professor(111, "João da Silva", "01/01/2010", 2000.00);
        professor1.imprimirDadosProfessor();
        professor1.aplicarAvaliacao();
        professor1.lancarFrequencia();

        System.out.println("====================================");

        CoordenadorCurso coordenadorCurso1 = new CoordenadorCurso(222, "Maria de Oliveira", "02/02/2020", 2000.00, 1000.00);
        coordenadorCurso1.imprimirDadosCoordenador();
        coordenadorCurso1.aplicarAvaliacao();
        coordenadorCurso1.lancarFrequencia();
        coordenadorCurso1.analisarPedidoSegundaChamada();
        coordenadorCurso1.analisarPedidoSuficiencia();
        coordenadorCurso1.participarReunioesDirecao();
    }
}
```

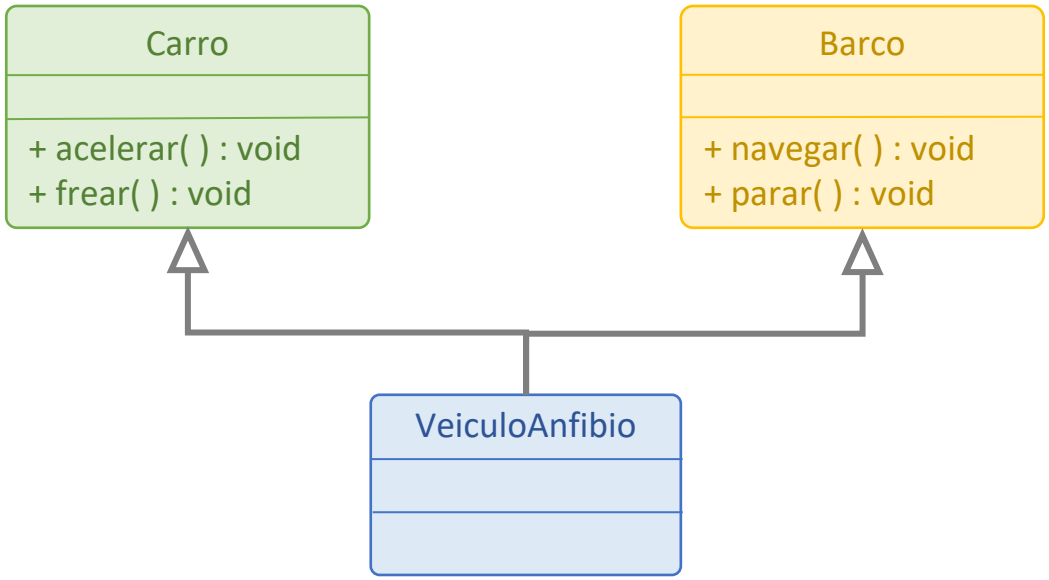
Herança Múltipla

Herança

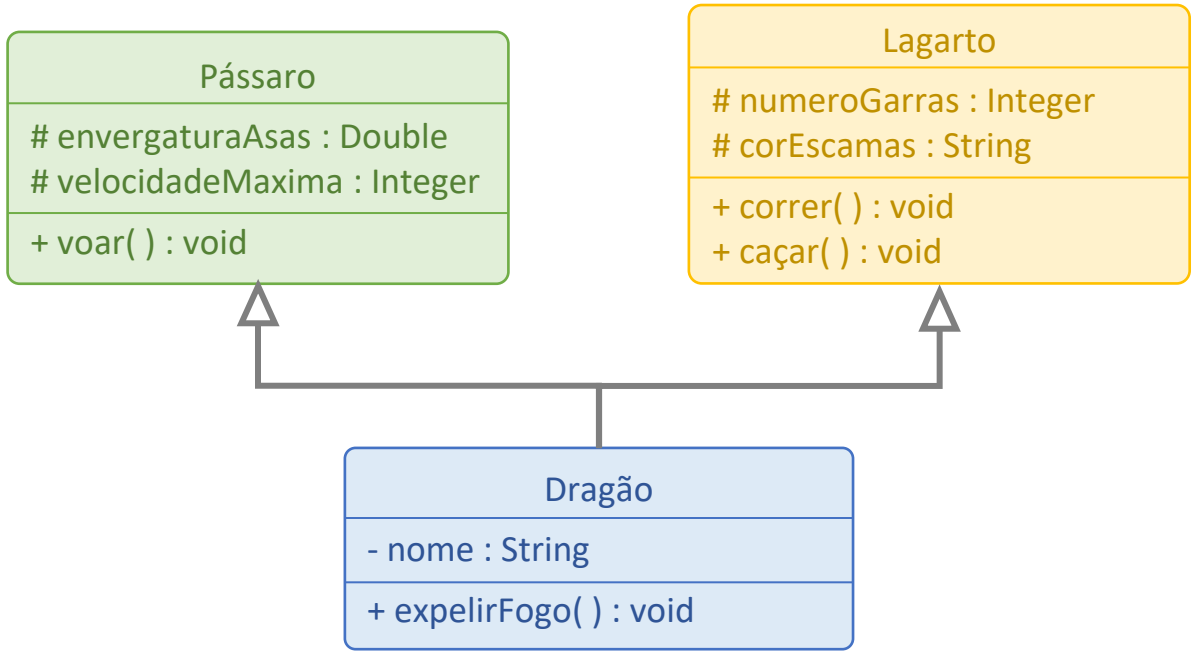
Herança Múltipla

Criação de uma subclasse a partir de mais de uma classe, herdando as características de todas elas.

Herança Múltipla



Herança Múltipla



Implementação em Java

Java não implementa herança múltipla!

Como saber quando a Relação é Herança?

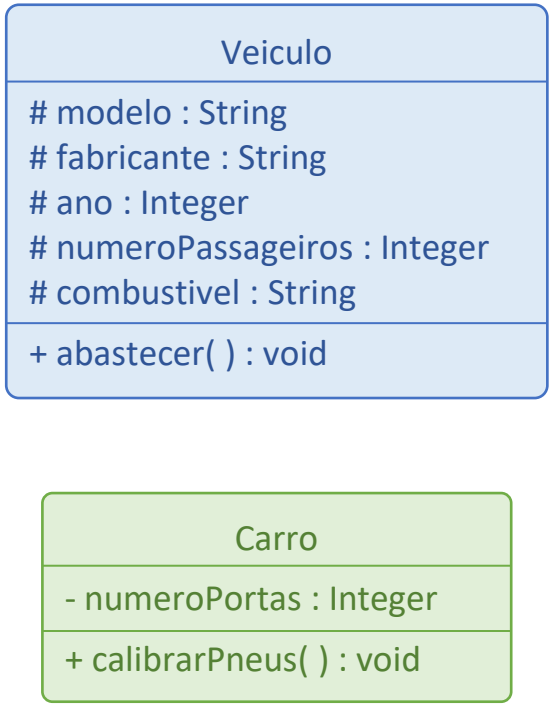
Herança?

Faça a seguinte pergunta:

Classe A é um tipo de Classe B?

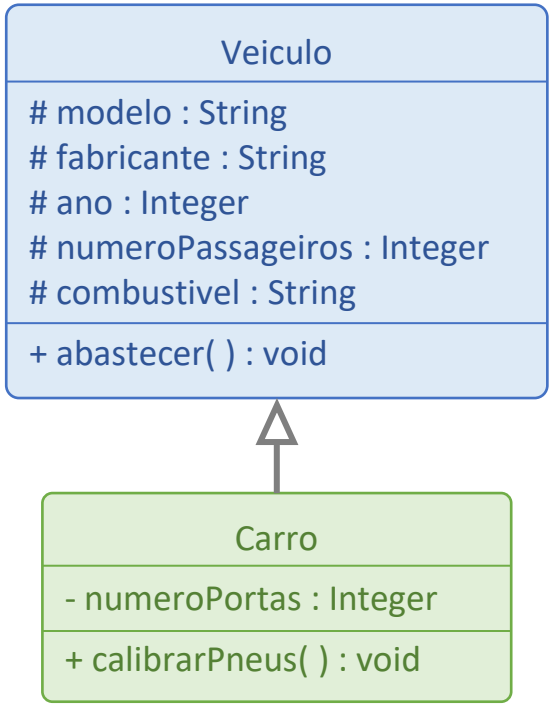
Herança?

Carro é um tipo de Veículo?



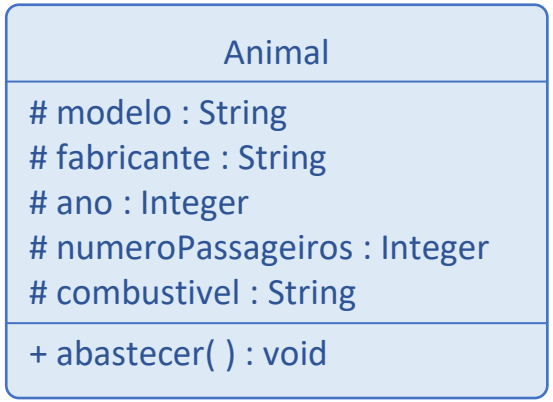
Herança?

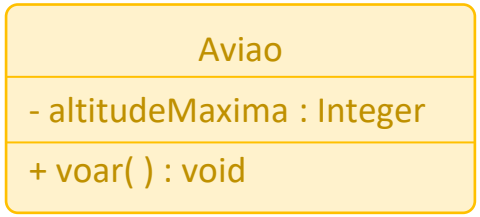
Carro é um tipo de Veículo? Sim.



Herança?

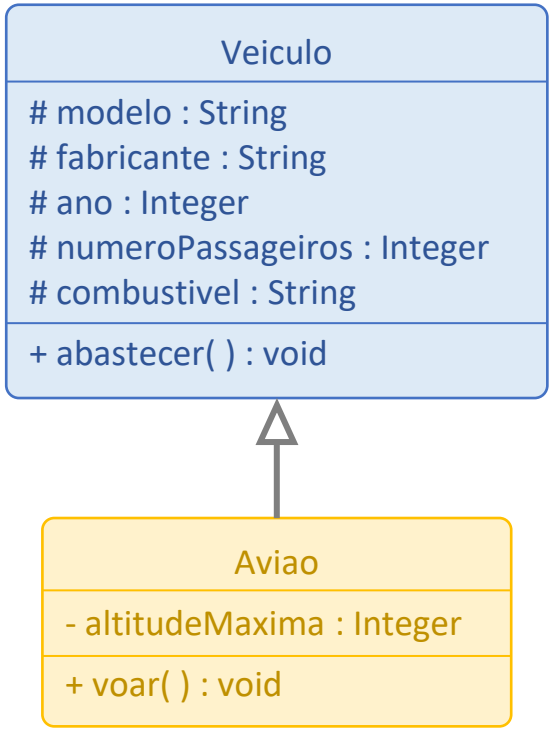
Avião é um tipo de Veículo?





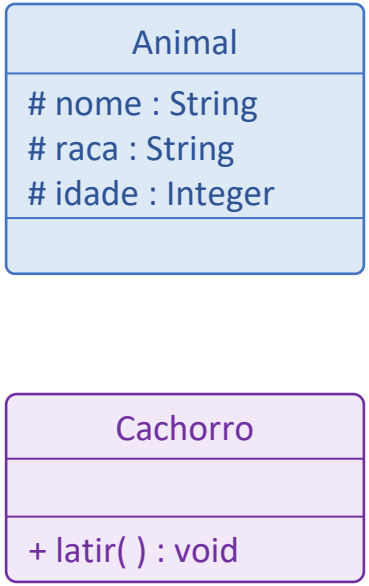
Herança?

Avião é um tipo de Veículo? Sim.



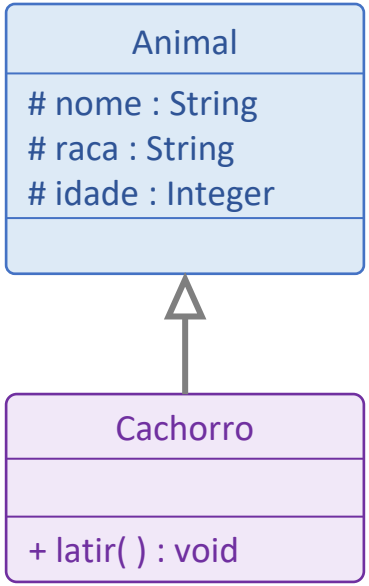
Herança?

Cachorro é um tipo de Animal?



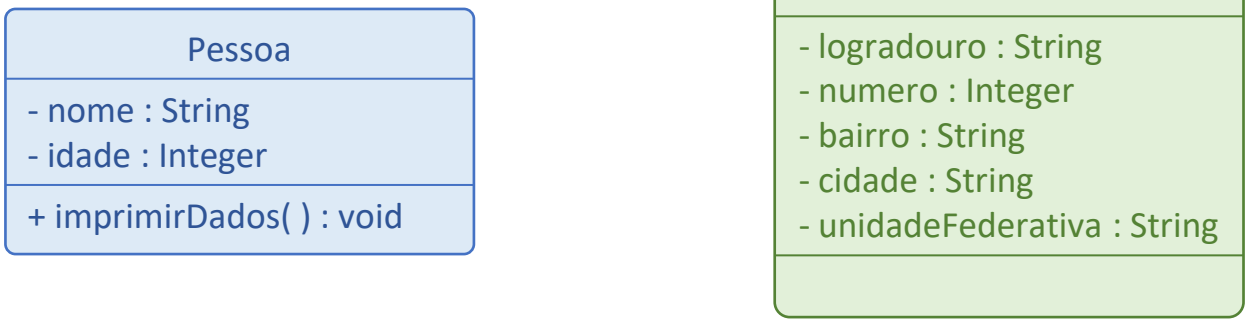
Herança?

Cachorro é um tipo de Animal? Sim.



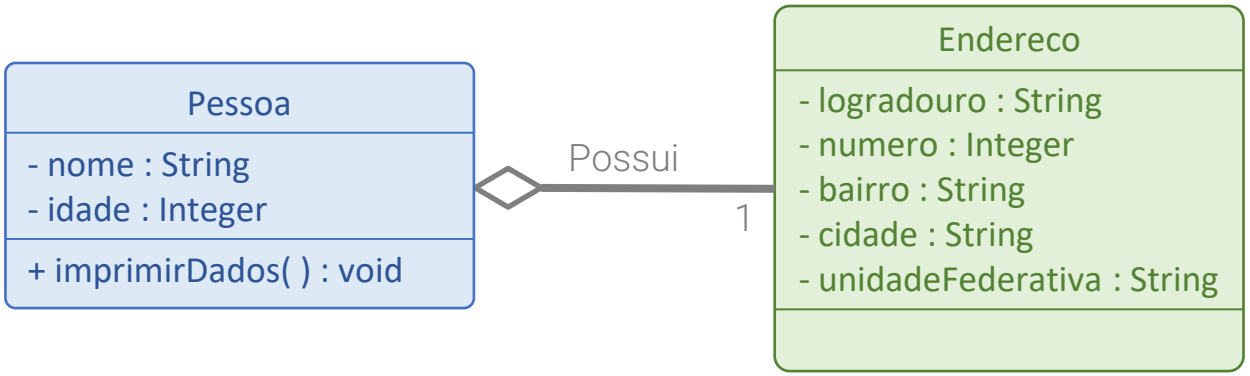
Herança?

Pessoa é um tipo de Endereço?



Herança?

Pessoa é um tipo de Endereço? Não.



Generalização

Herança

Generalização

Generalização é o ato de tornar um objeto geral (abstrair), agrupar características comuns para objetos dentro de um mesmo contexto.

Especialização

Herança

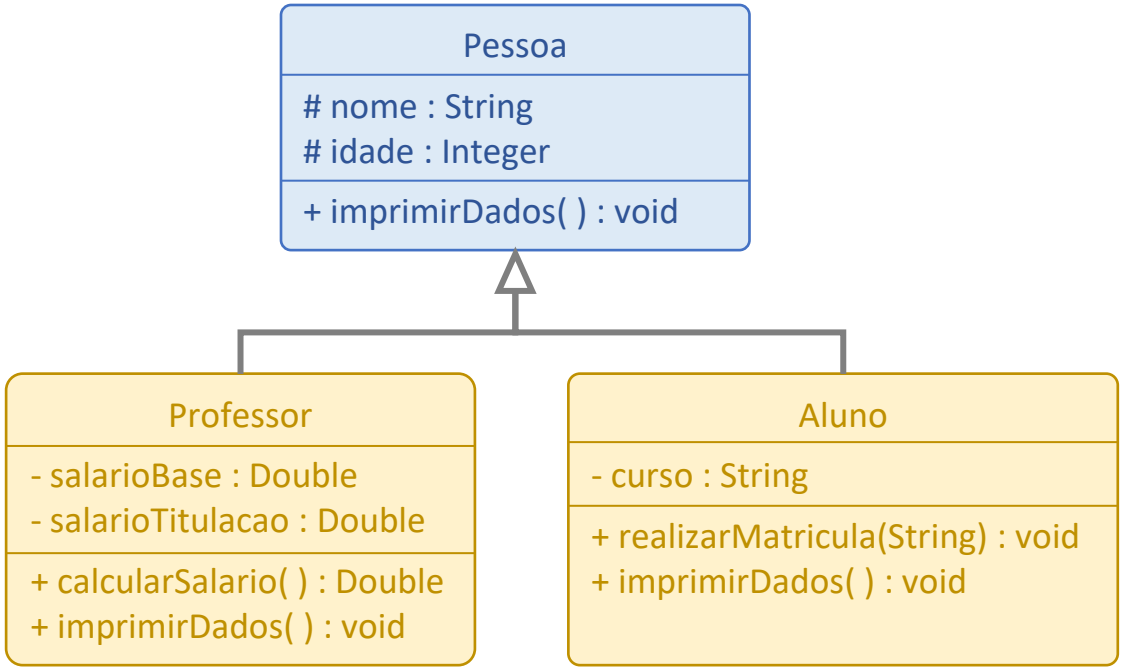
Especialização

Especialização nada mais é do que a parte que "especializa" o objeto vindo de uma Generalização, trazer características próprias para o objeto.

Exercício 1

Exercício 1

Codifique o exemplo do modelo abaixo com seus respectivos métodos construtores:



Exercício 1

- *Classe Professor*
  - *calcularSalario( ): retorna o resultado da soma o salário base e o salário titulação.*
  - *imprimirDados( ): imprime todos os dados de professor, inclusive o salário calculado.*
- *Classe Aluno:*
  - *realizarMatricula( ): recebe o curso (String) que o aluno será matriculado e então o curso é armazenado na variável correspondente.*
  - *imprimirDados( ): imprime todos os dados de aluno, inclusive o curso que o mesmo está matriculado.*

Exercício 2

Exercício 2

E se tanto o professor quanto o aluno possuírem um endereço?