

JAVA

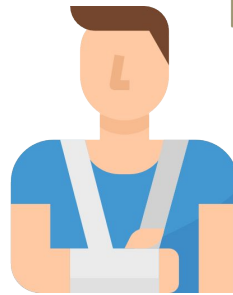
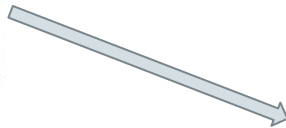
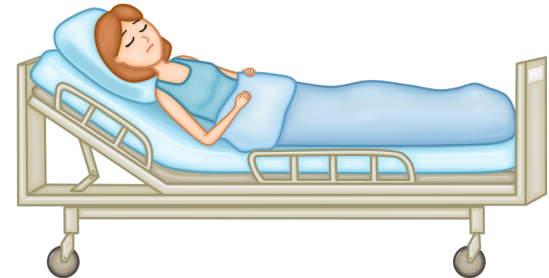
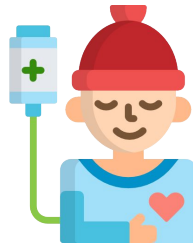
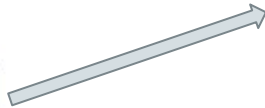
ORIENTAÇÃO A OBJETOS



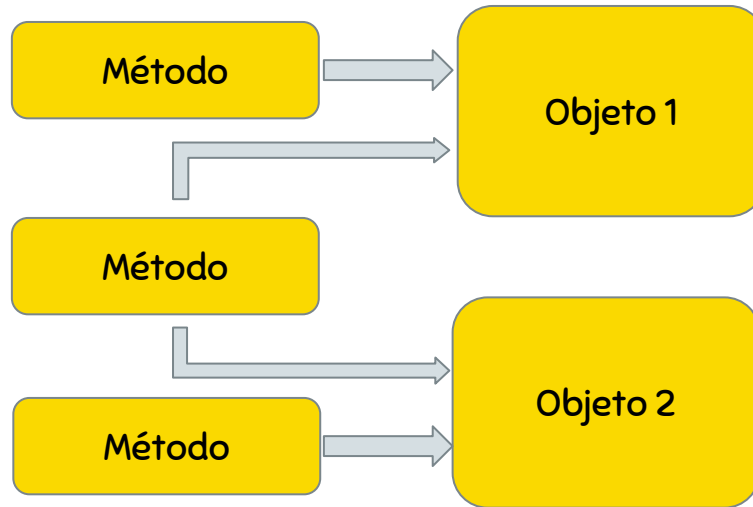
PROGRAMAÇÃO ORIENTADA A OBJETOS

- × Pensamento humano
- × Sequência lógica mas pensando em modelar problemas de uma forma mais natural e parecido com o mundo real

PILARES DA ORIENTAÇÃO A OBJETOS



PROGRAMAÇÃO ORIENTADA A OBJETO

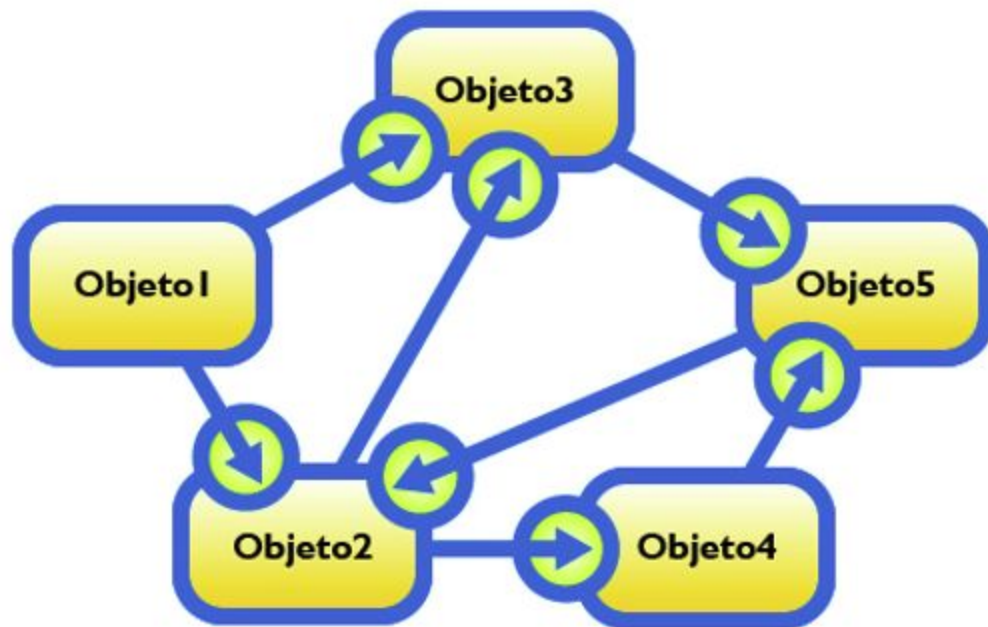


PROGRAMAÇÃO ESTRUTURADA

1	Desligar o interruptor
2	Pegar uma escada
3	Montar a escada
4	Subir na escada
5	Desenroscar a lâmpada queimada
6	Descer da escada
7	Jogar a lâmpada queimada no lixo
8	Pegar uma lâmpada nova
9	Subir na escada
10	Rosquear a nova lâmpada
11	Descer da escada
12	Ligar o interruptor para verificar se a nova lâmpada acende

PROGRAMAÇÃO ORIENTADA A OBJETO

Objetos	Características
Pessoa	Pegar, descer, subir, jogar
Lâmpada	Inteira, quebrada, ligada, desligada





CARACTERÍSTICAS

(dados, atributos)

Tipo: Ferrari

Cor: Vermelho

Placa: KZE-1018

Número de portas: 2

COMPORTAMENTOS

(operações, métodos)

Ligar

Desligar

Acelerar

Frear



CARACTERÍSTICAS

(dados, atributos)

Tipo: Camila

Cor do cabelo: Negro

Biotipo: Magro

COMPORTAMENTOS

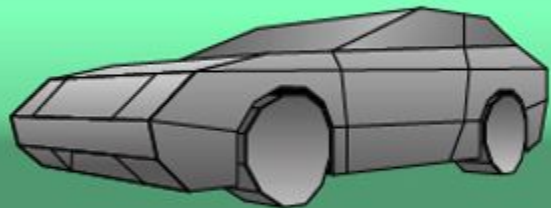
(operações, métodos)

Andar

Correr

Dirigir carro

Objetos



Tipo: ?

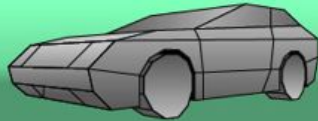
Cor: ?

Placa: ?

Número de portas: ?

Classes

CLASSE



Tipo: ?
Cor: ?
Placa: ?
Número de portas: ?

OBJETOS



Tipo: Porsche
Cor: Branco
Placa: MHZ-4345
Número de portas: 4



Tipo: Ferrari
Cor: Vermelho
Placa: JKL-0001
Número de portas: 2

Conceito de
POO

HAM ?



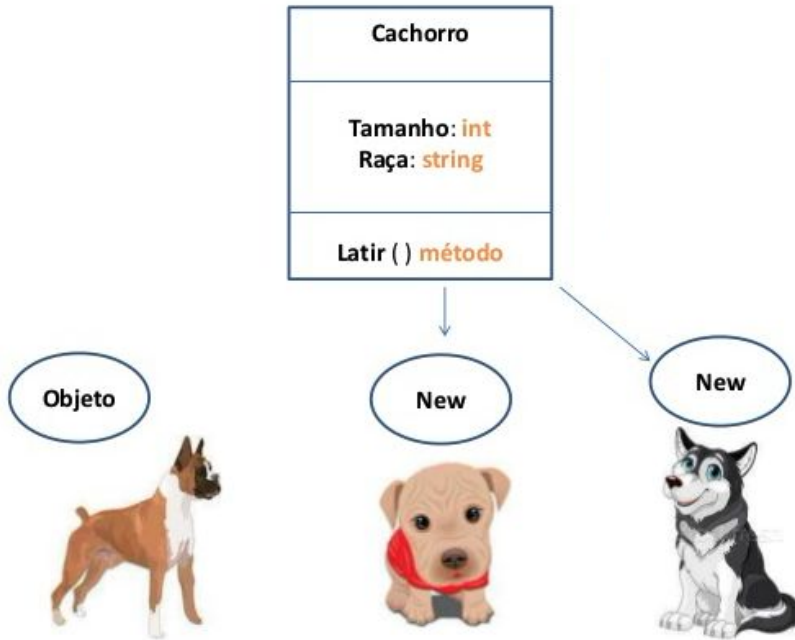


Cachorro
Tamanho: <i>int</i> Raça: <i>string</i>
Latir () <i>método</i>

Tipo
Classe

Atributos
Variáveis

Ações
Métodos

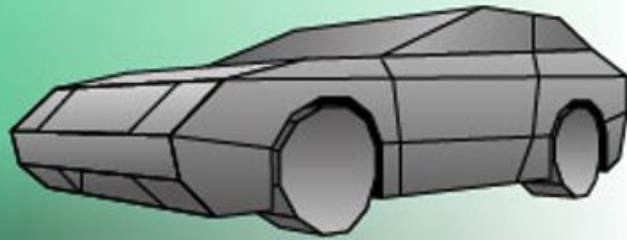


Classe nada mais é
que um projeto de
objeto podendo
instanciar vários
objetos



***COMO
CRIAMOS UMA
CLASSE ?***

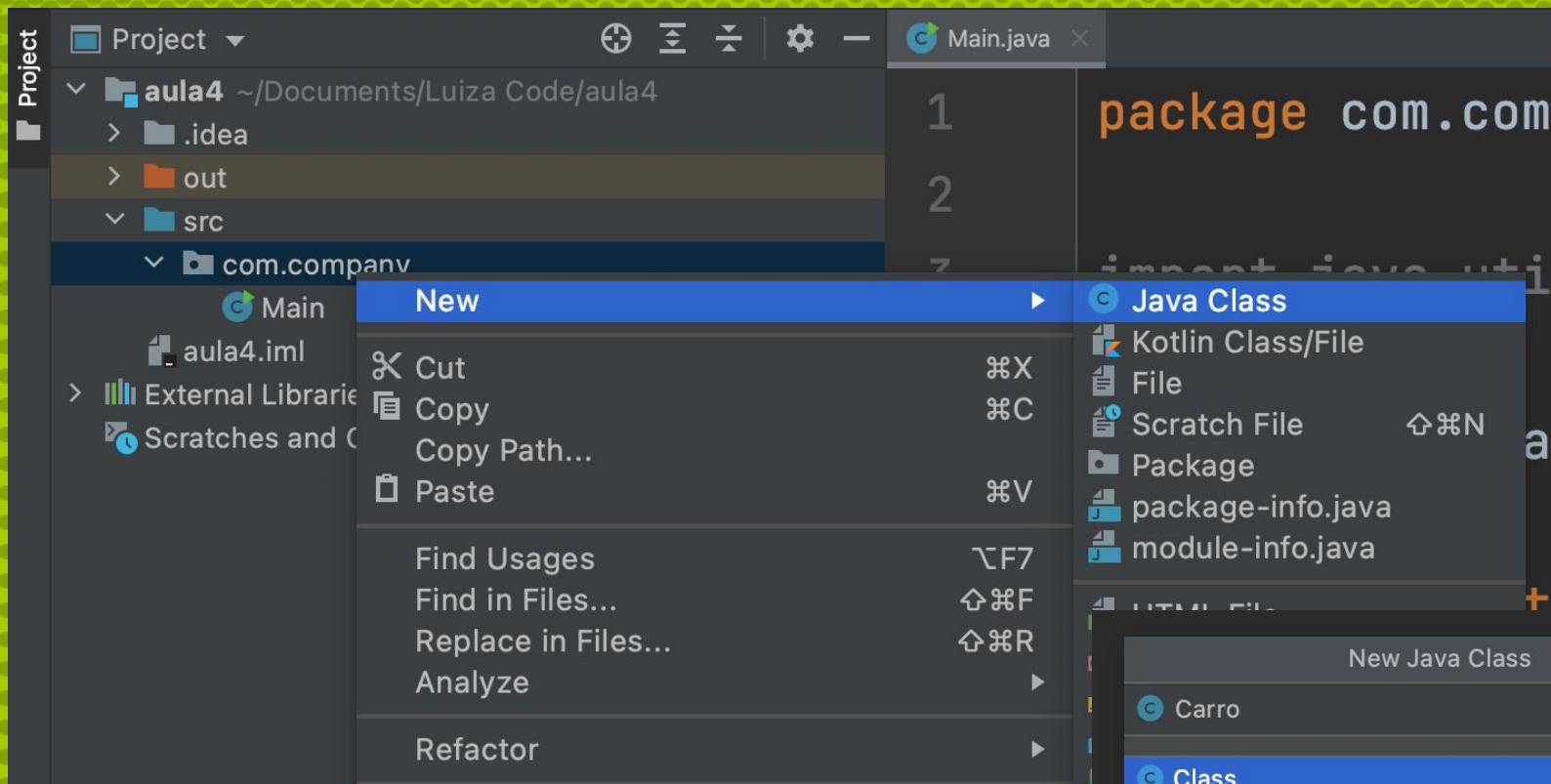




```
Class Carro {  
  
    String tipo;  
    String cor;  
    String placa;  
    int numPortas;  
  
    void setCor(String cor){  
        this.cor = cor;  
    }  
  
    String getCor(){  
        return this.cor;  
    }  
}
```



MÃO NA MASSA



```
1 package com.company;
2
3 public class Carro {
4     String tipo;
5     String cor;
6     String placa;
7     int numPortas;
8
9     Carro() {
10    }
11
12    Carro(String placa, String cor) {
13        this.placa = placa;
14        this.cor = cor;
15    }
16 }
```

Constructors

São responsáveis por
criar o objeto na
memória


```
9      public Carro() {  
10     }  
11  
12     public Carro(String placa, String cor) {  
13         this.placa = placa;  
14         this.cor = cor;  
15     }  
16  
17     public Carro(String tipo, String cor, String placa, int numPortas) {  
18         this.tipo = tipo;  
19         this.cor = cor;  
20         this.placa = placa;  
21         this.numPortas = numPortas;  
22     }
```

***COMO
CRIAMOS UMA
OBJETO ?***



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

The diagram illustrates the components of the Java code line `Carro meuCarro = new Carro();`. The code is presented in a light blue rounded rectangle. Two brown arrows originate from this rectangle: one points to the left side (`Carro meuCarro`) and the other points to the right side (`= new Carro();`). Each arrow points to a corresponding explanatory text box below.

O lado esquerdo da instrução declara uma variável do tipo Carro, uma referência a um objeto Carro.

O lado direito da expressão cria o objeto na memória e retorna a sua referência para a variável.



MÃO NA MASSA



The image shows a screenshot of an IDE with two tabs: 'Main.java' and 'Carro.java'. The 'Main.java' tab is active, displaying the following code:

```
1 package com.company;  
2  
3 public class Main {  
4  
5     public static void main(String[] args) {  
6  
7         Carro meuPrimeiroCarro = new Carro();  
8  
9         Carro meuSegundoCarro = new Carro( placa: "ABC1234", cor: "Laranja");  
10  
11        Carro meuTerceiroCarro = new Carro( tipo: "Conversivel", cor: "Amarelo", placa: "DEF5678", numPortas: 4);  
12  
13    }  
14 }
```

The code defines a package 'com.company' and a public class 'Main'. Inside the 'main' method, three 'Carro' objects are created: 'meuPrimeiroCarro' (no arguments), 'meuSegundoCarro' (with 'placa' and 'cor'), and 'meuTerceiroCarro' (with 'tipo', 'cor', 'placa', and 'numPortas'). The IDE interface includes a line number margin on the left, a gutter with icons, and a top bar with tabs and a warning icon indicating 3 warnings.

***COMO
ACESSAMOS AS
INFORMAÇÕES DE
UM OBJETO ?***



MÉTODOS GET E SET

Carro carro = 32;

Get em number

Number = 32
Retorno para
usuário vai ser 32

Set em number

Programa passa um
novo valor para
number
Number = 1

MÉTODOS GET E SET

`int number = 32;`

Get em number

Set em number

`getNumber()`

Função


`setNumber(int n)`

Procedimento



MÃO NA MASSA

```
Main.java x Carro.java x
24 public String getTipo() {
25     return tipo;
26 }
27
28 public String getCor() {
29     return cor;
30 }
31
32 public String getPlaca() {
33     return placa;
34 }
35
36 public int getNumPortas() {
37     return numPortas;
38 }
39 }
```

```
39
40  public void setTipo(String tipo) {
41      this.tipo = tipo;
42  }
43
44  public void setCor(String cor) {
45      this.cor = cor;
46  }
47
48  public void setPlaca(String placa) {
49      this.placa = placa;
50  }
51
52  public void setNumPortas(int numPortas) {
53      this.numPortas = numPortas;
54  }
55  }
```

***COMO
ACESSAMOS
ESSAS VARIÁVEIS
DO OBJETO ?***



```
Main.java x Carro.java x
1 package com.company;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         //Instanciando um objeto Carro
7         Carro meuPrimeiroCarro = new Carro();
8
9         //Atribuindo valores aos atributos do carro
10        meuPrimeiroCarro.setCor("Laranja");
11        meuPrimeiroCarro.setNumPortas(4);
12        meuPrimeiroCarro.setPlaca("ABC1234");
13        meuPrimeiroCarro.setTipo("Conversível");
14
15        //Acessando os valores dos atributos do carro
16        System.out.println("A cor é: " + meuPrimeiroCarro.getCor());
17        System.out.println("A quantidade de portas é: " + meuPrimeiroCarro.getNumPortas());
18        System.out.println("A placa é: " + meuPrimeiroCarro.placa);
19        System.out.println("O tipo é: " + meuPrimeiroCarro.getTipo());
20    }
21 }
```

```
Run: Main x
↑ /Library/Java/JavaVirtualMachin
↓ A cor é: Laranja
↺ A quantidade de portas é: 4
↻ A placa é: ABC1234
⌂ O tipo é: Conversível
```

***COMO CRIAMOS
METODOS
ESPECIFICOS PARA
UM OBJETO ?***



MÉTODOS

Um método é um trecho de código que realiza uma função específica e pode ser chamado por qualquer outro método ou classe para realizar a referida função num determinado contexto



CARACTERÍSTICAS

(dados, atributos)

Tipo: Ferrari

Cor: Vermelho

Placa: KZE-1018

Número de portas: 2

COMPORTAMENTOS

(operações, métodos)

Ligar

Desligar

Acelerar

Frear

MÉTODO SEM RETORNO

`void ligar()`

Quando é
procedimento.

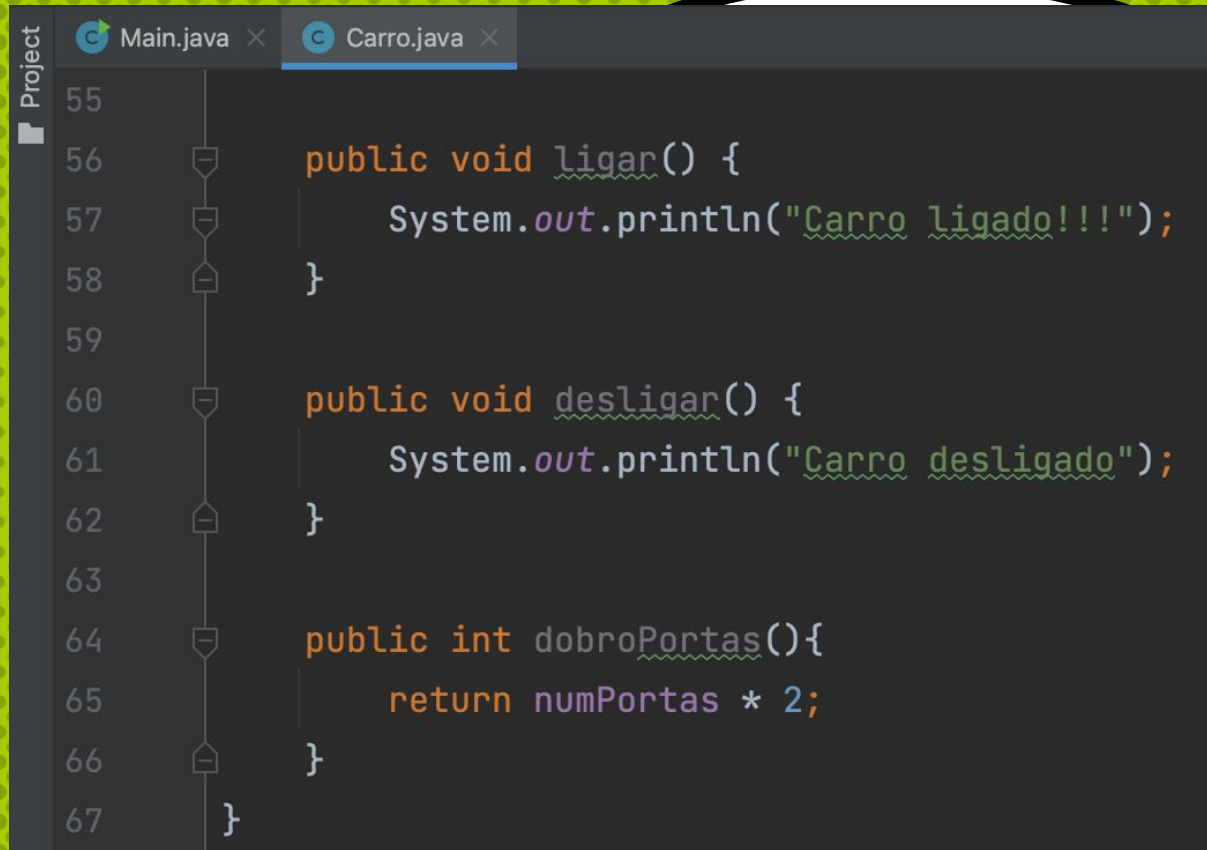
MÉTODO COM RETORNO

`int dobroPortas()`

Int corresponde ao
tipo de retorno
dessa função



MÃO NA MASSA



```
Project
Main.java x Carro.java x
55
56 public void ligar() {
57     System.out.println("Carro ligado!!!");
58 }
59
60 public void desligar() {
61     System.out.println("Carro desligado");
62 }
63
64 public int dobroPortas() {
65     return numPortas * 2;
66 }
67 }
```



***MÃO NA MASSA
COM VOCÊ***

PRÁTICA 01

Crie uma classe para pessoa com as seguintes características:

- Atributos:
 - Nome, idade, CNH
- Metodos:
 - Dirigir, dormir, comer

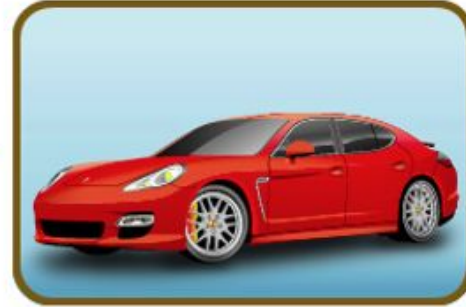

```
1 package com.company;
2
3 public class Pessoa {
4
5     String nome;
6     int idade;
7     String cnh;
8
9     public Pessoa(String nome, int idade, String cnh) {
10         this.nome = nome;
11         this.idade = idade;
12         this.cnh = cnh;
13     }
14
15     public String getNome() {
16         return nome;
17     }
18
19     public int getIdade() {
20         return idade;
21     }
22
23     public String getCnh() {
24         return cnh;
25     }
26 }
```

```
26
27     public void setNome(String nome) {
28         this.nome = nome;
29     }
30
31     public void setIdade(int idade) {
32         this.idade = idade;
33     }
34
35     public void setCnh(String cnh) {
36         this.cnh = cnh;
37     }
38
39     public void dirigir() {
40         System.out.println("Pessoa dirigindo");
41     }
42
43     public void comer() {
44         System.out.println("Pessoa comendo");
45     }
46
47     public void dormir() {
48         System.out.println("Pessoa dormindo");
49     }
50 }
```

***COMO FAZEMOS
PARA ESSES
OBJETOS SE
RELACIONAREM?***



RELACIONAMIENTO ENTRE OBJETOS



OBJETOS NÃO EXISTEM ISOLADOS

- × São formados por outros objetos
- × Objetos usam outros objetos
- × Um programa OO possui vários objetos que interagem entre si

TIPOS DE RELACIONAMENTO

Composição

Um objeto pode
ser formado por
outros objetos

Agregação

Um objeto pode
conter outros
objetos

Associação

Um objeto pode
usar outros
objetos

COMPOSIÇÃO

Um livro é composto de capítulos



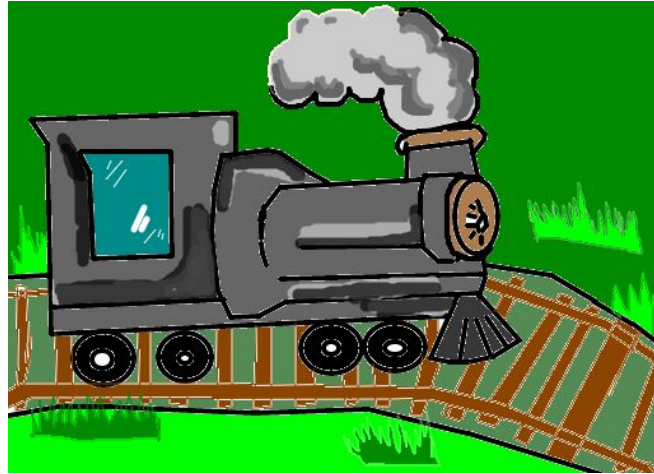
AGREGAÇÃO

Carro possui Porta



ASSOCIAÇÃO

Trem usa estrada de ferro





MÃO NA MASSA



Main.java ×



Carro.java ×



Pessoa.java ×

```
1 package com.company;
```

```
2
```

```
3 public class Carro {
```

```
4     String tipo;
```

```
5     String cor;
```

```
6     String placa;
```

```
7     int numPortas;
```

```
8     Pessoa dona;
```

```
9
```

```
57
```

```
58
```

```
59
```

```
60
```

```
61
```

```
62
```

```
63
```



```
public Pessoa getDona() {
```

```
    return dona;
```

```
}
```

```
public void setDona(Pessoa dona) {
```

```
    this.dona = dona;
```

```
}
```



```
1 package com.company;
2
3 public class Main {
4
5     public static void main(String[] args) {
6         //Instanciando um objeto Carro
7         Carro meuPrimeiroCarro = new Carro();
8
9         //Instanciando um objeto Pessoa
10        Pessoa pessoa = new Pessoa( nome: "Taina", idade: 33, cnh: "123456789");
11
12        //Atribuindo valores aos atributos do carro
13        meuPrimeiroCarro.setCor("Laranja");
14        meuPrimeiroCarro.setNumPortas(4);
15        meuPrimeiroCarro.setPlaca("ABC1234");
16        meuPrimeiroCarro.setTipo("Conversível");
17        meuPrimeiroCarro.setDona(pessoa);
18
19        //Acessando os valores dos atributos do carro
20        System.out.println("A cor é: " + meuPrimeiroCarro.getCor());
21        System.out.println("A quantidade de portas é: " + meuPrimeiroCarro.getNumPortas());
22        System.out.println("A placa é: " + meuPrimeiroCarro.placa);
23        System.out.println("O tipo é: " + meuPrimeiroCarro.getTipo());
24        System.out.println("A dona desse carro é: " + meuPrimeiroCarro.getDona().getNome());
25    }
26 }
```



```
26 //Acessando os metodos
27 meuPrimeiroCarro.ligar();
28 meuPrimeiroCarro.desligar();
29 System.out.println(meuPrimeiroCarro.dobroPortas());
30 }
```

```
19 //Acessando os valores dos atributos do carro
20 System.out.println("A cor é: " + meuPrimeiroCarro.getCor());
21 System.out.println("A quantidade de portas é: " + meuPrimeiroCarro.getNumPortas());
22 System.out.println("A placa é: " + meuPrimeiroCarro.placa);
23 System.out.println("O tipo é: " + meuPrimeiroCarro.getTipo());
24 System.out.println("A dona desse carro é: " + meuPrimeiroCarro.getDono().getNome());
25
26 //Acessando os metodos
27 meuPrimeiroCarro.ligar();
28 meuPrimeiroCarro.desligar();
29 System.out.println(meuPrimeiroCarro.dobroPortas());
```

Run: Main ×

```
↑ /Library/Java/JavaVirtualMachin
↓ A cor é: Laranja
↕ A quantidade de portas é: 4
⌕ A placa é: ABC1234
⌕ O tipo é: Conversível
⌕ A dona desse carro é: Taina
⌕ Carro ligado!!!
⌕ Carro desligado
8
```

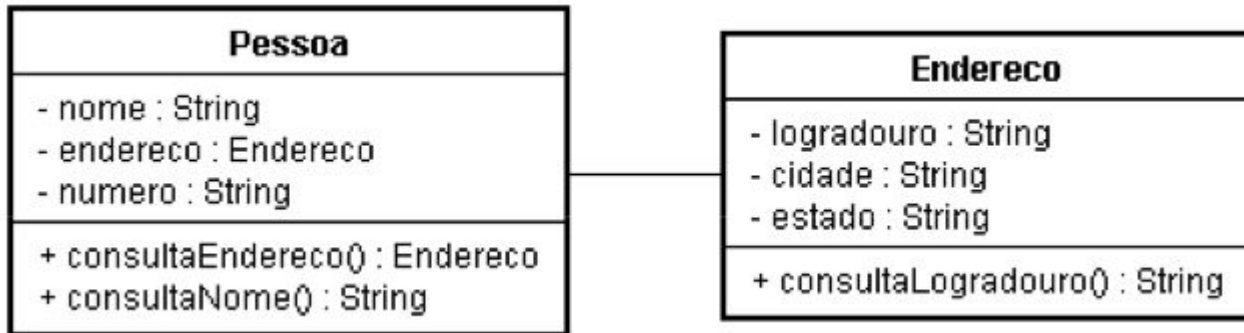




***MÃO NA MASSA
COM VOCÊ***

PRÁTICA 02

Desenvolva o seguinte relacionamento

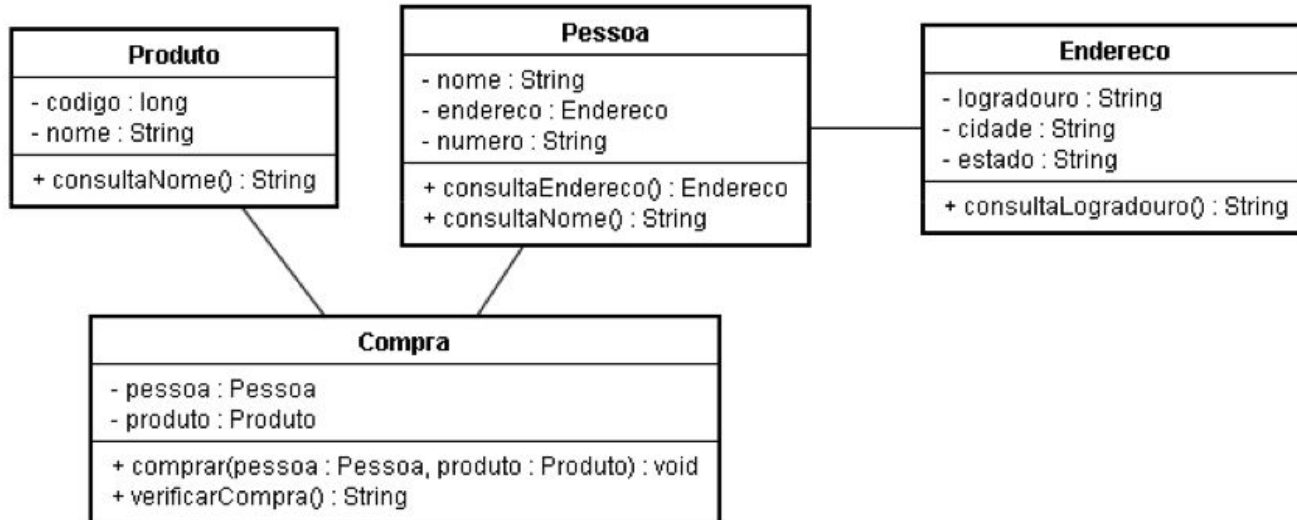


Pessoa vai ter um objeto do tipo Endereço

Lembre-se de criar os construtores parametrizados

PRÁTICA 03

Agora adicione as seguintes classes no resultado da Prática anterior.



PRÁTICA 03 (CONTINUAÇÃO)

A classe Compra vai conter um objeto do tipo Pessoa e outro do tipo Produto.

Será possível comprar acessando o método comprar, passando como parâmetro dois objetos, um do tipo Pessoa e outro do tipo Produto.

PRÁTICA 04

Agora desenvolva a classe Main:

- × Inicialmente, a classe deve criar 2 objetos do tipo Produto (crie os objetos com as informações que desejar)
- × O usuário faz um cadastro (criando um objeto do tipo Pessoa)
- × Logo após, o usuário seleciona entre os 2 produtos cadastrados anteriormente
- × Dependendo da escolha, é acessado o método comprar da classe Compra, e passado como parâmetro o objeto Pessoa que ele cadastrou e o do Produto escolhido
- × Logo após, é exibido no console ao usuário uma mensagem de confirmação, exibindo o nome dele e do produto comprado (acesso ao método verificarCompra()) e finaliza a aplicação

THE END