Engenharia De Software

AULA 1 E 2 - INTRODUÇÃO E REVISÃO DE POO

Inatel

Apresentação

Professor: Phyllipe Lima

Monitores:

- João Pedro Giacometti de Souza
- Vanessa Swerts Esteves

Atendimento: Terça-Feira - 17hr30

Contato:

vanessaswerts@gec.inatel.br

Inatel

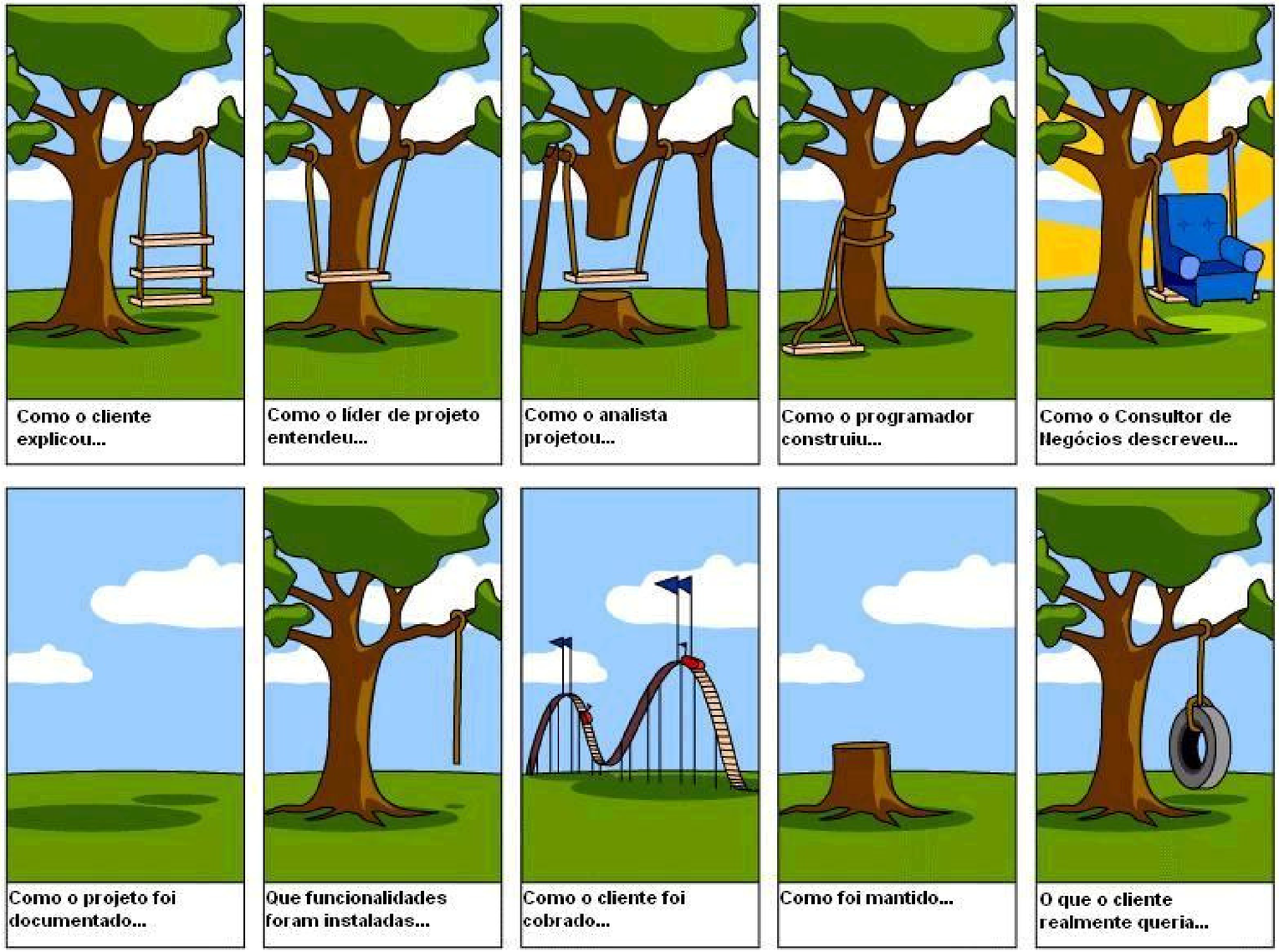
Definição

“A Engenharia de Software assume papel crítico para garantir que tarefas, dados, pessoas e tecnologias estejam apropriadamente alinhadas para produzir um sistema efetivo e eficiente.”

Ana Lúcia de Oliveira, Ana Paula Eckel, Závia Roselita e Cateane Scarpa.

Inatel

Importância da Engenharia de Software



Inatel

Cronograma

Cronograma Laboratório de C214		
	Tema	**L2 (Quarta-Feira - 15:30)**
Semana 1	Apresentação do laboratório / Revisão POO	24/02/2021
Semana 2	Revisão POO	03/03/2021
Semana 3	Gerência de Dependências	10/03/2021
Semana 4	Git	17/03/2021
Semana 5	Git	24/03/2021
Semana 6	Testes	31/03/2021
Semana 7	Testes	07/04/2021
Semana 8	**Apresentação - PT1**	14/04/2021
Semana 9	Auxílio no Projeto	21/04/2021
Semana 10	**Apresentação - PT2**	28/04/2021
Semana 11	CI - Travis e GitHub Action	05/05/2021
Semana 12	Auxílio no Projeto	12/05/2021
Semana 13	Auxílio no Projeto	19/05/2021
Semana 14	Auxílio no Projeto	26/05/2021
Semana 15	**Apresentação - PT3**	02/06/2021
Semana 16	Auxílio no Projeto	09/06/2021
Semana 17	Auxílio no Projeto	16/06/2021
Semana 18	Auxílio no Projeto	23/06/2021
Semana 19	**Apresentação Final do Projeto**	30/06/2021
Semana 20	NP3	07/07/2021

Inatel

Avaliação - NPL

Avaliação das Entregas	Porcentagem (100 pontos)
Apresentação PT1	10%
Apresentação PT2	20%
Apresentação PT3	20%
Apresentação Final	50%

Projeto Final	Porcentagem (50 pontos)
README	30%
Implementações do conteúdo	65%
Apresentação	5%

Inatel

Ferramentas do Laboratório

Linguagem de Programação : JavaScript - React js

IDE: Visual Studio Code

Configurando o ambiente:

VSCode : https://code.visualstudio.com

React js: Tutorial para a configuração do ambiente

Node.js https://nodejs.org/en/

Inatel

Revisão de POO

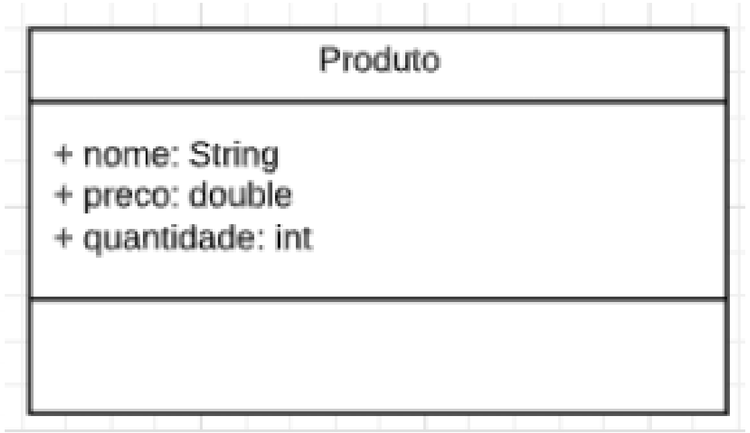
Construtores:

- É uma operação específica de uma classe, que executa sempre que ela é instanciada.
- Pra que serve?
  - Receber parâmetros de classes externas
  - Obrigar a inicialização de variáveis
  - Injeção de dependências.
- É possível que uma classe tenha mais de um construtor porém recebendo parâmetros diferentes (sobrecarga)

Inatel

Revisão de POO

```java
package model.entities;

public class Produto {

    //Atributos
    public String nome;
    public double preço;
    public int quantidade;

    //Construtores
    //Construtor com campos vazios {Default}
    public Produto() {

    }

    public Produto(String nome, double preço, int quantidade) {
        this.nome = nome;
        this.preço = preço;
        this.quantidade = quantidade;
    }


}
```



Obriga que os atributos da classe Produto sejam passados como referência.

Inatel

Revisão de POO

```java
package application;

import java.util.Scanner;

import model.entities.Produto;

public class Programa {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        System.out.println("Entre com o nome do produto");
        String nome = sc.nextLine();

        System.out.println("Entre com o preço do produto");
        double preco = sc.nextDouble();

        System.out.println("Entre com a quantidade de produtos");
        int quantidade = sc.nextInt();

        Produto produto = new Produto(nome, preco, quantidade);

        sc.close();
    }

}
```

Inatel

Revisão de POO

Encapsulamento:

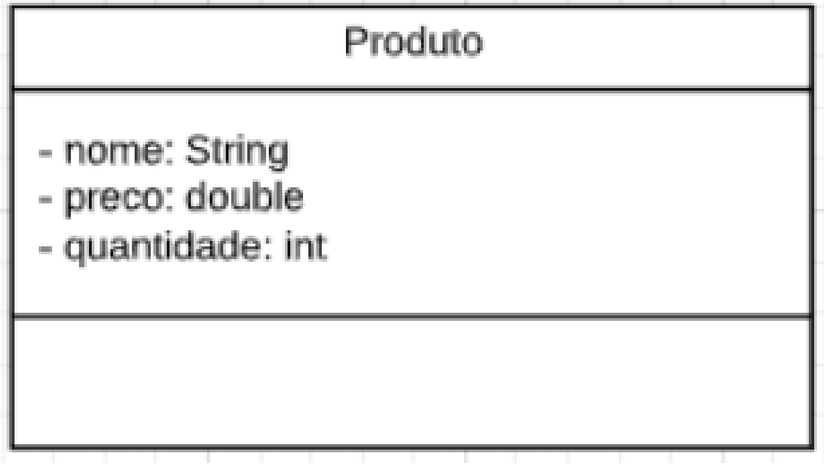
- Esconde os detalhes e informações de uma classe, expondo somente operações seguras.
- Regras importantes
    - Todo atributo não deve ser exposto (private)
    - Devem ser acessados através de métodos get e set

Inatel

Revisão de POO

```java
//Atributos
private String nome;
private double preço;
private int quantidade;

//Construtores
public Produto(String nome, double preço, in

//Get e Set
public String getNome() {
    return nome;
}

public void setNome(String nome) {
    this.nome = nome;
}

public double getPreço() {
    return preço;
}

public void setPreço(double preço) {
    this.preço = preço;
}

public int getQuantidade() {
    return quantidade;
}

public void setQuantidade(int quantidade) {
    this.quantidade = quantidade;
}
```

Mudança de public
=> private



Inatel

Revisão de POO

Modificadores de acesso:

- **(default):** o membro só pode ser acessado nas classes do mesmo pacote
- **public:** o membro é acessado por todas classes (ao menos que ele resida em um módulo diferente que não exporte o pacote onde ele está)
- **protected:** o membro só pode ser acessado no mesmo pacote, bem como em subclasses de pacotes diferentes.
- **private:** o membro só pode ser acessado na própria classe

Inatel

Revisão de POO

Modificadores de acesso:

MODIFICADOR	CLASSE	MESMO PACOTE	PACOTE DIFERENTE (SUBCLASSE)	PACOTE DIFERENTE(GLOBAL)
Public	✓	✓	✓	✓
Protected	✓	✓	✓	✗
Default	✓	✓	✗	✗
Private	✓	✗	✗	✗

Inatel

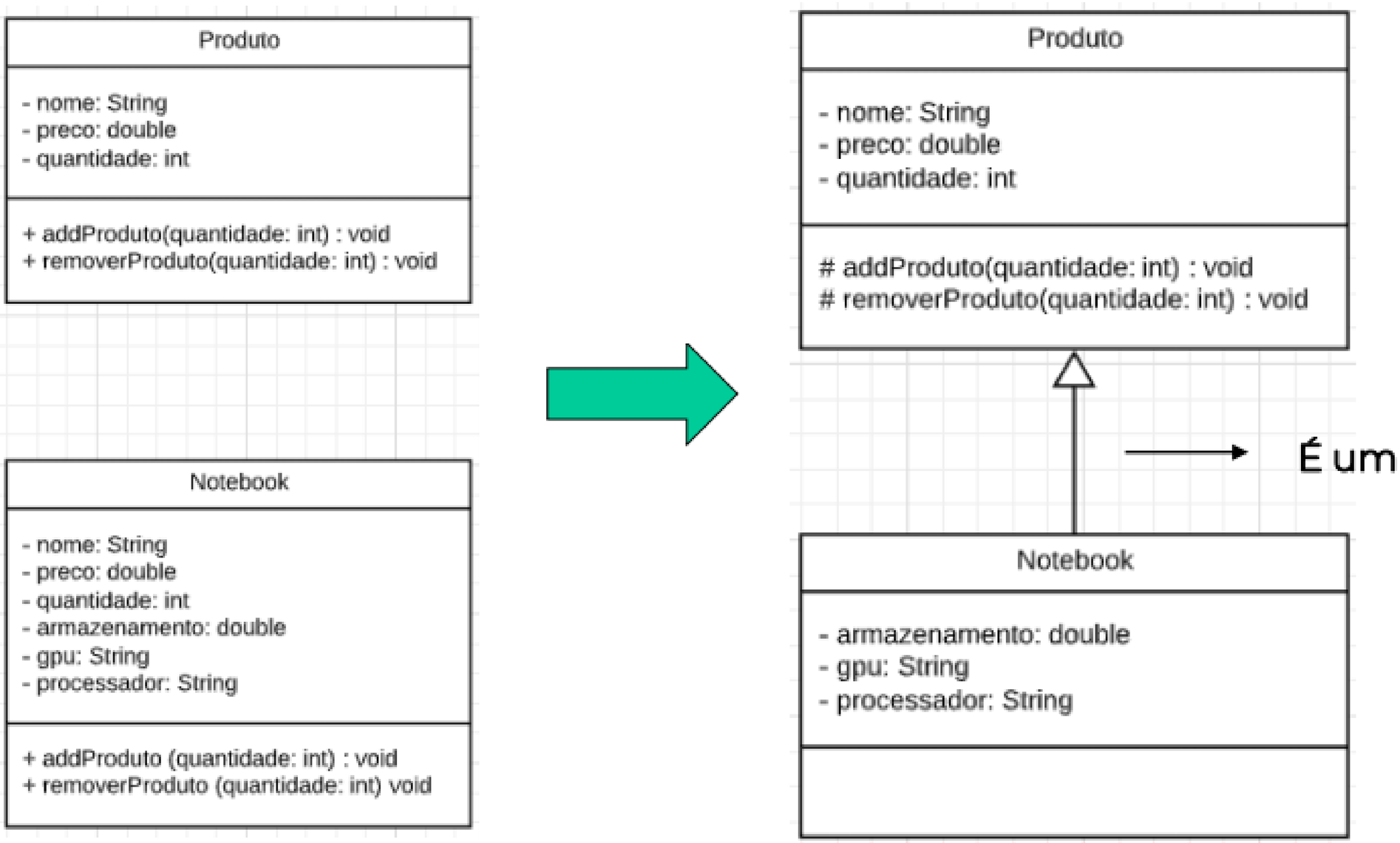
Revisão de POO

Herança:

- É um tipo de associação entre uma classe mais genérica (superclasse), com classes mais específicas
- Permite que uma classe herde os comportamentos e dados da superclasse.
- **Vantagens:**
  - Reuso de código;
  - Polimorfismo.

Inatel

Revisão de POO



Inatel

Revisão de POO

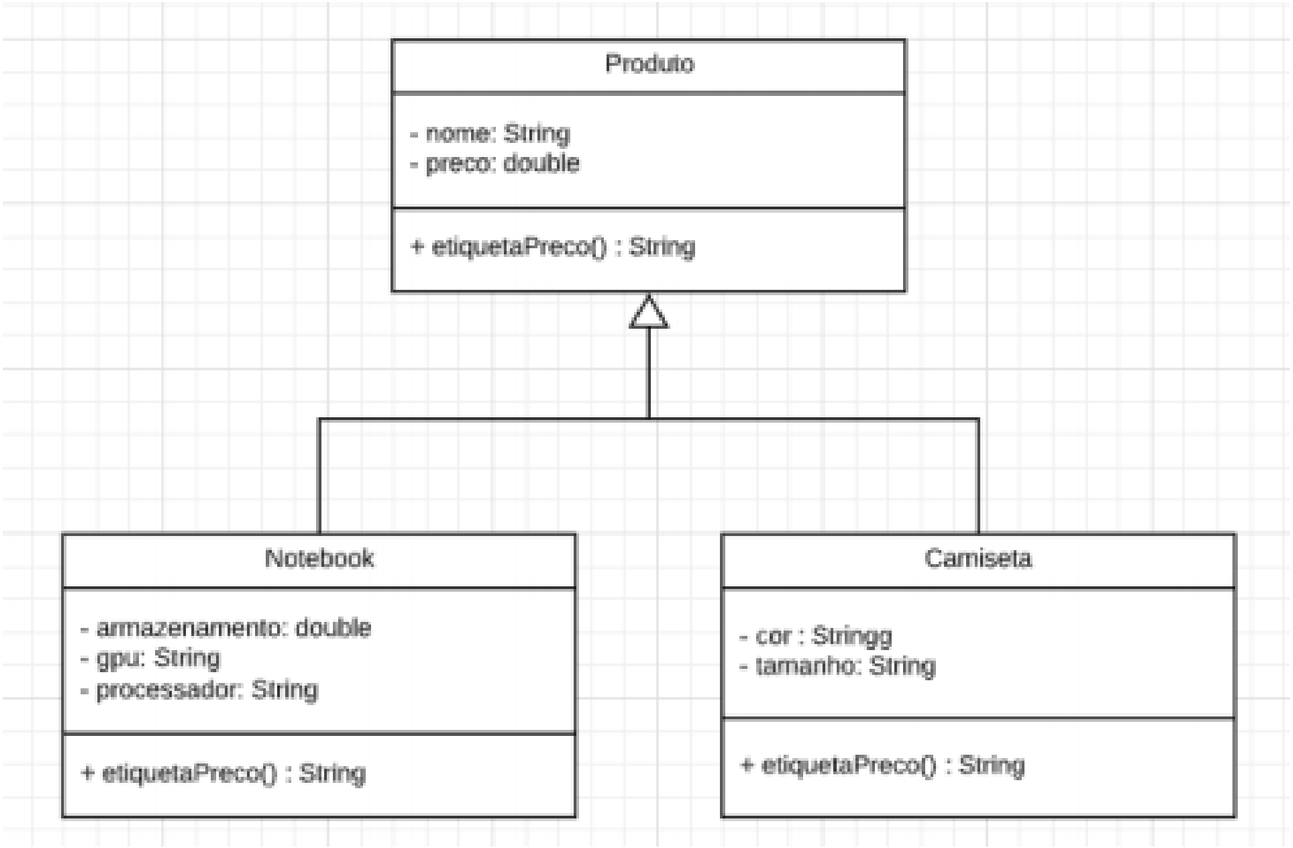
Exercício de fixação:

Fazer um programa para ler os dados de N produtos. Ao final, mostrar a etiqueta de preço de cada produto na mesma ordem em que foram digitados.

Todo produto possui nome e preço. Notebook possui armazenamento, gpu e um processador e camiseta possui tamanho e cor. Estes dados específicos dever ser acrescentados na etiqueta de preço conforme o exemplo.

Revisão de POO

```
Numero de produtos
2
Produto #1
Notebook ou Camiseta? (n/c)
n
Entre com o nome do produto
Notebook Dell
Entre com o preço do produto
3500
Entre com o armazenamento do Notebook
1000
Entre com a GPU Notebook
NVIDIA GTX 1050
Entre com o processador do Notebook
i5
Produto #2
Notebook ou Camiseta? (n/c)
c
Entre com o nome do produto
Camiseta Nike
Entre com o preço do produto
80
Entre com a cor da Camiseta
Preta
Entre com o tamanho da Camiseta
P

Etiquetas:
Notebook Dell $3500,00 1000.0 NVIDIA GTX 1050 i5
Camiseta Nike $80,00 Preta P
```



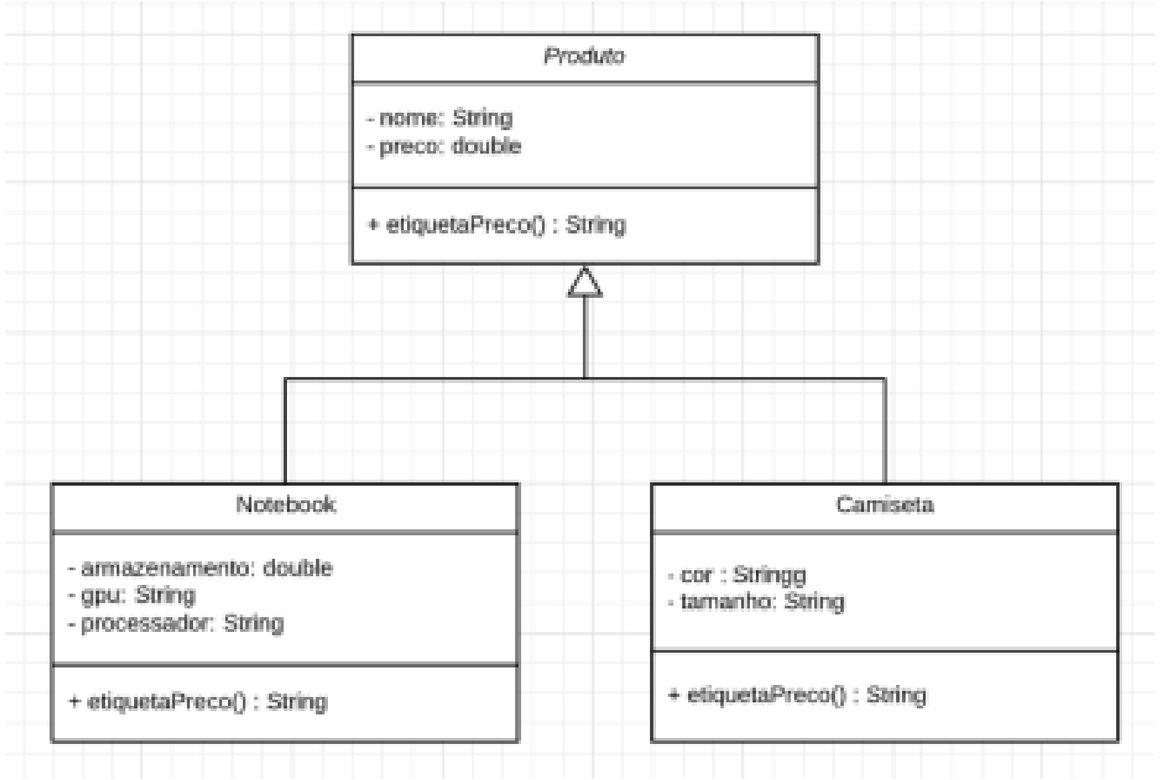
Inatel

Revisão de POO

Polimorfismo:

- É um recurso que permite que variáveis de um mesmo tipo genérico possam apontar para objetos de tipo específico.

- Quando a instanciação do tipo específico com o tipo genérico acontece, isso é chamado de ***upcasting***.



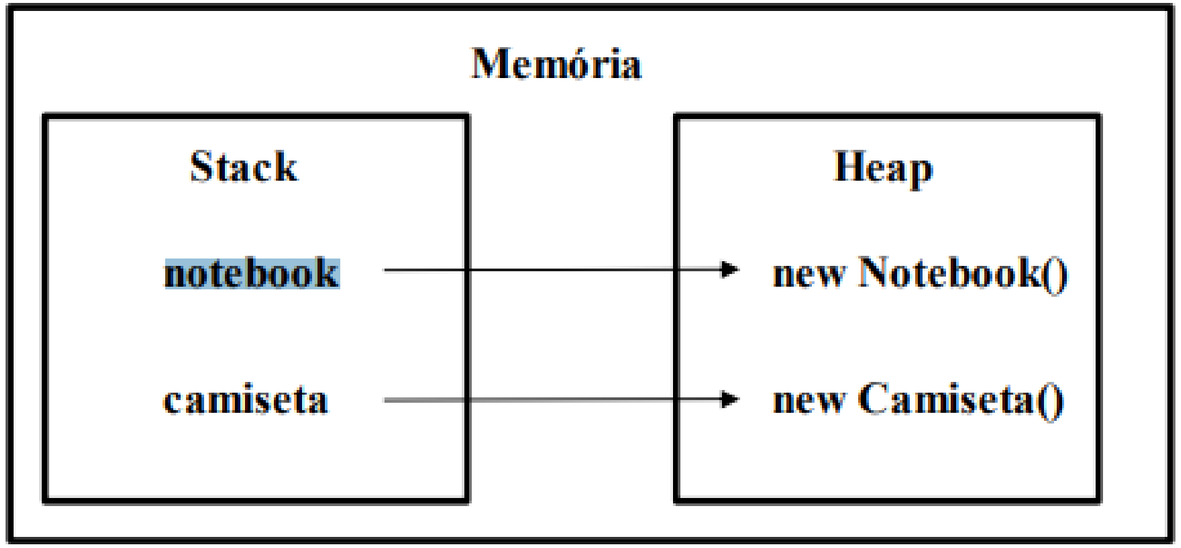
Inatel

Revisão de POO

Polimorfismo:

```
public class Programa {

    public static void main(String[] args) {

        Produto produto = new Produto("Iphone XR", 2999.99);

        Produto notebook = new Notebook("Notebook Dell", 2500, 1000, "GTX 1050", "core i5");
        Produto camiseta = new Camiseta("Camiseta Nike", 80.00, "Preta", "P");

    }

}
```

Upcasting



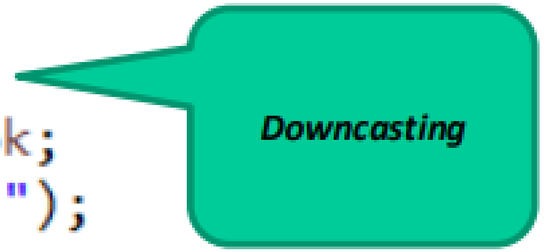
Inatel

Revisão de POO

Polimorfismo:

- Mas e se quisermos acessar um atributo específico e fazer uma alteração?
- Por mais que as variáveis apontem para o tipo específico, elas continuam sendo do tipo Produto
- Então teremos que realizar um **downcasting** na variável.

```
// Editando as informações
if (notebook instanceof Notebook) {
    Notebook notebookAux = (Notebook) notebook;
    notebookAux.setProcessador("Intel core i5");
}

System.out.println(notebook.etiquetaPreco());
```



Inatel

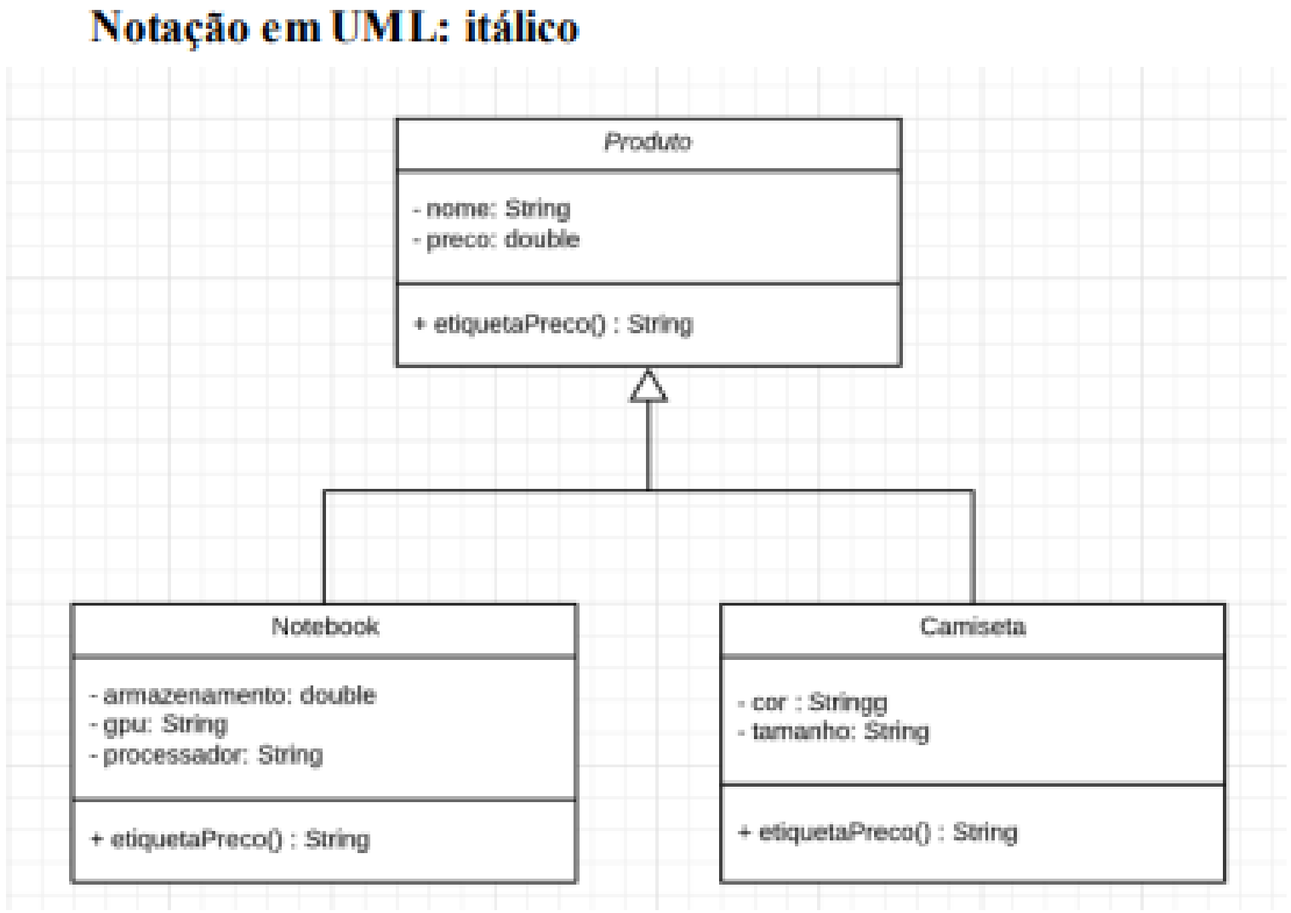
Revisão de POO

Abstração:

- É uma maneira de se garantir uma herança total, portanto somente subclasses podem ser instanciadas.

```
public abstract class Produto

Produto produto = new Produto("Iphone XR", 2999.99);
```

Cannot instantiate the type Produto



Inatel

Revisão de POO

Questionamento:

Se Produto não pode ser instanciado, por que simplesmente não criar somente Notebook e Camiseta ?

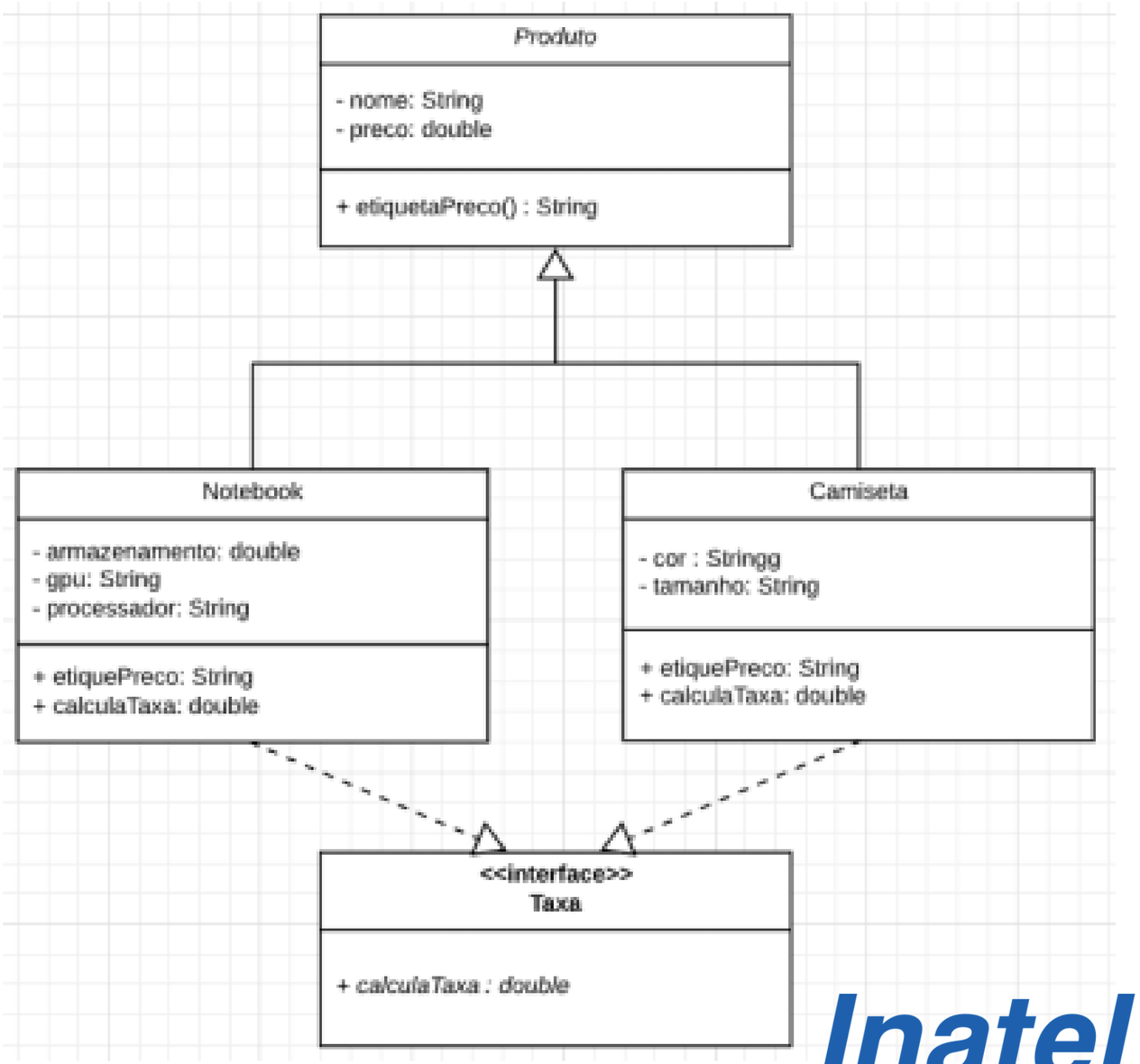
- Reuso de código;
- Polimorfismo: A classe genérica vai nos permitir tratar de forma fácil todo tipo de produto e colocar todos os tipos de produto em uma só coleção.

Inatel

Revisão de POO

Interface:

- É um tipo que define apenas os métodos que uma classe deve implementar.
- Declaramos apenas o escopo desses métodos.
- A interface estabelece um contrato com a classe.



Inatel

Revisão de POO

Interface:

- Pra que serve?
  - Criar sistemas com baixo acoplamento e flexíveis.
- Sintaxe

```
public class Notebook extends Produto implements Taxa{

public class Camiseta extends Produto implements Taxa{
```

Inatel

Revisão de POO

Métodos estáticos:

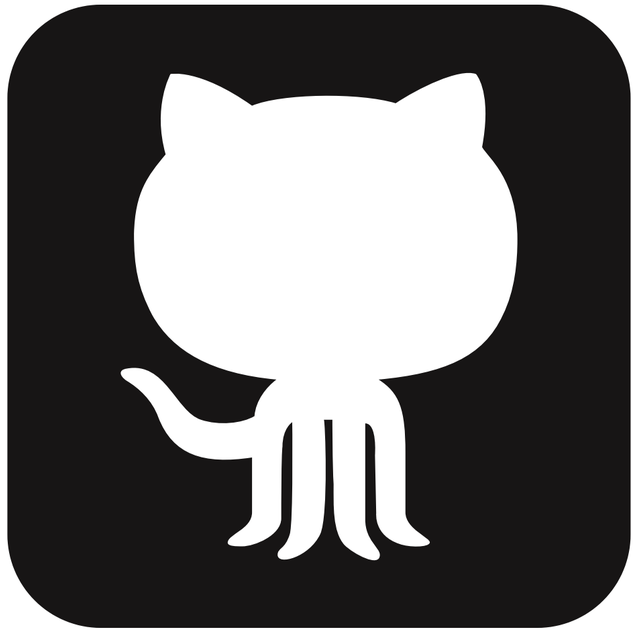
- São métodos que fazem sentido sentido independentemente da instanciação do objeto.
- São chamaos a partir do próprio nome da classe.
- Aplicação:
  - Classes utilitárias
  - Exemplo:

```
Math.sqrt(4);
```

Notação em UML: Sublinhado

Inatel

Código da aula



https://github.com/monitoria-C214/aula01

Inatel