Programação Orientada a Objetos

Aula 3 – Professor Diogo Orlando Nunes de Almeida

Conteúdo

- Conceitos do paradigma da Programação orientada a objetos;
- Diagrama de classes (UML);
- Introdução a classes e objetos;
- Atributos, métodos e interação entre objetos;
- Construtores e destrutores;
- Agregação e composição de objetos;
- Encapsulamento;
- Herança e polimorfismo;
- Tratamento de Exceções.

Encapsulamento

- A diretriz de encapsulamento (ocultar informações de implementação de visualização) sugere que somente as informações sobre o que uma classe pode fazer devem ser visíveis externamente, não como ela é. (BARNES; KÖLLING; FURMANKIEWICZ, 2022, p. 188).

Encapsulamento no Java

- Para privar um atributo ou método usamos a palavra private, para ela poder ser usada fora do pacote vamos declarar como pública e para usar internamente não precisamos chamar ela como pública e nem como privada.

```
private String nome;
private double peso;
private double preco;
private int quantidade;
private int estoque_maximo;
public double margem_lucro;
```

Encapsulamento no Java

- Os métodos também podem ser encapsulados e protegidos para uso dentro da classe.

```
private double valor_total() {
    return preco * quantidade;
}

public double lucro_total() {
    double valor, lucro;
    valor = this.valor_total();
    lucro = valor * margem_lucro;
    return lucro;
}
```

Construtores

- Os construtores de uma classe têm um papel especial a cumprir; é sua responsabilidade colocar cada objeto dessa classe, quando ele está sendo criado em um estado previamente definido para ser utilizado. (BARNES; KÖLLING; FURMANKIEWICZ, 2022, p. 22).

Construtor no Java

- Apenas crie um método sem declarar retorno com o nome da Classe.

```java
class Produto{
    private String nome;
    private double peso;
    private double preco;
    private int quantidade;
    private int estoque_maximo;
    public double margem_lucro;

    public Produto(String nome, double peso, double preco, int estoque_maximo){
        this.nome = nome;
        if (peso > 500){
            System.out.println("Aviso: Produto excede 500 kilos");
        }
        this.peso = peso;
        this.preco = preco;
        this.estoque_maximo = estoque_maximo;
        this.quantidade = 0;
        this.margem_lucro = 0;
    }
```

Construtor no Java

- Para instanciar uma variável com construtor no programa principal faça o seguinte:

```
public static void main(String[] args) {
    Produto sorvete = new Produto("Sorvete de flocos", 0.800, 19.90, 10);
```

Sobrecarga de nome

- A razão para utilizar essa construção é que temos uma situação que é conhecida como sobrecarga de nome (name overloading) – o nome é usado para duas entidades diferentes. (BARNES; KÖLLING; FURMANKIEWICZ, 2022, p. 69).
- A expressão this referencia o objeto atual em Java. (BARNES; KÖLLING; FURMANKIEWICZ, 2022, p. 69).
- Em Python se usa self ao invés de this.

Destrutores

- Para destruir uma instância de um objeto é só atribuir “null” a ele.
- O destrutor vai liberar a memória.

```
58  class supermercado {
59      public static void main(String[] args) {
60          Produto sorvete = new Produto("Sorvete de flocos", 0.800, 19.90, 10);
61          sorvete.adiciona_produto(5);
62          sorvete.valor_total();
63          sorvete.margem_lucro = 0.2;
64          sorvete.lucro_total();
65          sorvete.imprime_informacao();
66          sorvete = null; //Destrutor
67          sorvete.lucro_total();
```

Problems @ Javadoc Declaration Console

```
<terminated> New_configuration [Java Application] C:\Program Files\Java\jdk1.8.0_202\bin\javaw.exe (13 de
Os dados são:

Nome: Sorvete de flocos
Peso: 0,800
Preço: 19,900
Quantidade: 5
Limite de Estoque: 10
Margem de lucro: 0.2
Exception in thread "main" java.lang.NullPointerException
        at teste.supermercado.main(supermercado.java:67)
```

Métodos de acesso (getters)

- É o retorno de informações ao chamador sobre o estado de um objeto. (BARNES; KÖLLING; FURMANKIEWICZ, 2022, p. 27).
- É um método usado para coletar e tratar os atributos de uma classe.

```
public String get_nome() { //get
    if (this.quantidade < 0) { //tratativa
        System.out.println("Atenção: Produto em falta!");
    }
    return this.nome;
}
```

Métodos modificadores (Mutatores / setters)

- Os métodos modificadores alteram o estado de um objeto. (BARNES; KÖLLING; FURMANKIEWICZ, 2022, p. 28).

```
public void set_quantidade(int valor) { //set
    int nova_quantidade;
    nova_quantidade = valor + this.quantidade; //tratamento de excessão
    if (nova_quantidade>estoque_maximo) {
        System.out.println("Excedeu o máximo!");
    } else {
        this.quantidade = nova_quantidade;
    }
}
```

Exercícios

- Faça uma classe para uma frutaria.
- Faça uma classe para uma montadora de automóveis.
- Faça uma classe para um hospital (pode ser cadastro de paciente, cadastro de medicamento).

Referências

- BARNES, David J.; KÖLLING, Michael; FURMANKIEWICZ, Edson trad. Programação orientada a objetos com Java/ uma introdução prática usando o BlueJ. 4. ed. São Paulo: Pearson Prentice Hall, 2009. Ebook ISBN 978-85-7605-187-9. Disponível em: https://plataforma.bvirtual.com.br/Leitor/Publicacao/434/pdf. Acesso em: 2 mar. 2022. [On-line]