JAVA BÁSICO

LABORATÓRIO I

1. Abra o **Eclipse** e selecione para “File > New > Java Project”; em **Project Name** coloque “Lab.Classes” e então confirme.

2. Busque a **pasta** com o nome “src” no lado esquerdo no **Package Explorer**

   (a) Clique com o botão direito na pasta “src” e selecione “New > Class”

   (b) No campo nome digite “Produto” e confirme pressionando **Finish**

3. Caso o novo código não tenha sido aberto automaticamente, busque no **Package Explorer** dentro da pasta “src” o arquivo de classe recém criado “Produto.java”

4. Observe o codigo gerado

```
public class Produto {
        [TUDO AQUI DENTRO FARÁ PARTE DESSA CLASSE]
}
```

5. Observe que uma classe pública foi gerada (explicaremos o que público significa mais tarde) de nome “Produto”

   (a) De maneira simplificada, uma classe é composta por um

conjunto de funções (métodos) e dados (atributos)

(b) Devemos criar um método dentro da classe produto com a seguinte assinatura e conteúdo:

```
public static void main(String[] args)
{

      Produto agua = new Produto();
      Produto leite = new Produto();
      Produto qualquer = null;

      System.out.println(agua == qualquer);
      System.out.println(agua == leite);
      System.out.println(leite == qualquer);

}
```

6. O que será impresso no terminal caso o programa acima seja executado?

7. O que seria impresso no caso abaixo?

```
      Produto agua = new Produto();

      Produto leite = new Produto();
      Produto qualquer = null;

      qualquer = agua;
      agua = leite;

      System.out.println(agua == qualquer);
      System.out.println(agua == leite);
      System.out.println(leite == qualquer);
```

8. Crie dois atributos para a classe Produto:

   (a) Nome

   (b) Preço

```
public class Produto {


	// Não apague o método MAIN!

	String nome;
	double preco;
}
```

9. Agora crie dois construtores para a classe Produto, um construtor padrão (sem parâmetros) e outro que recebe um **nome** e um **preço**

```
	public Produto()

	{
		System.out.println("Construtor padrão");
		this.nome = "";
		this.preco = 0.0;
	}

	public Produto(String nome, double preco)
	{
		System.out.println("Construtor de dois parâmetros");
		this.nome = nome;
		this.preco = preco;
	}
```

10. Dentro do método main teste se os construtores funcionam utilizando o código abaixo

```
Produto p1 = new Produto("nome", 15.0);

Produto p2 = new Produto();
System.out.println("Programa rodou sem problemas");
```

11. Agora devemos criar maneiras de acessar a informação sobre **nome** e **preço** do produto. Para isso criaremos "acessors", ou seja, os métodos get/set

```
public String getNome() {

	return nome;
}

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

public double getPreco() {
	return preco;
}

public void setPreco(double preco) {
	this.preco = preco;
}
```

12. **DESAFIO:** Qual a necessidade de ter os métodos get/set? Qual a diferença entre esses métodos e acessar os atributos de maneira direta?

13. Utilize os acessors para obter o valor do **preco** e **nome** do produto. Para isso modifique o método main como mostrado abaixo

```
public static void main(String[] args)

{
	Produto p1 = new Produto("Água" , 1.50);
	Produto p2 = new Produto("Leite", 2.50);
	System.out.println(p1.getNome() + " " + p1.getPreco());
	System.out.println(p2.getNome() + " " + p2.getPreco());
	System.out.println("Programa rodou sem problemas");
}
```

---

14. Agora criaremos uma variável estática para indicar se as vendas de produtos estão permitidas. Adicione o seguinte atributo a classe Produto:

```
public static boolean PermiteVenda = true;
```

15. Comente o código que atualmente ocupada o método **main** e adicione o código abaixo

```
public static void main(String[] args)

{
	if (Produto.PermiteVenda)
	{
		System.out.println("Venda dos produto é permitida");
	}
	else
	{
		System.out.println("Venda dos produto está bloqueada");
	}
}
```

16. Observe que não foi necessário criar uma instância da classe produto utilizando **new** o atributi existe independente da instância

17. **DESAFIO:** Não é bom exagerar nas variáveis estatícas, ainda mais se forem mutáveis, por qual motivo?

18. **DESAFIO:** O que acontece quando tentamos acessar um método ou atributo **não-estático** sem ter uma instância (ver exemplo abaixo)

```
Produto.getNome();
```

19. **DESAFIO:** Por qual motivo o método main é estático?