

# Linguagem de Programação II

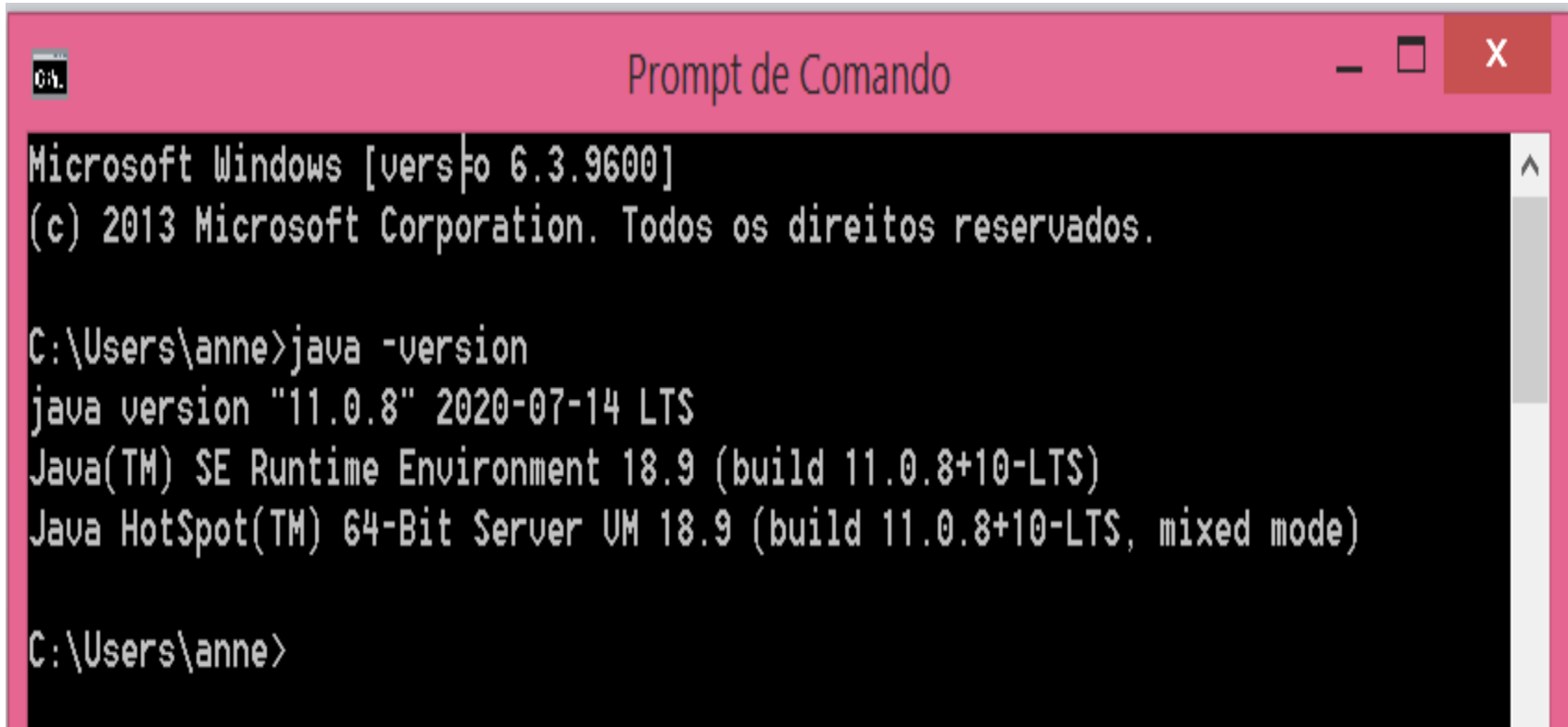
## IMD0040

### Aula 02 – Criando Classes e Objetos

# Vamos criar uma classe em Java?

- ❑ Editor de código:
  - ❖ Qualquer um.
  - ❖ Precisa salvar o código com extensão **.java**, e mesmo **nome da classe**.
- ❑ Compilação:
  - ❖ Abrir uma janela do terminal.
  - ❖ Digitar **javac** <nome do código.java>.
- ❑ Execução:
  - ❖ Na janela do terminal, digitar **java** <nome do código> sem extensão.

# Vamos começar???



```
Microsoft Windows [vers|fo 6.3.9600]
(c) 2013 Microsoft Corporation. Todos os direitos reservados.

C:\Users\anne>java -version
java version "11.0.8" 2020-07-14 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.8+10-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.8+10-LTS, mixed mode)

C:\Users\anne>
```

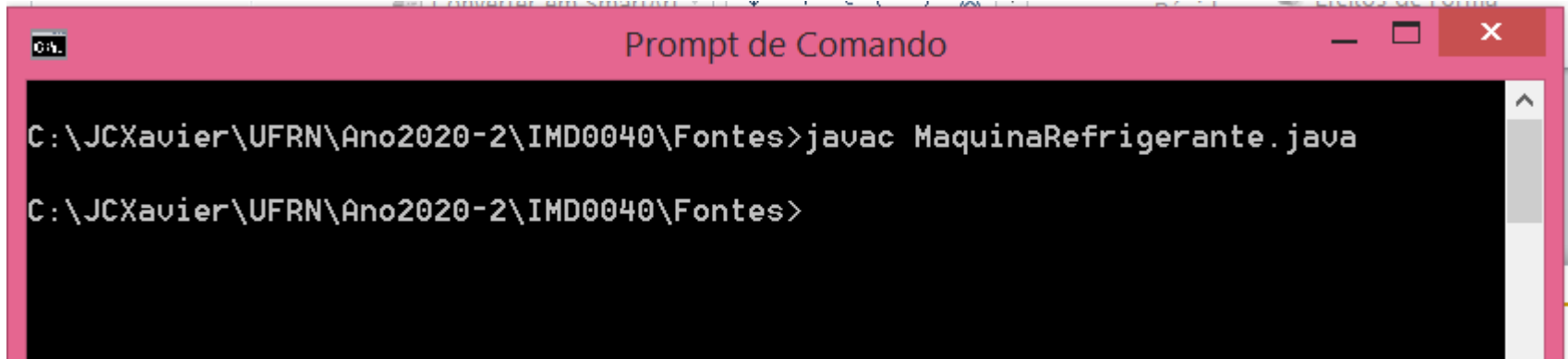
# Vamos começar???

```
1 public class MaquinaRefrigerante {
2
3     // Atributos.
4     private int preco;
5     private int balanco;
6     private int total;
7
8     // Método construtor para inicializar os atributos
9     public MaquinaRefrigerante(int valor) {
10         preco = valor;
11         balanco = 0;
12         total = 0;
13     }
14
15     // Retorna o preço
16     public int getPreco() {
17         return preco;
18     }
19
20     // Retorna o balanço corrente
21     public int getBalanco() {
22         return balanco;
23     }
24 }
```

# Vamos começar???

```
24
25 // Recebe um valor em dinheiro
26 public void inserirDinheiro(int valor){
27     balanco += valor;
28 }
29
30 // Imprimir o preço do refrigerante
31 public void imprimirPreco(){
32     System.out.println("#####");
33     System.out.println("# Preço #####");
34     System.out.println("# Refrigerante");
35     System.out.println("# R$ " + preco);
36     System.out.println("#####");
37 }
38 }
```

# Compilando a classe



```
C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>javac MaquinaRefrigerante.java
C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>
```

# Compilando a classe

```
C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>dir
O volume na unidade C é OS
O Número de Série do Volume é AE0E-1F41

Pasta de C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes

20/01/2021  18:21    <DIR>          .
20/01/2021  18:21    <DIR>          ..
20/01/2021  18:21                1.392 MaquinaRefrigerante.class
21/08/2016  12:00                1.159 MaquinaRefrigerante.java
                2 arquivo(s)                2.551 bytes
                2 pasta(s) 256.560.422.912 bytes disponíveis

C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>
```

# Erros de Sintaxe



A screenshot of a Windows Command Prompt window titled "Prompt de Comando". The window has a pink title bar with standard Windows window controls (minimize, maximize, close). The command prompt shows the following text:

```
C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>javac MaquinaRefrigerante.java
MaquinaRefrigerante.java:5: error: ';' expected
    private int balanco
                        ^
1 error

C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>
```

The error message indicates a syntax error in the file `MaquinaRefrigerante.java` at line 5, where a semicolon is expected after the declaration `private int balanco`. The error is highlighted with a caret (^) under the end of the line.



# Erros de Sintaxe

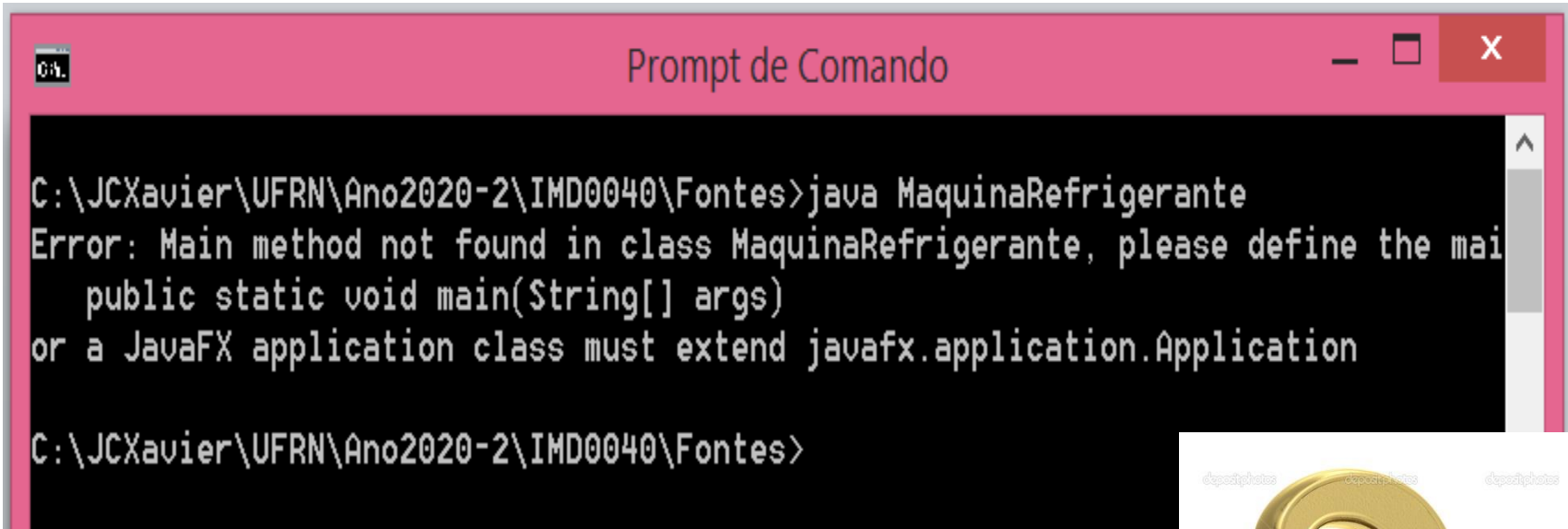


The screenshot shows a Windows Command Prompt window titled "Prompt de Comando". The command prompt is open at the directory `C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes`. The user has entered the command `javac MaquinaRefrigerante.java`. The output shows a compilation error on line 10 of `MaquinaRefrigerante.java`: `error: cannot find symbol`. The error points to the variable `valor` in the assignment `preco = valor;`. The error details are: `symbol: variable valor` and `location: class MaquinaRefrigerante`. The command prompt shows `1 error` and then returns to the prompt.

```
C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>javac MaquinaRefrigerante.java
MaquinaRefrigerante.java:10: error: cannot find symbol
    preco = valor;
              ^
    symbol:   variable valor
    location: class MaquinaRefrigerante
1 error

C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>
```

# Executando a classe



```
C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>java MaquinaRefrigerante
Error: Main method not found in class MaquinaRefrigerante, please define the main
  public static void main(String[] args)
or a JavaFX application class must extend javafx.application.Application

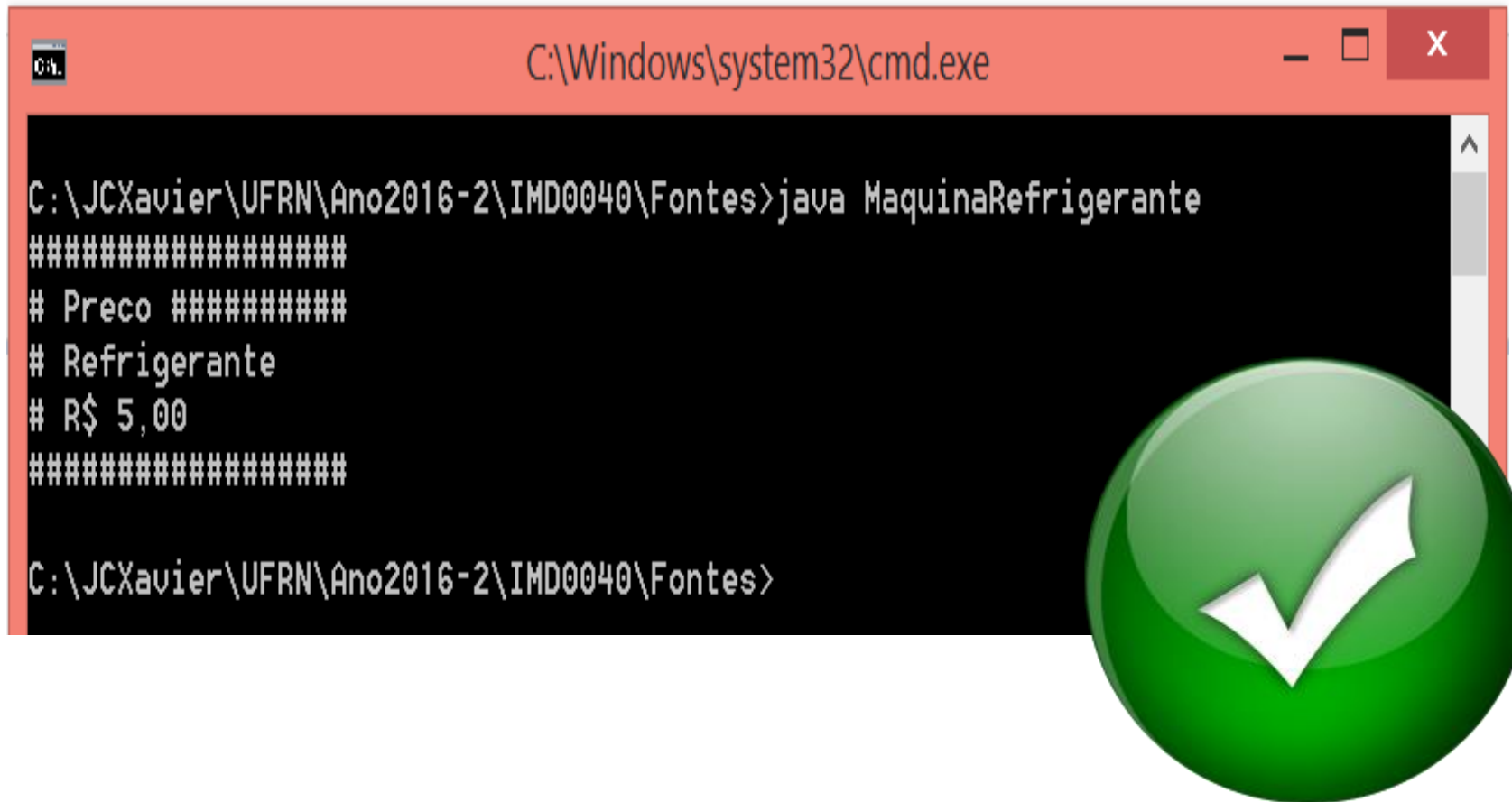
C:\JCXavier\UFRN\Ano2020-2\IMD0040\Fontes>
```



# Criando o Método Principal

```
30 // Imprimir o preço do refrigerante
31 public void imprimirPreco(){
32     System.out.println("#####");
33     System.out.println("# Preço #####");
34     System.out.println("# Refrigerante");
35     System.out.println("# R$ " + preco + ",00");
36     System.out.println("#####");
37 }
38
39 // Criando o método executável da classe
40 public static void main(String args[]){
41
42     // Vamos instanciar a classe
43     MaquinaRefrigerante maquina = new MaquinaRefrigerante(5);
44
45     // Chamando um método
46     maquina.imprimirPreco();
47
48 }
49
50 }
```

# Executando a classe



```
C:\Windows\system32\cmd.exe

C:\JCXavier\UFRN\Ano2016-2\IMD0040\Fontes>java MaquinaRefrigerante
#####
# Preco #####
# Refrigerante
# R$ 5,00
#####

C:\JCXavier\UFRN\Ano2016-2\IMD0040\Fontes>
```

# Boas Práticas



**Mas, porque?**

# Criando Classe de Visão

```
public class MaquinaRefrigeranteView {  
  
    // Criando o método executável da classe  
    public static void main(String args[]){  
  
        // Instanciando a classe -- criando objeto  
        MaquinaRefrigerante maquina = new MaquinaRefrigerante(5);  
  
        // Chamando um método  
        maquina.imprimirPreco();  
  
        // Inserir dinheiro na máquina  
        System.out.println("");  
        maquina.inserirDinheiro(20);  
  
        // Retornando o balanço corrente  
        System.out.println("");  
        System.out.println("Balanço atual => " + maquina.getBalanco());  
  
    }  
}
```

# Executando Classe de Visão

```
Windows PowerShell
PS C:\Users\annec\Downloads> java MaquinaRefrigeranteView
#####
# Preço #####
# Refrigerante
# R$ 5,00
#####

Valor inserido com sucesso!!!!

Balanco atual => 20
PS C:\Users\annec\Downloads>
```



# Máquina de Refrigerante

- ❑ Quais foram os problemas encontrados?
- ❑ Algum comportamento inadequado?



# Máquina de Refrigerante

- ❑ Quais foram os problemas encontrados?
- ❑ Algum comportamento inadequado?
  - ❖ Sem verificação de quantias inseridas:
    - Valor suficiente para comprar um bilhete?
    - Valor negativo.
  - ❖ Sem restituições (troco).

# Máquina de Refrigerante

- ❑ Método de Inserção de dinheiro:
  - ❖ Parâmetro do método `inserirDinheiro`
  - ❖ `int valor`
  - ❖ Quais são os valores válidos para um `int`?
- ❑ Como melhorar?
  - ❖ Não aceitar valores negativos.....

# Melhorando o código

## ❑ Uso de estruturas condicionais:

```
25 // Recebe um valor em dinheiro
26 public void inserirDinheiro(int valor){
27     if (valor > 0){
28         balanco += valor;
29         System.out.println("Valor inserido com sucesso!!!!");
30     }
31     else {
32         System.out.println("Use um valor positivo!!!");
33         System.out.println("Tente outra vez!!!");
34     }
35 }
```

# Melhorando o código

## ❑ Testando o código:

```
// Criando o método executável da classe
public static void main(String args[]){

    // Instanciando a classe -- criando objeto
    MaquinaRefrigerante maquina = new MaquinaRefrigerante(5);

    // Chamando um método
    maquina.imprimirPreco();

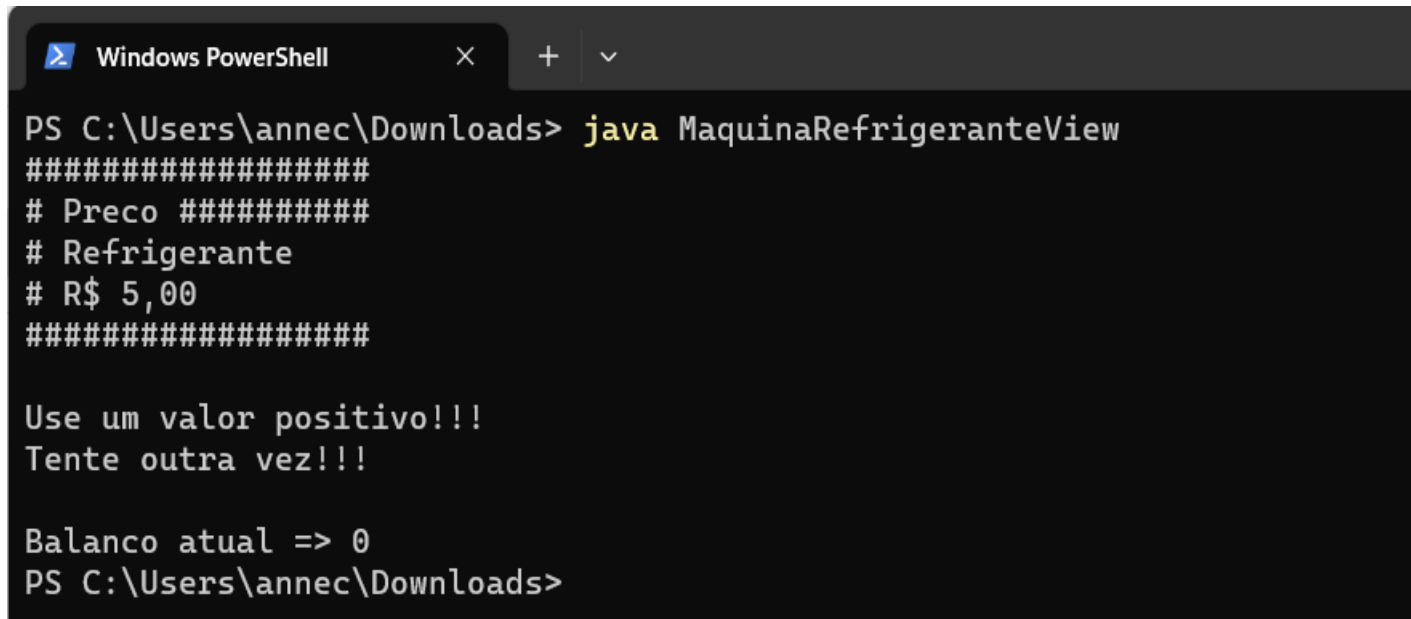
    // Inserir dinheiro na máquina
    System.out.println("");
    maquina.inserirDinheiro(0);

    // Retornando o balanço corrente
    System.out.println("");
    System.out.println("Balanço atual => " + maquina.getBalanco());

}
```

# Melhorando o código

## ❑ Testando o código:



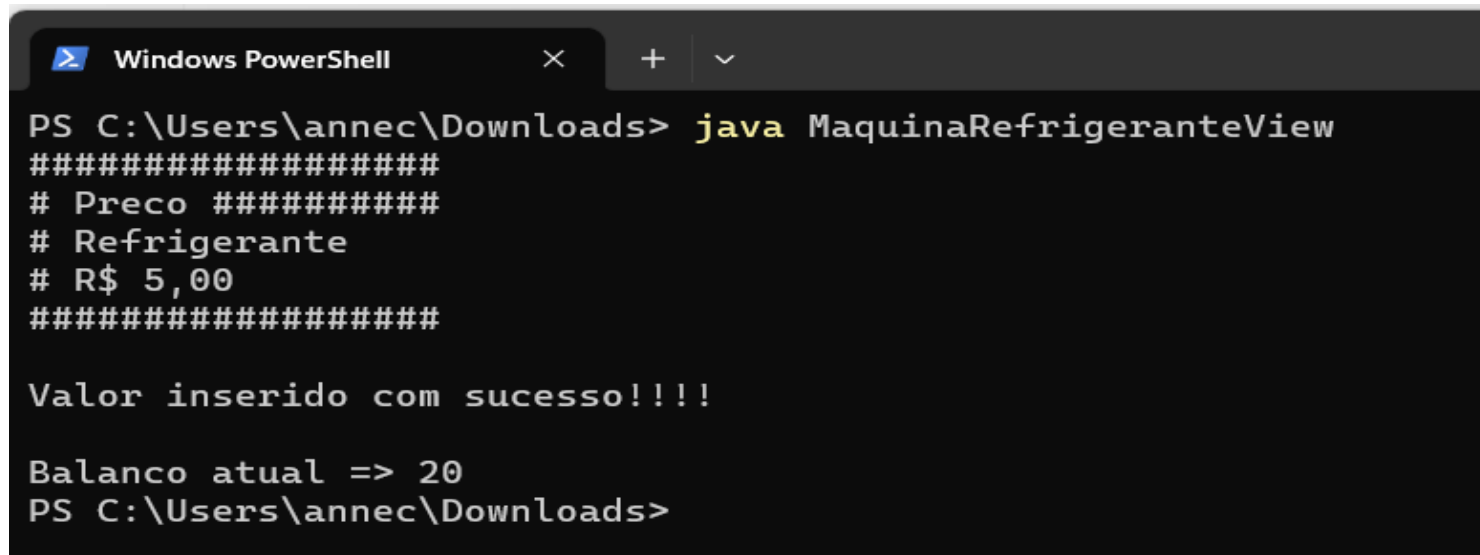
```
Windows PowerShell
PS C:\Users\annec\Downloads> java MaquinaRefrigeranteView
#####
# Preco #####
# Refrigerante
# R$ 5,00
#####

Use um valor positivo!!!
Tente outra vez!!!

Balanco atual => 0
PS C:\Users\annec\Downloads>
```

# Melhorando o código

❑ Testando o código:



```
Windows PowerShell
PS C:\Users\annec\Downloads> java MaquinaRefrigeranteView
#####
# Preco #####
# Refrigerante
# R$ 5,00
#####

Valor inserido com sucesso!!!!

Balanco atual => 20
PS C:\Users\annec\Downloads>
```

# Máquina de Refrigerante

- ❑ Método comprarRefrigerante ( ):
  - ❖ Não existe!!!!
  - ❖ O que ele deve fazer?
- ❑ Implemente o referido método.
  - ❖ Observação: verificar se tem dinheiro suficiente na máquina antes de imprimir o recibo.

# Melhorando o código

## ❑ Uso de estruturas condicionais 02:

```
// Comprar um refrigerante
public void comprarRefrigerante() {
    if(balanco >= preco) {

        System.out.println("#####");
        System.out.println("## Recibo ##");
        System.out.println("## Refrigerante ##");
        System.out.println("## R$ " + preco + ",00");
        System.out.println("#####");
        System.out.println();

        // Atualizar Montante Total de vendas da máquina.
        total = total + preco;

        // Atualizar o montante do cliente.
        balanco = balanco - preco;
    }
    else {
        System.out.println("Voce precisara inserir pelo menos: R$ " +
            (preco - balanco) + ",00");
    }
}
```



# Melhorando o código

## ❑ Testando o código:

```
// Criando o método executável da classe
public static void main(String args[]){

    // Instanciando a classe -- criando objeto
    MaquinaRefrigerante maquina = new MaquinaRefrigerante(5);

    // Chamando um método
    maquina.imprimirPreco();

    // Inserir dinheiro na máquina
    System.out.println("");
    maquina.inserirDinheiro(20);

    // Retornando o balanço corrente
    System.out.println("");
    System.out.println("Balanço atual => " + maquina.getBalanco());

    // comprar na máquina
    maquina.comprarRefrigerante();

}
```

# Melhorando o código

## ❑ Testando o código:

```
Windows PowerShell
PS C:\Users\annec\Downloads> java MaquinaRefrigeranteView
#####
# Preco #####
# Refrigerante
# R$ 5,00
#####

Valor inserido com sucesso!!!!

Balanco atual => 20
#####
## Recibo ##
## Refrigerante ##
## R$ 5,00
#####

PS C:\Users\annec\Downloads>
```

# Melhorando o código

## ❑ Testando o código:

```
// Chamando um método
maquina.imprimirPreco();

// Inserir dinheiro na máquina
System.out.println("");
maquina.inserirDinheiro(4);

// Retornando o balanço corrente
System.out.println("");
System.out.println("Balanço atual => " + maquina.getBalanco());

// comprar na máquina
maquina.comprarRefrigerante();
```

# Melhorando o código

## ❑ Testando o código:

```
Windows PowerShell
PS C:\Users\annec\Downloads> java MaquinaRefrigeranteView
#####
# Preco #####
# Refrigerante
# R$ 5,00
#####

Valor inserido com sucesso!!!!

Balanco atual => 4
Voce precisara inserir pelo menos: R$ 1,00
PS C:\Users\annec\Downloads>
```

# Melhorando o código 03

- ❑ Ainda podemos melhorar o código?
- ❑ O que **ainda** está faltando????

# Melhorando o código 03

- ❑ Ainda podemos melhorar o código?
- ❑ O que **ainda** está faltando????
- ❑ Um método para restituir dinheiro (devolver o troco).

# Melhorando o código 03

- ❑ Como escrever o método **devolverTroco()** ?
  - ❖ Qual o objetivo desse método?
  - ❖ Como fazer?
  - ❖ O que acontece com o saldo atual da máquina quando uma restituição é realizada?

# Melhorando o código 03

❑ O método **devolverTroco()**:

```
83 public int devolverTroco() {  
84     int valorParaDevolver;  
85     valorParaDevolver = balanco;  
86     balanco = 0;  
87     return valorParaDevolver;  
88 }
```

```
// Atributos.  
private int preco;  
private int balanco;  
private int total;  
private boolean darTroco;
```

```
// Método construtor para inicializar os atributos  
public MaquinaRefrigerante(int valor) {  
    preco = valor;  
    balanco = 0;  
    total = 0;  
    darTroco = false;  
}
```



# Melhorando o código 03

## ❑ Alterando o método **comprarRefrigerante()**:

```
// Comprar um refrigerante
public void comprarRefrigerante() {
    if (balanco >= preco) {

        System.out.println("#####");
        System.out.println("## Recibo ##");
        System.out.println("## Refrigerante ##");
        System.out.println("## R$ " + preco + ",00");
        System.out.println("#####");
        System.out.println();

        // Atualizar Montante Total de vendas da máquina.
        total = total + preco;

        // Atualizar o montante do cliente.
        balanco = balanco - preco;

        if (balanco > 0 && darTroco == true) {
            System.out.println("#####");
            System.out.println("## Troco ##");
            System.out.println("## R$ " + devolverTroco() + ",00");
            System.out.println("#####");
        }
    }
}
```

# Melhorando o código 03

## ❑ Inserindo o método **setDarTroco()**:

```
public class MaquinaRefrigerante02 {  
  
    // Atributos ....  
    private int preco;  
    private int balanco;  
    private int total;  
    private boolean darTroco;  
  
    // Método construtor para inicializar os atributos  
    public MaquinaRefrigerante02(int valor) {  
        preco = valor;  
        balanco = 0;  
        total = 0;  
        darTroco = false;  
    }  
  
    // Setar darTroco  
    public void setDarTroco() {  
        darTroco = true;  
    }  
}
```

# Melhorando o código 03

## ❑ Alterando a **View**:

```
// Inserir dinheiro na máquina
System.out.println("");
maquina.inserirDinheiro(15);

// Retornando o balanço corrente
System.out.println("");
System.out.println("Balanço atual => " + maquina.getBalanco());

// comprar na máquina
maquina.comprarRefrigerante();

maquina.setDarTroco();

// comprar na máquina
maquina.comprarRefrigerante();
```

# Melhorando o código

## ❑ Testando o código:

```
Windows PowerShell
PS C:\Users\annec\Downloads> java MaquinaRefrigeranteView
#####
# Preco #####
# Refrigerante
# R$ 5,00
#####

Valor inserido com sucesso!!!!

Balanco atual => 15
#####
## Recibo ##
## Refrigerante ##
## R$ 5,00
#####

#####
## Recibo ##
## Refrigerante ##
## R$ 5,00
#####

#####
## Troco ##
## R$ 5,00
#####
PS C:\Users\annec\Downloads>
```

# Perguntas ...



# Obrigado!!

