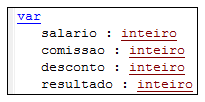
VARIÁVEIS: Tipos de Dados

**Explicação**

Em nosso programa, temos 4 variáveis definidas: 

Essas 4 variáveis são do tipo **inteiro**.

Isso indica que elas podem armazenar números inteiros, ou seja, sem casas decimais.

**Explicação**

O erro aconteceu pois a variável **salario** está definida como tipo de dados "inteiro". Ao informar o valor "120,55", informamos um valor com casas decimais, gerando o erro na atribuição do valor para a variável.

**Tipos de Dados**

**Tipos de Dados**

Quando estamos prevendo a execução de um programa, temos de prever também qual o **tipo de dados** a variável vai receber.

Por exemplo: se você deseja que o usuário informe o "nome da pessoa", a variável **nome** deverá ser do tipo **caractere**, não poderá ser do tipo inteiro.

Em nosso programa, estamos tratando de salário. Logo, as variáveis tem que ser preparadas para receberem dados do tipo **"real"**. O tipo de dados **"real"** aceita valores decimais, separados por vírgula.

**Tipos de Dados**

**Tipos de Dados**

O resultado foi apresentado no tipo de dados **real**. Esse tipo de dados permite a digitação de números com casas decimais.

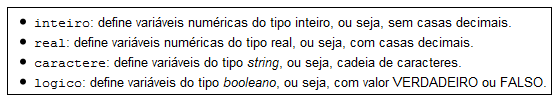
**Tipos de Dados**

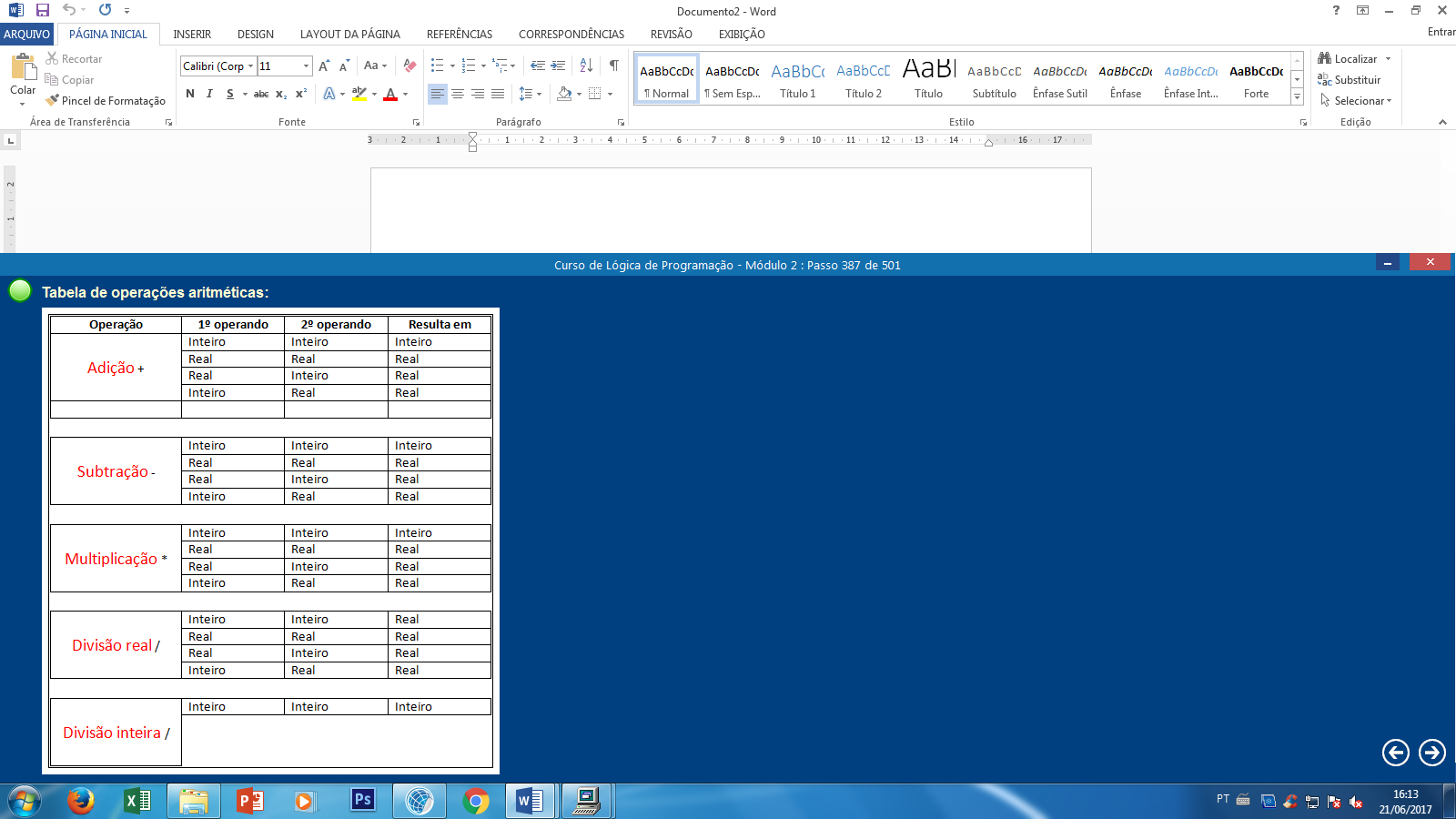
**Tipos de Dados nas Linguagens de Programação**

Existem centenas de linguagens de programação. Cada uma delas possui uma estrutura para o "tipo de dados" das variáveis. Então, quando você começar a estudar uma determinada linguagem de programação, é interessante procurar pelos tipos de dados aceitos naquela linguagem.

Isso é uma variação que vai existir de linguagem para linguagem.

**No programa VisuAlg** que estamos utilizando para o aprendizado, são suportados 4 tipos de dados para as variáveis:





**Explicação**

Resumo do algoritmo:

- O algoritmo solicita ao usuário informar 2 números  
- Os números são armazenados nas variáveis **primeiro** e **segundo**  
- Após receber, o programa atribui um valor para a variável **resultado**  
- A variável *resultado* recebe a operação: **primeiro \* segundo** - é uma multiplicação

**Explicação**

O programa executou corretamente pois recebeu 2 valores do tipo **inteiro**.

Segundo a tabela de operações aritméticas, quando se trata da multiplicação de 2 números inteiros, o resultado gerado também será um número inteiro.

Verifique: 

**Tipos de Dados**

Lembre-se:

Para evitar problema com o tipo de dados das variáveis, é sempre útil consultar a tabela de operações aritméticas. Através dela conseguimos visualizar qual o tipo de dados mais recomendado para cada operação.

Para operações de cálculo, o tipo *Real* será sempre o utilizado, pois a grande maioria dos cálculos de um programa vai envolver números decimais.

O tipo *Inteiro* deve ser utilizado para dados específicos, por exemplo: idade da pessoa; número de filhos; número de carros que possui, etc.

**Tipos de Dados**

Dica: para as 4 variáveis das notas, utilizaremos o tipo **real** pois as notas podem ser digitadas com casas decimais

**Tipos de Dados**

**Como calcular a Média Aritmética**

A média aritmética simples é a mais utilizada no nosso dia-a-dia.

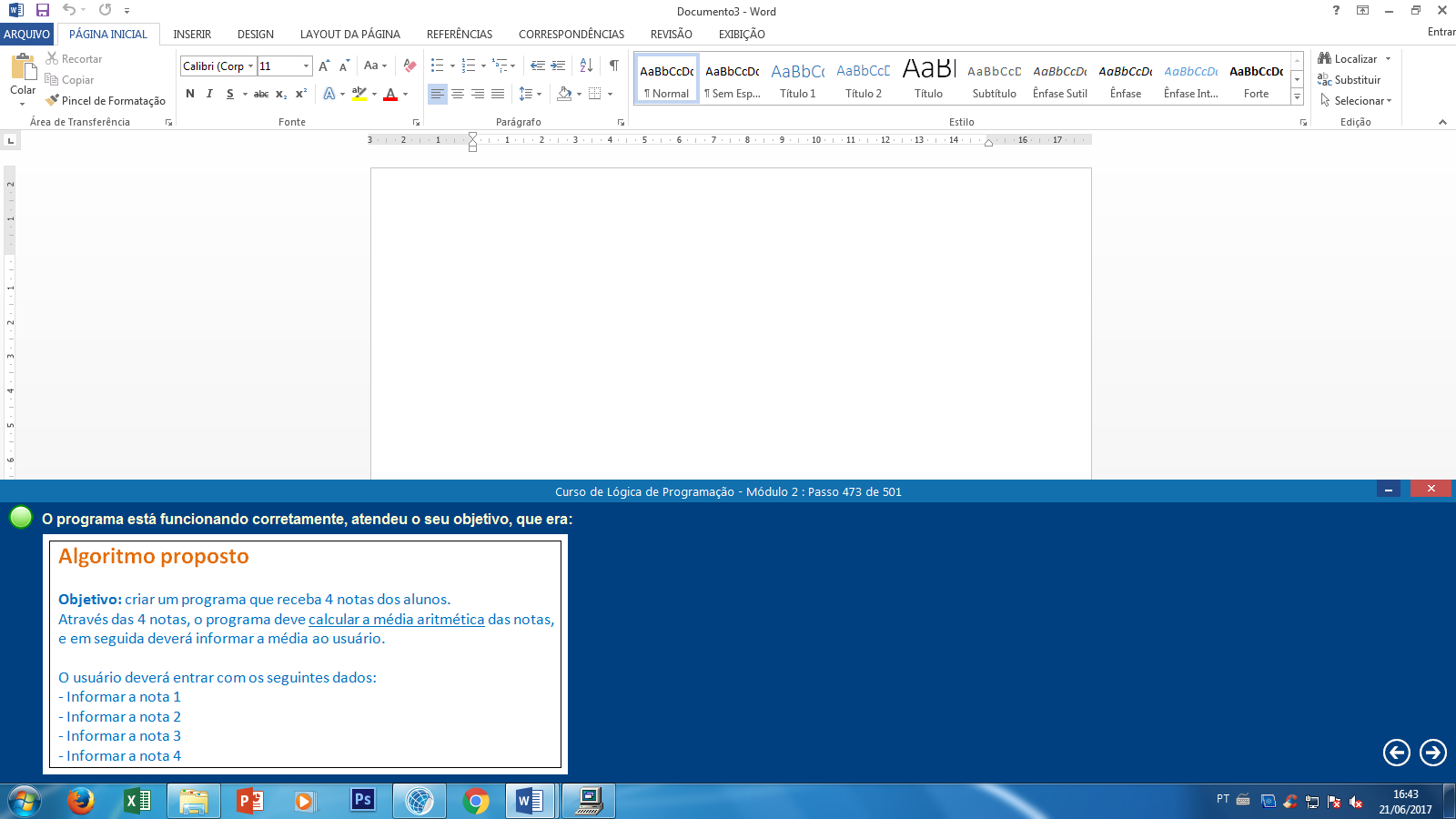
No caso das notas ela é obtida dividindo-se a soma de todas as notas pelo número notas (avaliações).

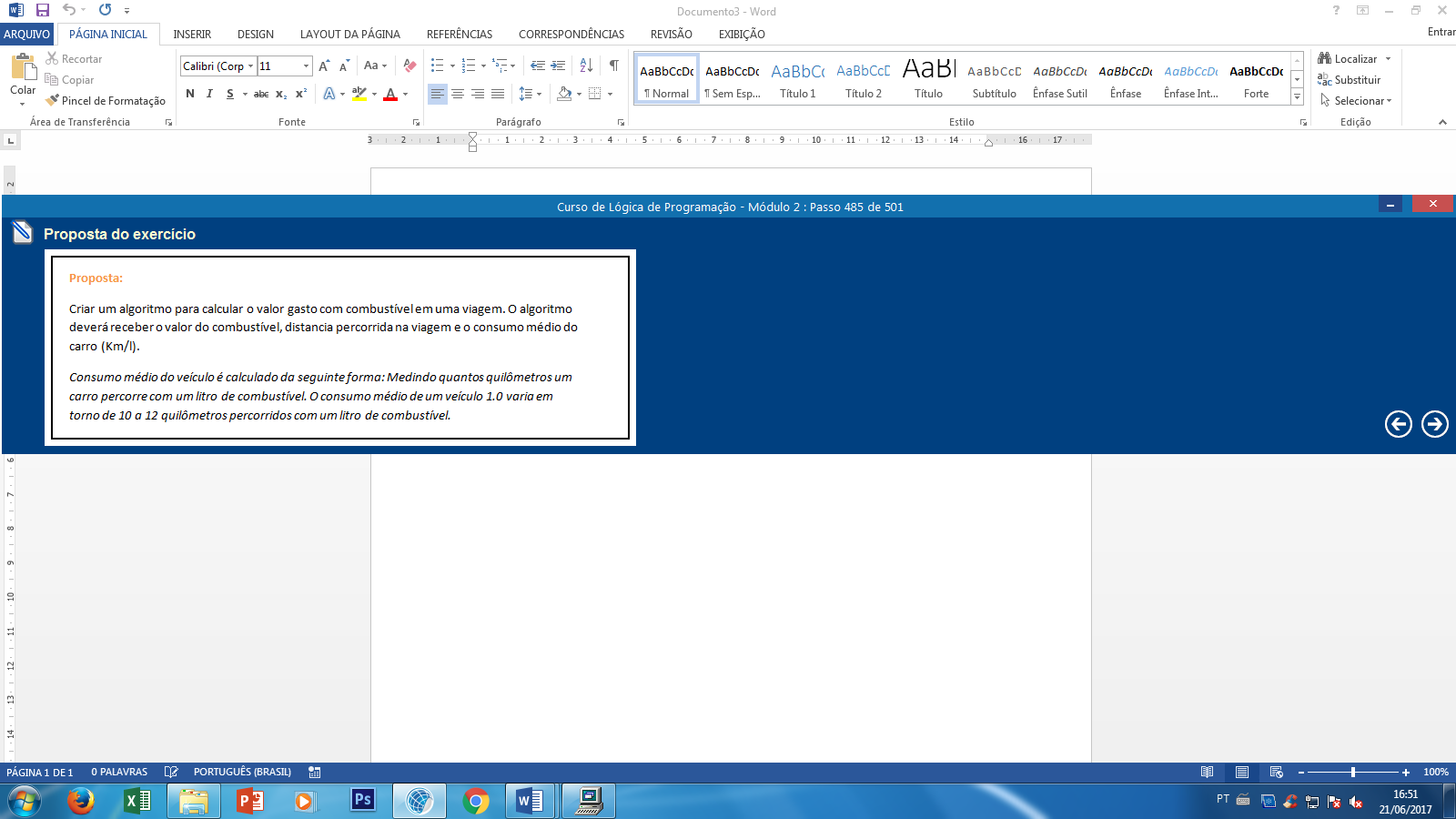
Exemplo:

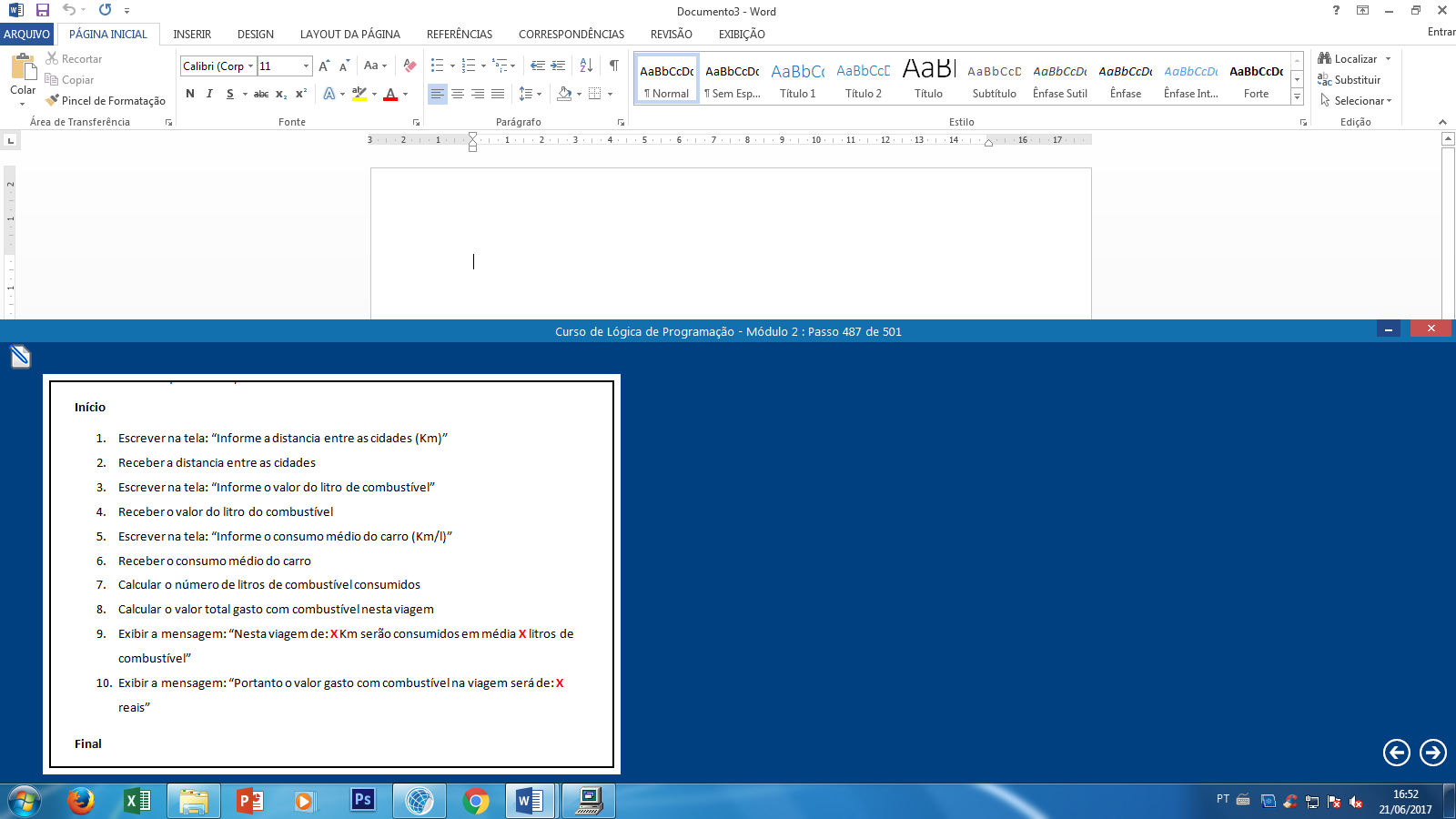
Um aluno tirou as notas 5, 7, 9 e 10 em quatro provas.   
A sua média será (5 + 7 + 9 + 10) / 4 = 7.75

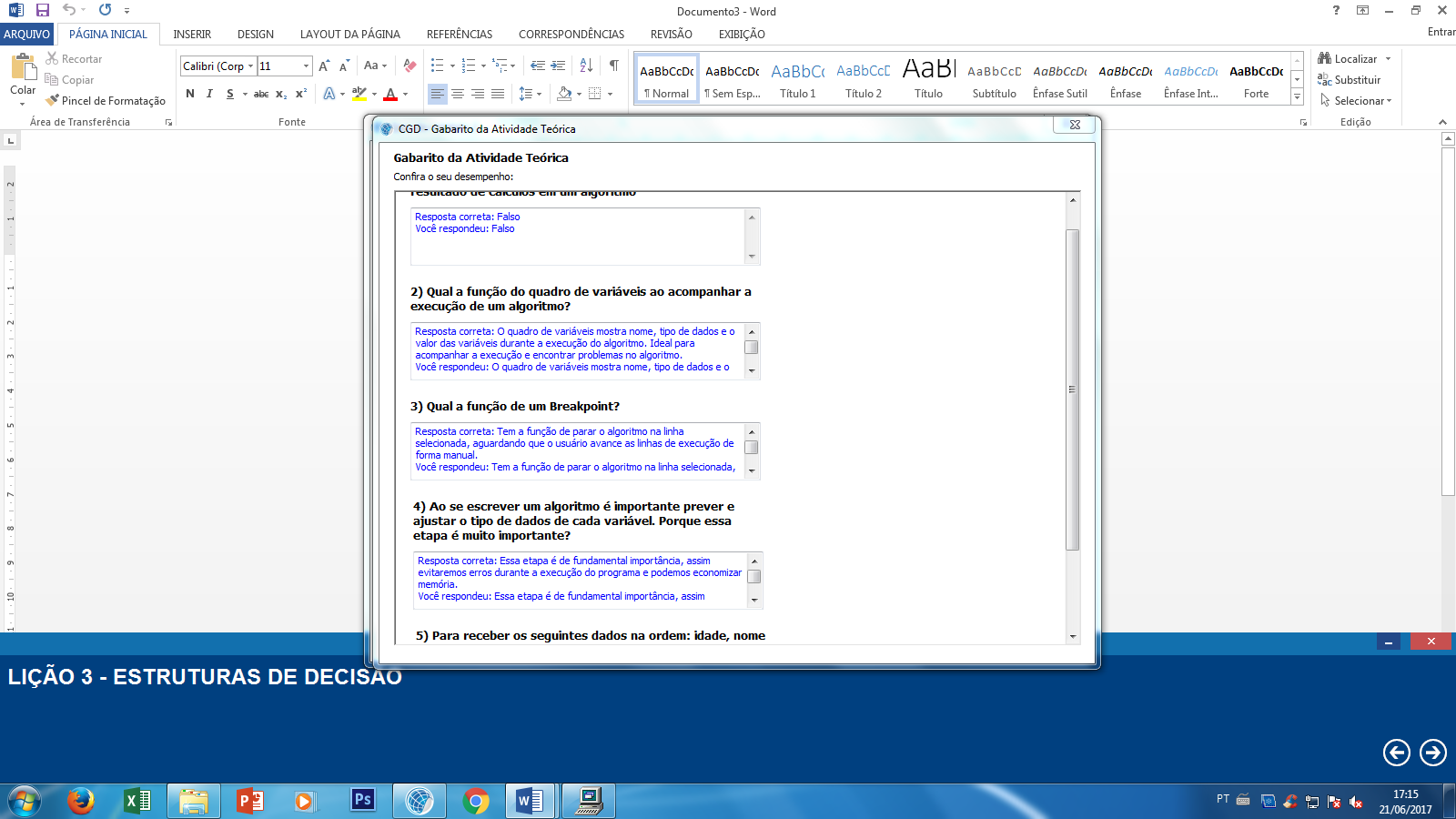
Temos de dividir a soma das notas, pelo número de notas somadas.

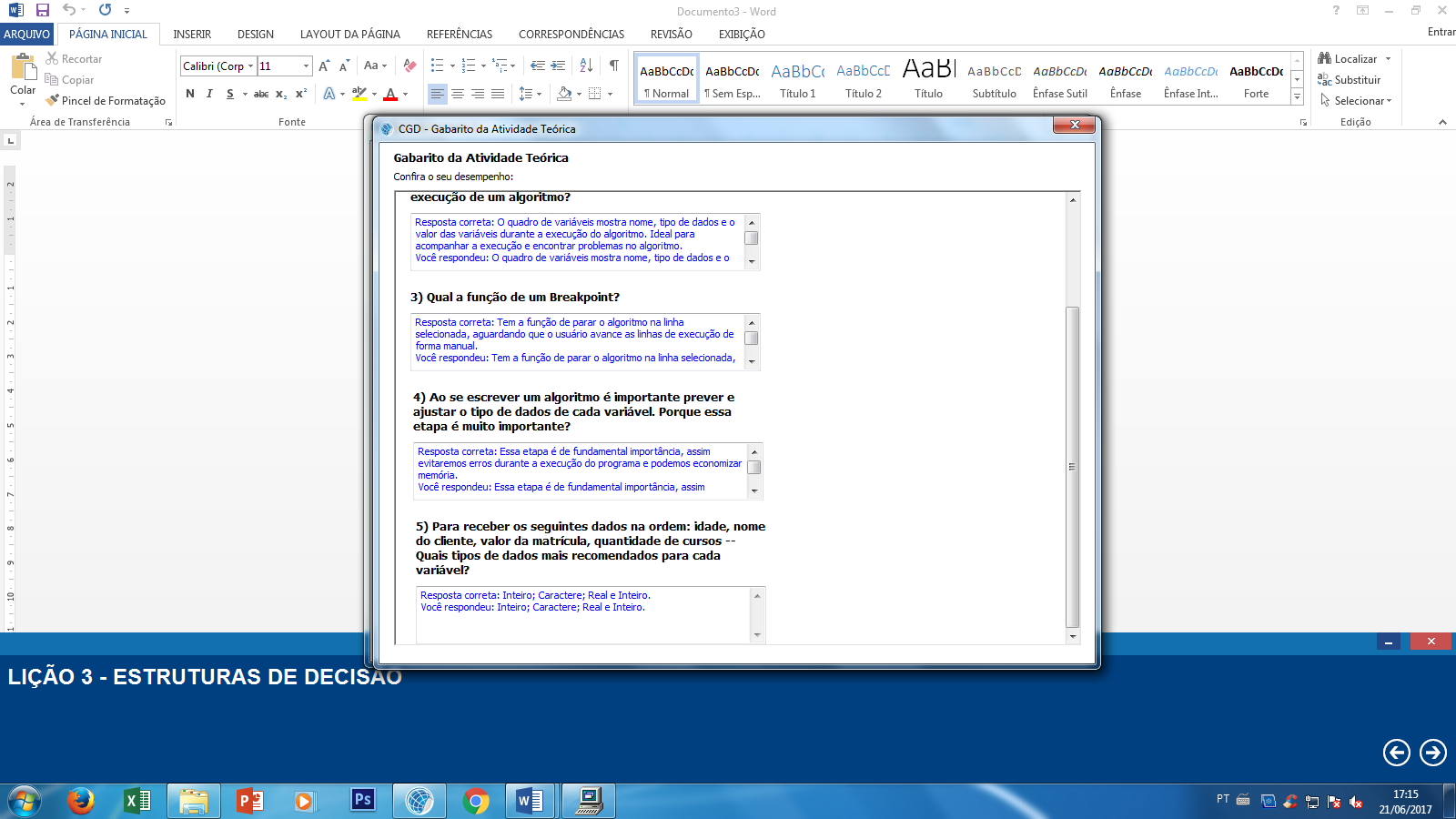
AO UTILIZAR O SINAL <- ESTAMOS INDICANDO UMA ATRIBUIÇÃO











ESTRUTURAS DE DECISÃO

ESTRUTURA SEQUENCIAIS

**Estruturas Sequenciais**

**Estruturas Sequenciais**

Origem: Wikipédia, a enciclopédia livre.

Em ciência da computação, uma estrutura sequencial é uma estrutura de desvio do fluxo de controle presente em linguagens de programação, que realiza um conjunto predeterminado de comandos de forma sequencial, de cima para baixo, na ordem em que foram declarados.

Sua estrutura básica é a repetição de comandos sequencialmente:

Ação1;  
Ação2;  
Ação3;  
Ação4;  
Ação5;

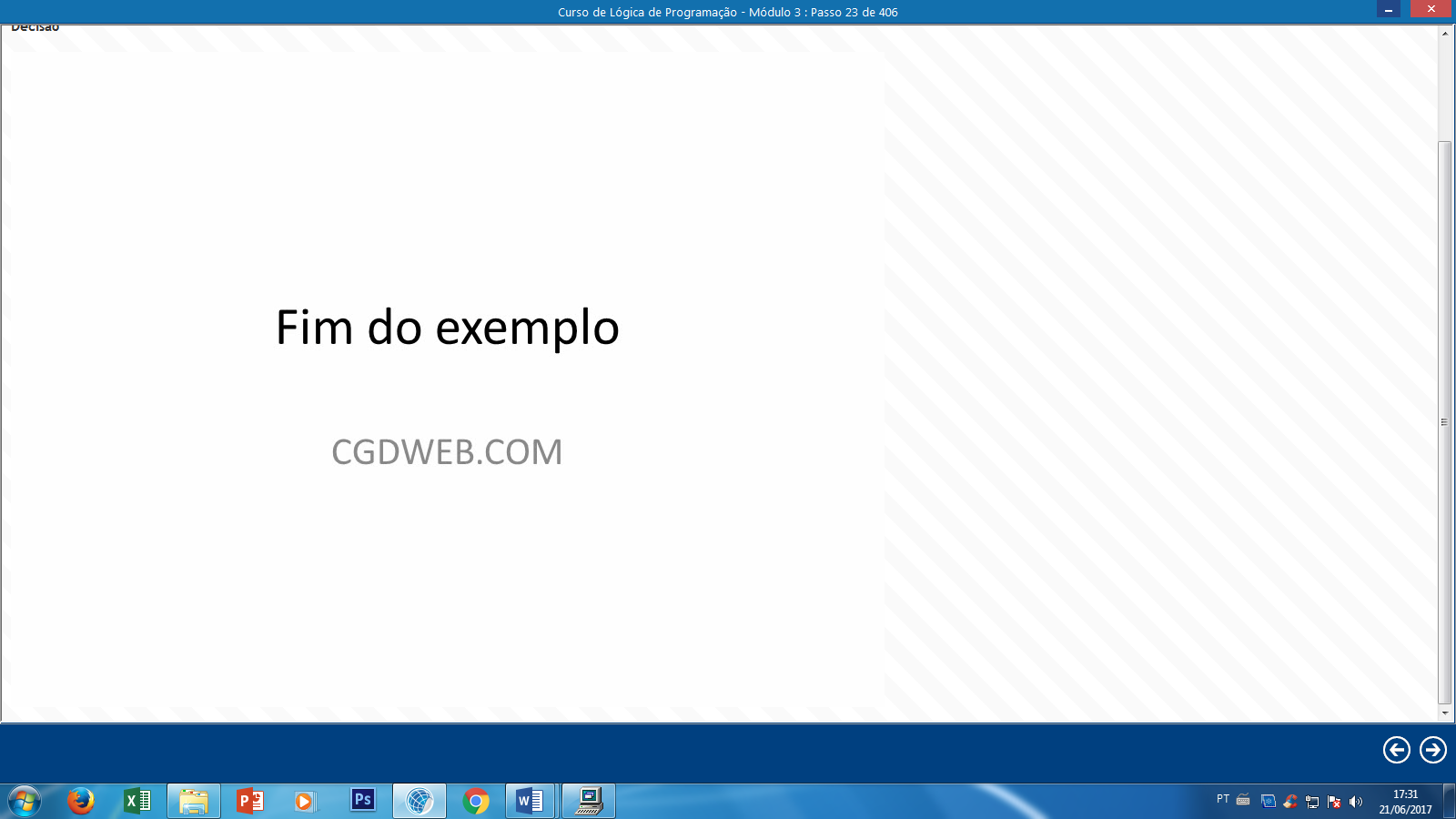
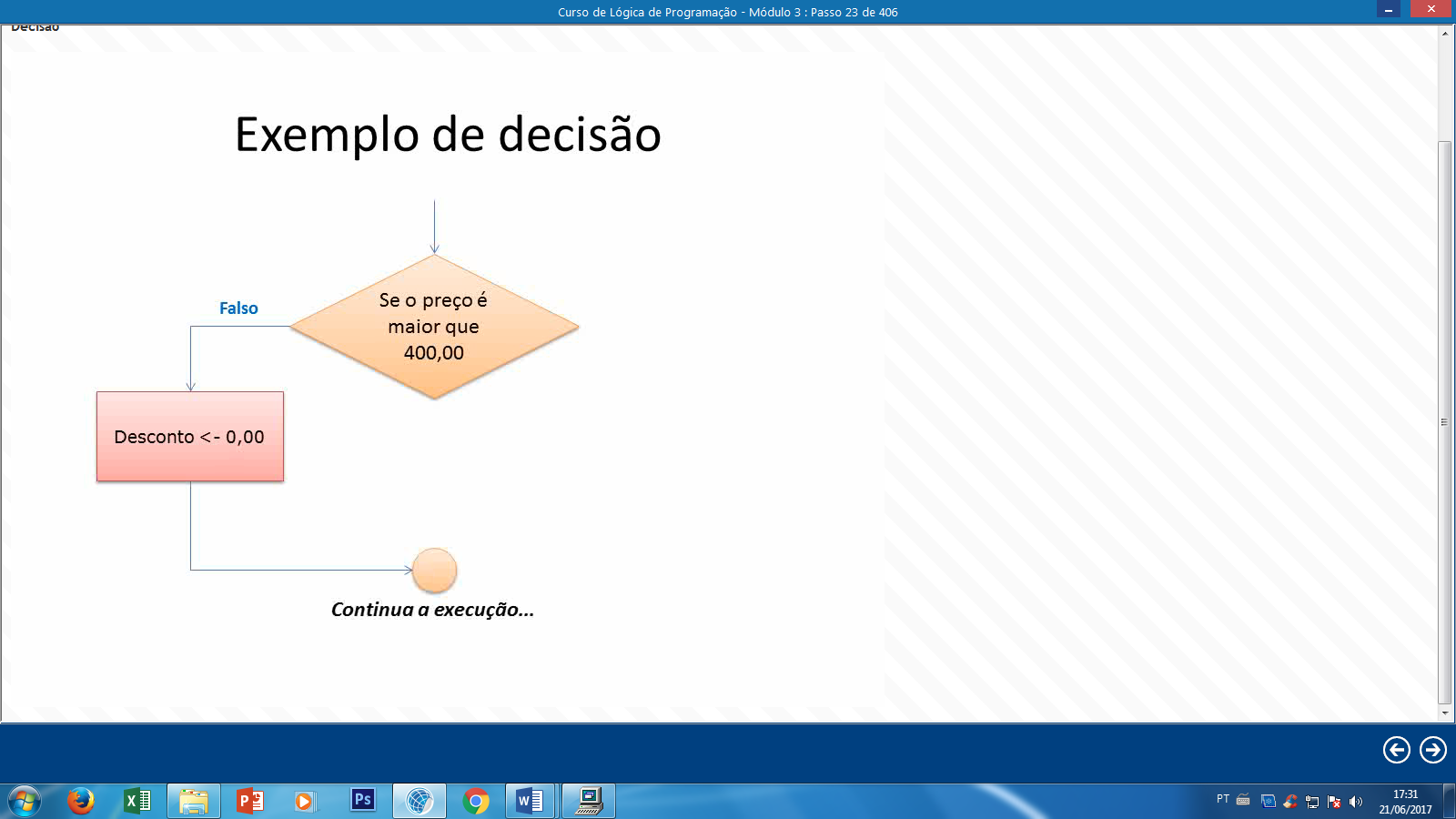
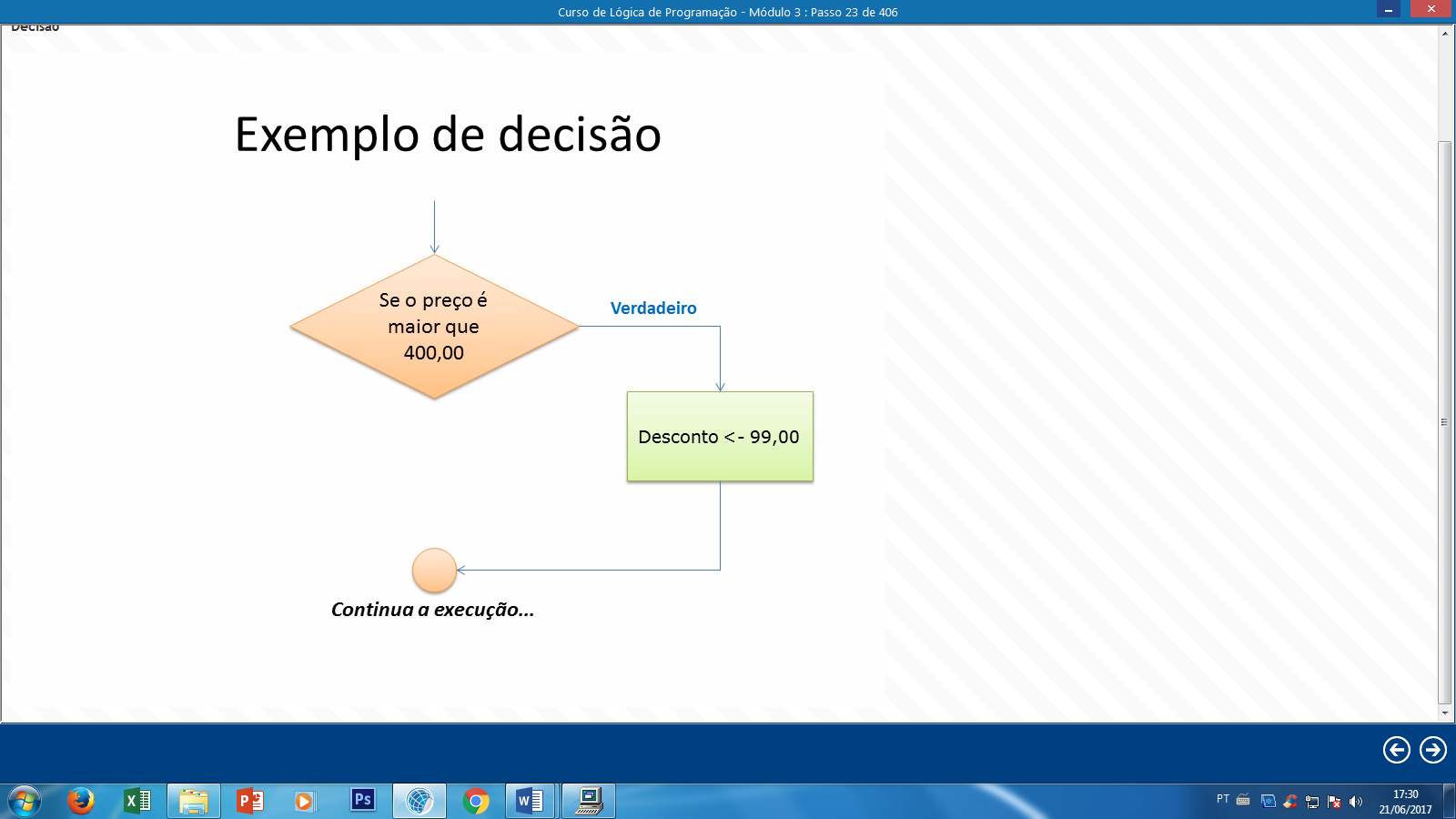
**Estruturas Sequenciais**

Esse algoritmo é uma **estrutura sequencial**. Ele nunca será diferente disso que está apresentado, ou seja, ele não contém desvios em seu código.

Um programa de computador pode conter um **desvio** no código, quando ele possui uma condição.

Por exemplo: *Pense um programa que calcula o preço da entrada no cinema...*   
Se a idade é maior ou igual a 18 anos, então, um preço será oferecido.   
Caso contrário, idade menor que 18, um outro preço será oferecido.

Quando um programa não possui desvios (condições) e nem repetições, então dizemos que o programa é uma estrutura sequencial. O código será sempre executado da mesma maneira, imperativamente, na sequencia como foi escrito.

ESTRUTURA DE DECISÃO/ ESTRUTURA DE SELEÇÃO

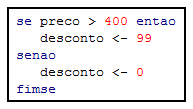
As estruturas de decisão permitem que um programa tome uma decisão no código, isto é, de acordo com uma condição. O programa faz o teste com o resultado de uma expressão lógica. De acordo com o resultado, o programa sabe qual o caminho deve seguir.

**Explicação**

**Analise o exemplo acima:**

O programa obtém o resultado da expressão lógica. A expressão lógica é a seguinte: **"O preço do produto é maior que 400?"**

**Se** Verdadeiro: então o Desconto passa a ser de R$ 99,00  
**Senão** (Falso): então o Desconto passa a ser de R$ 0,00

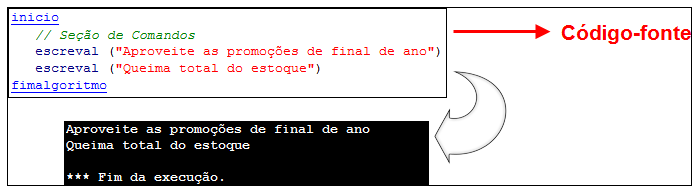
Pensando no código do algoritmo, ficaria assim: 

Um exemplo de estrutura de decisão do tipo SE, é lembrar-se da função SE utilizada no programa Excel. No Excel utilizamos a função SE para "testar uma condição", e, dependendo do resultado da condição, apresentar um determinado valor.

ESTRUTURAS DE DECISÃO: CONDICIONAL SIMPLES – SE

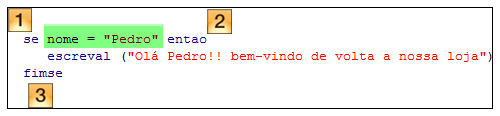
**Explicação**

Esse algoritmo possui apenas **instruções sequenciais**, isto é, não existe decisão no código-fonte do programa. Com isso, sempre que você o executar, vai obter o seguinte resultado:



**Explicação**

A estrutura de decisão SE, funciona assim:



**1.**Inicia a condição SE

Destaque em verde = é a condição lógica que está sendo testada. Nesse exemplo, testa se a variável nome tem o valor igual a "Pedro"

**2.** Então, indica a continuação da execução  
**3.** Indica o final da condição SE

Temos que imaginar a leitura da condição SE da seguinte forma:

**SE o valor da variável nome for igual a ""Pedro"", então escreva "Olá Pedro!! Bem-vindo de volta à nossa loja!"**