OPERADOR LÓGICO DO TIPO *NOT*

**Operador Lógico NOT**

Um operador lógico do tipo **NOT** é utilizado para realizar a negação de um valor. Ele não possui uma tabela verdade aplicável, pois é mais simples. Esse operador apenas faz a negação de um valor, por exemplo:

Operador A tem o valor "Verdadeiro" (true)

Se aplicar o operador NOT... ficaria assim: **Not Operador A** *= Falso* (false)

Ele sempre retorna o contrário, vejamos outro caso.

A agora tem o valor "false". Se utilizarmos: NOT A, então A passa a ter o valor "true".

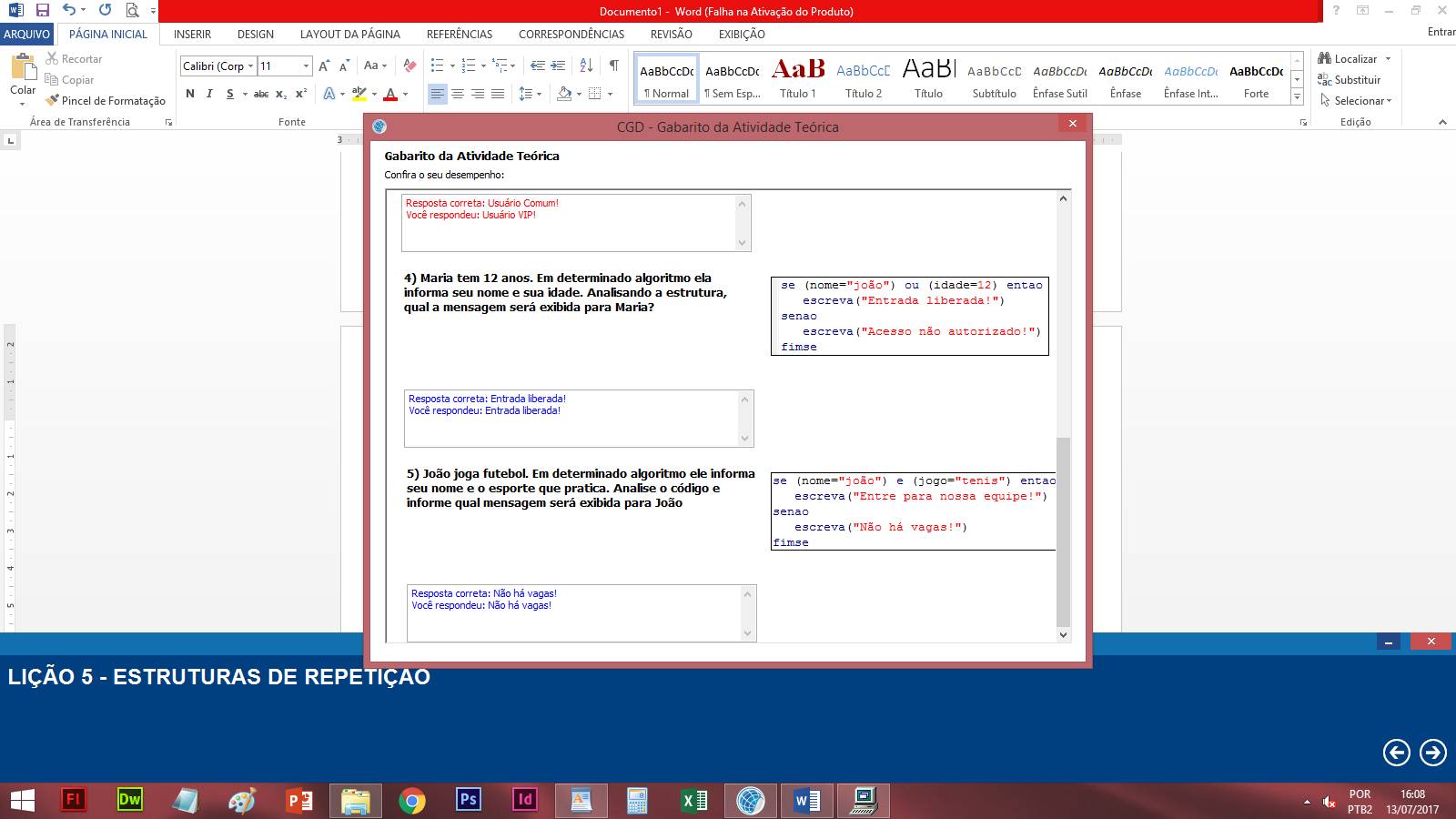
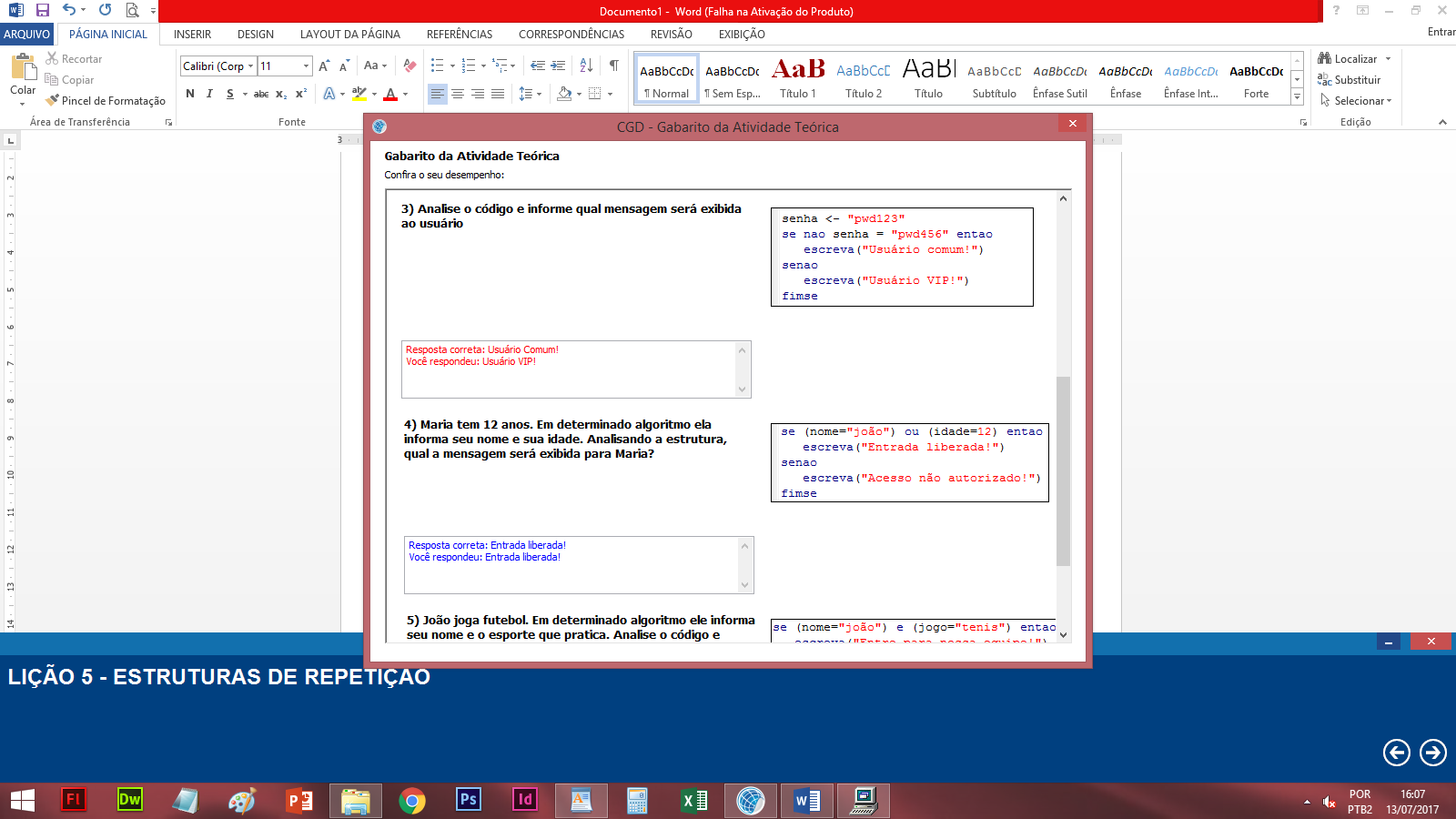
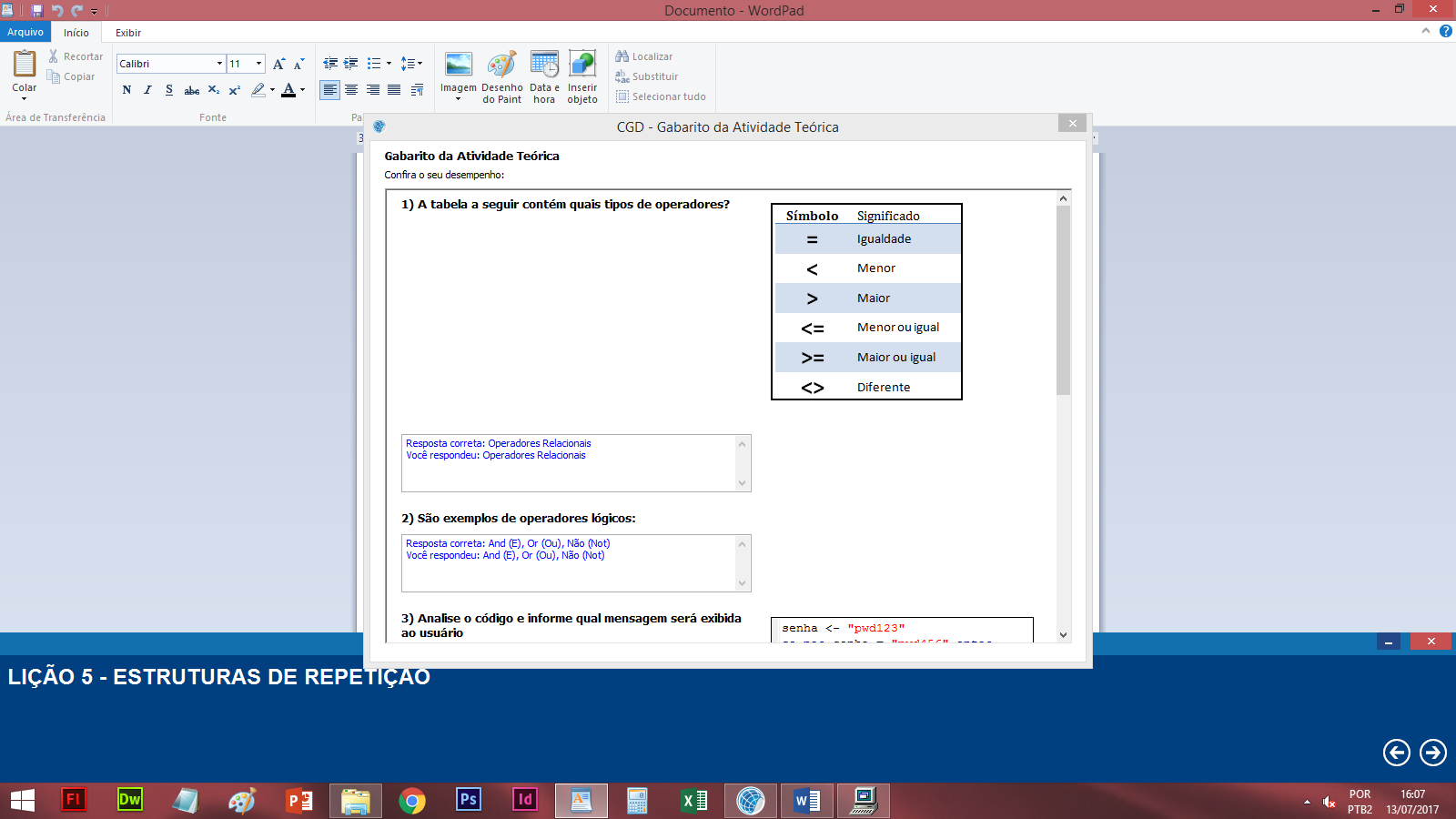
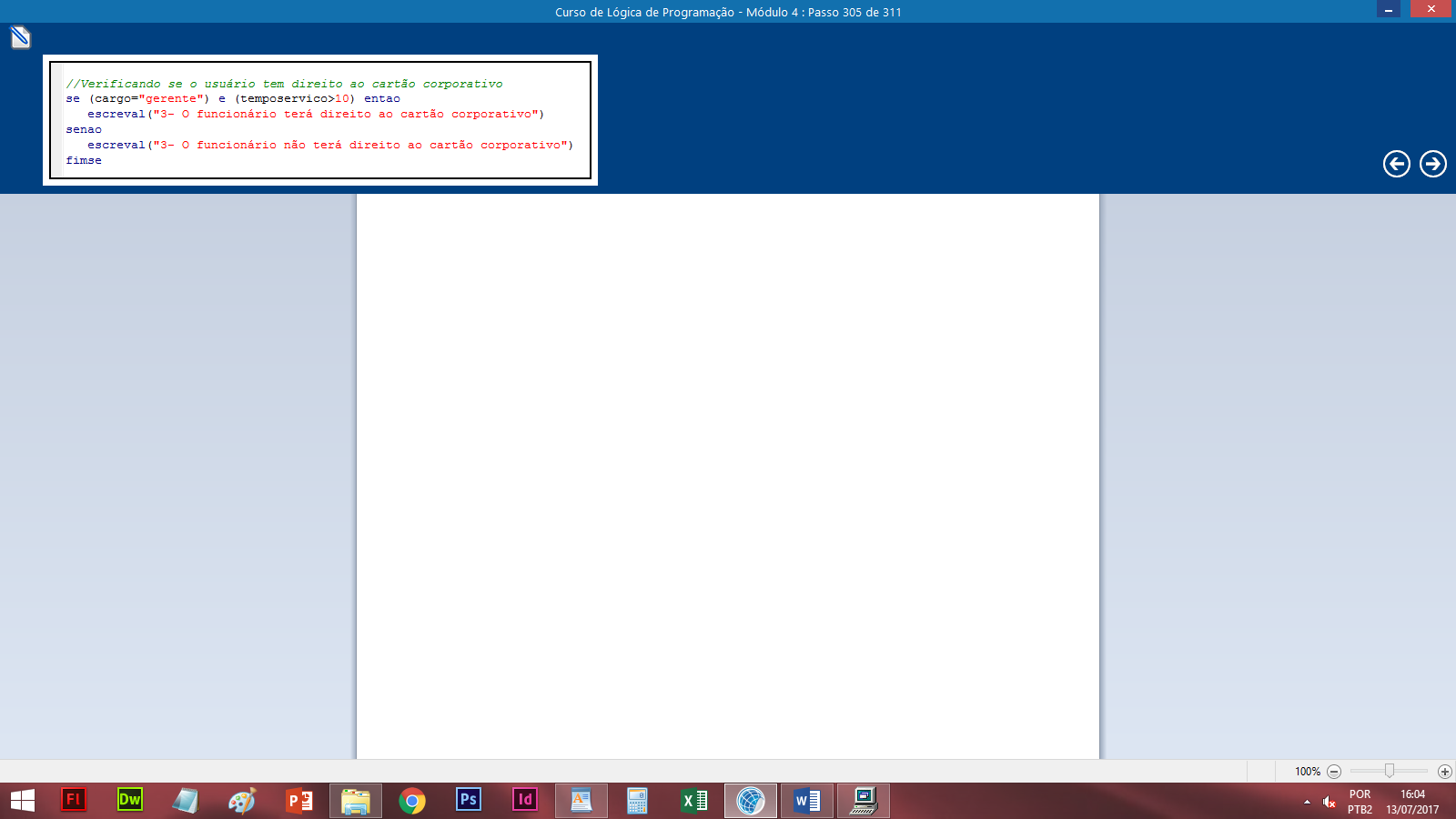
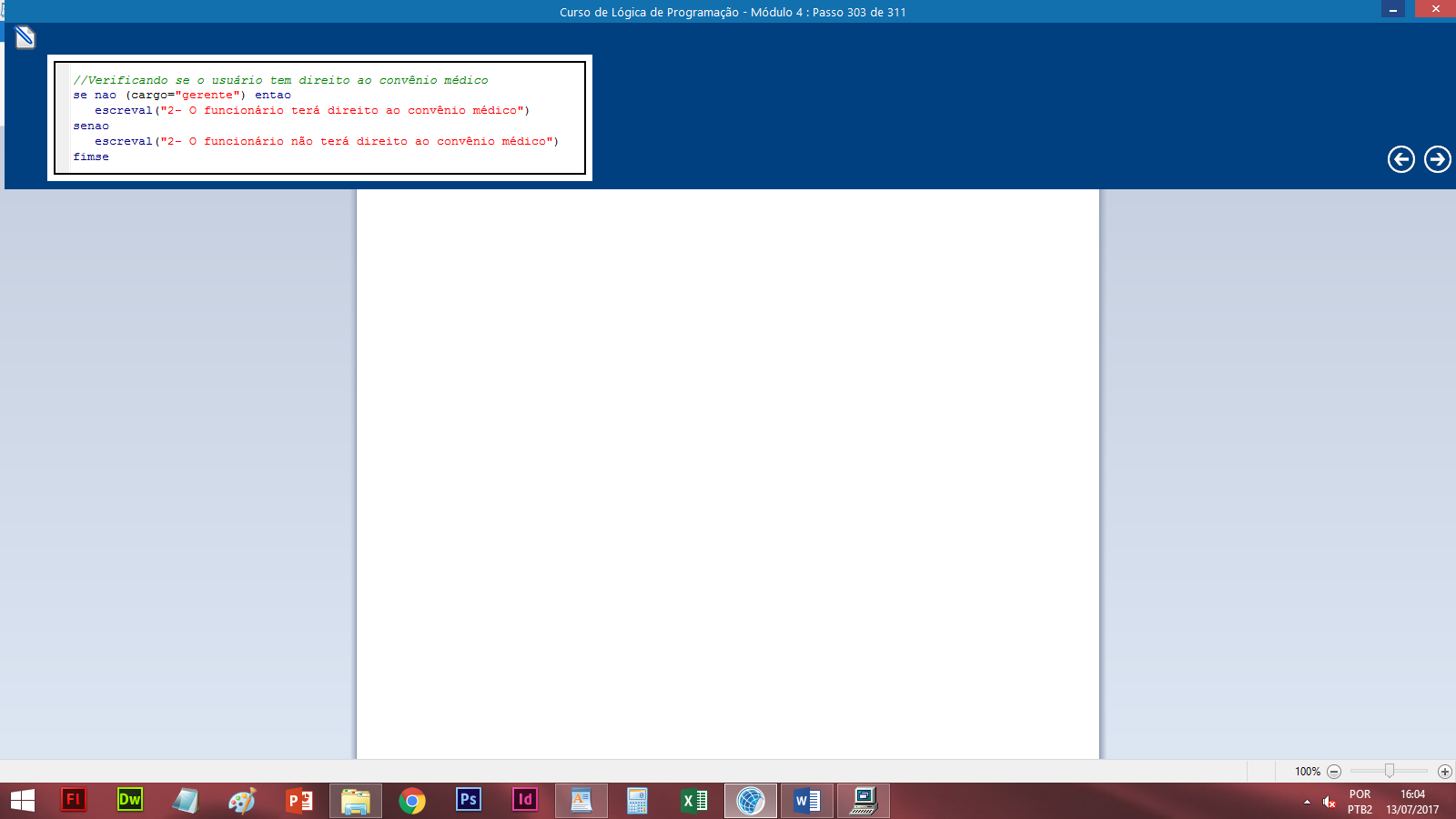
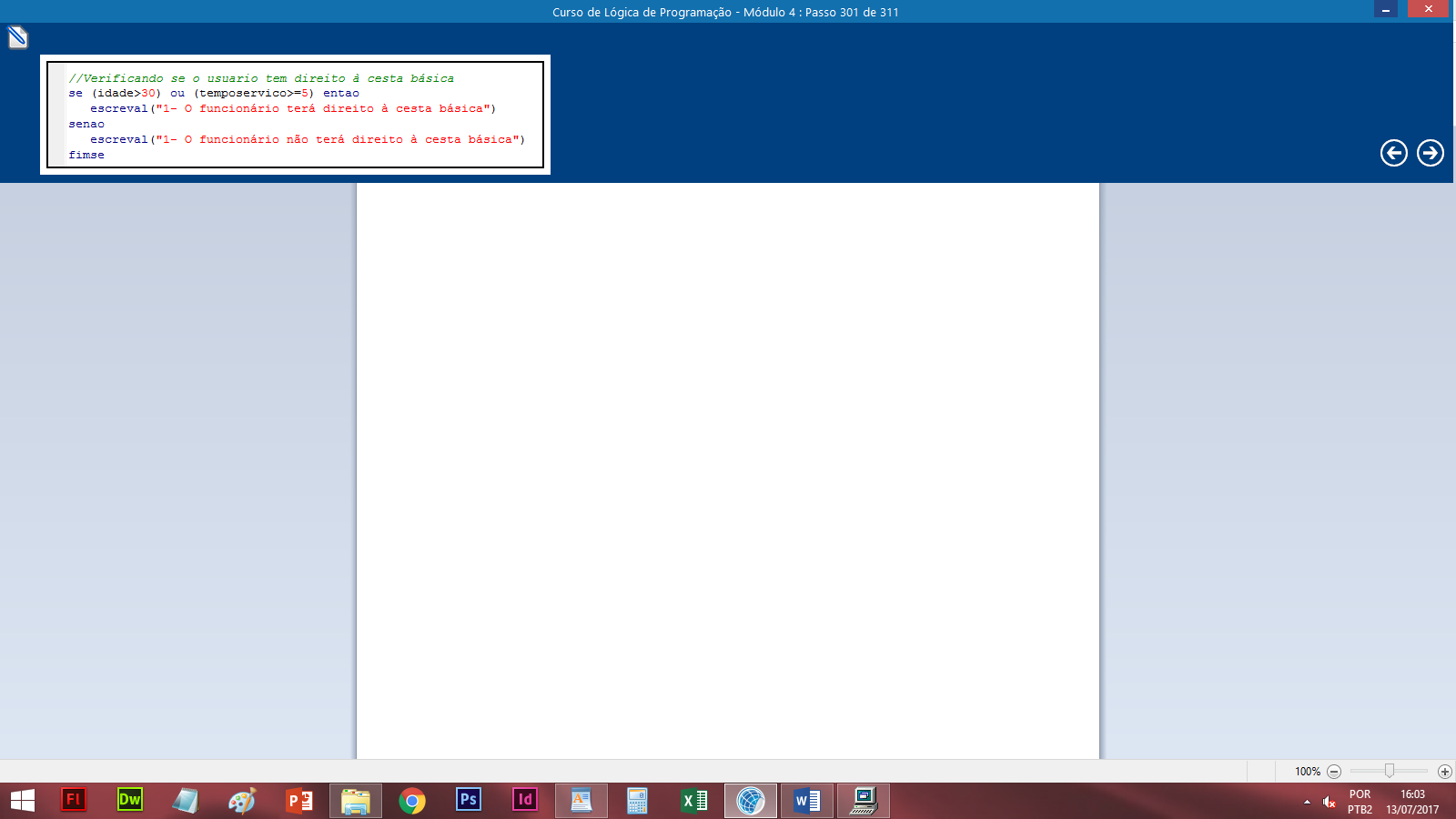
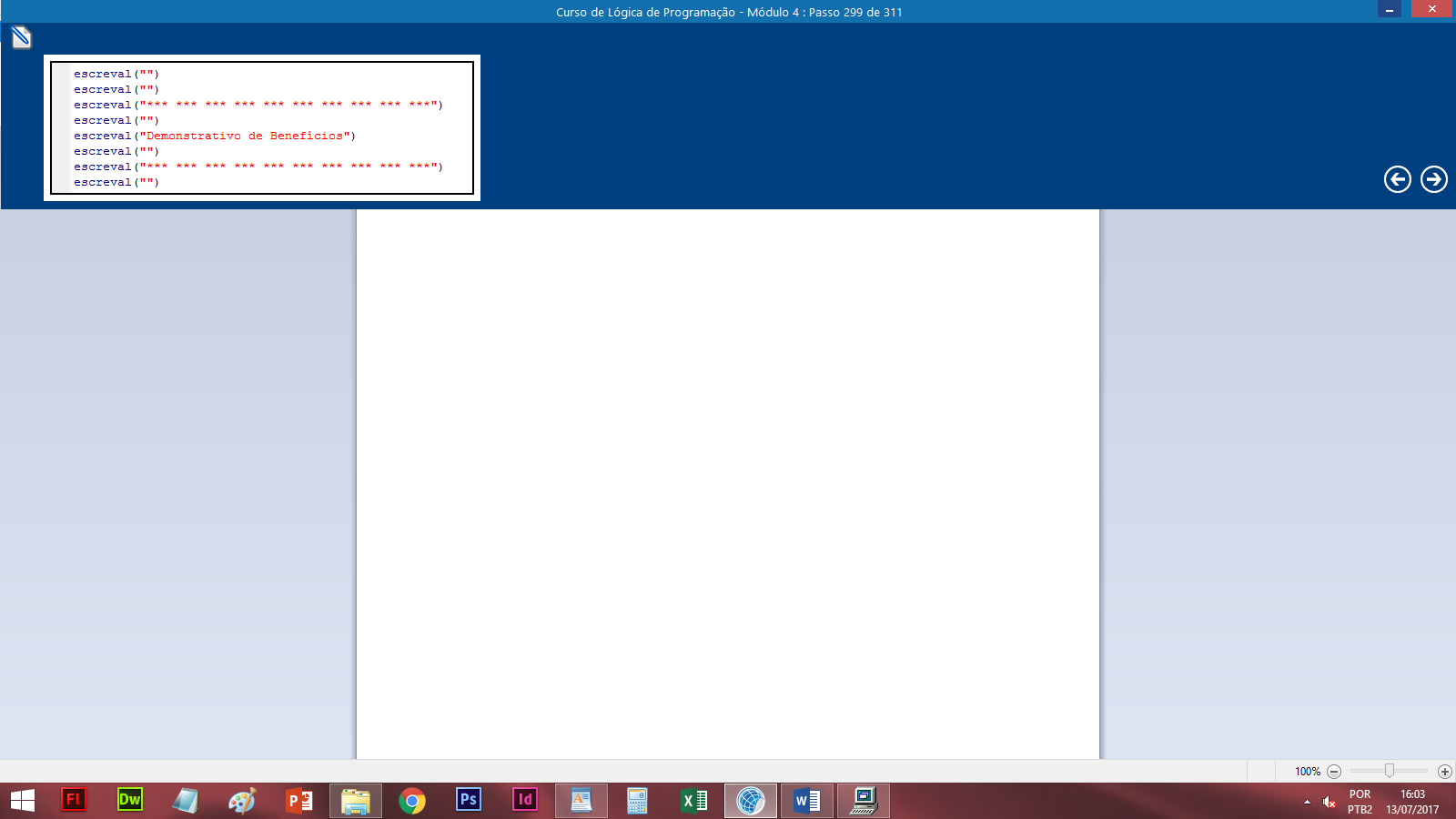
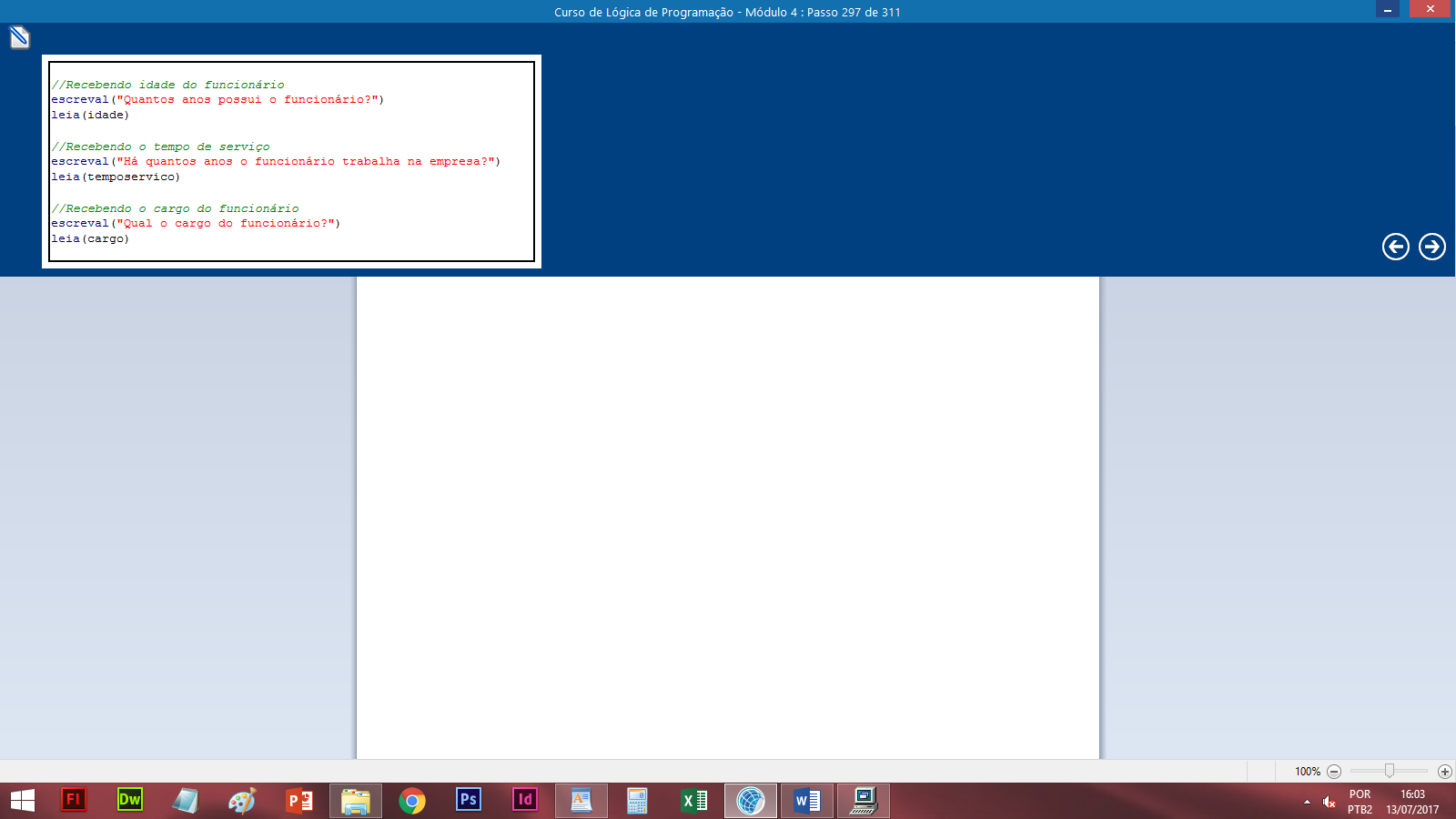
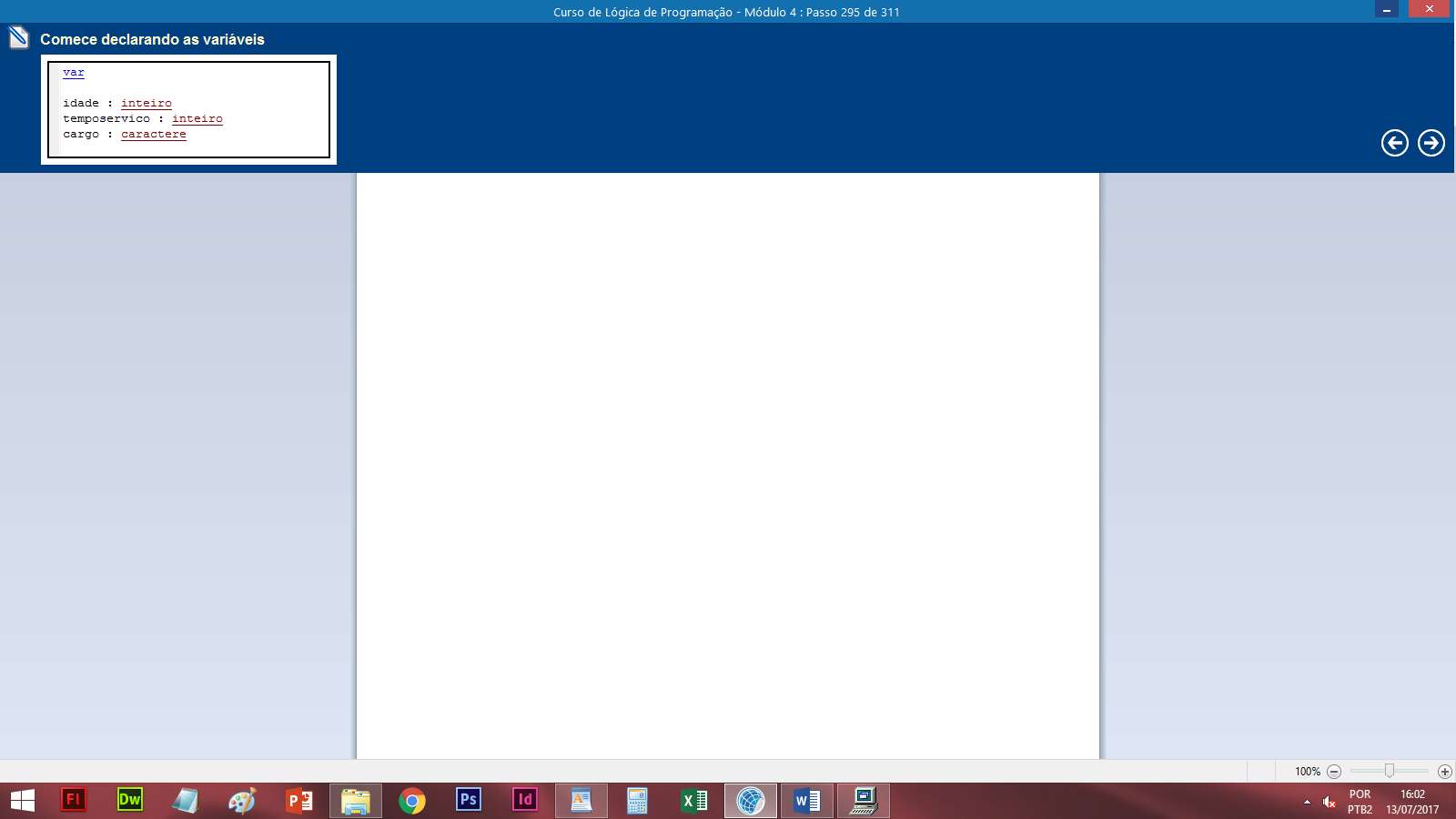
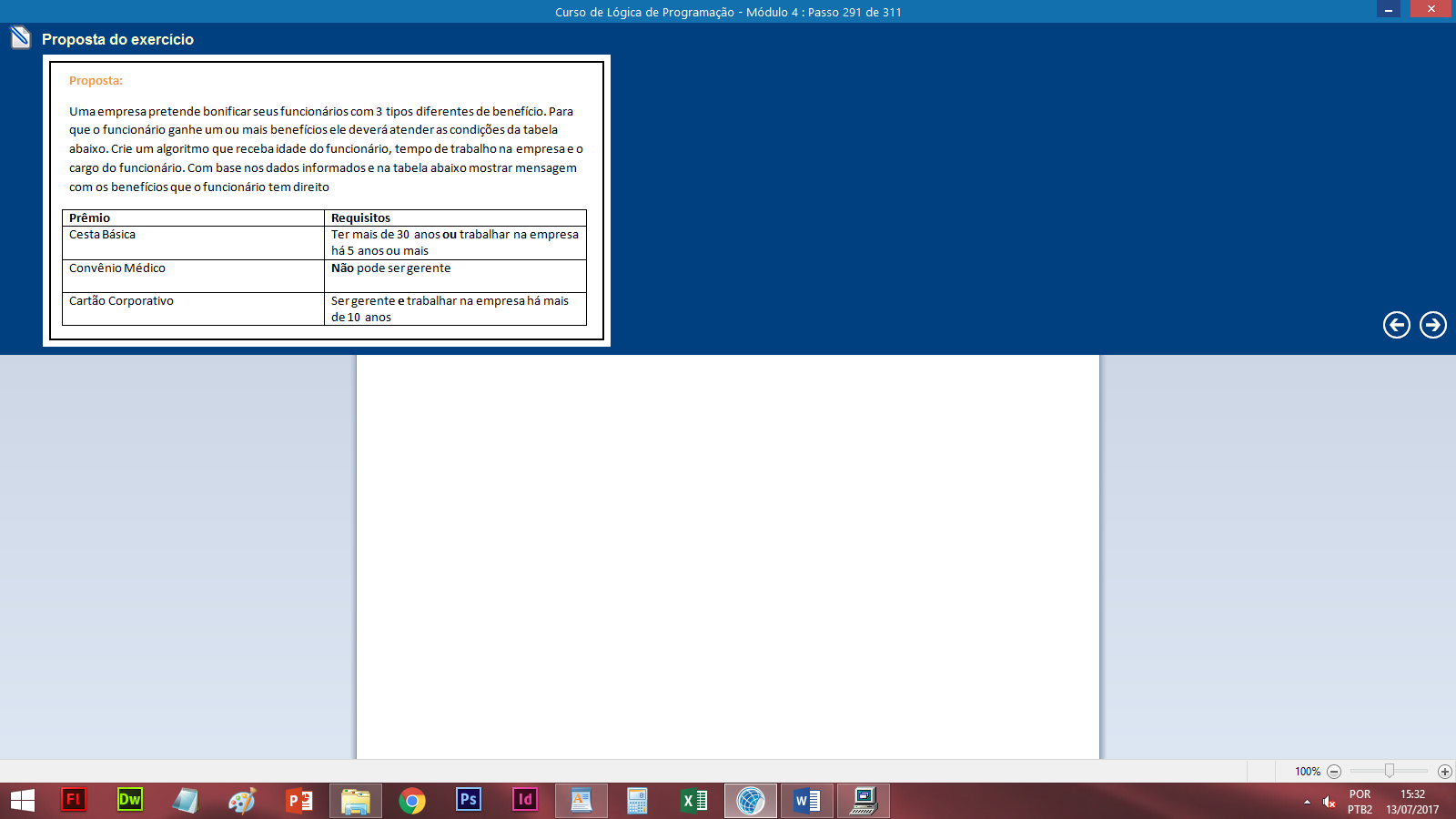
**Explicação**

Note que utilizando o operador lógico NOT é possível negar uma situação.

Esse é um recurso muito utilizado em programação. Por exemplo, para exibir a lista de clientes de uma loja, essa listagem só pode ser exibida caso o banco de dados **NÃO** esteja vazio. Se ele estiver vazio, nada será mostrado.

Sempre que for necessário fazer a negação de um valor de uma variável, utilize o operador NOT (não).

Ele sempre vai inverter o teste, se era Verdadeiro, será Falso... e assim por diante.



ESTRUTURA DE REPETIÇÃO

**Estruturas de Repetição**

Iremos aprender agora a manipular Estruturas de Repetição, que são estruturas que proporcionam que um mesmo bloco de código do algoritmo, seja processado "n" vezes, de acordo com a necessidade estipulada.

Uma estrutura de repetição possui sempre uma condição de controle, isto é, uma espécie de teste lógico que verifica se o bloco de código deve ser repetido mais uma vez, ou o programa já pode encerrar a repetição.

Esse tipo de estrutura é muito utilizada em programação, para que o algoritmo possa efetuar repetições, sem a necessidade de linhas adicionais desnecessárias no programa.

**Explicação**

Esse algoritmo é do tipo **estrutura sequencial**. Quantas vezes ele for executado, sempre vai retornar o valor "6", até que se *modifique a sua estrutura* . Ele não possui estruturas para decisão (condicional SE) e por isso o valor sempre será o mesmo.

Nesse algoritmo, temos um valor para a variavel **numero** (vale 1) inicialmente. Em seguida, ele executa uma operação de soma, usando o valor da variável **soma** (vale 5), logo 1 + 5 = 6.

ESTRUTURA DE REPETIÇÃO ENQUANTO...FAÇA

**Estrutura Enquanto... Faça**

Esta estrutura repete uma sequência de comandos enquanto uma determinada condição (especificada através de uma expressão lógica) for satisfeita. Enquanto a expressão lógica for true (verdadeira) o bloco de comandos será repetido e reiniciado ao início. O bloco só será interrompido, ou finalizado, quando a expressão lógica for false (falso).

*CONTADOR <- CONTADOR + 1*

**Linha de Incremento**

Ela é a chamada linha de incremento, ela que faz com que em cada execução do laço, a variável contador some +1 em seu valor. Se não for colocado o incremento, o laço vai se repetir para sempre, sem nunca terminar...

**Explicação**

A variável "contador" começa sempre com o **valor = 0**.

O laço obedece a seguinte condição -> **contador é menor que 20?** -> caso seja *verdadeiro* , o laço é executado pela primeira vez. Durante sua execução, a variável "contador" soma + 1... e assim sucessivamente.

O laço **só termina** quando a variável contador está com o **valor = 20**

Nesse momento o laço percebe que: **contador é menor que 20?** -> resulta em **FALSO**e encerra o laço.