Aula 10 – Comandos Repetitivos (Laços condicionais)

Esta estrutura de repetição possui como característica principal o fato de que **não se conhece previamente** quantas vezes os comandos existentes dentro do laço serão executados. Neste sentido, existe a **necessidade clara de se utilizar uma condição** (expressão lógica na forma de Verdadeiro ou Falso)..

Por exemplo, quando o problema coloca a leitura de certa quantidade de elementos, ou existência de certa quantidade de dados, etc, ou ainda mais quando é importante a necessidade de se colocar uma condição de parada (isto será visto posteriormente).

Os comandos em algoritmo que implementa estas estruturas são: **`do until..loop`** e **`do while..loop`.**

Um laço condicional possui como características duas formas de controle:

a) Controle por uma **variável**, no caso de um laço contado como o comando **for** que necessita de contador explicito. Nesta situação é importante a presença de uma variável que conta quantas vezes o laço foi executado

b) Controle por uma **condição**, como é o caso dos comandos **`do until..loop`** e **`do while..loop`** que necessitam satisfazer estas condições para continuar no laço. Aqui é importante a presença de uma condição na forma de **V**erdadeiro ou **F**also.

Observamos que: é possível implementar o comando **`for..next`** utilizando-se comandos **`do while..loop`** ou **`until..loop`** ou vice-versa.

Sintaxe dos comandos:

Sintaxe 01

```
Do [{While | Until} condição]
[comandos]
[Exit Do]
[comandos]
Loop
```

Ou

Sintaxe 02

```
Do
[comandos]
[Exit Do]
[comandos]
Loop [{While | Until} condição]
```

Utilizamos essa estrutura lógica para repetir um trecho de código **enquanto uma determinada condição for verdadeira** (teste da condição no **inicio** do loop – Sintaxe 01), ou **até que uma determinada condição torne-se verdadeira** (teste da condição no **final** do loop – Sintaxe 02).

Podemos usar dois operadores condicionais diferentes para o desenvolvimento desta estrutura de comando, que podem aparecer no **início** ou no **final** do trecho de código, são eles: `While` e `Until`. Com isso, temos algumas formas distintas de escrever o código, conforme a situação que o problema requer.

O loop `While .. Wend` existe para tornar o VBA compatível com um código mais antigo. No entanto, a Microsoft recomenda que se faça uso da função `Do Loop`, isto porque ela é mais "estruturada e flexível".

Observe os formatos da tabela abaixo:

Formato do Loop	Exemplo

`Do [condição]`
`...`
`Loop`
	`Do`
	
	`Loop While Res = Verdadeiro`

`Do while [condição]`
 	
`Loop`
	`Do while Res = Verdadeito`
	
	`Loop`

`Do Until [condição]`
`...`
`Loop`
	`Do Until res= Verdadeiro`
	
	`Loop`

`Do`
`...`
`Loop [condição]`
	`Do`
	`...`
	`Loop Until res = Verdadeiro`

`Do`
`...`
`Loop Until [condição]`
	`Do`
	`...`
	`Loop Until res=Verdadeiro`

`Do`
`...`
`Loop while [condição]`
	`Do`
	`...`
	`Loop While res=Verdadeiro`

`While`
`....`
`Wend`
	`While res = Verdadeiro`
	`....`
	`Wend`

Tem-se duas situações a serem verificadas:

a) Instruções de repetição enquanto uma condição for Verdadeira

Há duas maneiras de usar a palavra-chave `while` para verificar uma condição em uma estrutura `Do... Instrução Loop`.

Você pode verificar a condição antes de inserir o loop ou pode verificá-la depois que o loop tiver sido executado pelo menos uma vez.

No procedimento `Comando_Do_while` verifica a condição antes de inserir o *loop*. Se `vezes` for definido como 9 em vez de 30, as instruções dentro do loop nunca serão executadas.

Observe a pergunta `faça enquanto vezes>20`.

Outra observação importante é que para executar o bloco de comando a condição deve estar valida pelo menos uma vezes (para que ele possa entrar no loop)

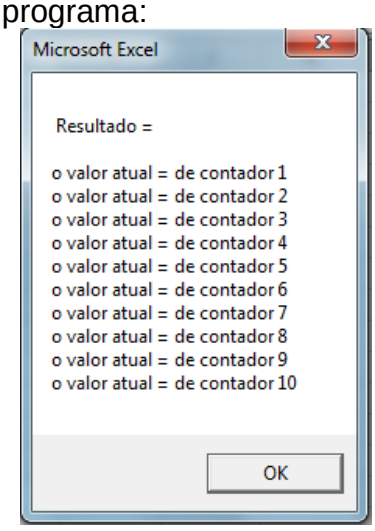
```
Sub Comando_Do_While()
Dim cont As Integer, vezes As Integer
    cont = 0
    vezes = 30
    Do While vezes > 20
        vezes = vezes - 1
        cont = cont + 1
    Loop
    MsgBox "O Loop executou " & cont & " vezes."
End Sub
```

Outro exemplo de código com `While` no início do trecho de código.

Neste exemplo, vamos mostrar o conteúdo de uma variável contador que inicia em 1 e termina em 10. Observe que o contador foi inicializado em 1 e a condição de parada é que este contador seja maior que 10. Em algum momento dentro do laço a variável contador deve ser atualizada.

```
Sub Do_Loop_While_no_início()
Dim contador As Integer, s As String
' inicialização das variaveis
contador = 1
s = ""
'Armazena os dez primeiros números maiores que zero.

Do While contador <= 10
s = s & vbCrLf & "o valor atual = de contador " & contador
contador = contador + 1
Loop

MsgBox " Resultado = " & vbCrLf & s
End Sub
```

O resultado da execução do programa:



Um outro exemplo Somando os primeiros 10 números inteiros positivos usando o VBA

```
Sub Comando_Do_While2()
Dim i As Integer, result As Integer
Dim s As String
i = 1
result = 0
Do While i <= 10
result = result + i
s = s & vbCrLf & " O valor de result = " & result
i = i + 1
Loop
MsgBox s
End Sub
```

No próximo procedimento, chamado de `Comando_do_Loop_while` as instruções dentro do loop são executadas apenas uma vez antes de a condição se tornar falsa.

```
Sub Comando_do_Loop_While()
Dim cont As Integer, vezes As Integer
    cont = 0
    vezes = 9
    Do
        vezes = vezes - 1
        cont = cont + 1
    Loop While vezes > 10
    MsgBox "O Loop executou " & cont & " vezes."
End Sub
```

Observe a pergunta `faça os comandos enquanto vezes>10`

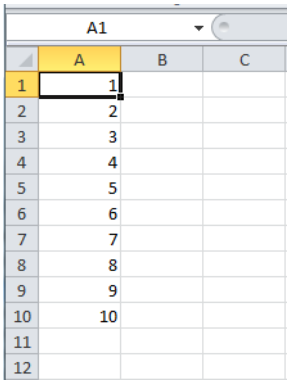
Neste caso, observa-se que os comandos que estão dentro do loop são executados pelo menos 1 vezes e depois é que ele realiza o teste.

Exemplo de código com `While` no final do trecho de código.

Este exemplo mostra o preenchimento de células A1 até A10 com os valores da variável Z que inicia em 1 e termina em 10. Observe que Z deve ser inicializada no inicio da iteração e a condição é testada somente no final do loop.

```
Sub Do_Loop_While_no_final()
Dim Z As Integer
Z = 1

Do
Range("A" & Z).Value = Z
Z = Z + 1
Loop While Z <= 10

End Sub
```

Resultado:



b) Instruções de repetição ATÉ uma condição se tornar Verdadeira

Há duas maneiras de usar a palavra-chave `until` para verificar uma condição em um comando `do... Instrução loop`.

Na primeira forma, verifica-se a condição antes de inserir o loop (conforme mostrado no procedimento `Comando_Do_Until_Loop`) O loop continua enquanto a condição permanece falsa.

```
Sub Comando_Do_Until_Loop()
Dim cont As Integer, vezes As Integer
    cont = 0
    vezes = 20
    Do Until vezes = 10
        vezes = vezes - 1
        cont = cont + 1
    Loop
    MsgBox "O Loop executou " & cont & " vezes."
End Sub
```

Um outro exemplo: Somando os primeiros 10 números inteiros positivos usando o VBA

Suponha que você queira somar os primeiros dez inteiros positivos usando o loop `Do Until` no VBA.

Para fazer isso, você pode usar o loop `Do Until` até que o próximo número seja menor ou igual a 10. Assim que o número for maior que 10, o loop será interrompido.

```
Sub Comando_Do_Until_Loop2()
Dim i As Integer, result As Integer
Dim s As String
i = 1
result = 0
Do Until i > 10
 result = result + i
 s = s & vbCrLf & " O valor de result = " & result
 i = i + 1
Loop
MsgBox s
End Sub
```

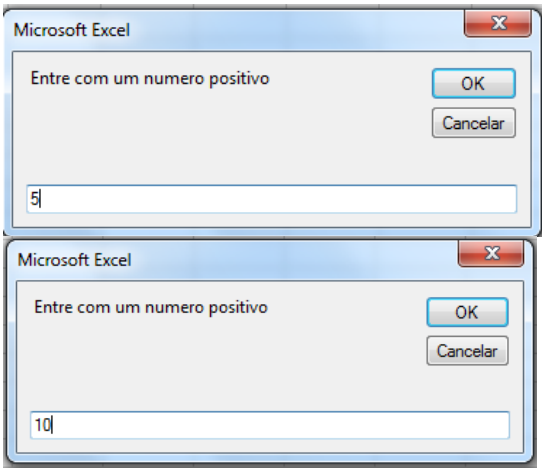
Exemplo de código com **`Until`** no início do trecho de código:

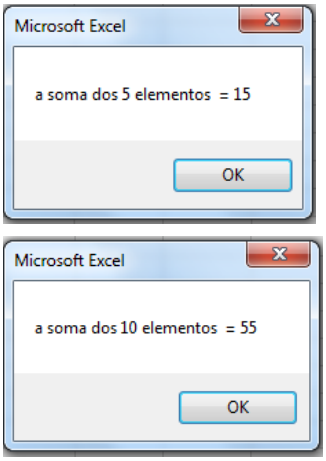
Considere somar termos inteiros até um determinado numero. Por exemplo. Se o numero for 5 soma= 1 + 2 + 3 + 4 + 5 = 15. Se o numero for 7 = 1 + 2 + 3 + 4 + 5 + 6 + 7

Resultado: Observe o código abaixo

```
Sub Do_Loop_Until_no_início()

Dim x As Integer, Cont As Integer, acumulador As Integer
x = CInt(InputBox(" Entre com um numero positivo "))
Cont = 1
acumulador = 0
' Efetua a soma dos dez primeiros números maiores que zero.

Do Until Cont > x
' uso de um acumulador
acumulador = acumulador + Cont
'uso de contador
Cont = Cont + 1
Loop ' fim do laço

MsgBox " a soma dos " & x & " elementos  = " & acumulador

End Sub
```

Assim, executando este código teremos:





Na segunda forma, verifica-se a condição após o loop ser executado pelo menos uma vez (conforme mostrado no procedimento `Comando_Do_Loop_Until`). O Loop continua enquanto a condição permanece falsa.

```
Sub Comando_Do_Loop_Until()
Dim cont As Integer, vezes As Integer
    cont = 0
    vezes = 1
    Do
        vezes = vezes + 1
        cont = cont + 1
    Loop Until vezes = 10
    MsgBox "O Loop executou " & cont & " vezes."
End Sub
```

Exemplo de código com **`Until`** no final do trecho de código:

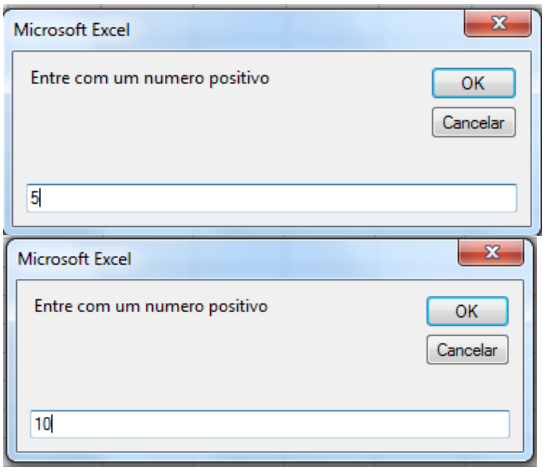
Considere somar termos inteiros até um determinado numero. Por exemplo. Se o numero for 5 soma= 1 + 2 + 3 + 4 + 5 = 15. Se o numero for 7 = 1 + 2 + 3 + 4 + 5 + 6 + 7

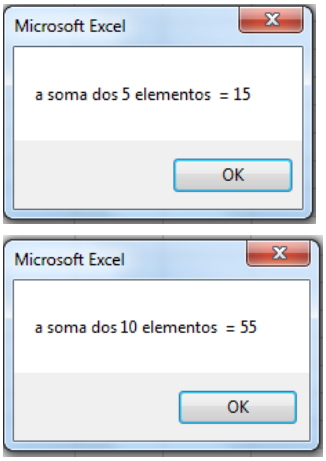
Observe que a estrutura de condição não muda com relação a um exemplo, ou outro.

Resultado: Observe o código abaixo

```
Sub Do_Loop_Until_no_Fim()
Dim x As Integer, Cont As Integer, acumulador As Integer

x = CInt(InputBox(" Entre com um numero positivo "))

Cont = 1
acumulador = 0
' Efetua a soma dos dez primeiros números maiores que zero.

Do
' uso de um acumulador
acumulador = acumulador + Cont
'uso de contador
Cont = Cont + 1
Loop Until Cont > x ' fim do laço

MsgBox " a soma dos " & x & " elementos  = " & acumulador
End Sub
```

Assim, executando este código teremos:





Exercícios:

Resolvido 19 - Faça um program que leia um número não determinado de pares de valores [M, N], todos inteiros e positivos, um par de cada vez, e que calcule e mostre a soma de todos os numeros inteiros entre M e N (inclusive). A digitação de pares terminará quando M for maior ou igual a N.

Resolvido 21 – Faça um programa que receba vários números, calcule e mostre:

a) A soma dos numeros digitados
b) A quantidade de numeros digitados
c) A média dos numeros digitados
d) O maior numero digitado
e) O menor numero digitado
f) A média dos numeros pares
g) A porcentagem dos numeros digitados

Para esse exercicio podemos utilizar algumas estruturas de comandos condicionais

- Inicialmente vamos utilizar uma estrutura de comado condicional com teste no inicio do laço, utilizando o comando WHILE
- Neste caso, vamos realizar uma leitura de dados antes do loop para que possaos garantir que ele vai entrar somente se o numero for positivo. Depois será realizado uma outra leitura de dados para sair do loop.
- No caso do numero ser negativo, o algoritmo encerra
- Observe o algoritmo abaixo:

```
Sub exerci02()
'declaração de variaveis
Dim n As Integer, soma As Integer, s As String
'calculos e operações
soma = 0
n = CInt(InputBox("Entre com um numero > 0 "))
s = " O numero lido = " & n
Do While (n > 0) ' inicio do loop
  soma = soma + n

  n = CInt(InputBox("Entre com um numero > 0 "))
  s = s & vbCrLf & " O numero lido = " & n
Loop ' fim do loop

s = s & vbCrLf & " A soma dos numeros " & soma

MsgBox s
End Sub
```

Vamos considerar agora que a condição a ser testada no loop seja no final. Assim, vamos utilizr o comando UNTIL.

- Inicialmente vamos utilizar uma estrutura de comado condicional com teste no final do laço, utilizando o comando UNTIL
- Neste caso, vamos realizar uma leitura de dados dentro do loop para que possaos garantir que ele leia o dado pelo menos UMA vez, antes de testar condição de parada no final do loop.
- O comando condicional IF é para garantir que o dado negativo lido não seja utilizado para acumulo dos numeros
- No caso do numero ser negativo, o algoritmo encerra
- Observe o algoritmo abaixo:

```
Sub exerc03()
'declaração de variaveis
Dim n As Integer, soma As Integer, s As String
'calculos e operações
soma = 0
s = ""
Do
  n = CInt(InputBox("Entre com um numero >0 "))
  s = s & vbCrLf & " O numero lido = " & n
  If n > 0 Then
    soma = soma + n

   End If
Loop Until (n <= 0)
s = s & vbCrLf & " A soma dos numeros = " & soma

MsgBox s
End Sub
```

Uma outra forma de garantir que os numeros lidos sejan utilizados para operação, é mostra uma mensagem no final de cada loop “ Deseja Continuar(s/n)?”. Conforme o usuário clicar, o numero 7(sete) refere ao botão não.

O algoritmo pode ser observado abaixo:

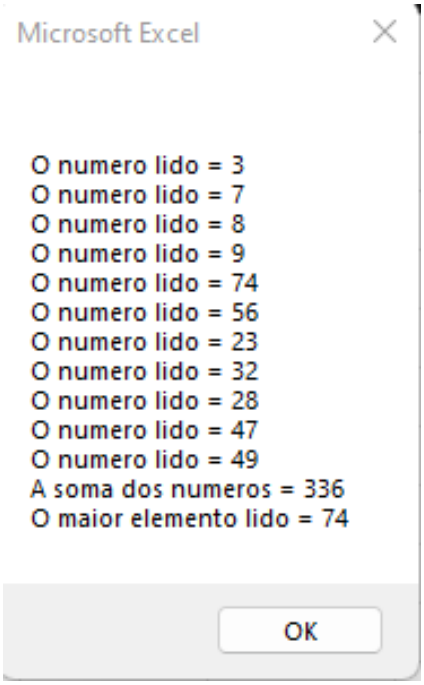
```
Sub exerc03a()
'declaração de variaveis
Dim n As Integer, soma As Integer, s As String
Dim op As Integer
'calculos e operações
soma = 0
s = ""
Do
  n = CInt(InputBox("Entre com um numero >0 "))
  s = s & vbCrLf & " O numero lido = " & n
  soma = soma + n

   op = MsgBox("Deseja continuar?", vbInformation + vbYesNo, " Informação")
Loop Until (op = 7)
s = s & vbCrLf & " A soma dos numeros = " & soma

MsgBox s
End Sub
```

Para o calculo do maior ou menor numero utilizando-se essa problematica, obsevamos que para inicializar a variavel maior/menor, vamos utilizar o primeiro elemento do conjunto

Assim temos o seguinte algoritmo:

```
Sub exerc04a()
'declaração de variaveis
Dim n As Integer, soma As Integer, s As String
Dim op As Integer, cont As Integer, maior As Integer
'calculos e operações
soma = 0
s = ""
cont = 0
Do
  n = CInt(InputBox("Entre com um numero >0 "))
  s = s & vbCrLf & " O numero lido = " & n
  soma = soma + n
  If cont = 0 Then
    maior = n
    cont = cont + 1
    Else
    If n >= maior Then
       maior = n
     End If
   End If

   op = MsgBox("Deseja continuar?", vbInformation + vbYesNo, " Informação")
Loop Until (op = 7)
s = s & vbCrLf & " A soma dos numeros = " & soma
s = s & vbCrLf & " O maior elemento lido = " & maior
MsgBox s
End Sub
```

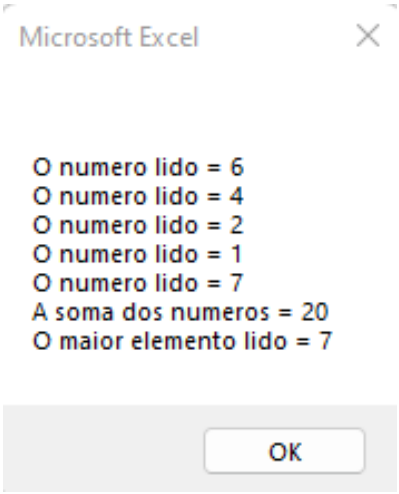
Saida do algoritmo:



Para o algoritmo anterior podemos utilizar variaveis booleanas, ou seja, variaveis logicas
Assim, modificando o algoritmo temos

```
Sub exerc04a()
'declaração de variaveis
Dim n As Integer, soma As Integer, s As String
Dim op As Integer, flag As Boolean, maior As Integer
'calculos e operações
soma = 0
s = ""
flag = True
Do
  n = CInt(InputBox("Entre com um numero >0 "))
  s = s & vbCrLf & " O numero lido = " & n
  soma = soma + n
  If flag Then
    maior = n
    flag = False
    Else
    If n >= maior Then
       maior = n
     End If
   End If

   op = MsgBox("Deseja continuar?", vbInformation + vbYesNo, " Informação")
Loop Until (op = 7)
s = s & vbCrLf & " A soma dos numeros = " & soma
s = s & vbCrLf & " O maior elemento lido = " & maior
MsgBox s
End Sub
```


Saida do novo algoritmo



Proposto 07 -Faça um programa que receba a idade, altura e o peso de cinco pessoas. Calcule e mostre:

a) A quantidade de pessoas com idade superior a 50 anos
b) A média das alturas das pessoas com idade entre 10 e 20 anos
c) A porcentagem de pessoas com peso inferior a 40Kg entre todas as pessoas analisadas