

Insper

Visual Basic for Applications

Aula 01

Raul Ikeda

2º semestre de 2017

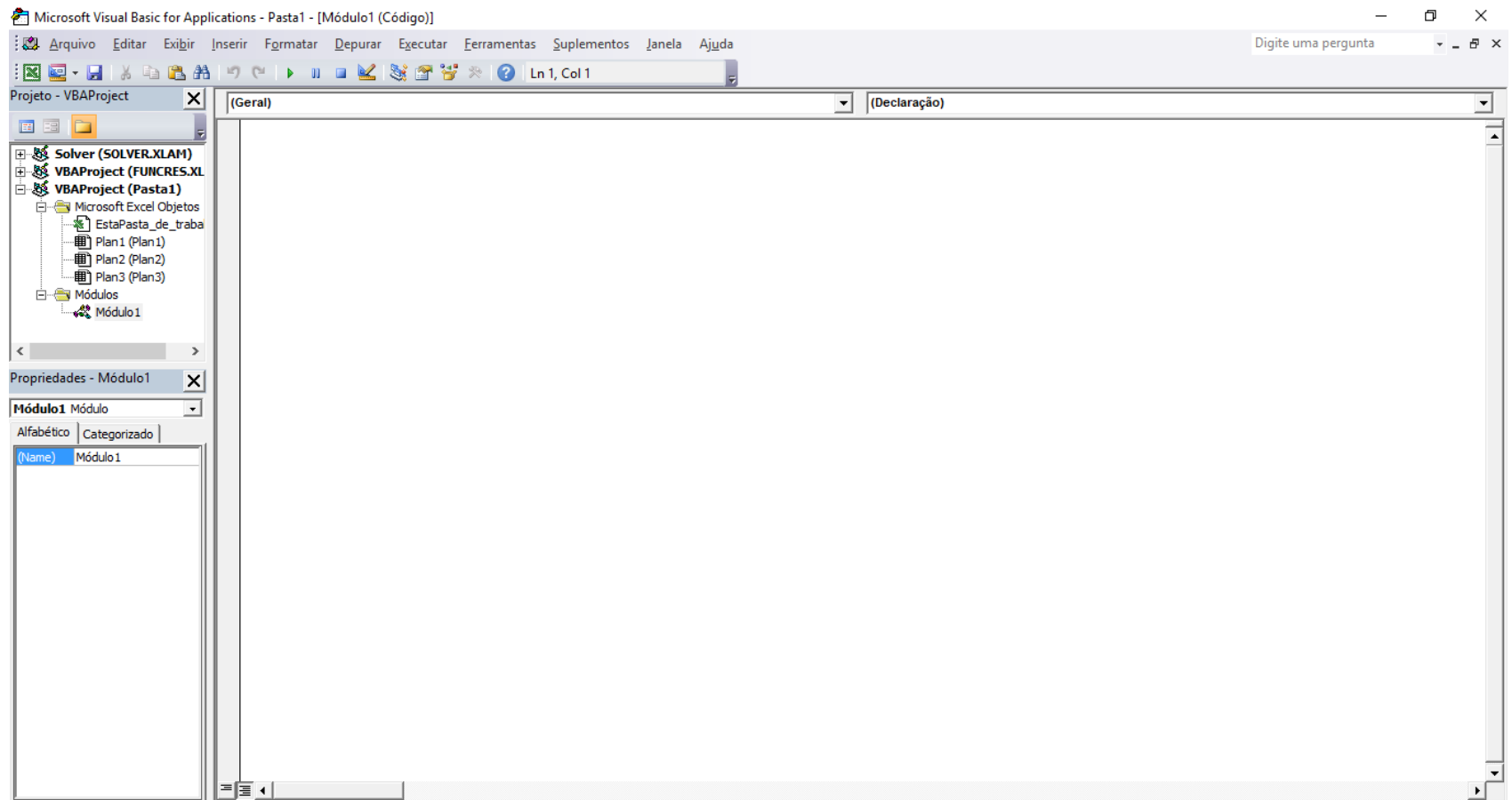
Para que serve?

- VBA é uma linguagem interpretada derivada do Visual Basic.
- Roda dentro dos aplicativos do pacote Office (Excel, Word, Access, Powerpoint, Outlook, etc).
- Principal funcionalidade é automatizar processos.
- Não concorre diretamente com outras linguagens por ter características diferentes.
- Pode-se utilizar outras linguagens para automatizar planilhas (de forma não nativa).

Características

- Possui tipagem forte¹ (obrigatório declarar o tipo das variáveis).
- Não implementa Classes de forma usual. Comumente chamado de pseudo-orientado a objetos.
- Identação é opcional.
- Não possui listas, dicionários ou tuplas nativamente.
- Não possui *list comprehension* (Desempacotamento).
- Não pode escrever duas instruções na mesma linha.

O Ambiente Excel/VBA



Analogia Python-VBA

1. Tipo de Dados	
VBA	Python
<p>-Pode declarar o tipo das Variáveis</p> <p>Exemplo: Dim x as Integer</p> <p>Tipos:</p> <ul style="list-style-type: none">• Integer• Single• Double• String• Date• Boolean <p>Função de Conversão:</p> <ul style="list-style-type: none">CInt()CSgn()CDbl()CStr()CDate()CBool() <p>CUIDADO: Dim x, y as Integer</p> <p>Nesse caso x é Variant e y é Integer</p>	<p>-Não precisa declarar</p>

2. Comentários	
VBA	Python
'Esse é um comentário	#Esse é um comentário

Analógia Python-VBA

3. Operadores	
3.1 Comparação	
VBA	Python
= < > <= >= <>	== < > <= >= !=
3.2 Aritméticos	
+ - * / \ mod ^	+ - * / // % **
3.3 Texto	
&	+
3.4 Atribuição	
=	= += -= *= /=
3.5 Atribuição (named arguments)	
:=	=
3.5 Lógicos	
And Or Not	And Or Not
3.6 Bitwise	
And Or Xor Not (Não é real bitwise)	& ^ ~

Analógia Python-VBA

4. Estruturas	
4.1 If	
VBA	Python
If x < 0 Then 'Faça algo Elseif x > 0 Then 'Faça algo Else 'Faça algo End If	if x < 0: #Faça algo elif x > 0: #Faça algo else: #Faça algo
4.2 While	
i = 1 While x <= 10 i = i + 1 Wend	i = 1 while i < 10: i += 1
4.3 For	
For i = 1 to 10 'Faça Algo Next i For Each v In A 'Faça algo Next	for i in range(0,10): #Faça algo for v in A: #Faça algo

Analógia Python-VBA

5. Estruturas Exclusivas	
5.1 Repeat	
VBA	Python
i = 1 Do 'Faça algo i = i + 1 Loop While i <= 10	-Não possui equivalente
5.2 Case	
Select Case x Case 0 'Faça algo Case 1 'Faça algo Case Else 'Faça algo End Select	-Não possui equivalente
5.3 Enum	
Enum valor primeiro = 1 segundo = 2 terceiro = 4 quarto = 8 'etc End Enum	-Não possui equivalente
5.4 With	
With Workbooks("Livro") With .Worksheets("Plan1") MsgBox .Cells(1, 1).Value End With End With	-Não possui equivalente

Analógia Python-VBA

6. Funções	
6.1 Sem retorno de valores	
VBA	Python
Public Sub Mover(ByVal x as Integer) 'Faça algo End Sub	def mover(x): #Faça algo
6.2 Com retorno	
Function Calcular(x as Integer) as Integer Calcular = x ^ 2 End Function	def calcular(x): return x ** 2
6.3 Detalhes	
<ul style="list-style-type: none">- ByVal é passagem por valor (equivalente a uma variável no Python)- ByRef é passagem por referência (equivalente a um objeto no Python - entretanto pode-se passar variáveis como referência)- É possível retornar 2 ou mais valores mediante um vetor- Aceita named argument, Exemplo: y = Calcular(x := 3)- Pode rodar uma Sub usando Call, Exemplo: Call Mover(2)- Se não usar Call, não pode usar '()': Mover 2	<ul style="list-style-type: none">- Pode retornar mais de um valor usando tuplas- Aceita named arguments, Exemplo: y = calcular(x = 3)

Analogia Python-VBA

7. Vetores, Listas e Dicionários	
7.1 Vetores e Matrizes	
VBA	Python
Dim x() as Integer, y() as Integer ReDim x(1 to n) as Integer ReDim y(1 to n, 1 to n) as Integer x(1) = 10 y(1, 1) = 15 Obs: Não possui operadores para vetores e matrizes.	-Não possui equivalente nativo - Usar Numpy
7.2 Listas	
-Não possui equivalente	A = [1, 2, 3] A.append(4)
7.3 Dicionários	
Dim x As Scripting.Dictionary x.Add "x1", 1 x.Add "x2", 2 x.Add "x3", 3 Obs: Adicionar Referência 'Microsoft Scripting Runtime'.	A = {'x1' : 1, 'x2' : 2} A['x3'] = 3

Analógia Python-VBA

8. Exception Handling	
8.1 Try Catch	
VBA	Python
<pre>Sub FazerAlgo() On Error GoTo catch 'Faça algo Exit Sub catch: MsgBox "Erro: " & Err.Description End Sub</pre>	<pre>def fazerAlgo(): try: #Faça algo except: print(sys.exc_info()[0])</pre>

Analógia Python-VBA

9. Detalhes Adicionais

VBA

- Não possui classes realmente. Implementa pseudo-orientação a objetos.
- Implementa escopo de Sub/Funções e variáveis. Pode ser Public ou Private.
- Possui Userforms para construir formulários.
- Pode interagir com as células na planilha:

```
x = Workbooks("Livro").Worksheets("Plan1").Cells(1, 1).Value
```

```
Workbooks("Livro").Worksheets("Plan1").Cells(1, 1).Value = x
```

- Possui Eventos:

```
Private Sub Workbook_Open()
```

```
    'Evento disparado quando abre um arquivo
```

```
End Sub
```

```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
```

```
    'Evento disparado quando seleciona uma célula
```

```
End Sub
```

- Pode usar API do Windows

1º Programa

- Antes vamos implementar uma função simples no Excel.
- Faça uma Function que retorna a soma de dois números inteiros.
- Dica 1: usar a cola de comandos.
- Dica 2: coloque em um módulo.
- Dica 3: teste nas células.

```
Function MySum(x As Integer, y As Integer) As Integer  
    MySum = x + y  
End Function
```

- **ACHTUNG!!!** Salvar o arquivo como “Habilitado para Macro”, ou seja .xlsm – ou perderá o seu código VBA.

2º Programa

- Faça uma função para calcular a combinação de 2 números.
- Lembrando que:

$$C_P^m = \frac{m!}{p! (m - p)!}$$

- TESTE a sua função.

2º Programa

```
Function MyFact(n As Long) As Long
    Dim i As Integer

    MyFact = 1
    For i = 1 To n
        MyFact = MyFact * i
    Next i

End Function
```

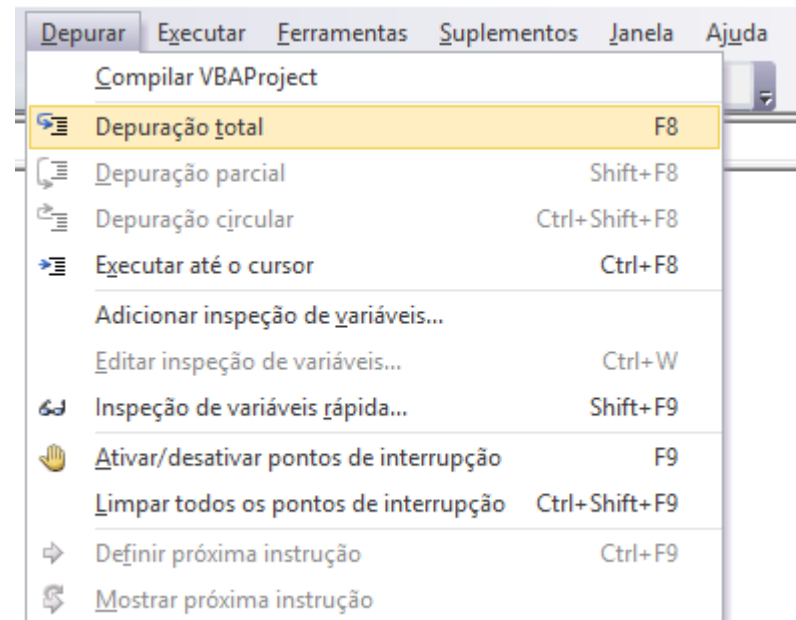
```
Function MyComb(m As Integer, p As Integer) As Integer

    MyComb = MyFact(m) / (MyFact(p) * MyFact(m - p))

End Function
```

Debugging

- F8 – Roda a instrução atual.
- Shift F8 – Roda a instrução atual sem entrar em funções.
- F9 – Insere um breakpoint na linha atual.
- Shift F9 – Verifica o valor de uma variável.
- Ctrl F9 – Define a próxima Instrução (CUIDADO!);
- F5 – Roda o programa até o final ou o próximo breakpoint.



Debugging – Breakpoint

```
Function MyFact(n As Integer) As Long  
    Dim i As Integer  
    ● MyFact = 1  
    For i = 2 To n  
        MyFact = MyFact * i  
    Next i  
End Function
```

Debugging – Variable Watch

Adicionar inspeção de variáveis


Expressão:
MyFact

Contexto
Procedimento: MyFact
Módulo: Módulo1
Projeto: VBAProject

Tipo de inspeção de variáveis
☒ Expressão de inspeção de variáveis
☐ Interromper quando o valor for verdadeiro
☐ Interromper quando o valor for alterado

OK
Cancelar
Ajuda



Inspeções de variáveis			
Expressão	Valor	Tipo	Contexto
 MyFact	1	Long	Módulo1.MyFact

Like a Pro

- É possível passar um conjunto de células como argumento de entrada:

```
Function MyFunc(x As Range) As Integer  
End Function
```

- É possível retornar em mais de uma célula – Próxima aula.
- NÃO é possível alterar em VBA uma célula que não foi passada na entrada ou faz parte da saída (dica do milhão: em C ou .net pode).