



Trabalhando

com **VBA** no **Excel®**

Módulo 1

Prof. Janderson Garcia

www.jandersongarcia.com.br

Sumário

I. INICIANDO A JORNADA	3
LÓGICA DE PROGRAMAÇÃO E ALGORÍTIMOS?	3
SALVANDO UM ARQUIVO PARA SUPORTAR MACROS	4
GUIA DESENVOLVEDOR.....	4
EDITOR DO VISUAL BASIC	5
MENU EXIBIR.....	5
MENU INSERIR.....	6
MENU EXECUTAR.....	6
MENU FERRAMENTAS.....	7
II. TRABALHANDO COM MÓDULOS	8
TIPOS DE MÓDULOS.....	8
CRIANDO MÓDULOS	8
DEU ERRO?	9
PERSONALIZANDO UMA CAIXA DE MENSAGEM – PARTE 1.....	9
ESTRUTURA DA MENSAGEM DE TEXTO	9
III. DECLARAÇÃO DE VARIÁVEIS.....	11
DECLARAÇÃO DE VARIÁVEIS.....	11
OPERADORES ARITMÉTICOS.....	12
IV. FUNÇÃO IF.....	13
EXEMPLO 1	13
EXEMPLO 2	14
EXEMPLO 3	15
EXERCÍCIOS	16
V. ESTRUTURA DE REPETIÇÃO	18
DO WHILE.....	18
EXEMPLO	18
FOR... NEXT.....	18
EXEMPLO 1	19
EXEMPLO 2	19
EXERCÍCIOS	20
VI. MANIPULAÇÃO DE CÉLULAS E PLANILHAS.....	21
OBJETO WORKSHEETS.....	21
EXEMPLO 1	21
EXEMPLO 2	22
NOVIDADES NA LINHA DE COMANDO	22
EXEMPLO 3	22
EXERCÍCIO.....	23
VII. EXERCÍCIOS.....	25
ATIVIDADE 1.....	25
ATIVIDADE 2	25
VIII. CRIAÇÃO DE FORMULÁRIOS	27
INSERINDO UM FORMULÁRIO	27
CRIANDO UM FORMULÁRIO SIMPLES	28

CAIXA DE TEXTO	28
RÓTULO OU LABEL.....	29
BOTÕES DE COMANDO.....	29
TABULAÇÃO.....	30
CHAMANDO O FORMULÁRIO	30
PROGRAMANDO O BOTÃO ADICIONAR	30
PROGRAMANDO O BOTÃO EXCLUIR	32
PROGRAMANDO O BOTÃO SAIR.....	33
IX. TRABALHANDO ENTRE FORMULÁRIOS	35
ESBOÇO DO PROJETO.....	35
FORMULARIO INICIAL	35
BOTÕES DO PROJETO	35
FORMULÁRIO DE PRODUTOS	38
X. PROJETO AGENDA.....	42
ESBOÇO DO PROJETO.....	42
MONTANDO O BANCO DE DADOS	42
CONTATOS.....	42
MONTANDO OS FORMULÁRIOS	44
FORMULÁRIO DE CONTATOS	44
FORMULÁRIO DE SENHA	45
FORMULÁRIO PRINCIPAL.....	45
FORMULÁRIO DE LOGIN	46
CRIANDO OS SCRIPTS	47
FORMULÁRIO PRINCIPAL.....	47
FORMULÁRIO DE CONTATOS	48
FORMULÁRIO DADOS DE ACESSO	51
FORMULÁRIO LOGIN	51
ABRIR O FORMULÁRIO E OCULTAR O EXCEL	52

I. INICIANDO A JORNADA

Antes de mergulharmos no mundo da criação de macros no Excel, precisamos ter ideia de alguns detalhes de suma importância para que o seu desenvolvimento seja completo.

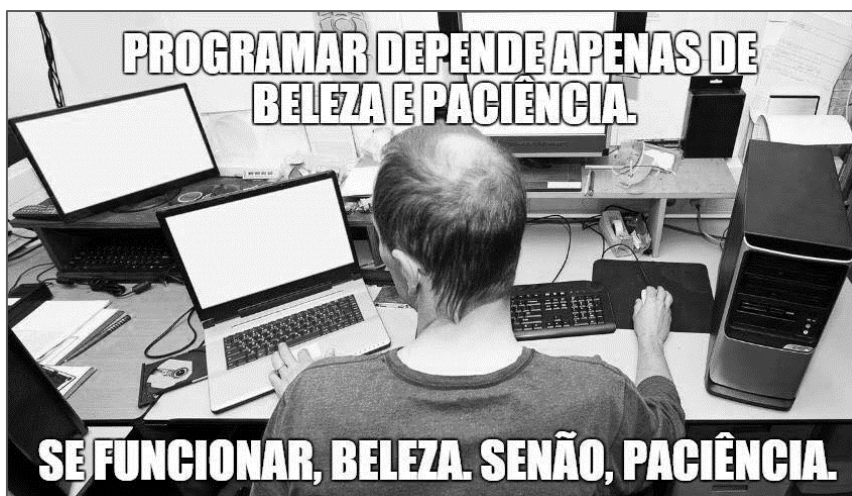
Você precisa ter uma pequena noção no que é lógica de programação e algoritmos. Fique tranquilo que isso não é nada de muito complicado (ao menos o que precisamos saber). Precisamos apenas de ter a noção para podermos desenvolver nossa planilha com perfeição e sem muita perda de tempo.

LÓGICA DE PROGRAMAÇÃO E ALGORÍTIMOS?

É a técnica de desenvolver sequências lógicas para atingir um determinado objetivo. Essas sequências lógicas são adaptadas para linguagem de computador por meio do programador a fim de produzir software.

Uma sequência lógica é denominada algoritmo. Então podemos dizer em linguagem mais coloquial, que um algoritmo é uma sequência de passos para atingir um determinado objetivo. Como podemos ver a lógica de programação trata basicamente de construir algoritmos que serão transformados em macros.

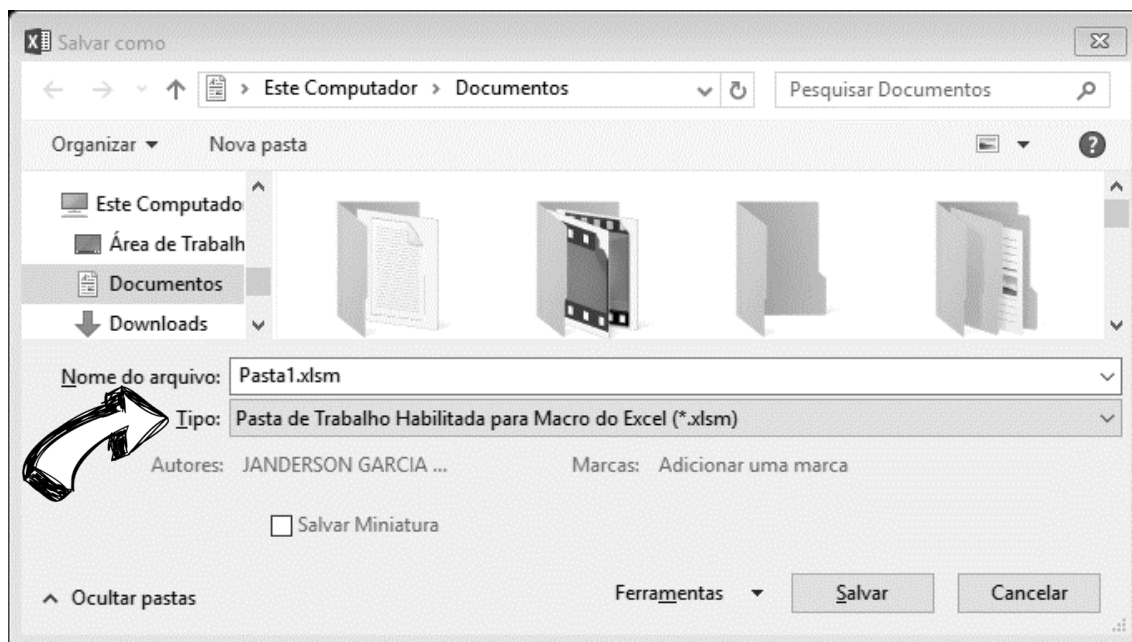
Macros, portanto, são instruções lógicas que fazem o processamento dos comandos criados pelo seu desenvolvedor. O programador necessita ter um pensamento lógico bastante desenvolvido, analisando sempre causa e efeito, e acima de tudo ser paciente e persistente.



Macro é uma sequência de comandos também denominado de Rotina que contém uma lista de instruções a serem realizadas no Excel, que pode ser classificada como: **Sub-rotina**, **Função** ou **Procedimento de evento**. Quando se trata de facilitar tarefas repetitivas, longas ou um conjunto de tarefas, as rotinas resolvem o problema. Pode ser composta por uma lista armazenada de dois ou mais comandos de aplicações que, quando acionada por um programa, reproduz os comandos que foram programados. As instruções que formam o corpo da rotina são escritas num código próprio para que o computador as possa entender, essa linguagem é designada por VBA – Visual Basic for Applications. O VBA é uma poderosa ferramenta, para automatizar alguns procedimentos que, geralmente facilitam nosso trabalho em diversas situações.

SALVANDO UM ARQUIVO PARA SUPORTAR MACROS

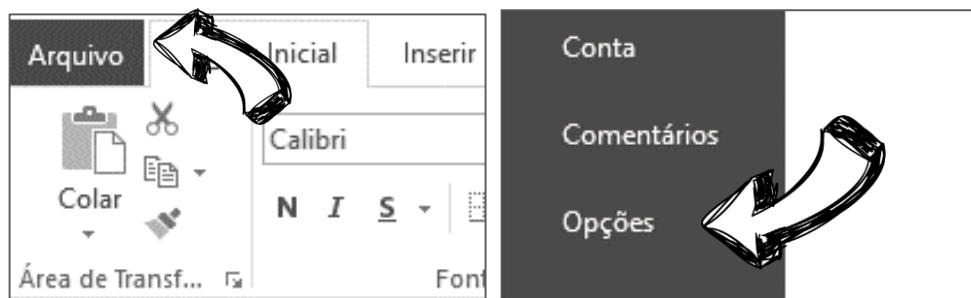
Para que suas macros funcionem de forma correta para uso posterior, você deve salvá-la de modo correto, ou seja, alterar o formato da pasta de trabalho de xlsx para xlsxm. Para realizar esta alteração, basta na hora de salvar seu documento, clicar na opção **Tipo** e selecionar a opção **Pasta de Trabalho Habilitada para Macros do Excel (*.xlsxm)**.



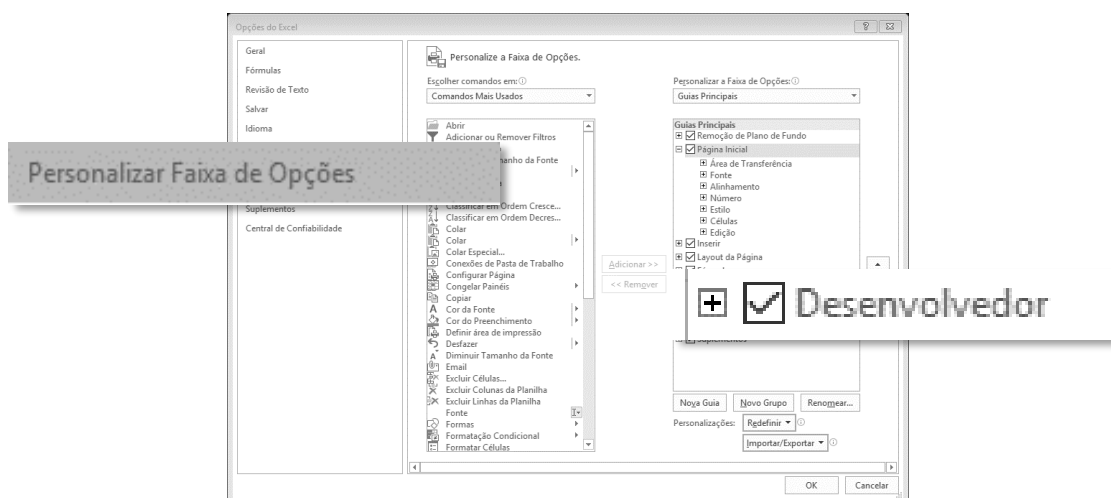
GUIA DESENVOLVEDOR

A guia Desenvolvedor é de extrema importância quando o objetivo é utilizar recursos avançados do Excel. Através deste menu é possível ativar diretamente os suplementos, ativar editar, excluir e gravar uma Macro, inserir controles de formulário, controles ActiveX, código-fonte, etc.

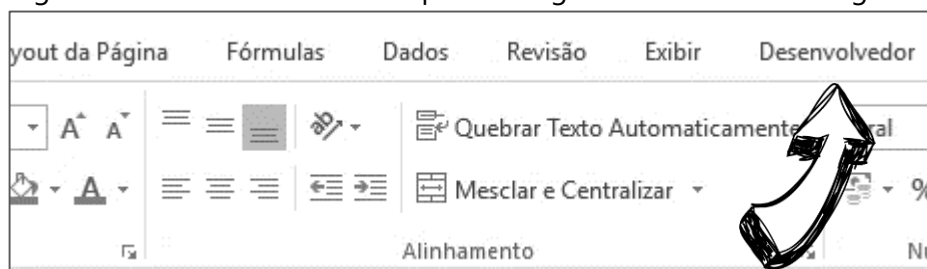
Para isso faça o seguinte: Clique sobre a guia **Arquivo e Opções**:



Na janela de opções do Excel, clique sobre a opção **"Personalizar faixa de Opções"** e selecione a opção **"Desenvolvedor"** e pressione o botão **OK**.

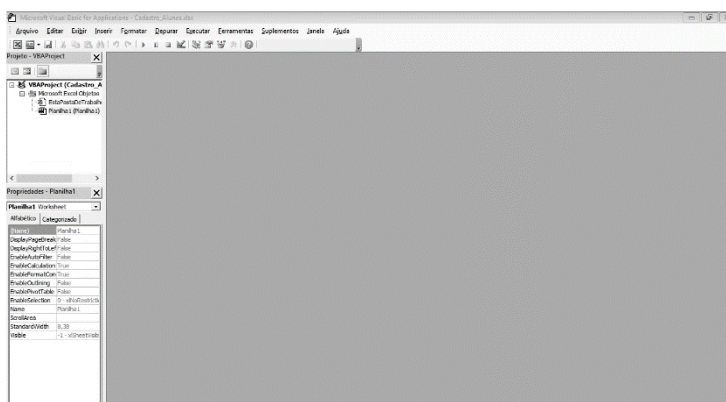


A guia **Desenvolvedor** deverá aparecer logo no final das demais guias no Excel.



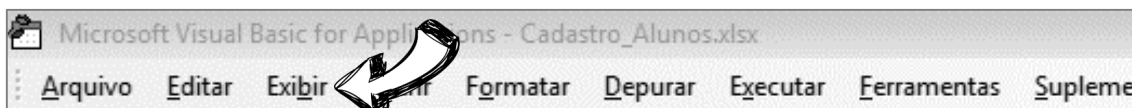
EDITOR DO VISUAL BASIC

Para ativar o editor do Visual Basic, você pode clicar sobre a guia **Desenvolvedor** e no canto esquerdo você encontrará o botão **VISUAL BASIC**. Caso prefira utilizar o teclado, podemos utilizar as teclas **CTRL + F11** para acessar a tela do *Visual Basic for Applications*.



Na parte superior temos a barra de menu, iremos aprender apenas os necessários e seus principais comandos.

MENU EXIBIR



Os principais comandos que utilizaremos serão os seguintes:

COMANDO

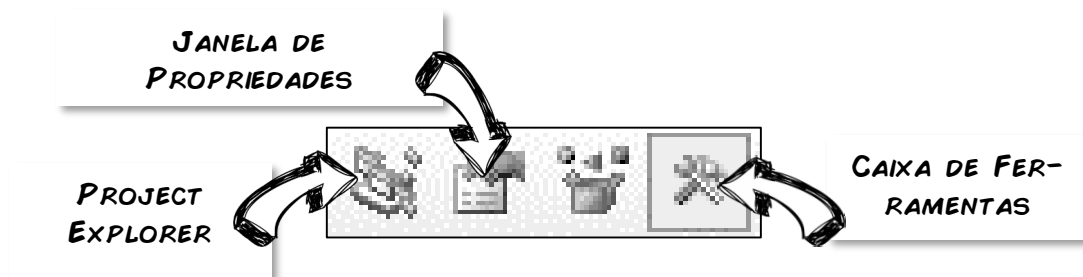
FUNCIONALIDADE

CODIGO

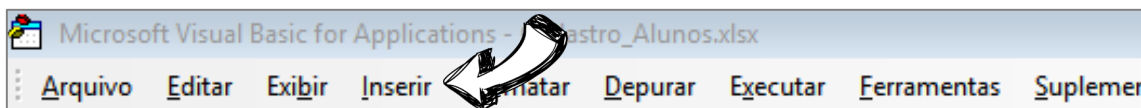
Atalho de Teclado: F7

Abre a janela de código de um objeto.

OBJETO <i>Atalho de Teclado: Shift + F7</i>	Retornar ao objeto
PROJECT EXPLORER <i>Atalho de Teclado: Ctrl + R</i>	Habilita a janela de projetos, que fica localizada geralmente no lado esquerdo do editor do Visual Basic.
JANELA DE PROPRIEDADES <i>Atalho de Teclado: F4</i>	Ativa a janela de propriedades de um objeto.
CAIXA DE FERRAMENTAS	Mostra todas as ferramentas para utilizar no desenvolvimento do sistema.



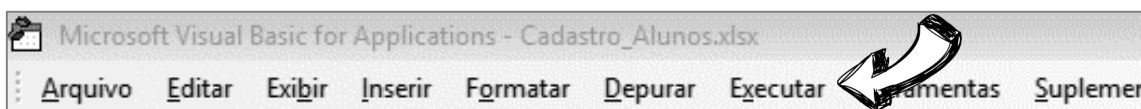
MENU INSERIR



COMANDO	FUNCIONALIDADE
USERFORM	Inserir uma janela de formulário ao projeto
MÓDULO	Utilizado para inserir módulos no projeto.



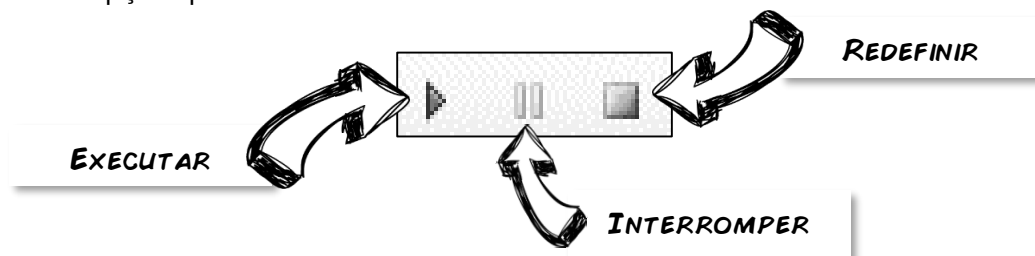
MENU EXECUTAR



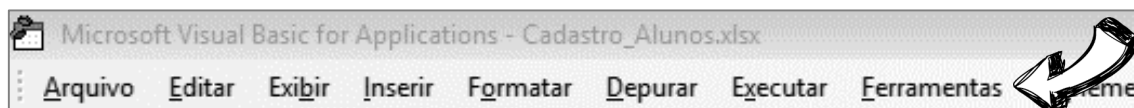
COMANDO	FUNCIONALIDADE
EXECUTAR USERFORM	Utilizamos para iniciar as macros.

INTERROMPER	Pausa a macro em execução.
REDEFINIR	Para todos com comandos em execução.

Estas opções podem ser localizadas também na barra de ferramentas do VBA.



MENU FERRAMENTAS



COMANDO	FUNCIONALIDADE
REFERÊNCIAS	Habilita ou desabilita bibliotecas se necessário.
MACROS	Mostra todas as macros criadas.

II. TRABALHANDO COM MÓDULOS

Os Módulos servem para armazenar códigos de procedimentos e códigos de funções. Eles podem ser utilizados em uma planilha no Excel como se fossem funções internas ou executados em resposta a um evento (clique de um botão, atalho de teclado, ativar uma planilha, etc.).

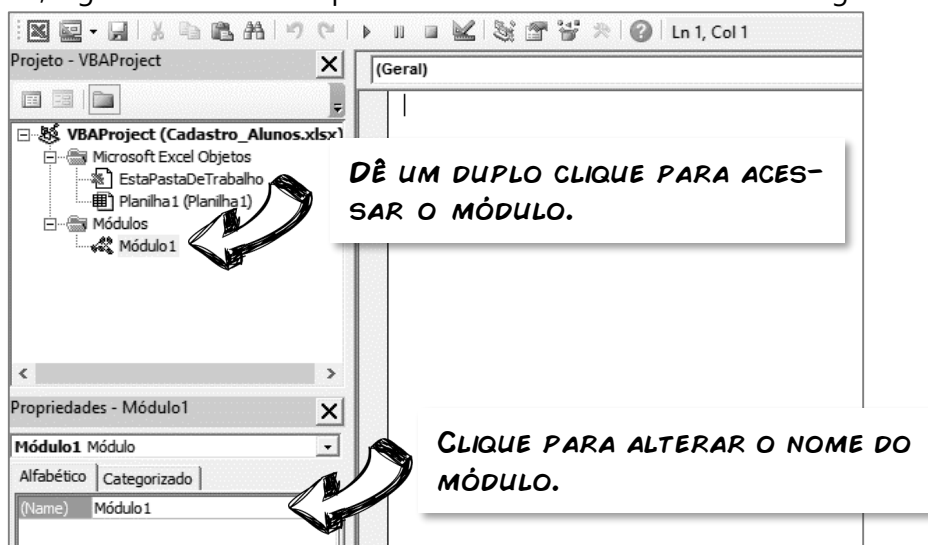
TIPOS DE MÓDULOS

Módulos padrão: contêm procedimentos de uso geral que podem ser executados a partir de qualquer lugar da pasta de trabalho e a partir de outros módulos.

Módulos de classe: estão associados a um formulário ou relatório. Em geral, contêm procedimentos de evento que são executados em resposta a um evento no formulário ou relatório, como por exemplo, o clique do mouse sobre um botão de comando.

CRIANDO MÓDULOS

No Editor do Visual Basic, clique em Inserir – Módulo. Você poderá renomear os módulos se necessário. Abra a janela Propriedades, se não estiver aberta, pressione F4 para abrir. Clique no módulo que queira renomear, localize o campo (Name) na janela Propriedades, digite o novo nome e pressione Enter conforme mostra a imagem a seguir.



Vamos criar uma linha de comando que mostre uma mensagem de texto. Digite o comando abaixo:


```
Sub Primeira_Macro()  
    MsgBox "Olá Mundo!"  
End Sub
```

Após realizar escrever o comando, clique dentro o texto e pressione o botão **F5**.

Se você digitou os comandos corretamente, deverá aparecer uma caixa de mensagem semelhante a imagem abaixo:



DEU ERRO?

Caso ao invés da mensagem, apareceu uma mensagem qualquer de erro. Provavelmente o código que você digitou está incorreto. Se este fato ocorrer contigo, pressione o botão de confirmação que aparecer, no geral será **OK** ou **Depurar** e em seguida, quando retornar a tela do VBA, clique menu **Executar** e selecione a opção **Redefinir**, ou pressione o botão presente na barra de ferramentas ().

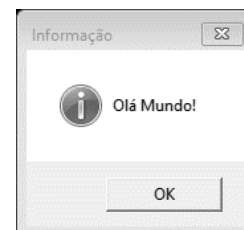
Após os scripts parados, verifique a escrita dos comandos. Acredite, um acento, letra, aspas ou ponto e vírgula fora do lugar pode fazer seu script não funcionar.

PERSONALIZANDO UMA CAIXA DE MENSAGEM – PARTE 1

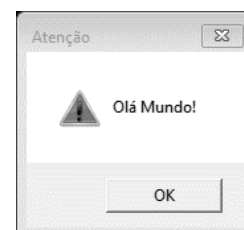
Um dos principais recursos quando o assunto é informar o usuário, a caixa de mensagem é um item fundamental para qualquer programador em qualquer tipo de linguagem. Este recurso possibilita a ele, enviar informações ou até mesmo alternativas para o usuário escolher.

A estrutura da caixa de mensagem pode ser alterada conforme a necessidade do programador, e pode ser personalizada alterando apenas alguns detalhes em sua linha de comando, conforme podem ser notadas nas imagens a seguir.

```
Sub Primeira_Macro()  
    MsgBox "Olá Mundo!", vbInformation, "Informação"  
End Sub
```



```
Sub Primeira_Macro()  
    MsgBox "Olá Mundo!", vbExclamation, "Atenção"  
End Sub
```



```
Sub Primeira_Macro()  
    MsgBox "Olá Mundo!", vbCritical, "Erro"  
End Sub
```



Estrutura da mensagem de texto

MSGBOX "CORPO DA MENSAGEM", TIPO DA MENSAGEM, "TÍTULO"

Utilizando algum conhecimento e caracteres a mais, podemos personalizar uma caixa de mensagem conforme nossa necessidade, como demonstrado a linhas de comandos abaixo:

```
Sub exibe_data_hora()  
    mes_atual = Month(Date)  
    mes_nome = MonthName(mes_atual, False)  
    mensagem = "Ano atual: " & Year(Date) & Chr(13)  
    mensagem = mensagem & "Mês atual: " & mes_nome & Chr(13)  
    mensagem = mensagem & "Hoje é dia: " & Day(Date) & Chr(13)  
    mensagem = mensagem & Chr(13) & "*****" & Chr(13) & Chr(13)  
    hora_atual = Hour(Time())  
    minuto_atual = Minute(Time())  
    segundo_atual = Second(Time())  
    mensagem = mensagem & "Hora atual: " & hora_atual & Chr(13)  
    mensagem = mensagem & "Minuto atual: " & minuto_atual & Chr(13)  
    mensagem = mensagem & "Segundo atual: " & segundo_atual  
    MsgBox mensagem  
End Sub
```

O comando **Chr(13)** é utilizando quando você deseja realizar uma quebra de linha dentro da mensagem. Já o caractere **&** é utilizando quando queremos juntar um valor com outro. Este tem a mesma funcionalidade quando se é utilizado para fazer referência a duplas sertanejas como Fernando & Sorocaba, Zezé & Luciano ou Chitãozinho & Chororó.

Existem várias outras formas de se trabalhar com uma caixa de mensagem que abordaremos em outros capítulos.



FAÇA SUAS ANOTAÇÕES AQUI

[illegible]

III. DECLARAÇÃO DE VARIÁVEIS

Agora vamos iniciar o estudo de uma série de comandos e conceitos básicos da linguagem VBA. Esses comandos serão utilizados nas lições do próximo capítulo, onde apresentaremos alguns exemplos práticos do uso do VBA para solução de problemas com o Excel.

DECLARAÇÃO DE VARIÁVEIS

Uma variável é um espaço criado na memória do computador, que fica reservado o armazenamento de informações ou valores. Fazemos referência a este espaço utilizando nomes. Para declararmos uma variável, utilizamos o comando Dim, conforme exemplificado abaixo:

Dim n

Dim nome

Neste caso a variável foi declarada apenas com o nome, sem ser declarado seu tipo (texto, inteiro, data, etc). Quando declaramos uma variável sem tipo, ela é considerada do tipo Variant (variável pode aceitar qualquer tipo de valor).

A sintaxe para o comando Dim é a seguinte:

Dim nome_da_variável As tipo_da_variável

Também podemos declarar mais do que uma variável, com um único comando Dim. Para isto, basta separar as variáveis por vírgula, conforme exemplo a seguir:

Dim x, bola, casa As String

IMPORTANTE: O Tipo da variável define quais dados podem ser armazenados nela. Por exemplo, variáveis que armazenam valores numéricos, não devem aceitar textos. Toda variável no VBA, é do tipo Variant, isto significa que a variável pode ser de qualquer tipo.

```
Sub Somatorio()  
    Dim a As Integer  
    Dim b As Integer  
    Dim C As Integer  
  
    a = 5  
    b = 2  
    C = a + b  
  
    MsgBox "A variável C vale: " & C  
End Sub
```

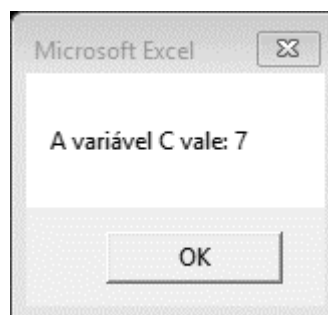


Tabela - Subtipos do tipo Variant disponíveis no VBA:

SUBTIPO	DESCRIÇÃO
Boolean	Aceita apenas valores: Verdadeiro ou Falso (True ou False).
Byte	Valor inteiro, na faixa de 0 até 255.
Integer	Valor inteiro, na faixa de -32768 até 32767.
Currency	Valores entre -923.337.203.685.447,5808 até 922.337.203.685.447,5807
Long	Valor inteiro, na faixa de -2.147.483.648 até 2.147.483.647.
Date(Time)	Valor de data a partir de 01 de Janeiro do ano 100.

String	Texto de tamanho variável.
Object	Pode conter um objeto qualquer, como um Activex
Error	Pode conter um número de erro.

OPERADORES ARITMÉTICOS

Para realizarmos cálculos entre variáveis ou valores, utilizamos os operadores. Na Tabela a seguir, temos uma descrição dos operadores que podemos utilizar:

OPERADOR	SÍMBOLO	DESCRIÇÃO
Adição	+	Soma o valor de duas ou mais variáveis.
Subtração	-	Subtração entre duas ou mais variáveis.
Multiplicação	*	Multiplica os valores de duas ou mais variáveis.
Divisão	/	Divide o valor de duas ou mais variáveis.
Inteiro da Divisão	\	Retorna a parte inteira, da divisão.
Exponenciação	^	(a^b) O valor de a, elevado na potência b
Modulo	Mod	Retorna o resto de uma divisão.

Exemplo utilizando todas as fórmulas de cálculo.

```
Sub calcula()
    Dim x As Integer
    Dim y As Integer
    Dim z As Integer

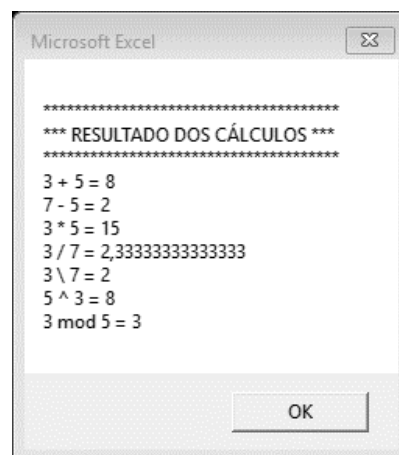
    x = 3
    y = 5
    z = 7

    soma = x + y
    subtrair = z - y
    multiplica = x * y
    divide1 = z / x
    divide2 = z \ x
    expoente = y ^ x
    resto = x Mod y
```

```
mensagem = "*****" & Chr(13)
mensagem = mensagem & "*** RESULTADO DOS CÁLCULOS ***" & Chr(13)
mensagem = mensagem & "*****" & Chr(13)
mensagem = mensagem & x & " + " & y & " = " & soma & Chr(13)
mensagem = mensagem & z & " - " & y & " = " & subtrair & Chr(13)
mensagem = mensagem & x & " * " & y & " = " & multiplica & Chr(13)
mensagem = mensagem & x & " / " & z & " = " & divide1 & Chr(13)
mensagem = mensagem & x & " \ " & z & " = " & divide2 & Chr(13)
mensagem = mensagem & y & " ^ " & x & " = " & soma & Chr(13)
mensagem = mensagem & x & " mod " & y & " = " & resto & Chr(13)
```

```
MsgBox mensagem
```

```
End Sub
```



IV. FUNÇÃO IF

Em determinadas situações, existe a conveniência de realizarmos comparações entre valores de variáveis ou expressões. Com base nesta comparação, temos os seguintes resultados (**Verdadeiro** ou **Falso**). Neste caso, utilizamos a função If, sua estrutura seria If (E se) Then (então) Else (senão).

Sintaxe da função

```
If Condição a ser verificada Then
    Código se a situação for verdadeira
Else
    Código caso a situação seja falsa
End If
```

EXEMPLO 1

Ao pressionarmos o botão, o Excel deverá verificar se o valor digitado na célula **C3** é maior do que 20 e retornar o resultado na célula **H3**.

	A	B	C	D	E	F	G	H	I	J
1										
2										
3			30					← O valor é maior que 20?		
4										
5										
6										

Sempre, antes de digitar o código, escreva-o em uma folha ou algo semelhante para eliminar possíveis erros.

No VBA, é possível utilizar o símbolo do apóstrofo (') para criar observações na linha código. Essas observações não aparecem durante a execução do sistema.

SE O VALOR DA CÉLULA C3 FOR MAIOR QUE 20 ENTÃO, A CÉLULA C3 SERÁ IGUAL AO TEXTO SIM. SENÃO, A CÉLULA C3 SERÁ IGUAL AO TEXTO NÃO. FIM DA CONDIÇÃO

```
Sub Maior_que_20()

    'Se o valor da célula C3 for maior que 20 Então,
    If Range("c3") > 20 Then
        'a célula C3 será igual ao texto SIM
        Range("H3") = "SIM"
    'Senão,
    Else
        'a célula C3 será igual ao texto NÃO
        Range("H3") = "NÃO"
    'Fecha o If
    End If

End Sub
```

O comando **Range** é utilizado quando queremos fazer uma referência a uma ou mais células.

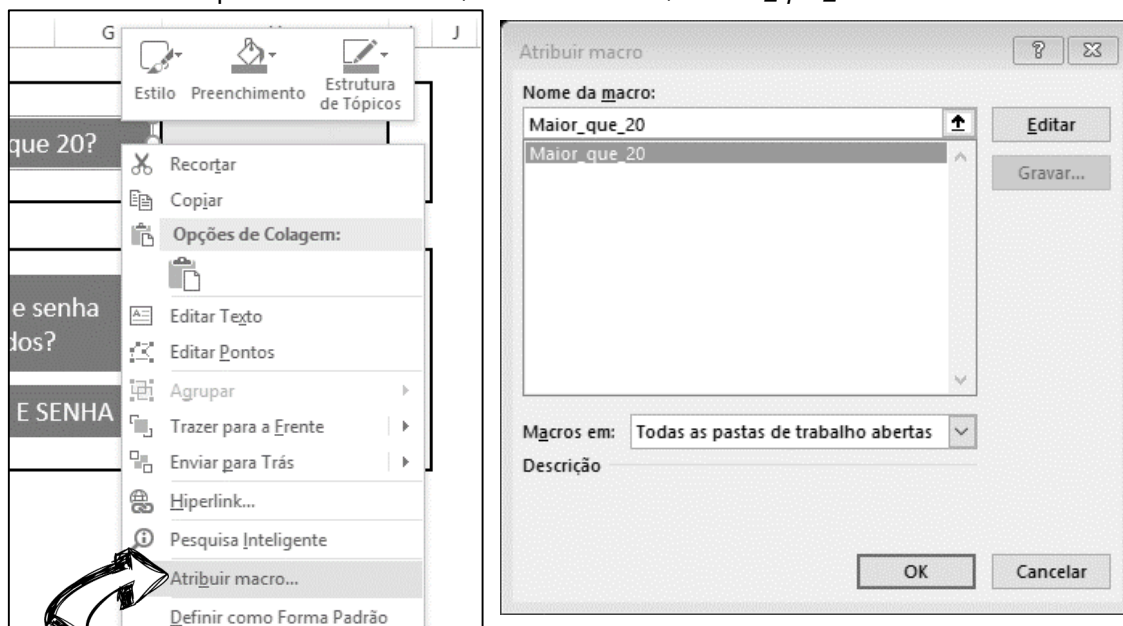
Exemplo: **Range("C3")** ou **Range("C3:C5")**

Não confunda o comando **Range** com **Ranger**.



Feito o comando, vamos atribuir a macro ao botão. Para isto, clique com o botão direito do mouse sobre a imagem que será o botão e selecione a opção **ATRIBUIR MACRO**.

Feito este processo, aparecerá a janela **Atribuir Macro**, aonde você deverá selecionar a macro que acabou de criar, em nosso caso, "*Maior_que_20*".



Terminado este processo, assim que você digitar algum valor na célula C3 e pressionar o botão, a *macro* será executada.

EXEMPLO 2

No segundo exemplo, iremos verificar se os campos de usuário e senha presente nas células C8 e C10 estão preenchidos. Perceba que neste caso, deveremos analisar dois campos distintos e retornar um resultado verdadeiro se um ou outro não estiver preenchido. Quando precisamos fazer duas verificações e enviar um resultado caso uma **ou** outra seja verdadeira, utilizamos a variável **OR**.

Se colocarmos os comandos no papel, ficaria da seguinte forma:

SE O VALOR DA CÉLULA C8 ESTIVER VAZIO OU O VALOR DA CÉLULA C10 ESTIVER VAZIA, ENTÃO, A CÉLULA H8 SERÁ IGUAL A MENSAGEM "PREENCHA TODOS OS CAMPOS" SENÃO, A CÉLULA H8 SERÁ IGUAL A MENSAGEM "OK"

```

Sub Verifica_Celulas()

    ' Se o valor da célula C8 estiver vazio ou
    ' o valor da célula C10 estiver vazia então
    If Range("C8") = "" Or Range("C10") = "" Then
        ' A célula H8 será igual a mensagem:
        ' "PREENCHA TODOS OS CAMPOS"
        Range("H8") = "PREENCHA TODOS OS CAMPOS"
    ' Senão
    Else
        ' A célula H8 será igual a mensagem "OK"
        Range("H8") = "OK"
    ' Fecha o If
    End If

End Sub

```

Atribua a macro e faça o teste.

EXEMPLO 3

Neste último exemplo, vamos comparar o usuário e senha digitados nas células C8 e C10 com suas referências presente nas células D14 e E14. Deverá aparecer a mensagem de "OK" somente se os valores das células forem iguais. Quando necessitar de analisar alguma situação em que um ou mais itens devem ser verdadeiros, utilizamos o recurso **AND**.

SE O CONTEÚDO DA CÉLULA C8 FOR IGUAL AO DA CÉLULA D14 E A CÉLULA C10 FOR IGUAL AO DA CÉLULA E14 ENTÃO, DEVERÁ APARECER O TEXTO "OK", SENÃO APARECERÁ O TEXTO "DADOS DO USUÁRIO INVÁLIDO! "

```

Sub Verifica_Login()

    'Se o conteúdo da célula c8 for igual ao da célula d14
    ' e o valor da célula c10 for igual ao da célula e14, então
    If Range("C8") = Range("D14") And Range("C10") = Range("E14") Then

```



```

        'deverá aparecer o texto "ok",
        Range("H10") = "OK"
    'Senão aparecerá o texto
Else
    '"dados do usuário inválido! "
    Range("H10") = "Dados do usuário inválidos!"
'Fechando o If
End If

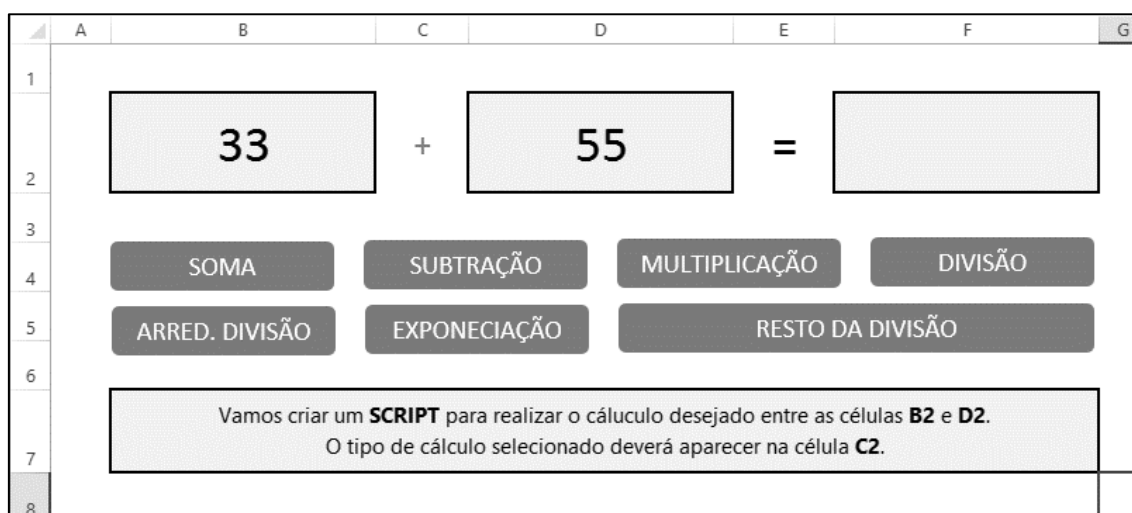
```

End Sub

Atribuindo a Macro no botão, você poderá testar seu **script**. Perceba que só aparecerá o texto **OK** apenas se os campos forem iguais. O VBA nota a diferença entre maiúsculas e minúsculas, logo admin é diferente de ADMIN ou Admin. Posteriormente, trataremos a respeito destes “detalhes”.

EXERCÍCIOS

Agora chegou a sua hora de colocar seu conhecimento em prática. Apesar de termos o vício de tentar resolver as coisas logo de cara, tente escrever primeiro o que cada script deverá fazer e em seguida os comandos.



Exemplo botão SOMA

DESCRIÇÃO DO COMANDO:

QUANDO PRESSIONAR O BOTÃO SOMA, O EXCEL DEVERÁ VERIFICAR SE AS CÉLULAS B2 E D2 ESTÃO PREENCHIDAS, SE ESTIVEREM, NA CÉLULA C2 DEVERÁ APARECER O SINAL DE SOMA (+) E O RESULTADO DA SOMA NA CÉLULA F2.

SCRIPT DO COMANDO:

```

SUB SOMAR()
    IF <COMPARAÇÃO> THEN
        <COMANDO>
    ELSE
        <COMANDO>
    END IF
END SUB

```



FAÇA SUAS ANOTAÇÕES AQUI

[illegible]

V. ESTRUTURA DE REPETIÇÃO

Neste capítulo veremos as funções de repetições de ações. Este recurso é de fundamental importância quando precisamos que o sistema tome alguma decisão e repita alguma ação.

Se houver a necessidade de repetir um grupo de comandos, podemos utilizar um loop (retorno), o que permitirá a execução dos comandos repetidamente. Alguns loops repetem instruções até que uma condição seja **FALSO**, já outros, repetem os comandos até uma condição seja **VERDADEIRA**. Também existem loops que repetem instruções um número específico de vezes.

DO WHILE

Usamos esta estrutura quando você deseja repetir um grupo de comandos um número indefinido de vezes, enquanto a condição especificada permanecer **VERDADEIRA** (True).

SINTAXE DA FUNÇÃO

```
Do While condição
    Grupo de comandos que será executado.
Loop
```

EXEMPLO

Em uma planilha, precisamos que o Excel escreva em uma sequência de células os valores de 1 até 9. Estes valores devem começar a serem digitados na célula B4 e depois passar pela B5, B6 e assim por diante, até chegar ao valor 9.

```
Sub Maior_que_20()

    Dim n As Integer
    Dim cel As Integer

    n = 1    'Informo o valor inicial da contagem
    cel = 4  'Informo o número da primeira célula

    Do While n <= 9
        Range("B" & cel) = "Número: " & n
        n = n + 1
        cel = cel + 1
    Loop

End Sub
```

Escreva os comandos, crie o botão na planilha, atribua a macro e execute-a.

Lembre-se que o comando **Do While** funciona até chegar a um valor negativo. Logo, o script irá funcionar enquanto o valor de **n** for menor ou igual a 9.

FOR... NEXT

Utilizamos a estrutura **FOR...NEXT** quando precisarmos repetir um conjunto de instruções durante um número definido de vezes.

Esta função trabalha com um contador e executa as função até que a quantidade de **loops** atinja o número do contador ou a função seja finalizada.

SINTAXE DA FUNÇÃO

```
For i = Valor inicial to Valor final
    Grupo de comandos que será executado
Next
```

EXEMPLO 1

Seguindo a mesma lógica da função Do While, vamos criar uma macro que crie uma numeração de 1 até 20 a partir da célula A1 até a célula A20.

O script seria o seguinte:

```
Sub Contador()

    For i = 1 To 20
        Range("A" & i) = i
    Next

End Sub
```

Escreva os comandos, crie o botão na planilha, atribua a macro e execute-a.

Neste caso, a variável **i** inicia com o valor zero e este valor vai sendo somado a mais um, sempre que ele chega ao final do script. Ou seja, ao comando NEXT.

EXEMPLO 2

Na planilha abaixo temos uma lista de produtos e dois botões: Um para procurar os dados na tabela e mostrar os valores e outra para realizar as alterações com base no código.

	A	B	C	D	E	F	G
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							

CÓDIGO	PRODUTO	QUANTIDADE
10	MARTELO	3
11	ALICATE	6
12	PARAFUSOS	9
13	PORCAS	8
14	CHAVE DE BOCA	1
15	CHAVE DE FENDA	4
16	MAÇARICO	3
17	FURADEIRA	7

CÓDIGO	15
PRODUTO	
QUANT.	

PROCURAR PELO CÓDIGO

ALTERAR DADOS

Primeiramente, vamos iniciar programando a **macro** para procurar um produto pelo código.

```
Sub procurar()

    'Valor inicial 3 porque o primeiro cadastro
    'inicia na linha 3 e 10 é a linha do último.
```

```

For i = 3 To 10
    If Range("B" & i) = Range("g2") Then
        Range("G3") = Range("C" & i)
        Range("G4") = Range("D" & i)
        End 'Finaliza a macro assim que
            'encontrar o valor
    End If
Next

End Sub

```

Quando rodar a macro, a função começa a procurar pelo código digitado na célula **G2** e assim que for localizado, as informações começarão a ser carregadas nas células que foram indicadas e o script é finalizado através do comando **END**.

O código para realizar as alterações no banco de dados, seria o seguinte:

```

Sub Gravar()

    'Valor inicial 3 porque o primeiro cadastro
    'inicia na linha 3 e 10 é a linha do último.

    For i = 3 To 10
        If Range("B" & i) = Range("g2") Then
            Range("C" & i) = Range("G3")
            Range("D" & i) = Range("G4")
            MsgBox "Alteração Realizada!"
            End 'Finaliza a macro assim que
                'encontrar o valor
        End If
    Next

End Sub

```

Perceba que o código teve uma variação mínima em comparação com o **script** anterior. Neste, as células que antes enviariam as informações, agora estão recebendo.

EXERCÍCIOS

Crie um pequeno banco de dados com CÓDIGO, PRODUTO, QUANTIDADE e VALOR. Em seguida, crie um pequeno formulário que adicione os dados no banco de dados, realize alterações e faça pesquisas.

Cada ação realizada, deverá aparecer uma mensagem de texto informando que foi concluída, com exceção da pesquisa, que deverá mostrar a mensagem apenas se o código do produto não for encontrado.

	FAÇA SUAS ANOTAÇÕES AQUI
---	---------------------------------

VI. MANIPULAÇÃO DE CÉLULAS E PLANILHAS

Realizar alterações, em planilhas, seja na inserção, edição ou exclusão de dados em uma célula, linha ou coluna no Excel é de extrema importância para quem deseja extrair o melhor desempenho desta ferramenta.

Neste capítulo, vamos aprender a trabalhar com alguns comandos para realizar a manipulação das planilhas no Excel.

OBJETO WORKSHEETS

Este objeto representa uma planilha e através dele conseguimos realizar alterações em qualquer planilha dentro da pasta de trabalho.

Abaixo, listarei alguns comandos e procedimentos que podem ser realizados ao se trabalhar com o WORKSHEETS. Mas antes, vamos entender a sintaxe deste objeto:

WORKSHEETS("PLANILHA").COMANDO

Dentro dos parênteses, você pode colocar o nome da planilha que deseja trabalhar ou o número da mesma com relação as outras na sua pasta de trabalho. Ex: Se você precisa realizar uma alteração na segunda planilha, ficaria **worksheets(2)**. Após o ponto, entraria a propriedade que gostaria de realizar na planilha.

Confira alguns comandos básicos do objeto WORKSHEETS

- **Worksheets.Add:** Adiciona uma nova planilha.
- **Worksheets("planilha1").Visible = False:** Oculta a planilha 1.
- **Worksheets("planilha1").Visible = True:** Deixa visível a planilha 1.
- **Worksheets("planilha1").Select:** Seleciona a Planilha 1.
- **Worksheets("planilha2").Delete:** Exclui a Planilha 2.
- **Worksheets(1).Name = "Novo Nome":** Altera o nome da primeira planilha para "Novo Nome".

EXEMPLO 1

Vamos criar uma Macro para deixar oculta ou visível a planilha "Banco de Dados" ao clicar no botão.

Lógica do que o exercício está pedindo:

SE A PLANILHA BANCO DE DADOS ESTIVER VISÍVEL ELA FICARÁ OCULTA, CASO CONTRÁRIO, ELA FICARÁ VISÍVEL.

Com base na lógica descrita acima, creio que você tenha percebido que utilizaremos a função If.



```
Sub Banco_de_Dados()
    If Worksheets("Banco de Dados").Visible = True Then
        Worksheets("Banco de Dados").Visible = False
    Else
        Worksheets("Banco de Dados").Visible = True
    End If
End Sub
```

OU

```
Sub Banco_de_Dados()
    If Worksheets(2).Visible = True Then
        Worksheets(2).Visible = False
    Else
        Worksheets(2).Visible = True
    End If
End Sub
```

Como neste modelo a planilha banco dedados é a segunda na ordem das planilhas, podemos informar apenas o número da planilha (2) ao invés de inserir o nome da mesma (Banco de Dados).

EXEMPLO 2

Nesta próxima atividade, vamos montar um script para alterar o nome da primeira planilha, mas, antes desta alteração, deverá aparecer uma janela perguntando qual o nome da planilha.

Lógica do que o exercício está pedindo:

DEVERÁ APARECER UMA JANELA PEDINDO PARA O USUÁRIO INSERIR O NOVO NOME DA PLANILHA. APÓS DIGITADO O NOME, O EXCEL IRÁ RENOMEAR A PRIMEIRA PLANILHA COM O NOME QUE O USUÁRIO ESCOLHEU.



```
Sub Altera_Nome()  
    Dim nome As String  
  
    nome = InputBox("Informe o novo nome da planilha", _  
        "Criar Planilha")  
  
    If nome <> "" Then  
        Worksheets(1).Name = nome  
    Else  
        MsgBox "Você não informou o nome da planilha!", _  
            vbInformation, "Operação Cancelada!"  
    End  
End If  
End Sub
```

NOVIDADES NA LINHA DE COMANDO

Inputbox: Utilizamos este comando quando queremos que o usuário insira alguma informação para ser processada. Neste caso, o novo nome da planilha.

Uso do underline (_): Este comando é utilizado quando queremos realizar uma quebra de linha no código. Caso você perceba que sua linha de código ficará muito extensa, poderá utilizar o underline _ para continuar seus comandos na próxima linha.

UCase: Utilizamos este comando quando queremos colocar algum texto todo em maiúscula, como podemos verificar no exemplo abaixo

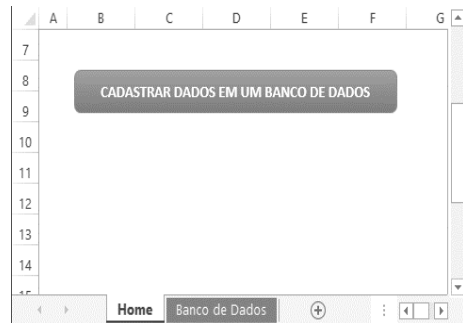
EXEMPLO 3

Vamos criar um cadastro simples onde o sistema irá armazenar os dados na planilha Banco de dados e retornar uma mensagem de texto informando que o cadastro foi adicionado com sucesso.

Lógica do que o exercício está pedindo:

DEVERÁ APARECER UMA JANELA PEDINDO PARA O USUÁRIO INSERIR O NOME DO NOVO CADASTRO. APÓS DIGITADO, O EXCEL DEVERÁ BUSCAR A PRÓXIMA LINHA YAZIA NA PLANILHA BANCO DE DADOS E INSERIR O NOME E CRIAR UMA CAIXA DE MENSAGEM INFORMANDO QUE OS DADOS FORAM SALVOS.

```
Sub Cadastro()  
    Dim nome As String  
    nome = UCase(InputBox("Informe o nome a ser adicionado"))  
    If nome = "" Then  
        MsgBox "Não foi informado nome para ser adicionado!"  
    Else  
        For i = 1 To 1000  
            If Worksheets(2).Range("A" & i) = "" Then  
                Worksheets(2).Range("A" & i) = nome  
                MsgBox nome & " adicionado com sucesso!"  
                End  
            End If  
        Next  
    End If  
End Sub
```



EXERCÍCIO

Abra a atividade do capítulo anterior, coloque o banco de dados em uma nova planilha e ajuste os scripts para funcionarem corretamente.



FAÇA SUAS ANOTAÇÕES AQUI

This image shows a blank sheet of white paper with horizontal ruling lines. The lines are evenly spaced and extend across the width of the page. There are no margins, text, or other markings on the paper.

VII. Exercícios

Chegou a hora de colocar seus conhecimentos em prática.

ATIVIDADE 1

Crie as macros necessárias para resolver as atividades da planilha abaixo. Link do arquivo: <https://goo.gl/vU6s9E>.

	A	B	C	D	E	F	G
1		DATA DE HOJE		MAIUSCULAS		MINUSCULAS	
2		Mostra uma caixa de mensagem com a data de hoje.		Deverá deixar os textos desta caixa todos em maiúsculas (UCASE) ou minúsculas (LCASE) quando pressionado o botão.		Deverá deixar todos os nomes da planilha NOMES em maiúsculos ou minúsculos.	
3							
4		RENAMEIA A 2ª PLANILHA		SOMAR		MOSTRAR NOME	
5		Deverá aparecer um inputbox perguntando qual o novo nome da segunda planilha e alterar o nome assim que o input for preenchido.		Em um inputbox vou digitar o primeiro valor, em outra informo o segundo valor. O resultado da soma deverá aparecer em uma mensagem de texto.		Quando pressionar o botão, deverá aparecer um inputbox pedindo qual a posição do nome. Após digitar o número (Exemplo: 5). Aparecerá uma caixa de mostrando o nome da planilha NOMES que está na posição informada.	
6							
7		MOSTRA OU OCULTA		MULTIPlicAR POR 9		PAR OU IMPAR	
8		Deve mostrar ou ocultar a segunda planilha.		Em um inputbox, eu digitarei um valor e logo em seguida, deverá aparecer uma mensagem de texto com o resultado.		Em um inputbox, eu digitarei um número e assim que digitar OK, deverá aparecer uma mensagem informando se o valor que eu digitei é par ou ímpar.	
9							
10							

ATIVIDADE 2

Faça o download da planilha abaixo através do link (<https://goo.gl/Sj3qeG>) e programe os botões conforme as especificações abaixo:

	A	B	C	D
1	NOME		TOTAL DE CADASTROS	
2	SEXO			
3	DATA DE NASCIMENTO		DATA ÚLTIMO BACKUP	
4	IDADE			
5	<div> <div>NOVO</div> <div>SALVAR</div> <div>VOLTAR</div> <div>AVANÇAR</div> <div>BACKUP</div> </div>			

	A	B	C	D
1	CÓDIGO	NOME	SEXO	NASCIMENTO
2	01			
3	02			
4	03			
5	04			
6	05			
7	06			
8	07			
9	08			
10	09			
11	10			
12	11			

Observações: A célula IDADE, está configurada para calcular automaticamente através da fórmula: **=SE(D6<>"";ANO(HOJE())-ANO(D6);"")**. Já no campo TOTAL DE CADASTROS, está com a fórmula **=CONT.VALORES('Banco de Dados'!B2:B51)** para contar os cadastros realizados na planilha "Banco de Dados".

Nesta atividade você irá criar a programação de cada botão para funcionar conforme as especificações a seguir:

BOTÃO NOVO: Limpa os campos de preenchimento da planilha REGISTRO (nome, sexo e data de nascimento).

BOTÃO SALVAR: Deve adicionar todos os dados na próxima linha vazia da planilha BANCO DE DADOS apenas se os campos (nome, sexo e data de nascimento) estiverem preenchidos. As informações a serem salvas, devem ser convertidas para a letras maiúsculas.

BOTÃO VOLTAR: Irá mostra os cadastros em ordem decrescente, um a uma na planilha REGISTRO. Caso chegue no primeiro cadastro, deverá aparecer uma mensagem do tipo **informação** com os seguintes dizeres: "NÃO EXISTEM MAIS CADASTRO ANTERIORES".

BOTÃO AVANÇAR: Irá mostra os cadastros em ordem crescente, um a uma na planilha REGISTRO. Caso chegue ao último cadastro, deverá aparecer uma mensagem do tipo **informação** com os seguintes dizeres: "ESTE É O ÚLTIMO CADASTRO REGISTRADO".

BOTÃO BACKUP: Deverá realizar uma cópia de todos os cadastros existentes na planilha BANCO DE DADOS para a planilha BACKUP. Feita esta tarefa, na célula F8 da planilha REGISTRO, deverá aparecer a data e hora do backup. O comando que retorna a data e hora no VBA é o **NOW**.

ANTES DE CRIAR OS CÓDIGOS, ESCREVA PASSO A PASSO O QUE CADA BOTÃO DEVERÁ FAZER. ORGANIZE SUA LÓGICA DE FORMA CORRETA E SÓ ENTÃO COMECE A CRIAR OS SCRIPTS.



FAÇA SUAS ANOTAÇÕES AQUI

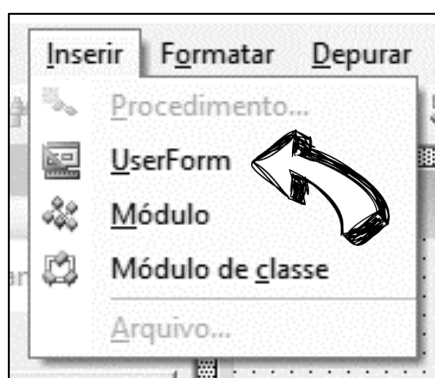
This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and extend across the width of the page. There are no margins, text, or other markings on the paper.

VIII. CRIAÇÃO DE FORMULÁRIOS

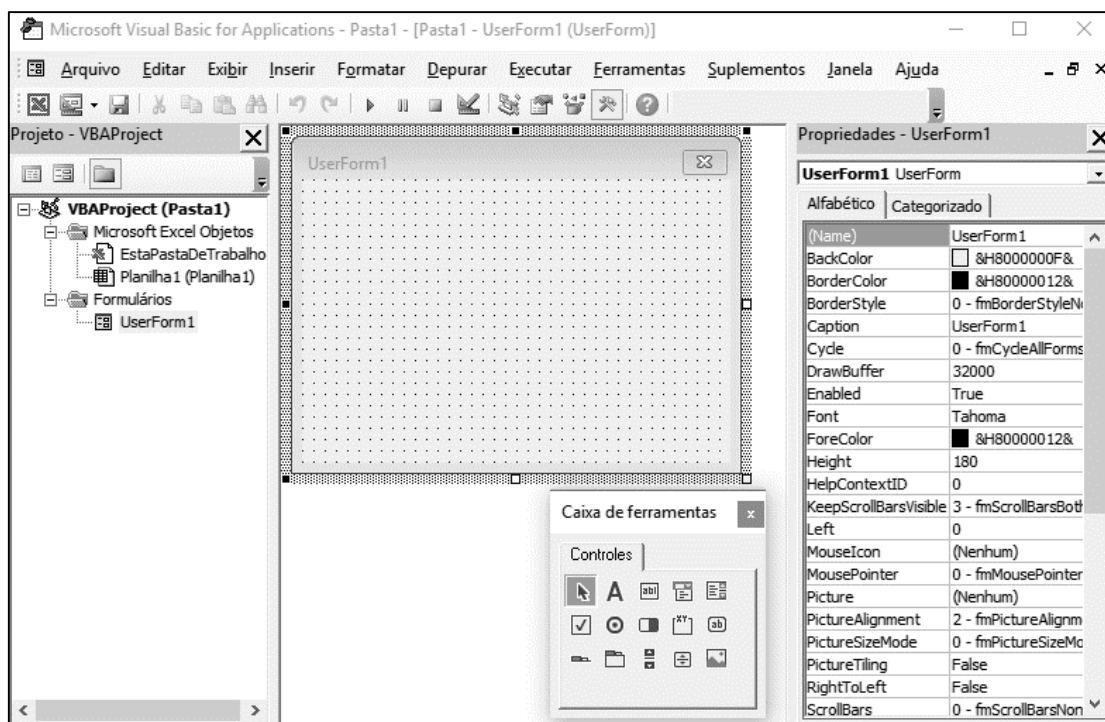
A utilização de formulários no Excel, são uma excelente ferramenta para a introdução de dados em uma planilha. Os objetos em um formulário (UserForm), como botões e caixas de texto, caixas de combinação e outros são chamados de controles. O formulário é bastante utilizado como uma plataforma para inserção de dados sem a necessidade de se manipular informações na planilha.


INSERINDO UM FORMULÁRIO

Para a inserção do nosso primeiro formulário, basta entrar na tela do Visual Basic for Application, nosso já conhecido VBA e no menu **INSERIR**, selecionar a opção **UserForm**.



A Janela que deverá aparecer será semelhante a imagem abaixo:



Caso a **Caixa de ferramentas** não apareça, basta clicar no menu **Exibir** e selecionar a opção **Caixa de ferramentas** ou clicar no botão () presente na barra de ferramentas padrão do VBA.

CRIANDO UM FORMULÁRIO SIMPLES

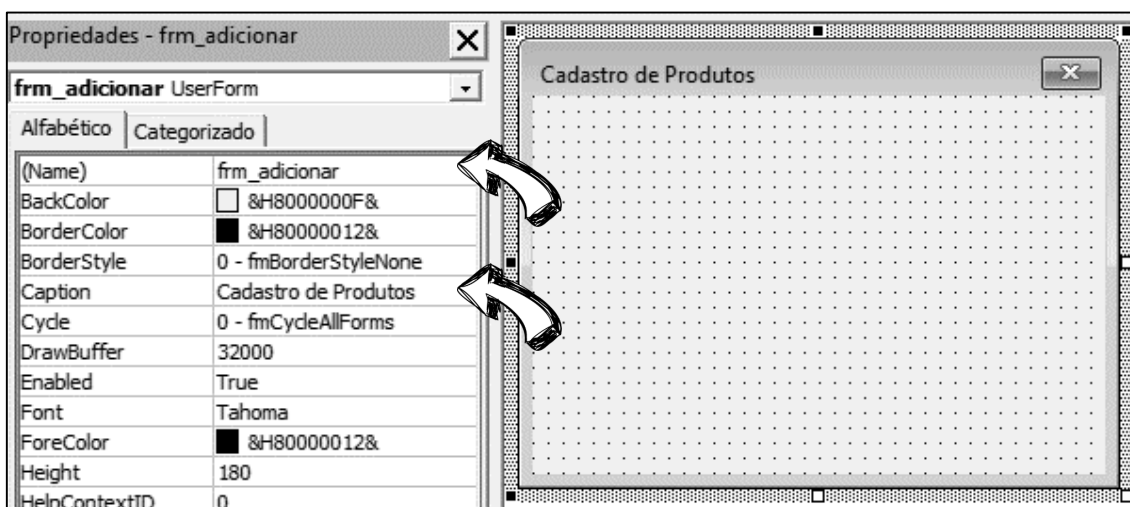
Após criar sua janela de formulário, vamos começar a personaliza-lo e inserir os botões de formulários que precisamos. Antes de começarmos, vamos criar uma planilha bem simples no Excel para servir como banco de dados.

	A	B	C	D
1	Código	Produto	Descrição	Quantidade
2				
3				

Terminada esta etapa, acesse a tela do VBA e peça um formulário conforme foi explicado no tópico anterior.

Com o formulário inserido, vamos personalizar seu nome e o texto o qual queremos que apareça na barra de título da janela. Clique no botão Janela 'Propriedades' na Barra de ferramentas Padrão ou pressione F4 para abrir a janela.

Para alterar o nome do formulário, dê um clique na barra de nome do formulário UserForm1 para selecioná-lo, em seguida, um duplo clique na propriedade (Name) e troque o valor desta propriedade para "frm_adicionar" e pressione Enter. Ainda na caixa de propriedades, clique sobre a opção Caption e altere o texto para "Cadastro de Produtos". Perceba que ao alterar esta propriedade, o nome que aparece na barra de título da janela é modificado, como podemos perceber na imagem abaixo:



CAIXA DE TEXTO

Clique no formulário para exibir a Caixa de ferramentas, caso ela não esteja visível, clicar no menu Exibir e selecionar a opção Caixa de ferramentas.

Assim que ela estiver aberta, selecione a opção Caixa de Texto. Após selecionar a opção, vá em qualquer parte do seu formulário, clique, segure e arraste para desenhar a caixa de texto.

Após desenhar a caixa. Clique sobre a janela Propriedades, selecione a opção (Name), mude o valor para txtCodigo e dê Enter.



Após realizar esta etapa, repita novamente este processo, desta vez, crie mais três caixas de textos com os nomes "txtProduto", "txtDescricao" e "txtQuantidade".

RÓTULO OU LABEL

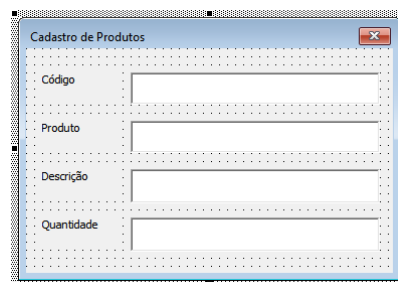
Para que possamos informar ao usuário qual o tipo de informação que cada caixa deverá receber, criamos um Rótulo ou Label. Botão este que está presente na Caixa de ferramentas como pode ser observado na imagem ao lado.

Depois de inserido, clique sobre ela e na janela Propriedades, altere a propriedade Caption para "Código" e tecle Enter.

Se necessário, redimensione o rótulo, arrastando a alça de seleção para perfazer um tamanho apropriado.

Repita o mesmo processo para e insira mais três rótulos "Produto", "Descrição" e "Quantidade".

Até o momento, seu formulário poderá estar semelhante a imagem ao lado.



BOTÕES DE COMANDO

Para que possamos executar ações, vamos incrementar nosso formulário com três botões de comando: um para inserir dados, outro para excluir um produto da lista de formulários e o terceiro para fechar a janela.

Para criar o primeiro botão, clique no Botão de comando na Caixa de ferramentas e em seguida no formulário.

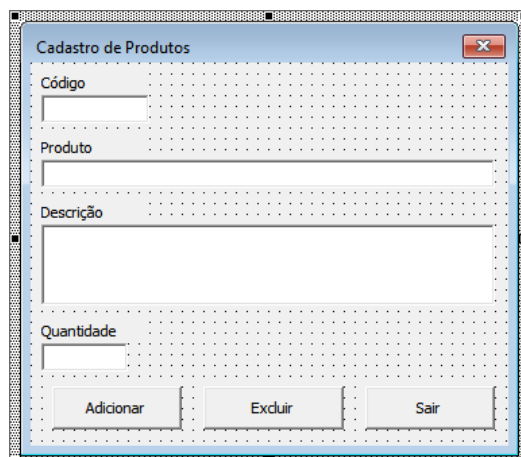
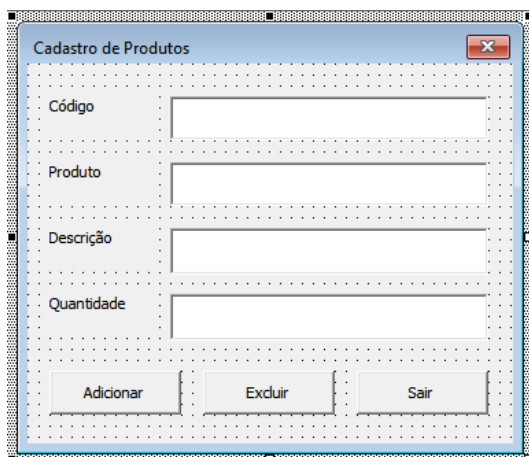
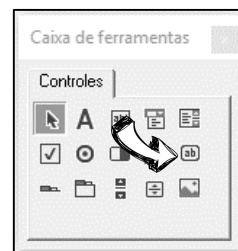
Com o botão selecionado, mude o valor da propriedade (Name) para "btnAdicionar" e a propriedade Caption para "Adicionar" e tecle Enter.

Redimensione o botão, clicando nele para selecioná-lo e, em seguida, arraste a alça do lado direito ou esquerdo do retângulo de seleção até chegar no tamanho desejado. Caso necessário, altere a posição do botão, arrastando-o de um local para outro.

Repita estes processos mais duas vezes para criar os botões "btnExcluir" e o "btnSair".

Sua janela final, poderá ficar semelhante a um dos modelos a seguir:

O interessante de se criar os formulários, é que eles podem ser criados de formas diferentes a ainda assim, atender uma única necessidade.

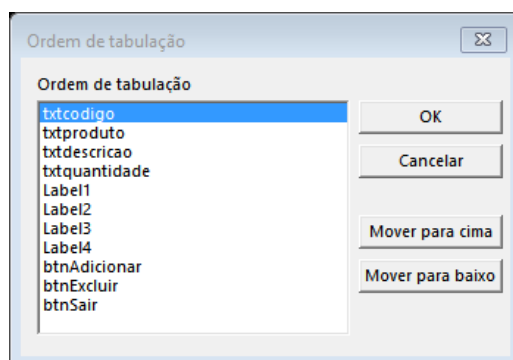


Pressione **F5** para ver o formulário em execução. Clique nos botões para testar seu funcionamento e depois feche a janela do formulário para voltar ao modo de desenvolvimento.

TABULAÇÃO

Pressionando repetidamente a tecla TAB podemos observar que a caixa de seleção se movimenta de controle para controle, mudando seu foco. Colocando o formulário em execução (**F5**) também é possível verificar a ordem de sequência do foco ao pressionar a tecla TAB.

A ordem de tabulação é importante para quem utiliza o teclado. Se a atual não for a ordem de tabulação lógica ela pode ser mudada. Com o formulário no modo de edição, clique no fundo do formulário acesse o menu Exibir clique na opção Ordem de tabulação. A caixa de diálogo, mostra nove controles cujas sequências podem ser alteradas, simplesmente, selecionando o controle desejado e clicando no botão Mover para cima ou para baixo para posicioná-lo corretamente.



CHAMANDO O FORMULÁRIO

Insira um módulo e nele, vamos criar uma macro chamada "Abrir_Form". A propriedade para se abrir um formulário criado é "Show". Logo, o script de criação seria o seguinte:

```
Sub Abrir_Form()
    frm_adicionar.Show
End Sub
```

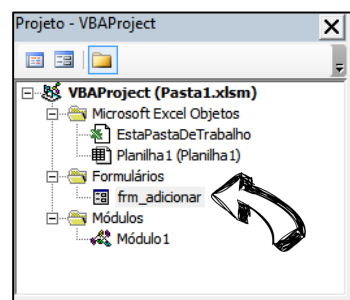
Após criar o script, desenhe um objeto ou botão no seu formulário, atribua a macro e teste.

PROGRAMANDO O BOTÃO ADICIONAR

Agora vamos colocar a mão na massa e criar o script de programação da nossa macro.

Para retornar ao formulário, caso esteja em um módulo ou algo do gênero, bata clicar na janela Projeto do VBA e dar dois cliques sobre o formulário "frm_adicionar".

Realizado este procedimento, vamos começar nossa programação com o botão "Adicionar". Para isso, dê um duplo clique no botão "Adicionar" e note que você será



redirecionado para uma janela semelhante à dos módulos, porém, a linha de comando será semelhante a imagem abaixo:



O script "Private Sub" é uma macro privada, direcionada apenas a um objeto e ação específicas:

Perceba no script que, o nome do botão vem seguido de "_Click()" ou seja, esta macro será executada somente quando o botão for pressionado. Existem outros modos e métodos de ação, que será abordado posteriormente.

ALGORÍTIMO

Lembrando sempre que, antes de iniciar qualquer tipo de programação, devemos ter em mente ou de preferência no papel as linhas com as tarefas que este script irá realizar que neste caso será a seguinte:

1. Se os campos código ou produtos estiverem vazios;
2. Então, aparecerá uma mensagem pedindo para preencher os campos;
3. Senão todos os dados serão adicionados na Planilha1 na primeira linha vazia sendo:
 - A. Coluna A = "Código"
 - B. Coluna B = "Produto"
 - C. Coluna C = "Descrição"
 - D. Coluna D = "Quantidade"
4. Aparece uma caixa de mensagem informando que o cadastro foi realizado;
5. Limpa os campos do formulário para adicionar outro código

Depois que rascunhei minha sequência de comandos, vamos montar o script poderá ser desenvolvido conforme abaixo:


```
Private Sub btnAdicionar_Click()
    '0. Descubra a primeira linha vazia
    For i = 1 To 1000
        'Se a linha da coluna A estiver vazia, Então
        If Range("A" & i) = "" Then
            'A variável linha receberá o número da linha vazia
            linha = i
            'Para o script
            i = 1000
        End If
    Next

    '1. Se os campos código ou produtos estiverem vazios então;
    If txtcodigo.Text = "" Or txtproduto.Text = "" Then
        '2. aparecerá uma mensagem pedindo para preencher os campos;
        MsgBox "Preencha os campos 'Código' e 'Produtos'", vbExclamation, "Atenção!"
    '3. Senão todos os dados serão adicionados na Planilha1 na primeira linha vazia sendo:
    Else
        'A.Coluna A = "Código"
        Worksheets("Planilha1").Range("A" & linha) = txtcodigo.Text
        'B.Coluna B = "Produto"
        Worksheets("Planilha1").Range("B" & linha) = txtproduto.Text
        'C.Coluna C = "Descrição"
        Worksheets("Planilha1").Range("C" & linha) = txtdescricao.Text
        'D.Coluna D = "Quantidade"
        Worksheets("Planilha1").Range("D" & linha) = txtquantidade.Text
        '4. Aparece uma caixa de mensagem informando que o cadastro foi realizado
        MsgBox "Cadastro adicionado com sucesso", vbInformation, "Concluído!"
        '5. Limpa os campos do formulário para adicionar outro código
        txtcodigo.Text = ""
        txtproduto.Text = ""
        txtdescricao.Text = ""
        txtquantidade.Text = ""
    End If
End Sub
```

PROGRAMANDO O BOTÃO EXCLUIR

Retorne ao formulário e dê um duplo clique no botão Excluir.

ALGORÍTIMO

1. Verifica se o campo código está vazio, então
 - A. Aparece uma mensagem pedindo para digitar um código;
 - B. A caixa de texto código deverá ser selecionada;
2. Se não,
 - A. Procura o código do cadastro nas linhas preenchidas
 - B. Encontrando
 - i. A linha será excluída;
 - ii. O campo código ficará vazio;
 - iii. Aparecerá uma mensagem informando;
 - C. Não Encontrando
 - i. Aparecerá uma mensagem dizendo que o código não existe;
 - ii. A caixa de texto código ficará em branco

SCRIPT

```
Private Sub btnExcluir_Click()
    '1. Verifica se o campo código está vazio, então
    If txtcodigo.Text = "" Then
        'A. Aparece uma mensagem pedindo para digitar um código;
        MsgBox "Informe o código do cadastro a ser excluído!", vbExclamation, "Atenção!"
        'A caixa de texto código deverá ser selecionada;
        txtcodigo.SetFocus 'Seleciona a célula
    '2. Se não,
    Else
    End If
End Sub
```

```
'A. Procura o código do cadastro nas linhas preenchidas
For i = 1 To 1000
    If Range("A" & i) = txtcodigo.Text Then
        produto_existe = "sim"
        linha = i
        i = 1000
    Else
        produto_existe = "não"
    End If
Next

'B.Encontrando
If produto_existe = "sim" Then
    'i. A linha será excluída;
    Worksheets("Planilha1").Range(linha & ":" & linha).Delete
    'ii. O campo código ficará vazio;
    txtcodigo.Text = ""
    'iii. Aparecerá uma mensagem informando;
    MsgBox "Produto excluído com sucesso!", vbInformation, "Concluído!"
'C.Não Encontrando
Else
    'i. Aparecerá uma mensagem dizendo que o código não existe;
    MsgBox "Código do cadastro não existe", vbInformation, "Não Informado!"
    'ii. A caixa de texto código ficará em branco
    txtcodigo.Text = ""
End If
End If
End Sub
```

PROGRAMANDO O BOTÃO SAIR

Retorne ao formulário e dê um duplo clique no botão Excluir.

ALGORÍTIMO

1. Aparecerá uma janela perguntando se o usuário deseja fechar a Janela;
2. Se sim, a janela será fechada

SCRIPT

```
Private Sub btnSair_Click()
    If MsgBox("Deseja realmente sair", vbQuestion + vbYesNo, "Sair?") = vbYes Then
        Unload Me
    End If
End Sub
```

Existem vários modos de se realizar a programação de uma macro ou um formulário, a pesquisa e dedicação são de extrema importância para o aprendizado.

	FAÇA SUAS ANOTAÇÕES AQUI
---	--------------------------

[illegible]

IX. TRABALHANDO ENTRE FORMULÁRIOS

Hora de colocar a mão na massa, neste capítulo, vamos colocar em prática grande parte do que vimos até o momento e ainda vamos aprender alguns comandos e propriedades mais ao desenvolver um sistema de gerenciamento de vendas simples.

Antes de iniciar nosso trabalho, é necessário lembrar que é de vital importância para o seu projeto, que você coloque no papel. Faça um rascunho do que você deseja alcançar, um esboço dos controles e o que cada botão irá fazer.

Os arquivos para o desenvolvimento desta atividade, estão presentes no seguinte endereço web: <https://goo.gl/sWFGmB>

ESBOÇO DO PROJETO

Nosso projeto vai contar com 4 planilhas que são: Cadastro de clientes, produtos, vendas e relatório de vendas.

De igual modo, o projeto contará com 6 formulários: Janela principal do projeto, cadastro de clientes, produtos, vendas, relatório de vendas e login de acesso a planilha.

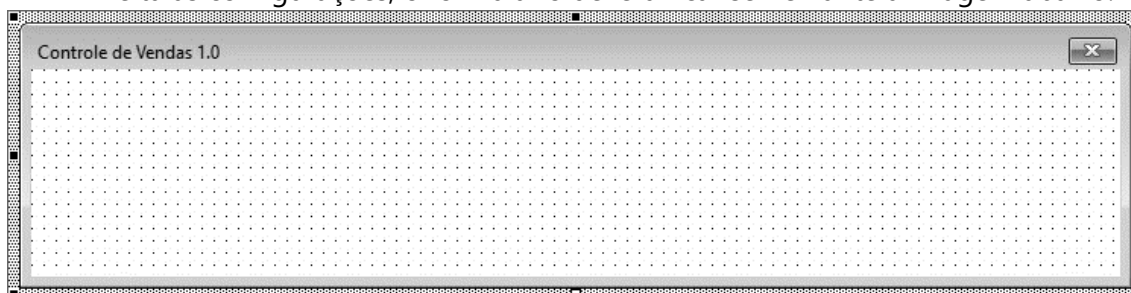
FORMULARIO INICIAL

Abra a tela do VBA e peça um formulário UserForm clicando em Inserir / User-Form.

Feito este processo, realize as alterações abaixo neste formulário:

Propriedade	Configuração
(Name)	frm_principal
Caption (Texto do formulário)	Controle de Vendas 1.0
BackColor (Cor de fundo)	&H00FFFFFF& (Branco)
Height (altura do formulário)	131
Width (largura do formulário)	550

Feita as configurações, o formulário deverá ficar semelhante a imagem abaixo:



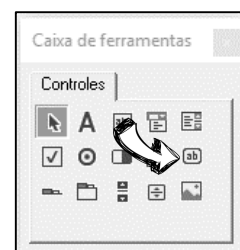
BOTÕES DO PROJETO

O formulário principal, será composto por 4 botões: cadastro de produtos, clientes, vendas e relatório de vendas.

Para criar o primeiro botão, clique no Botão de comando na Caixa de ferramentas e em seguida no formulário.

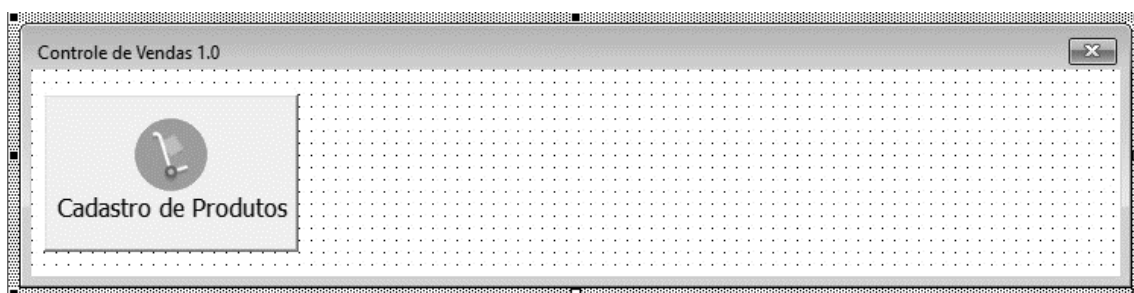
Com o botão criado, vamos realizar as configurações deste que será responsável pelo "Cadastro de Produtos".

Clique sobre o botão e altere suas propriedades para as listadas abaixo:



Propriedade	Configuração
(Name)	btn_produtos
Caption (Nome do Botão)	Cadastro de Produtos
Picture (imagem)	Clique no botão [...] e selecione o ícone: icon-boxes.ico
Picture Position	7 – Picture Position Above Center
Height (altura do botão)	78
Width (largura do botão)	126
Top (Distancia do topo)	12
Left (Distância da esquerda)	6

O resultado final, ficará idêntico ao modelo abaixo:



Seguindo os mesmos passos, crie mais 3 botões e coloque as seguintes configurações neles:

BOTÃO CLIENTE

Propriedade	Configuração
(Name)	btn_clientes
Caption (Nome do Botão)	Cadastro de Clientes
Picture (imagem)	Clique no botão [...] e selecione o ícone: icon-users.ico
Picture Position	7 – Picture Position Above Center
Height (altura do botão)	78
Width (largura do botão)	126
Top (Distancia do topo)	12
Left (Distância da esquerda)	138

BOTÃO VENDAS

Propriedade	Configuração
(Name)	btn_vendas
Caption (Nome do Botão)	Cadastrar Venda
Picture (imagem)	Clique no botão [...] e selecione o ícone: icon-market.ico
Picture Position	7 – Picture Position Above Center
Height (altura do botão)	78
Width (largura do botão)	126
Top (Distancia do topo)	12

Left (Distância da esquerda)	270
------------------------------	-----

BOTÃO RELATÓRIO DE VENDAS

Propriedade	Configuração
(Name)	btn_relatorio
Caption (Nome do Botão)	Relatório de Vendas
Picture (imagem)	Clique no botão [...] e selecione o ícone: icon-graph.ico
Picture Position	7 – Picture Position Above Center
Height (altura do botão)	78
Width (largura do botão)	126
Top (Distancia do topo)	12
Left (Distância da esquerda)	402

Ao término das configurações, nosso formulário ficará semelhante ao modelo abaixo:



MACRO

Terminado nosso formulário inicial, vamos criar uma macro para abrir este form. Crie um novo módulo e crie o seguinte script:

```
Sub abrir_form()
    frm_principal.Show
End Sub
```

Atribua a macro ao botão "Abrir formulário de controle" presente na primeira planilha do nosso arquivo e veja se funcionou.

Funcionando, o resultado deverá ser o seguinte:



FORMULÁRIO DE PRODUTOS

Analisando a planilha de cadastro de produtos, trabalhamos com 5 colunas (Código, produto, categoria, quantidade e valor) e para trabalhar de modo prático, vamos criar um formulário que realize todo o trabalho de gerenciamento da planilha.

Crie um novo UserForm, coloque as seguintes propriedades e configure as caixas conforme o esquema a seguir:

Propriedade	Configuração
(Name)	frm_produto
Caption (Texto do formulário)	Cadastro de Produtos
Height (altura do formulário)	216
Width (largura do formulário)	259

Monte o formulário abaixo seguindo as instruções:

O diagrama mostra um formulário de usuário com o título 'Cadastro de Produtos'. Ele contém os seguintes elementos e pontos de referência:

- 01:** Caixa de texto para 'Código'.
- 02:** Botão de lupa para buscar.
- 03:** Caixa de texto para 'Produto'.
- 04:** Caixa de texto para 'Categoria'.
- 05:** Caixa de texto para 'Quantidade'.
- 06:** Caixa de texto para 'Valor'.
- 07:** Botão 'Novo' com ícone de mais (+).
- 08:** Botão 'Gravar' com ícone de disquete.
- 09:** Botão 'Excluir' com ícone de menos (-).

01 Propriedade (Caixa de Texto)	Configuração
(Name)	txt_codigo
Top (Distancia do topo)	24
Left (Distancia da esquerda)	12
Height (altura do formulário)	18
Width (largura do formulário)	66
TabIndex (Ordem dos objetos)	0

02 Propriedade (Botão de comando)	Configuração
(Name)	btn_buscar
Top (Distancia do topo)	24
Left (Distancia da esquerda)	78

Height e Width	18
Caption	[Deixar em branco]
Picture	Selecione o ícone: icon-search.ico
Picture Position	Above Center
TabIndex (Ordem dos objetos)	1

03 Propriedade (Caixa de Texto)	Configuração
(Name)	txtproduto
Top (Distancia do topo)	60
Left (Distancia da esquerda)	12
Height (altura do formulário)	18
Width (largura do formulário)	228
TabIndex (Ordem dos objetos)	2

04 Propriedade (Caixa de Combinação)	Configuração
(Name)	txtcategoria
Top (Distancia do topo)	96
Left (Distancia da esquerda)	12
Height (altura do formulário)	18
Width (largura do formulário)	102
TabIndex (Ordem dos objetos)	3

05 Propriedade (Caixa de Texto)	Configuração
(Name)	txtquantidade
Top (Distancia do topo)	96
Left (Distancia da esquerda)	120
Height (altura do formulário)	18
Width (largura do formulário)	60
TabIndex (Ordem dos objetos)	4

06 Propriedade (Caixa de Texto)	Configuração
(Name)	txtvalor
Top (Distancia do topo)	96
Left (Distancia da esquerda)	186
Height (altura do formulário)	18
Width (largura do formulário)	54
TabIndex (Ordem dos objetos)	5

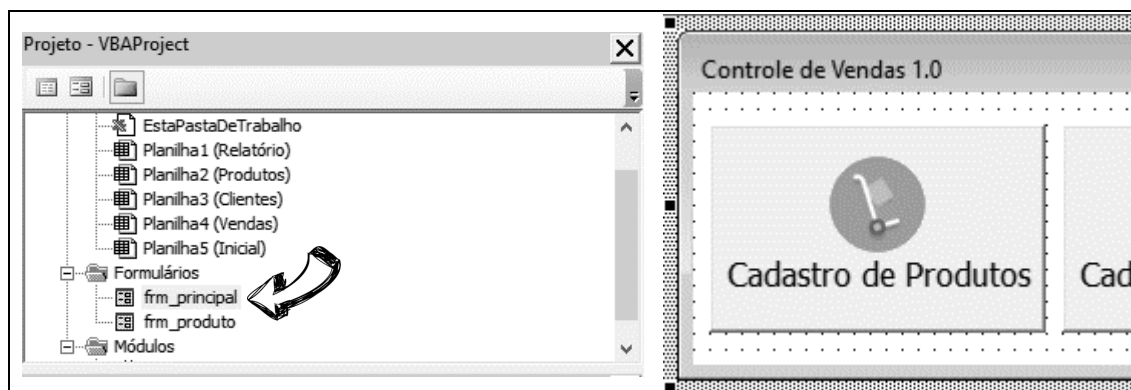
07 Propriedade (Botão de comando)	Configuração
(Name)	btn_novo
Top (Distancia do topo)	126
Left (Distancia da esquerda)	6
Height	54
Width	72

Caption	Novo
Picture	Selecione o ícone: icon-add.ico
Picture Position	Above Center
TabIndex (Ordem dos objetos)	6

08 Propriedade (Botão de comando)	Configuração
(Name)	btn_gravar
Top (Distancia do topo)	126
Left (Distancia da esquerda)	84
Height	54
Width	72
Caption	Gravar
Picture	Selecione o ícone: icon-save.ico
Picture Position	Above Center
TabIndex (Ordem dos objetos)	7

09 Propriedade (Botão de comando)	Configuração
(Name)	btn_excluir
Top (Distancia do topo)	126
Left (Distancia da esquerda)	168
Height	54
Width	72
Caption	Excluir
Picture	Selecione o ícone: icon-del.ico
Picture Position	Above Center
TabIndex (Ordem dos objetos)	8
Enabled (Botão ativo ou inativo)	False

Agora vamos configurar o botão “Cadastro de Produtos” que está na janela frm_produto. Na caixa de projeto do VBA, dê dois cliques sobre a opção “frm_principal” e em seguida, um duplo clique no botão “Cadastro de Produtos”.



LÓGICA DO BOTÃO

Ao pressionar o botão, deverá abrir a janela de cadastro de produtos.

```
Private Sub btn_produtos_Click()  
    'Abre o formulário  
    frm_produto.Show  
End Sub
```

Com base nas planilhas, crie os formulários **cadastro de clientes** e **cadastar venda**. Crie os scripts para que, quando clicar nos botões, abram as janelas.


 FAÇA SUAS ANOTAÇÕES AQUI

[illegible]

X. PROJETO AGENDA

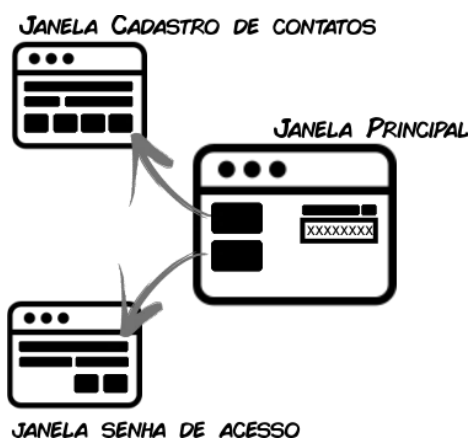
Chegou a hora de colocarmos nosso conhecimento em “cheque” e criar um sistema do zero. Neste capítulo, você poderá acompanhar todo o processo para a criação de um projeto e agenda telefônica e ainda aprender algumas coisas.

ESBOÇO DO PROJETO

Antes de iniciar um projeto, é de vital importância que se faça um desenho do mesmo, pode ser em uma folha sulfite, em um caderno, tanto faz. O principal é você ter ideia do que deseja montar, como será sua base de dados, quantos formulários vai precisar, etc...

Neste projeto, iremos precisar de duas planilhas (Banco de dados), uma para cadastrar os contatos e outra para armazenar os dados de acesso da planilha (usuário e senha).

No quesito formulários, iremos utilizar uma principal (pai) e outras duas (filhas): uma para cadastrar os contatos e outra para cadastrar os dados de acesso a própria planilha. Também criaremos um botão de login para acessar o formulário principal.



MONTANDO O BANCO DE DADOS

A principal peça do nosso projeto, o banco de dados deve ser pensado e analisado para que você tenha um melhor proveito do seu formulário. Neste caso, para o cadastro dos contatos, pensei em uma planilha com quatro campos: código, nome, telefone e e-mail.

Sempre que for criar um banco de dados, adicione uma coluna que não poderá ser alterada pelo usuário. Neste caso, criei a coluna código que será a “coluna chave” todos os outros campos podem ser alterados com exceção desta planilha. Ela será fundamental na hora de colocar nossa planilha para trabalhar

CONTATOS

Nomeie a primeira planilha como Contatos e monte a tabela a seguir:

	A	B	C	D	E	F	G
1						NOVA LINHA	
2	CÓDIGO	NOME	TELEFONE	E-MAIL		PROCURA	
3						RESULTADO DA BUSCA	
4	100	ANA	67 5555-9858	não tem			
5	101	JOÃO	67 5352-9865	joão@site.com			
6	102	PEDRO	67 9632-5236	pedro@site.com			

Perceba que na coluna F e G, foi criada uma planilha de referência para poder trabalhar melhor no desenvolvimento do VBA. Esses campos ajudam, pois diminuem as linhas de programação e melhoram a performance do nosso sistema.

NOVA LINHA: Este campo deverá informar em qual linha será adicionado um novo cadastro. Perceba que na linha 4 é onde deverá ser adicionado o primeiro cadastro. Na coluna A, existe apenas um valor, vamos utilizar esta coluna como referência para descobrir a linha vazia, utilizando a função **CONT.VALORES**:

E	F	G	H
	NOVA LINHA	=CONT.VALORES(A:A)+3	
	PROCURA		
	RESULTADO DA BUSCA		

PROCURA: Neste campo iremos adicionar o nome do usuário que iremos procurar.

RESULTADO DA BUSCA: Se o nome digitado na célula de PROCURA estiver na coluna dos nomes, o Excel deverá informar em que linha está este cadastro, senão, retornará o valor 0 (zero).

Para a realização desta, utilizaremos a função **CORRESP** juntamente com a **SEERRO**.

E	F	G	H
	NOVA LINHA		
	PROCURA		
	RESULTADO DA BUSCA	=SEERRO(CORRESP(G2;B:B;0);0)	

Se você colocar o nome ANA como o primeiro cadastro e colocar novamente na célula referente a PROCURA, verá que o Excel irá retornar o número 4. Ou seja, o nome ANA está na quarta linha da coluna B. Poderemos utilizar este valor quando formos buscar algum contato adicionado.

ACESSO

Crie uma segunda planilha e nomeie como **Acesso**. Nesta planilha, iremos colocar apenas três campos: Usuário, senha e dica de senha. Estes dados serão utilizados no acesso a lista de contatos ou quando for alterar alguma informação de acesso. Não foi criada uma coluna chave como código ou algo do gênero, pois o sistema irá trabalhar apenas com uma única linha de dados.

	A	B	C
1	USUÁRIO	SENHA	DICA DE SENHA
2	Fulano	123	Conte até 3

MONTANDO OS FORMULÁRIOS

Vamos iniciar montando os formulários que iremos utilizar no nosso sistema. Neste capítulo, será informado apenas as configurações principais de cada objeto do formulário. Com base no capítulo anterior, você conseguirá ajustar os objetos para ficar o mais semelhante possível com a imagem de cada formulário.

Os ícones utilizados neste capítulo, poderão ser encontrados no seguinte endereço: <https://goo.gl/sWFGmB>

FORMULÁRIO DE CONTATOS

Formulário	Configuração
(Name)	frm_contato
Caption	Cadastro de Contatos
Height	240
Width	252

01 – Caixa de Combinação	Configuração
(Name)	txt_contato
Font	Fonte: Segoe UI, tamanho 12
02 – Caixa de Texto	Configuração
(Name)	txt_telefone
Font	Fonte: Segoe UI, tamanho 12
03 – Caixa de Texto	Configuração
(Name)	txt_email
Font	Fonte: Segoe UI, tamanho 12
04, 5 e 6 – Botão de Comando	Configuração
(Name)	cmd_novo, cmd_salvar e cmd_excluir
Enable do botão excluir	False

FORMULÁRIO DE SENHA

O formulário 'Dados de Acesso' possui os seguintes elementos:

- 01**: Campo de texto para 'Nome do Usuário'.
- 02**: Botão 'Alterar Senha'.
- 03**: Campo de texto para 'Dica de Senha'.
- 04**: Botão 'Salvar' com ícone de disquete.

Formulário	Configuração
(Name)	frm_senha
Caption	Dados de Acesso
Height	230
Width	252

01 – Caixa de Texto	Configuração
(Name)	txt_usuario
Font	Fonte: Segoe UI, tamanho 12
02 – Caixa de Texto	Configuração
(Name)	txt_senha
Font	Fonte: Segoe UI, tamanho 12
03 – Caixa de Texto	Configuração
(Name)	txt_dica
Font	Fonte: Segoe UI, tamanho 12
04 – Botão de Comando	Configuração
(Name)	cmd_salvar

FORMULÁRIO PRINCIPAL

O formulário 'Agenda de Contatos' possui os seguintes elementos:

- 01**: Botão 'CONTATOS' com ícone de pessoa.
- 02**: Botão 'SEGURANÇA' com ícone de cadeado.
- 03**: Campo de texto 'Busca Rápida'.
- 04**: Campo de texto 'Telefones'.
- 05**: Campo de texto 'E-mail'.

Estrutura

Formulário	Configuração
(Name)	frm_principal
BackColor	[Cor Branco]
Caption	Agenda de Contatos
Height	195
Width	350

01 e 02 – Botão de Comando	Configuração
(Name)	cmd_contato e cmd_senha
3 – Caixa de Combinação	Configuração
(Name)	txt_contato
Font	Fonte: Segoe UI, tamanho 12
04 e 05 – Caixa de Texto	Configuração
(Name)	txt_telefone e txt_email
BackColor	&H00C0FFFF&
BorderStyle	1 – Border Style Single
Enable	False
Font	Fonte: Segoe UI, tamanho 12

FORMULÁRIO DE LOGIN



Estrutura

Formulário	Configuração
(Name)	frm_login
BackColor	[Cor Branco]
Caption	Login

Height	225
Width	225

01 – Rótulo	Configuração
BackStyle	0 - Back Style Transparent
Caption	Agenda de Contatos
Font	Fonte: Segoe UI, tamanho 16
TextAlign	2 – Text Align Center
02 e 03 – Caixa de Texto	Configuração
(Name)	txt_usuario e txt_senha
BackColor	&H00C0FFFF&
BorderStyle	1 – Border Style Single
Font	Fonte: Segoe UI, tamanho 12
04 – Botão de comando	Configurações
(Name)	cmd_login
Caption	Entrar
PicturePosition	1 – Position Left Center

CRIANDO OS SCRIPTS

Agora vamos colocar a mão na massa e desenvolver os scripts para cada elemento de cada formulário criado.

FORMULÁRIO PRINCIPAL

Este formulário servirá como um formulário pai, ou seja, responsável pelo acesso aos demais formulários. Sem este formulário, o usuário não conseguirá fazer nada na planilha.

BOTÃO CONTATOS



```
Private Sub cmd_contato_Click()
    frm_contato.Show
End Sub
```

BOTÃO SEGURANÇA

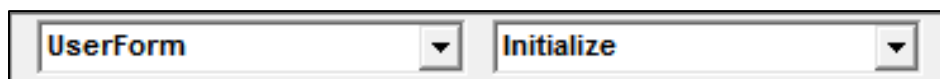


```
Private Sub cmd_senha_Click()
    frm_senha.Show
End Sub
```

PREENCHER O BOTÃO BUSCA RÁPIDA

Você já deve ter percebido que o VBA apesar de prático, não dá as coisas de mão beijada. Se você acessar a planilha dos contatos, fazer alguns cadastros e depois abrir o formulário de contatos, verá que a caixa "Busca Rápida" está vazia. Isto porque não informamos quais os dados que irão compor esta lista.

Dê um duplo clique em alguma área em branco do formulário. Na caixa de configuração, coloque do seguinte modo:



O que iremos fazer é informar que, ao **iniciar o formulário** (*initialize*), o VBA deverá pegar todos os dados que estão cadastrados na planilha "contatos" e listar no objeto *txt_contato*.

```
Private Sub UserForm_Initialize()
    'Cria uma variável para pegar a primeira e a última linha cadastrada
    célula = "Contatos!B4:B" & Worksheets("Contatos").Range("G1") - 1
    'Informa o grupo de células na planilha
    txt_contato.RowSource = célula
End Sub
```

REALIZANDO A BUSCA RÁPIDA

A dinâmica deste comando será a seguinte: Assim que selecionar um nome desta lista, ele deverá buscar no banco de dados e retornar os dados da pesquisa.

```
Private Sub txt_contato_Change()
    'Se a caixa contato está vazia, então
    If txt_contato.Text = "" Then
        'Limpa os campos telefone e email
        txt_telefone.Text = ""
        txt_email.Text = ""
    'Senão
    Else
        'Coloca a variável no campo pesquisa
        Worksheets("Contatos").Range("G2") = txt_contato.Text
        'Pega o resultado da pesquisa
        Resultado = Worksheets("Contatos").Range("G3")
        'Se encontrar o resultado, então
        If Resultado > 0 Then
            'Coloca os dados no formulário
            txt_telefone = Worksheets("Contatos").Range("C" & Resultado)
            txt_email = Worksheets("Contatos").Range("d" & Resultado)
        'Senão
        Else
            'Informa que o cadastro não existe
            MsgBox "Cadastro não existe", vbExclamation, "Concluído!"
        'Fecha If
        End If
    'Fecha If
    End If
End Sub
```

FORMULÁRIO DE CONTATOS

NOVO CADASTRO

Ao clicar no botão NOVO, o sistema deverá limpar os campos do formulário e o código do novo cadastro. Para que isto ocorra, devemos criar uma variável pública que se chamará código.

Para criar esta variável, basta clicar em qualquer parte do formulário, quando a abrir a tela de scripts, clique no início da primeira linha, crie a variável e dê Enter.

O script ficar semelhante ao modelo abaixo:

```
Dim COD As Integer
```

Em seguida, retorne ao formulário, e dê um duplo clique no botão novo e escreva o script abaixo:

```
Private Sub cmd_novo_Click()
    'Descobre qual a última linha preenchida
    celula = "A" & Worksheets("Contatos").Range("G1") - 1
    'COD será igual ao valor do último código cadastrado + 1
    COD = Worksheets("Contatos").Range(celula) + 1
    'Limpa os campos telefone e email
    txt_telefone.Text = ""
    txt_email.Text = ""
    cmd_excluir.Enabled = False
End Sub
```

CARREGAR LISTA DO CONTATO

Existem casos em que precisamos digitar uma sequência de comandos várias vezes para inúmeros botões. Como estamos trabalhando com programação, nada melhor do que fazer o VBA trabalhar a nosso favor.

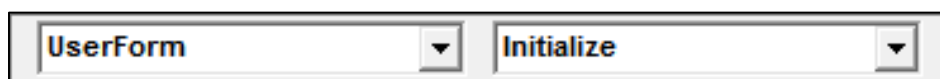
Vamos criar uma macro para atualizar a lista de contatos sempre que pedirmos. Dê um duplo clique no formulário, clique sobre a última linha e cria a macro *Atualizar*.

```
Sub Atualiza()
    'Cria uma variável para pegar a primeira e a última linha cadastrada
    célula = "Contatos!B4:B" & Worksheets("Contatos").Range("G1") - 1
    'Informa o grupo de células na planilha
    txt_contato.RowSource = célula
End Sub
```

CARREGAR NOME DO CONTATO

Ao abrir este formulário, ele deverá carregar todos os nomes cadastrados na planilha na caixa de listagem *txt_contato*.

Dê um duplo clique em alguma área em branco do formulário. Na caixa de configuração, coloque do seguinte modo:



Copie o script abaixo.

```
Private Sub UserForm_Initialize()
    'Atualiza a Lista
    Atualiza
    'Gera um novo código de cadastro
    cmd_novo_Click
End Sub
```

Perceba que o código é semelhante ao que fizemos no *formulário principal*. Com a única diferença que na última linha ele roda o script do botão novo.

REALIZANDO A BUSCA RÁPIDA

Logo que selecionar um nome desta lista, ele deverá buscar no banco de dados e retornar os dados da pesquisa e habilita o botão excluir.

```
Private Sub txt_contato_Change()  
    'Coloca a variável no campo pesquisa  
    Worksheets("Contatos").Range("G2") = txt_contato.Text  
    'Pega o resultado da pesquisa  
    resultado = Worksheets("Contatos").Range("G3")  
    'Se encontrar o resultado, então  
    If resultado > 0 Then  
        'A variável COD vai ficar com código do cadastro  
        COD = Worksheets("Contatos").Range("A" & resultado)  
        'Coloca os dados no formulário  
        txt_telefone = Worksheets("Contatos").Range("C" & resultado)  
        txt_email = Worksheets("Contatos").Range("D" & resultado)  
        'Habilita o botão excluir  
        cmd_excluir.Enabled = True  
    'Senão  
    Else  
        'Limpa os campos e gera o código do novo cadastro  
        cmd_novo_Click  
    End If  
End Sub
```

BOTÃO SALVAR

Se o cadastro existir, o sistema apenas altera as informações, caso contrário ele adiciona os dados na primeira linha vazia.

```
Private Sub cmd_salvar_Click()  
    'Coloca a variável no campo pesquisa  
    Worksheets("Contatos").Range("G2") = txt_contato.Text  
    'Pega o resultado da pesquisa  
    resultado = Worksheets("Contatos").Range("G3")  
    'Se o cadastro existir, então  
    If resultado > 0 Then  
        'Coloca os dados na linha existente  
        Worksheets("Contatos").Range("C" & resultado) = txt_telefone.Text  
        Worksheets("Contatos").Range("D" & resultado) = txt_email.Text  
        MsgBox "Cadastro alterado com sucesso!", vbInformation, "Concluído!"  
    'Senão  
    Else  
        'Coloca os dados em uma nova linha  
        nova_linha = Worksheets("Contatos").Range("G1")  
        Worksheets("Contatos").Range("A" & nova_linha) = COD  
        Worksheets("Contatos").Range("B" & nova_linha) = txt_contato.Text  
        Worksheets("Contatos").Range("C" & nova_linha) = txt_telefone.Text  
        Worksheets("Contatos").Range("D" & nova_linha) = txt_email.Text  
        MsgBox "Cadastro adicionado com sucesso!", vbInformation, "Concluído!"  
    End If  
    'Atualiza lista de dados  
    Atualiza  
End Sub
```

BOTÃO EXCLUIR

A linha só será excluída mediante a aprovação do usuário.

```
Private Sub cmd_excluir_Click()  
    'Se o usuário desejar excluir o cadastro, então  
    If MsgBox("Deseja excluir esse cadastro?", vbQuestion + vbYesNo, _  
        "Excluir Cadastro!") = vbYes Then  
        'Procura a linha do cadastro  
        'Coloca a variável no campo pesquisa  
        Worksheets("Contatos").Range("G2") = txt_contato.Text
```

```
'Pega o resultado da pesquisa  
linha = Worksheets("Contatos").Range("G3")  
'Excluir a linha  
Worksheets("Contatos").Range(linha & ":" & linha).Delete  
'Informa ao usuário  
MsgBox "Cadastro excluído com sucesso!", vbInformation, "Concluído"  
'limpa os dados do cadastro  
Atualiza  
End If  
End Sub
```

FORMULÁRIO DADOS DE ACESSO

CARREGAR DADOS DO USUÁRIO

Os dados do usuário deverão ser carregados assim que abrir o formulário. Para isto, utilizaremos novamente o *initialize*.

```
Private Sub UserForm_Initialize()  
    txt_usuario.Text = Worksheets("Acesso").Range("A2")  
    txt_senha.Text = Worksheets("Acesso").Range("B2")  
    txt_dica.Text = Worksheets("Acesso").Range("C2")  
End Sub
```

BOTÃO SALVAR

Tem a função de atualizar os dados da planilha de acordo com o formulário.

```
Private Sub cmd_salvar_Click()  
    Worksheets("Acesso").Range("A2") = txt_usuario.Text  
    Worksheets("Acesso").Range("B2") = txt_senha.Text  
    Worksheets("Acesso").Range("C2") = txt_dica.Text  
    MsgBox "Dados de acesso alterados!", vbInformation, "Concluído!"  
End Sub
```

FORMULÁRIO LOGIN

Este formulário será responsável pela liberação ou não da planilha mediante o nome do usuário e senha cedidos pelo usuário.

BOTÃO ENTRAR

O botão entrar terá um segredo. Ele liberará o acesso tanto para o usuário, quanto para o programador. O Objetivo deste é o seguinte: O **usuário** poderá acessar apenas os formulários, não as planilhas. Já com a senha de **administrador**, terá acesso total as planilhas deste documento.

```
Private Sub cmd_login_Click()  
    'Verifica se todos os campos foram preenchidos  
    If txt_usuario.Text = "" Or txt_senha.Text = "" Then  
        MsgBox "Preencha todos os campos.", vbCritical, "Atenção!"  
    Else  
        'Pega os dados do usuario  
        USER = Worksheets("Acesso").Range("A2")  
        PASS = Worksheets("Acesso").Range("A2")  
        'Se usuário for igual a user e senha igual a pass, então  
        If USER = txt_usuario.Text And PASS = txt_senha.Text Then  
            'Oculta o login  
            Me.Hide  
            'Abre o formulário principal  
            frm_principal.Show  
        End If  
    End If  
End Sub
```

```

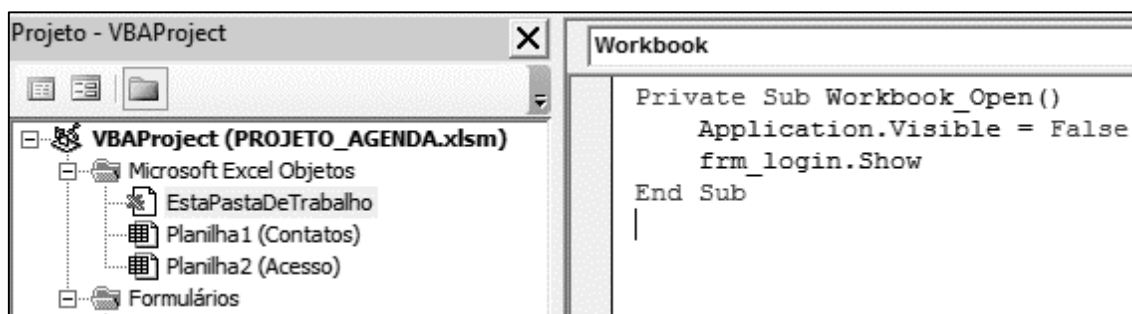
'Senão Se txt_usuario igual a ADMIN e txt_senha igual a ADMIN
ElseIf txt_usuario.Text = "ADMIN" And txt_senha.Text = "ADMIN" Then
    'Fecha o formulário de login
    Unload Me
    'Deixa o excel visível
    Application.Visible = True
Else
    'Envia mensagem
    MsgBox "Usuário ou senha inválidos", vbCritical, "Erro!"
End If
End If
End Sub

```

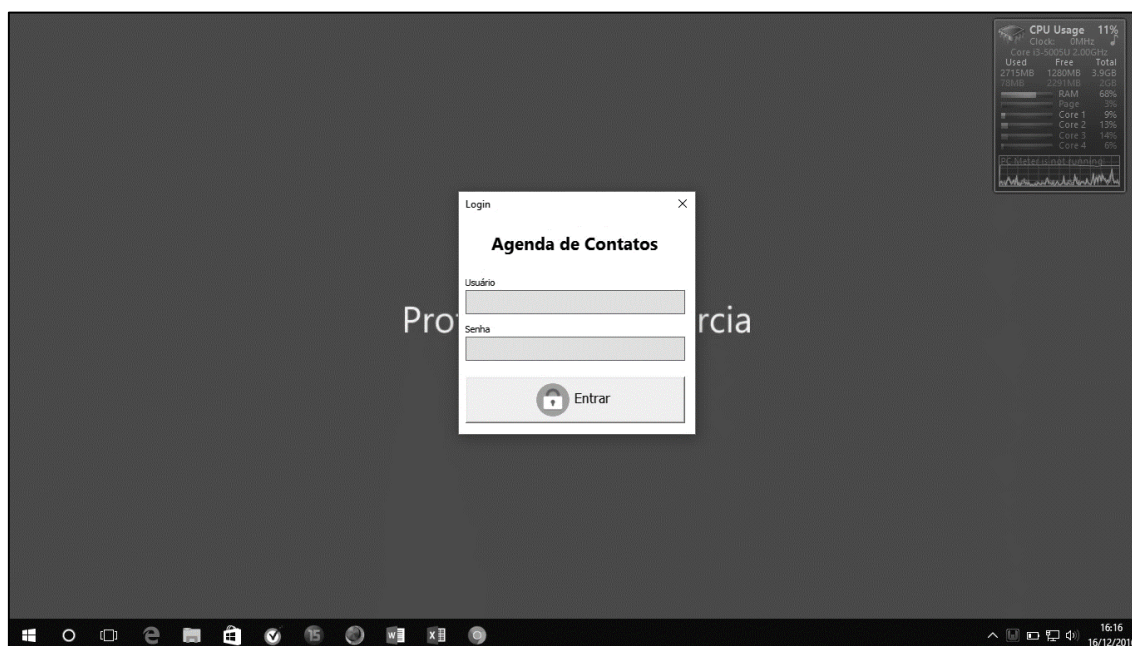
ABRIR O FORMULÁRIO E OCULTAR O EXCEL

Finalizando nosso projeto, vamos criar a macro que irá iniciar junto com o Excel, fazendo com que a tela de **login** abra e oculte as planilhas.

Na Janela de Projetos, localize o ícone “Esta pasta de trabalho” e dê um duplo clique. Abrindo a janela, selecione a opção **Workbook** e dentro dela, digite os comandos abaixo:



Feche e abra este documento novamente e você verá que o sistema entrará direto na tela de login.



Este sistema é bem simples e pode ser aprimorado conforme você aprofunda seu conhecimento na área de desenvolvimento no Excel com o VBA.