
Excel
70 FÓRMULAS INCRÍVEIS
AS FUNÇÕES MAIS PODEROSAS QUE VOCÊ PRECISA SABER
PREMIUM
Good Book
BEST PRICE
Awesome
Aprenda o PROCV()
Descubra o ÍNDICE + CORRESP
Manipule cadeias de Texto
Aprenda com dezenas de Exemplos
Luiz Felipe Araujo
51 DICAS E TRUQUES
51 MACROS VBA
70 FÓRMULAS INCRÍVEIS
Excel

Domine o Excel® (3 em 1): Excel - 70 Fórmulas Incríveis: As Funções mais Poderosas que Você Precisa Saber, Excel - 51 Macros incríveis: Rotinas Prontas Para Usar e Facilitar a Sua Vida e Excel - 51 Dicas e Truques Incríveis

Por Luiz Felipe Araujo

Publicado por:

Luiz Felipe Araujo

Copyright © 2018, Todos os direitos reservados. Nenhuma parte deste livro poderá ser arquivada, reproduzida ou transmitida por qualquer meio, físico ou digital.

1 2 3 7 6 4 4 2 2 1

E-mail para contato:

exceldevbr@gmail.com

Sumário



INTRODUÇÃO

DEFINIÇÃO

FUNÇÕES CONDICIONAIS

- SE ()
- E()
- OU()
- SEERRO()

FUNÇÕES DE MANIPULAÇÃO DE CADEIA DE TEXTO

- ESQUERDA()
- EXT.TEXTO()
- DIREITA()
- LOCALIZAR()
- NÚM.CARACT()
- SUBSTITUIR ()
- COMBINANDO AS FUNÇÕES
- TEXTO()
- ARRUMAR()
- LETRAS MAIÚSCULAS
- VERIFICAR CÉLULA

FUNÇÕES DE LOCALIZAÇÃO

- ÍNDICE()
- CORRESP()
- ÍNDICE(CORRESP())
- INDICE(CORRESP(); CORRESP())

PROCV()
PROCH()
PROCV(MAIOR())
DESLOC()
DESLOC(CORRESP())
INDIRETO()
CONT.VALORES()
INDICE(CONT.VALORES())
SEERRO() + FUNÇÕES DE PROCURA
FUNÇÕES DE SOMA E MÉDIA
SOMA()
SOMARPRODUTO ()
SUBTOTAL()
SOMASE()
SOMASES()
MEDIA()
MEDIASE()
OUTRAS FUNÇÕES MATEMÁTICAS
MÍNIMO ()
MÁXIMO ()
MENOR ()
MAIOR()
FATORIAL()
ARREDONADAMENTOS
ARRED()
ARREDONDAR.PARA.BAIXO()
ARREDONDAR.PARA.CIMA()
NÚMEROS ALEATÓRIOS
ALEATÓRIO()
ALEATÓRIOENTRE()

CONVERSÃO
- CONVERTER()
- MUDAR BASE NUMÉRICA

DATA E HORA
- DIASTRABALHOTOTAL()
- FUNÇÕES DE HORÁRIO
- FUNÇÕES DE DATA



INTRODUÇÃO

ABRIR SALVAR E FECHAR
1. Abrir um arquivo, caso já esteja aberto, maximiza-lo.
2. Salvar automaticamente antes de fechar
3. Copiar uma aba, converter para valor e salvar saída
4. Salvar backup rápido
5. Salvar uma cópia backup ao fechar o arquivo
6. Salvar cada planilha como arquivo Excel
7. Salvar cada aba como um arquivo PDF
8. Salvar o arquivo em diferentes pastas
9. Quando fechar o arquivo informar o tempo gasto

INTERAÇÃO COM GRÁFICOS
10. Automaticamente ajustar os rótulos dos gráficos
11. Redimensionar todos os gráficos

VÍNCULOS COM ARQUIVOS EXTERNOS
12. Atualizar todos os links
13. Quebrar todos os vínculos
14. Alterar vínculo

CONTROLE E SEGURANÇA DAS INFORMAÇÕES

15. Computar dados de usuários que acessaram a planilha
16. Marcar todas as células editadas pelo usuário
17. Proteger todas as planilhas
18. Desproteger todas as planilhas
19. Proteger uma aba com senha
20. Impedir o usuário salvar o arquivo
21. Simples Sistema de login e senha para acessar o arquivo

CONTROLE DE ERROS

22. Verificar todas as abas para encontrar erros
23. Verificar uma seleção para encontrar erros
24. Verificar todas as abas para contabilizar erros

OCULTAR E MOSTRAR INFORMAÇÕES

25. Exibir todas as linhas e colunas ocultas
26. Ocultar e mostrar todas as abas

OUTROS

27. Aplicar cores alternadas em uma seleção
28. Consolidar todos as abas na primeira
29. Cronometrar o tempo de outras macros
30. Copar e colar valor em todas as abas
31. Remover espaços em branco dentro das células
32. Atualizar todas as tabelas dinâmicas
33. Remover duplicatas em todas as abas
34. Consolidar dados na primeira aba
35. Deletar abas que estão vazias
36. Ordenar as abas em ordem alfabética
37. Trocar o nome de todas as abas
38. Destacar linha e coluna da célula selecionada

INTERAÇÃO COM WINDOWS

39. Salvar uma seleção como imagem
40. Converter todos os arquivos de uma pasta para PDF

41. Listar todos os arquivos de uma pasta
42. Copiar os arquivos de uma pasta para outra

INTERAÇÕES COM OUTOLOOK

43. Enviar um e-mail simples com VBA
44. Enviar arquivo como anexo por E-mail
45. Enviar planilha ativa como anexo por E-mail
46. Enviar e-mail com uma seleção como anexo
47. Enviar e-mail com arquivo externo como anexo

Interações com PowerPoint

48. Exportar gráficos para Microsoft PowerPoint
49. Exportar seleção para Microsoft PowerPoint

INTERAÇÕES COM WORD

50. Exportar seleção para o Microsoft Word
51. Exportar dados para o Microsoft Word



INTRODUÇÃO

DICAS DE FÓRMULAS

1. Utilize o Índice(corresp()) ao invés do ProcV()
2. Trave as suas fórmulas
3. Fórmula ordem com desempate
4. Auxiliar de fórmula
5. Somar maiores ou menores valores
6. Formula para aplicar letra maiúscula
7. Formula para diferentes formatações de data

DICAS DE ATALHOS

8. Atalhos de Menu
9. Colar especial

10. Aplicar filtro rapidamente
11. Navegar no filtro rapidamente
12. Adicionar e remover linhas/colunas rapidamente
13. Atalhos de Seleção
14. Selecionar uma região de células
15. Selecionar linhas e colunas
16. Atalhos para formatos de números
17. Criar gráficos de forma instantânea
18. Inserir Data e Hora

TRUQUES VARIADOS

19. Procurar termo com caractere desconhecido
20. Cálculo automático e manual
21. Verificar Erros
22. Rastrear precedentes e dependentes
23. Valores iniciados por zero
24. Redimensionar simultaneamente diversas colunas ou linhas
25. Copiar seleção ou gráfico como vínculo no PowerPoint ou Word
26. Quebre os vínculos antes de enviar um relatório
27. Recuperar arquivo após travamento
28. Ferramenta do Excel para capturar tela
29. Padronize e organize os gráficos e demais objetos
30. Ocultar formulas
31. Extensões de arquivo Excel

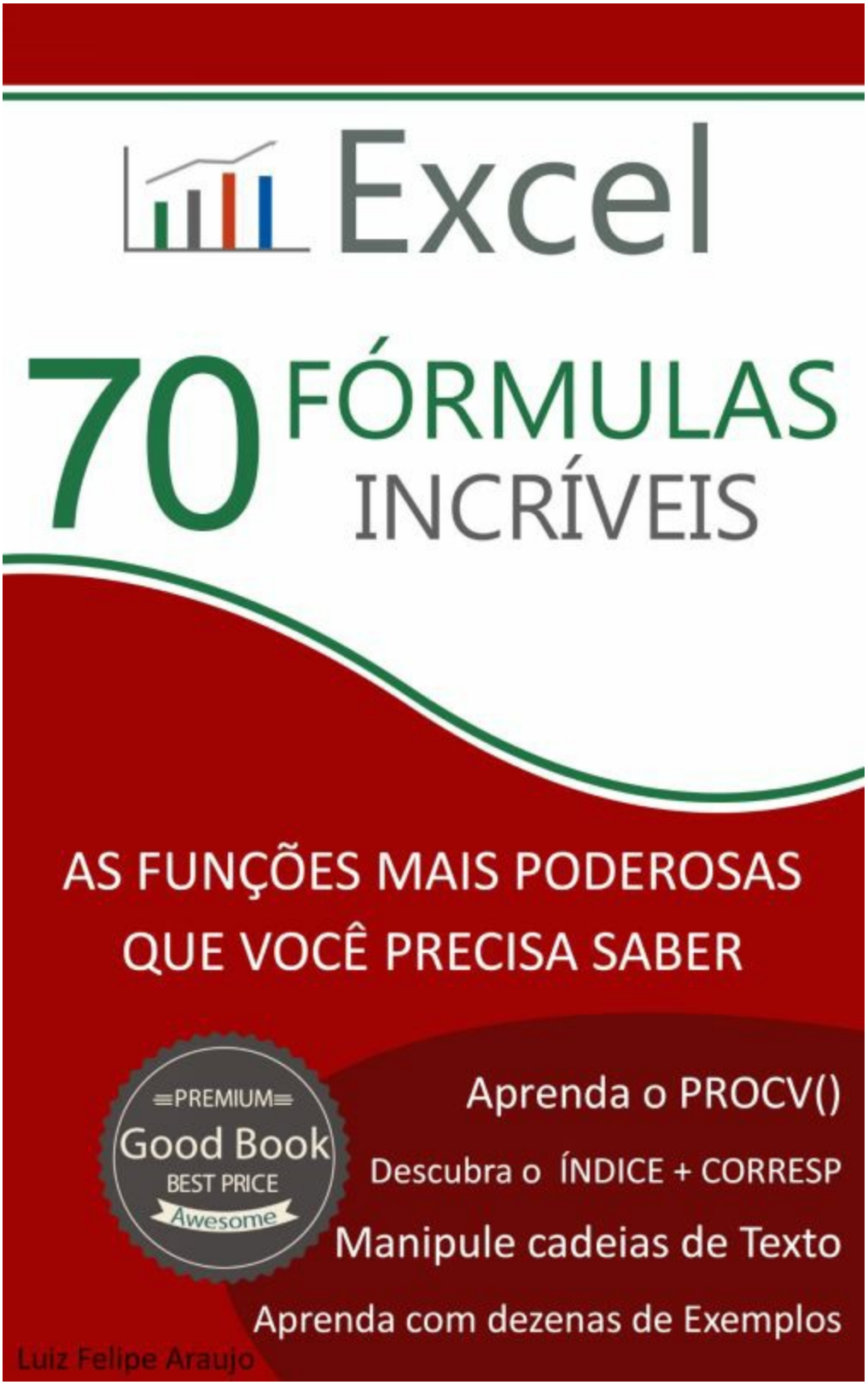
DICAS DE MACROS E VBA

32. Traduzir fórmulas de português para inglês usando VBA
33. Utilização do “With Statement” para simplificar o código
34. Desativar atualização de tela (Melhorar desempenho)
35. Calculo manual (Melhorar desempenho)
36. Copiar e colar corretamente (Melhorar desempenho)
37. Desabilitar eventos (Melhorar desempenho)

38. Impedir notificações
39. Chamar uma macro quando uma célula for alterada
40. Selecionar informações do início ao fim em uma direção
41. Selecionar informações do início ao fim em duas direções
42. Abrir planilhas na web com PHP
43. Controlar mouse e teclado com SendKeys
44. Esperar X segundos
45. Melhorar a visualização do código

DICAS RÁPIDAS

46. Para pular de linha dentro de uma célula
47. Ao invés de “concatenar ()” utilize “&”
48. Auto completar formula
49. Somar linhas com atalho
50. Gerar valores aleatórios
51. Utilizar uma célula pra cálculo rápido


Excel
70 FÓRMULAS INCRÍVEIS
AS FUNÇÕES MAIS PODEROSAS QUE VOCÊ PRECISA SABER
PREMIUM
Good Book
BEST PRICE
Awesome
Aprenda o PROCV()
Descubra o ÍNDICE + CORRESP
Manipule cadeias de Texto
Aprenda com dezenas de Exemplos
Luiz Felipe Araujo

O Excel não é apenas um software de planilhas eletrônicas, é uma plataforma que possui as mais variadas utilidades, as quais vão desde elaboração de relatórios simples, criação de painéis e análise de dados, até funções mais complexas com aplicações em banco de dados, soluções de engenharia, criação de soluções coorporativas, sistemas e as mais variadas funções que variam de acordo com o objetivo do usuário.

Para dominar esta ferramenta, é preciso passar por uma série de etapas, as quais vão desde o aprendizado básico até as funcionalidades mais complexas, para tanto, este livro tem o objetivo de facilitar o aprendizado do leitor em relação as funções. Uma ferramenta do fundamental para ter um maior entendimento das possibilidades que o Excel oferece.

INTRODUÇÃO

A estrutura deste livro, consiste na apresentação das funções seguida de uma série de exemplos, os quais serão minuciosamente explorados para apresentar de forma clara e objetiva o funcionamento das fórmulas. Para diversos casos, serão apresentadas fórmulas importantes que dependem de uma combinação de outras mais simples, para tanto, estas serão apresentadas inicialmente, para posteriormente demonstrar suas combinações e implicações em resultados poderosos e fascinantes.

Ao longo do livro serão demonstradas inúmeras funções que se encaixam dentro de uma categoria e depois as mesmas poderão reaparecer em combinações com outras, criando assim formulas mais elaboradas as quais podem atingir os mais diversos objetivos. Como por exemplo a categoria manipulação de texto, a qual possui diversas funções simples e depois suas combinações são exploradas para demonstrar seu real potencial.

Este livro não precisa ser necessariamente lido de maneira linear, pode-se utilizado de acordo com a necessidade específica de cada usuário, porém para funções que se encaixam dentro de uma mesma categoria, é recomendado ler todas que pertencem a um mesmo grupo, para facilitar o entendimento de suas combinações.

DEFINIÇÃO

As funções no Excel, também conhecidas como fórmulas, são basicamente instruções matemática que o Excel interpreta de maneira prática para cumprir os mais diversos objetivos. Através delas, pode-se realizar desde os cálculos mais simples até os mais complexos, pode-se também realizar tratamento de dados, manipular cadeias de textos e até mesmo realizar operações lógicas avançadas.

A versatilidade das funções do Excel possibilita realizar diversas tarefas tanto para objetivo pessoal como profissional, abaixo são listado alguns exemplos de possibilidades através da utilização deste incrível recursos:

- Controle de finanças pessoais
- Cálculos de finanças empresarias
- Tratamento e processamento de dados
- Elaborar relatórios de forma rápida e prática
- Criar cálculos sofisticados de engenharia física e matemática

FUNÇÕES CONDICIONAIS

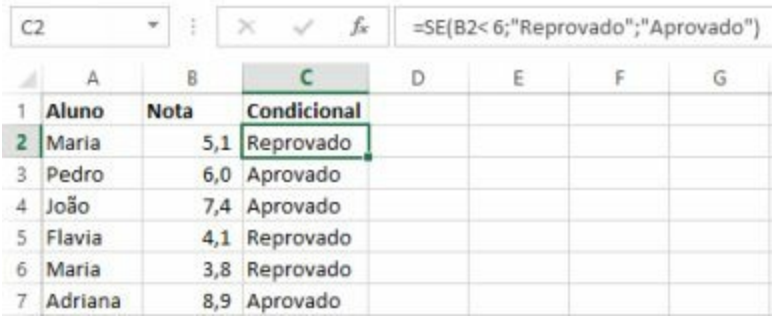
As funções condicionais são primordiais para o controle de informações no Excel, através delas pode-se realizar uma série de análises, estabelecer parâmetros para realizar operações e criar condições para aplicação de outras funções. São funções simples, diretamente ligadas ao significado de seus nomes.

Primeiramente, é preciso apresentar de forma breve os operadores lógicos que serão constantemente utilizados para as funções condicionais:

>	Maior
<	Menor
>=	Maior Igual
<=	Menor Igual
=	Igual
<>	Diferente

SE ()

Esta função é fundamental para o domínio do Excel, basicamente, fornecendo uma condição, pode-se estabelecer um retorno para o caso verdadeiro, e um retorno para o caso falso.



	A	B	C	D	E	F	G
1	**Aluno**	**Nota**	**Condicional**				
2	Maria	5,1	Reprovado				
3	Pedro	6,0	Aprovado				
4	João	7,4	Aprovado				
5	Flavia	4,1	Reprovado				
6	Maria	3,8	Reprovado				
7	Adriana	8,9	Aprovado				

=SE(B2< **6**;"**Reprovado**";"**Aprovado**")

B2**< 6 -** Condicional que será aplicada

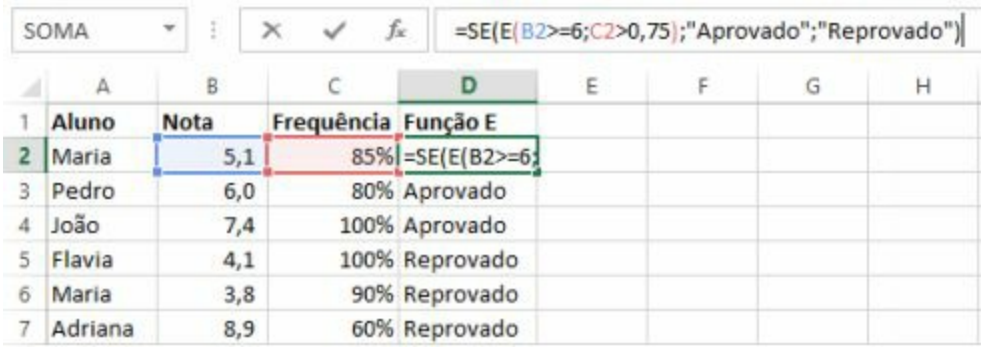
Reprovado – Retorno para condição atendida

Aprovado – Retorno para condição não atendida

Uma vez que a condição seja atendida, ou seja, a nota seja inferior ao valor determinado (B2**< 6**), existirá um retorno para célula, o qual será o primeiro argumento determinado de valor “Reprovado”, caso contrário, ou seja, condição não atendida, o valor retornado será “Aprovado”.

E()

O principal objetivo desta função, é unir dois argumentos ou mais dentro de uma condicional, ela pode ser utilizada isoladamente para retornar verdadeiro ou falso, porém sua utilidade se comprova de fato dentro de outras funções, conforme exemplo a seguir:



	A	B	C	D	E	F	G	H
1	**Aluno**	**Nota**	**Frequência**	**Função E**				
2	Maria	5,1	85%	=SE(E(B2>=6;				
3	Pedro	6,0	80%	Aprovado				
4	João	7,4	100%	Aprovado				
5	Flavia	4,1	100%	Reprovado				
6	Maria	3,8	90%	Reprovado				
7	Adriana	8,9	60%	Reprovado				

=SE(E(B2>=**6**;C2>**0,75**);"**Aprovado**";"**Reprovado**")

B2>=**6** – Primeira condicional

C2>**0,75** – Segunda condicional

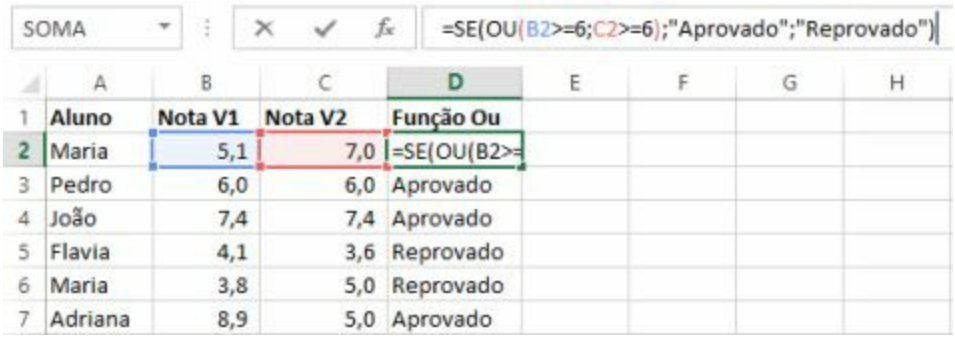
Aprovado - Retorno para as duas condicionais atendidas

Reprovado - Retorno para as duas condicionais não atendidas

Neste exemplo, diferente do anterior, a condicional foi invertida, desta vez foi apresentada a condição para nota ser maior ou igual a 6 (B2>=**6**), adicionando desta vez uma segunda condição, C2>**0,75**, ou seja, a frequência precisa também ser acima de 75%. Apenas com as duas condicionais atendidas, o retorno será o valor “Aprovado”, caso contrário, o retorno será “Reprovado”.

OU()

Novamente, da mesma maneira que a função E(), a função OU() pode ser utilizada isoladamente, para retornar o valor verdadeiro ou falso, porém sua utilidade se mostra mais presente com a sua utilização dentro da função SE(), conforme exemplo a seguir:



	A	B	C	D	E	F	G	H
1	**Aluno**	**Nota V1**	**Nota V2**	**Função Ou**				
2	Maria	5,1	7,0	=SE(OU(B2>=				
3	Pedro	6,0	6,0	Aprovado				
4	João	7,4	7,4	Aprovado				
5	Flavia	4,1	3,6	Reprovado				
6	Maria	3,8	5,0	Reprovado				
7	Adriana	8,9	5,0	Aprovado				

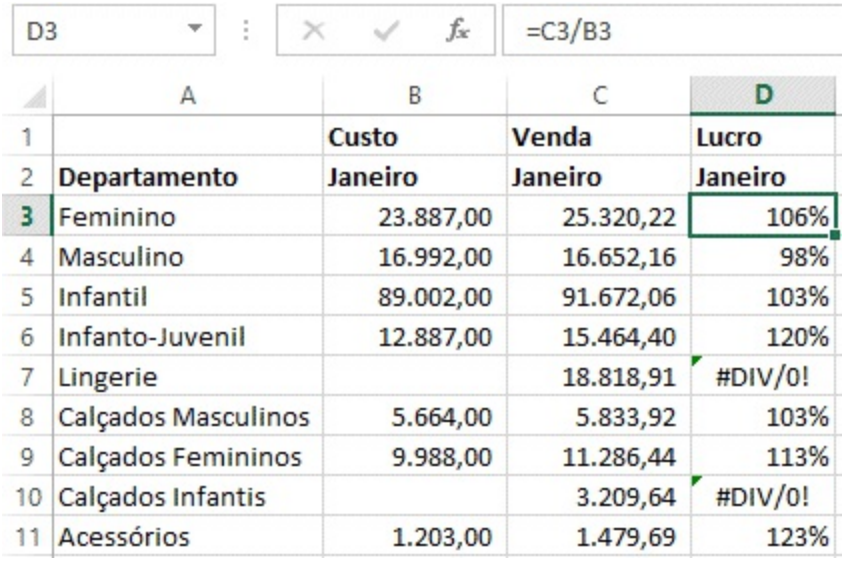
=SE(OU(**B2**>=**6**;**C2**>**6**);"**Aprovado**";"**Reprovado**")

Este exemplo simula uma situação onde um aluno necessita da nota maior que 6 em pelo menos uma das avaliações, desta maneira, utiliza-se a função OU(), com as duas condicionais relacionadas a primeira coluna **B2**>=**6**,e depois aplica-se o mesmo critério para segunda coluna **C2**>**6**, caso uma das duas seja atendida, o valor retornado será "**Aprovado**", caso contrário "**Reprovado**".

SEERRO()

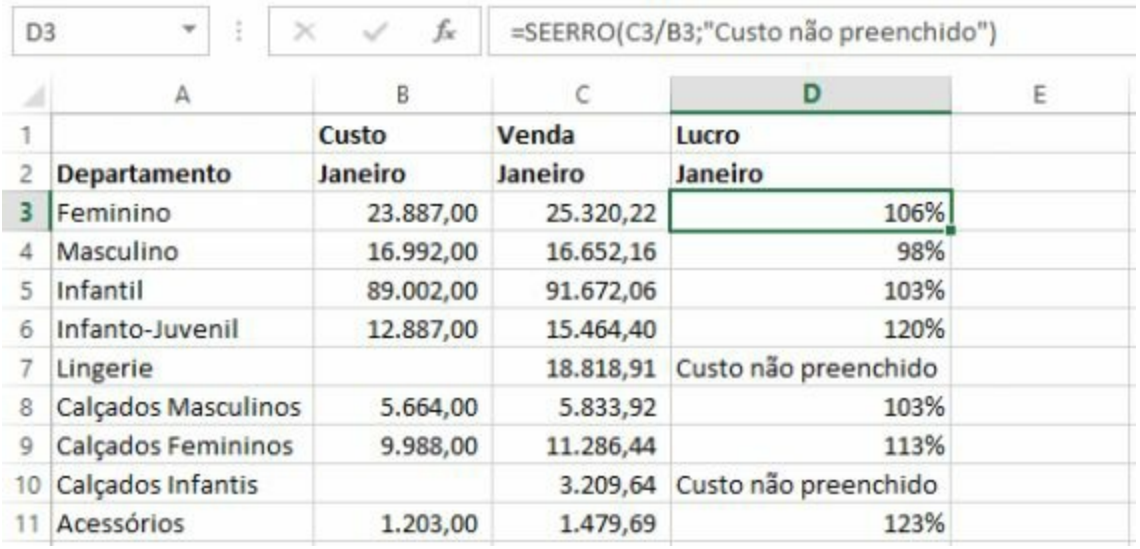
Esta é uma função importantíssima, porém que deve ser utilizada com cautela, uma vez que ela pode ocultar erros que se deseja enxergar. Sua principal aplicação é em fórmulas onde os erros são esperados para determinados casos, porém não deseja-se apresentar os valores com #N/D, #VALOR e etc. Ou para casos, onde caso um erro seja encontrado, realize-se um outro procedimento.

Basicamente o funcionamento, se dá por uma condicional, que caso seja atendida, fará algum outro procedimento, da seguinte maneira:



	A	B	C	D
1		**Custo**	**Venda**	**Lucro**
2	**Departamento**	**Janeiro**	**Janeiro**	**Janeiro**
3	Feminino	23.887,00	25.320,22	106%
4	Masculino	16.992,00	16.652,16	98%
5	Infantil	89.002,00	91.672,06	103%
6	Infanto-Juvenil	12.887,00	15.464,40	120%
7	Lingerie		18.818,91	#DIV/0!
8	Calçados Masculinos	5.664,00	5.833,92	103%
9	Calçados Femininos	9.988,00	11.286,44	113%
10	Calçados Infantis		3.209,64	#DIV/0!
11	Acessórios	1.203,00	1.479,69	123%

Observa-se neste exemplo que a coluna D relacionada aos lucros, realiza uma divisão entre os valores de custo pelos valores de venda. Porém caso o valor de custo não seja preenchido, a fórmula aplicada na coluna D, apresentará um erro, pois não se pode dividir 0 por nenhum número. Para contornar este erro, pode-se utilizar a função SEERRO()



	A	B	C	D	E
1		**Custo**	**Venda**	**Lucro**	
2	**Departamento**	**Janeiro**	**Janeiro**	**Janeiro**	
3	Feminino	23.887,00	25.320,22	106%	
4	Masculino	16.992,00	16.652,16	98%	
5	Infantil	89.002,00	91.672,06	103%	
6	Infanto-Juvenil	12.887,00	15.464,40	120%	
7	Lingerie		18.818,91	Custo não preenchido	
8	Calçados Masculinos	5.664,00	5.833,92	103%	
9	Calçados Femininos	9.988,00	11.286,44	113%	
10	Calçados Infantis		3.209,64	Custo não preenchido	
11	Acessórios	1.203,00	1.479,69	123%	

Desta forma, ao invés de apresentar a mensagem de erro, o texto: “Custo não preenchido” será apresentado.

FUNÇÕES DE MANIPULAÇÃO DE CADEIA DE TEXTO

Existem diversas funções para manipular textos, onde o objetivo é basicamente obter frações do valor de uma célula. São de grande utilidade para usuários que lidam com grandes quantidades de informação, onde muitas vezes se obtém os dados sem formatação, extrações de banco de dados e de demais origens diversas.

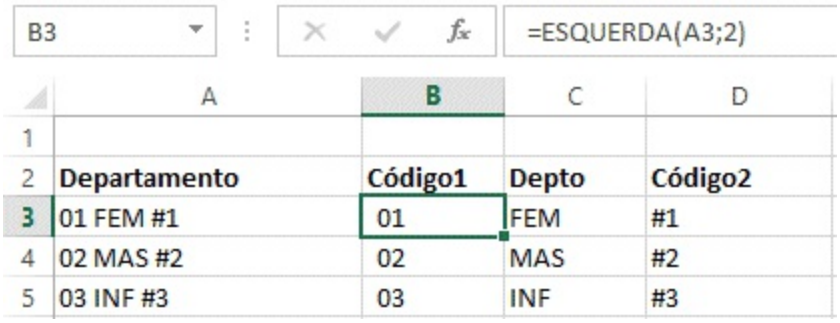
As funções ESQUERDA(), DIREITA() e EXT.TEXTO(), por si só já são de extrema utilidade, quando combinadas com LOCALIZAR() e/ou NÚM.CARACT(), podem trazer uma função adicional de grande utilidade para manipular as cadeias de textos.

Primeiramente, vamos apresentar de maneira breve a utilização das funções ESQUERDA(), DIREITA(), EXT.TEXTO(), LOCALIZAR() e NÚM.CARACT(), para então apresentar as suas combinações.

ESQUERDA()

Esta função é utilizada para se obter o prefixo de um texto.

Exemplo 1:



	A	B	C	D
1				
2	**Departamento**	**Código1**	**Depto**	**Código2**
3	01 FEM #1	01	FEM	#1
4	02 MAS #2	02	MAS	#2
5	03 INF #3	03	INF	#3

=ESQUERDA(A3;2)

A3 – Célula que será aplicada a função

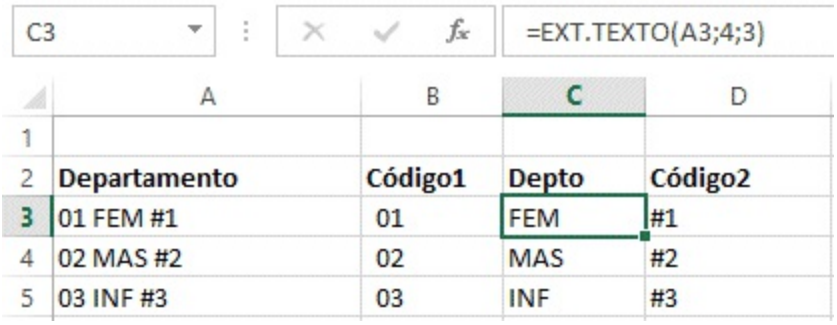
2 – Significa que será retornado os 2 primeiros caracteres da esquerda

Neste exemplo, o departamento está acompanhado de seu código inicial e final, porém deseja-se separa-los. Para se obter o código inicial (código1), se utiliza a função esquerda, com 2 caracteres.

EXT.TEXTO()

Esta função é utilizada para se extrair a parte central de uma célula.

Exemplo 1:



	A	B	C	D
1				
2	**Departamento**	**Código1**	**Depto**	**Código2**
3	01 FEM #1	01	FEM	#1
4	02 MAS #2	02	MAS	#2
5	03 INF #3	03	INF	#3

=EXT.TEXTO(**A3**; 3;3)

A3 – Célula que será aplicada a função

4 – Significa que será retornado a partir no terceiro caractere

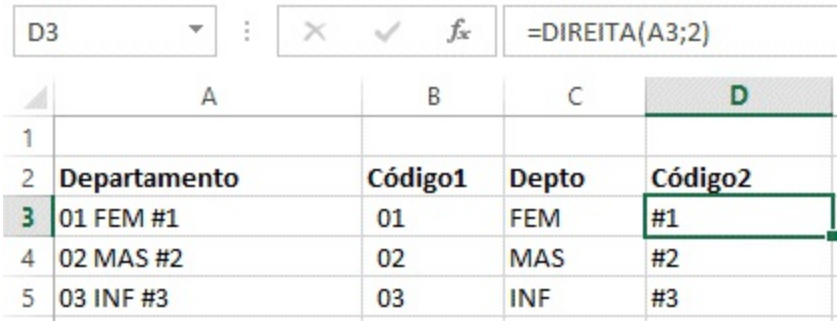
3 – Significa que será retornado 3 caracteres.

Agora deseja-se obter a abreviação FEM, desta forma, se utiliza o EXT.TEXTO(). Nesta função, é preciso informar a posição do caractere inicial e a posição final. Neste exemplo foi utilizado **4**, para inicial, visto que se deseja extrair a partir do quarto caractere (o espaço conta como 1 caractere), por último é informado o número 3, pois deseja-se retornar 3 dígitos.

DIREITA()

Esta função é utilizada para se obter o sufixo de uma célula.

Exemplo:



	A	B	C	D
1				
2	**Departamento**	**Código1**	**Depto**	**Código2**
3	01 FEM #1	01	FEM	#1
4	02 MAS #2	02	MAS	#2
5	03 INF #3	03	INF	#3

=DIREITA(**A3**;2)

A3 – Célula que será aplicada a função

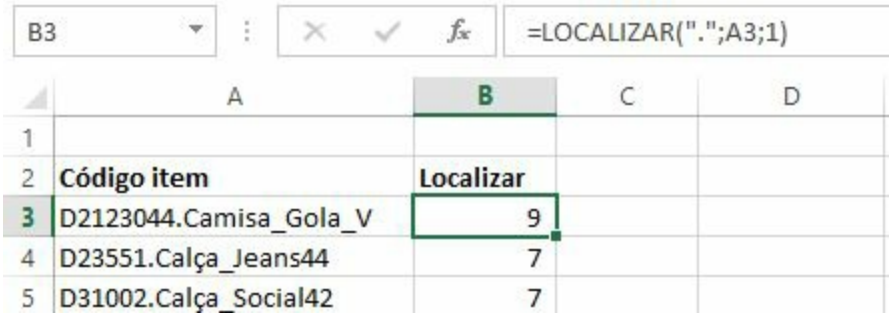
2 – Significa que será retornado os 2 primeiros caracteres da direita

Esta função tem o mesmo comportamento da ESQUERDA(), porém ao invés de contar a partir do primeiro caractere, ela conta a partir do último caractere, retornando os caracteres a partir da direita.

LOCALIZAR()

Esta função é extremamente simples, o objetivo dela é apenas retornar a posição de um caractere desejado.

Exemplo:



=LOCALIZAR(“.”;**A3**;1)

“.” – Texto que se deseja localizar, neste exemplo deseja-se localizar o ponto.

A3 – Célula que será aplicada a função

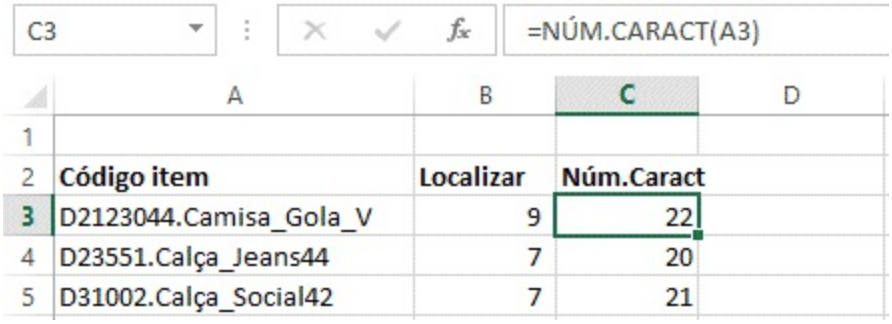
1 – A partir de qual caractere será localizado.

Neste exemplo deseja-se buscar o caractere ponto, desta maneira coloca-se entre aspas (“.”), como se deseja localizar a partir do início, a função é finalizada com o número **1**. O valor encontrado será 9, pois o ponto se encontra na nona posição.

NÚM.CARACT()

Assim como a anterior, esta função também é extremamente simples, o seu objetivo é simplesmente contar o número total de caracteres de uma célula.

Exemplo:



	A	B	C	D
1				
2	**Código item**	**Localizar**	**Núm.Caract**	
3	D2123044.Camisa_Gola_V	9	22	
4	D23551.Calça_Jeans44	7	20	
5	D31002.Calça_Social42	7	21	

=NÚM.CARACT(**A3**)

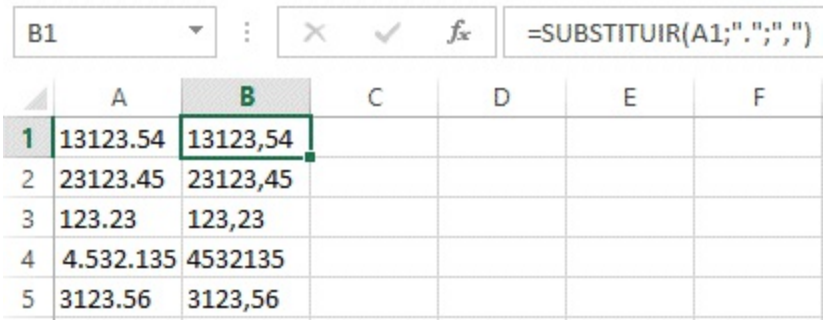
A3 – Célula que será aplicada a função

Neste exemplo, a função é aplicada na célula **A3** e o valor encontrado é 22, visto que o item tem 22 caracteres.

SUBSTITUIR ()

Este função, apesar de não ter utilidade em combinação com as demais apresentadas, possui uma grande utilidade para tratamento de dados, uma vez que ela pode substituir um termo procurado, por outro informado.

Exemplo 1:



	A	B	C	D	E	F
1	13123.54	13123,54				
2	23123.45	23123,45				
3	123.23	123,23				
4	4.532.135	4532135				
5	3123.56	3123,56				

=SUBSTITUIR(A1;"**.**";"**,**")

A1 – Célula onde a função será aplicada

"**.**" – Texto que será substituído

"**,**" **–** Texto novo que será implementado

Este exemplo demonstra a aplicação da função substituir para trocar o ponto de uma célula pela virgula, desta forma, sempre que uma virgula for encontrada, ela será substituída por ponto.

Obs: O argumento final de ocorrência não foi declarado por não ter necessidade neste exemplo.

Exemplo 2:



	A	B	C	D	E
1	2018 - Trimestre 2	2018 - Trimestre 3			
2					

=SUBSTITUIR(A1;"**2**";"**3**";**2**)

A1 – Célula onde a função será aplicada

"**2**" – Texto que será substituído

"3" – Texto novo que será implementado

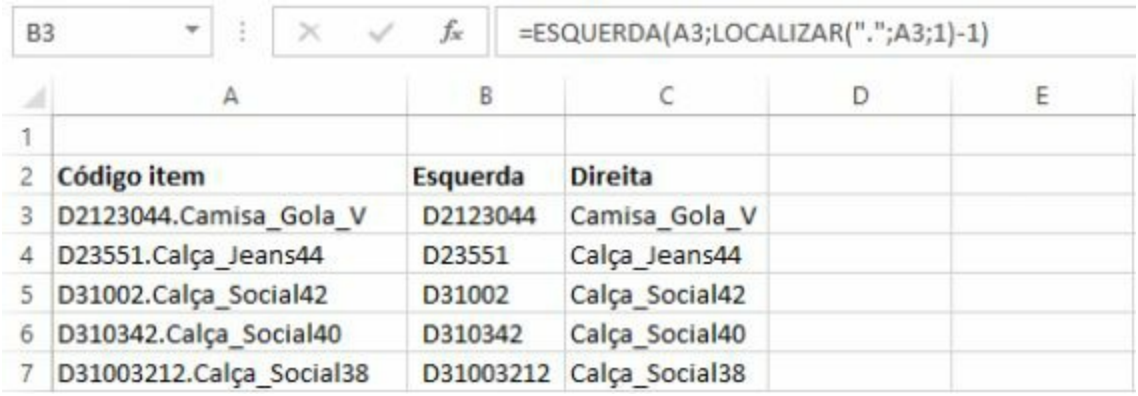
2 – Ocorrência onde a troca será realizada

Neste exemplo, deseja-se substituir o número **2** por **3**, porém é preciso informar em qual ocorrência esta troca irá ocorrer, para não alterar erroneamente o ano. Desta maneira é informado o argumento final de número **2**, o que significa que a troca só ocorrerá na segunda ocorrência.

COMBINANDO AS FUNÇÕES

As combinações vão variar de acordo com a necessidade do usuário para realizar o tratamento dos dados ou textos, estes exemplos, demonstram as possibilidades de combinações, porém podem ser criadas diversas outras de acordo com os cenários encontrados.

Exemplo:



	A	B	C	D	E
1					
2	**Código item**	**Esquerda**	**Direita**		
3	D2123044.Camisa_Gola_V	D2123044	Camisa_Gola_V		
4	D23551.Calça_Jeans44	D23551	Calça_Jeans44		
5	D31002.Calça_Social42	D31002	Calça_Social42		
6	D310342.Calça_Social40	D310342	Calça_Social40		
7	D31003212.Calça_Social38	D31003212	Calça_Social38		

=ESQUERDA(A3;LOCALIZAR(".";A3;1)-1)

A3 – Célula que será aplicada a função esquerda

"." – Texto que se deseja localizar, neste exemplo deseja-se localizar o ponto.

A3 – Célula que será aplicada a função localizar

1 – A partir de qual caractere será localizado.

- 1 – Numero subtraído do localizar, para se obter a posição do ponto -1 unidade.

O leitor pode-se questionar porque não utilizar "=ESQUERDA(A3;8)", afinal se obteria o mesmo resultado de forma mais simples. Porém quando deseja-se aplicar esta função para diversos elementos que possuem caracteres variados como neste exemplo, é preciso variar o número dentro da função esquerda, para tanto se utilizar a função: "LOCALIZAR(".";A3;1)-1" neste exemplo no lugar do número 8, para desta maneira, sempre se localizar a quantidade de caracteres até o ponto ("."), subtraindo uma unidade, para se aplicar esquerda até antes do mesmo. Desta forma pode-se arrastar a função para quantos elementos forem necessários.

Exemplo 2:



	A	B	C	D	E	F
1						
2	**Código item**	**Esquerda**	**Direita**			
3	D2123044.Camisa_Gola_V	D2123044	Camisa_Gola_V			
4	D23551.Calça_Jeans44	D23551	Calça_Jeans44			
5	D31002.Calça_Social42	D31002	Calça_Social42			
6	D310342.Calça_Social40	D310342	Calça_Social40			
7	D31003212.Calça_Social38	D31003212	Calça_Social38			

=DIREITA(**A3**;NÚM.CARACT(**A3**)-LOCALIZAR(".";**A3**;1))

A3 – Célula que será aplicada a função direita

A3 – Célula que será aplicada a função núm.caract

"." – Texto que se deseja localizar, neste exemplo deseja-se localizar o ponto.

A3 – Célula que será aplicada a função localizar

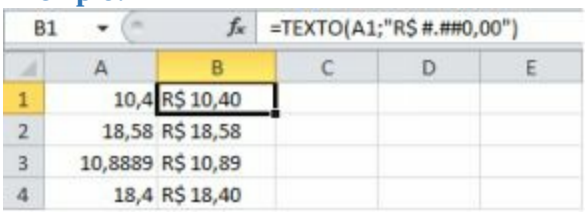
1 – A partir de qual caractere será localizado.

Com a formula direita, utiliza-se também o elemento NÚM.CARACT(), uma vez que o LOCALIZAR() conta a partir da esquerda, é preciso utilizar a subtração do total de caracteres com a função localizar, para se obter o número de caracteres desejados para aplicar a função DIREITA().

TEXTO()

A função texto é extremamente importante para lidar com dados, os quais muitas vezes estão em formatos indesejados, seja uma data que está como número, ou para padronizar formatações e etc.

Exemplo:



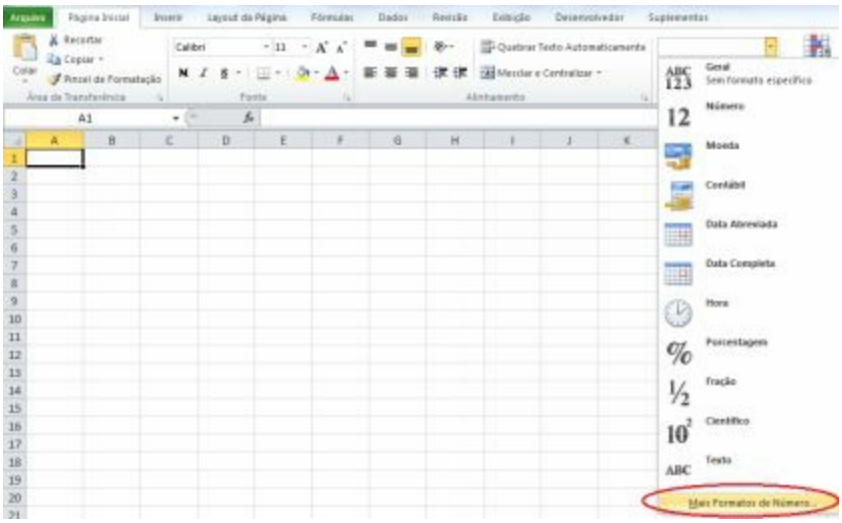
=TEXTO(A1;"**R$ #.##0,00**")

A1 – Célula que receberá a formatação

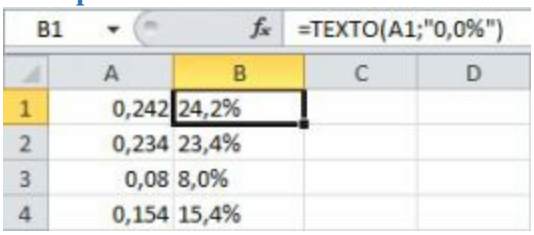
"**R$ #.##0,00**" – Formatação aplicada

A função é extremamente simples, exigindo apenas duas entradas de dado, uma é a célula que será formatada (Neste exemplo A1) e a outra é o código da formatação (Neste exemplo o código: **"R$ #.##0,00"**, formato de moeda) . Desta forma, é possível garantir que uma coluna estará sempre com a formatação desejada através da função.

Obs 1: Diversos exemplos de código de formatação podem ser consultados no menu através da opção “Mais Formatos de Números”, Número, Personalizado.



Exemplo:



=TEXTO(A1;"**0,0%**")

A1 – Célula que receberá a formatação

"**R$ #.##0,00**" – Formatação aplicada

O exemplo é extremamente semelhante ao anterior, porém ao invés de aplicar a formatação de moeda, é aplicada a formatação de percentual (**R$ #.##0,00**") na célula A1.

Dica:

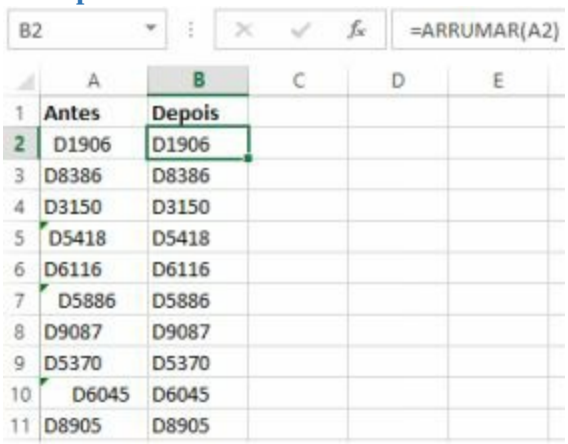
É possível transformar datas em textos de diversas maneiras diferentes, para tanto basta utilizar a fórmula texto e utilizar os códigos de formatação, segue abaixo alguns exemplos de aplicações da função texto para datas.

Data	Texto	Fórmula Utilizada
05/01/2018	05/01/18	=TEXTO(B3;"dd/mm/aa")
05/01/2018	05 jan 2018	=TEXTO(A3;"dd mmm aaa")
05/01/2018	sex 05 jan	=TEXTO(A4;"ddd dd mmm")
05/01/2018	janeiro 2018	=TEXTO(A5;"mmmm aaaa")
05/01/2018	05 janeiro 2018	=TEXTO(A6;"dd mmmm aaaa")

ARRUMAR()

Esta função é essencial para usuários que lidam com muitos dados. É sempre comum lidar com informações que estão desajustadas e possuem espaços no início ou no final da célula que não deveriam ter. Para solucionar este problema, utiliza-se a função ARRUMAR()

Exemplo:



	A	B	C	D	E
1	**Antes**	**Depois**			
2	D1906	D1906			
3	D8386	D8386			
4	D3150	D3150			
5	D5418	D5418			
6	D6116	D6116			
7	D5886	D5886			
8	D9087	D9087			
9	D5370	D5370			
10	D6045	D6045			
11	D8905	D8905			

=ARRUMAR(**A2**)

A2 – Célula onde a função será aplicada

A fórmula é extremamente simples, seu único argumento é a célula desejada para que os espaços de prefixo e sufixo sejam retirados. Desta forma observa-se que após a aplicação da função na coluna A, ela fica sem os problemas de desajustes na coluna B.

LETRAS MAIÚSCULAS

Existem três funções básicas para controlar se as letras serão maiúsculas, minúsculas ou se terão apenas o seu primeiro caractere maiúsculo. Qualquer necessidade além destes três casos, deverás ser solucionada com as combinação das funções de manipulação de texto.

Para os três casos deve se utilizar as seguintes funções:

	A	B	C
1	**Exemplo**	**Fórmula**	**Retorno**
2	planilha	=MAIÚSCULA(A2)	PLANILHA
3	planilha	=PRI.MAIÚSCULA(A3)	Planilha
4	PLANILHA	=MINÚSCULA(A4)	planilha

As funções são comumente utilizadas em extrações de dados, onde os textos extraídos de uma base de dados normalmente estão sem nenhuma formatação.

VERIFICAR CÉLULA

Existem funções que funcionam para verificar qual tipo de conteúdo de uma célula, seus retornos são basicamente verdadeiro ou falso. O único argumento deste conjunto de fórmulas é a célula que se deseja verificar.

	A	B	C
1	**Exemplo**	**Fórmula**	**Retorno**
2	Texto	=ÉTEXTO(A2)	VERDADEIRO
3	10	=ÉNÚM(A3)	VERDADEIRO
4	2	=ÉFÓRMULA(A4)	VERDADEIRO
5		=ÉCÉL.VAZIA(A5)	VERDADEIRO
6	#DIV/0!	=ÉERROS(A6)	VERDADEIRO
7	3	=ÉIMPAR(A7)	VERDADEIRO
8	4	=ÉPAR(A8)	VERDADEIRO
9	FALSO	=ÉLÓGICO(A9)	VERDADEIRO

Essas são as principais funções que verificam o conteúdo de uma célula, no exemplo acima todas foram utilizadas com o propósito de retornar seus valores como verdadeiro.

FUNÇÕES DE LOCALIZAÇÃO

As funções de localização, tem por objetivo retornar, valores, textos ou termos baseados em um valor de busca. Existem diversas funções que executam esse procedimento, as mais conhecidas são PROCV() e a combinação ÍNDICE(CORRESP()), esta última sendo considerada a mais importante, porém para apresentar esta, primeiramente serão apresentados o ÍNDICE() e o CORRESP() individualmente.

ÍNDICE()

Esta função é extremamente simples, através de coordenadas, ela retorna o valor da célula desejada. Este processo funciona em uma direção, ou seja, informando a região de uma linha ou coluna e a posição desejada ou informando uma matriz e duas coordenadas.

Exemplo 1:



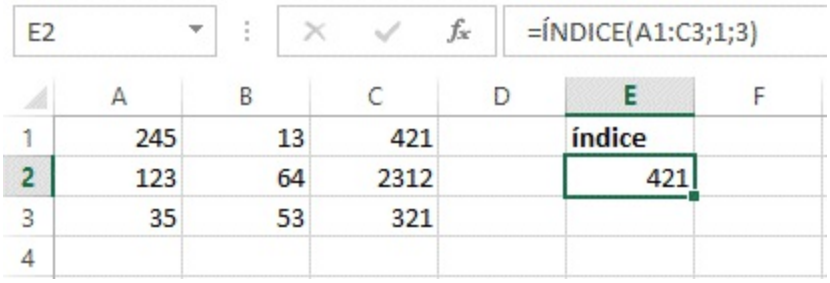
	A	B	C	D	E
1	**Departamento**	**Vendas**			**Índice**
2	Feminino	23.887,00			Masculino
3	Masculino	16.992,00			
4	Infantil	89.002,00			
5	Infanto-Juvenil	12.887,00			
6	Lingerie	19.009,00			
7	Calçados Masculinos	5.664,00			
8	Calçados Femininos	9.988,00			
9	Calçados Infantis	3.821,00			
10	Acessórios	1.203,00			

=ÍNDICE(**A2:A10**;**2**)

B2:B10 – Região que retorna os valores
2 – Linha de retorno do valor

Neste exemplo, utiliza-se a formula índice para encontrar quem está na segunda linha, **B2:B10** é a região, e 2 é a posição da linha retornada. O resultado consequentemente será Masculino.

Exemplo 2:



	A	B	C	D	E	F
1	245	13	421		índice	
2	123	64	2312		421	
3	35	53	321			
4						

=ÍNDICE(**A1:C3**;**1;3**)

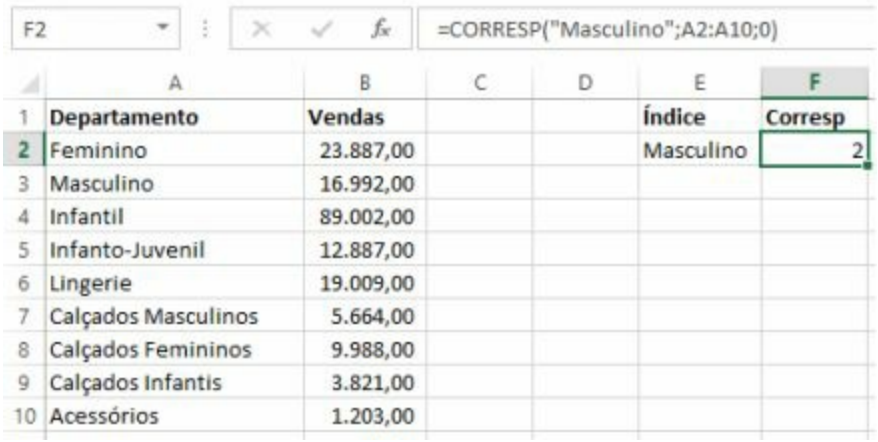
A1:C3 – Região que retorna os valores
1 – Linha onde será retornado o valor
3 – Coluna onde será retornado o valor

Esta formula, aplica o índice para uma matriz de região **A1:C3**, como se trata de uma região de matriz, é preciso informar duas coordenadas para retornar o valor, primeiro a linha e depois a coluna, neste exemplo de valor **1** e **3**, respectivamente.

CORRESP()

Esta função funciona de forma inversa ao índice, ao informar um valor e uma região, a posição é retornada, ao contrário do índice, esta funciona apenas em uma direção.

Exemplo:



	A	B	C	D	E	F
1	**Departamento**	**Vendas**			**Índice**	**Corresp**
2	Feminino	23.887,00			Masculino	2
3	Masculino	16.992,00				
4	Infantil	89.002,00				
5	Infanto-Juvenil	12.887,00				
6	Lingerie	19.009,00				
7	Calçados Masculinos	5.664,00				
8	Calçados Femininos	9.988,00				
9	Calçados Infantis	3.821,00				
10	Acessórios	1.203,00				

=CORRESP("**Masculino**";**A2:A10**;**0**)

"**Masculino**" – Nome procurado
A2:A10 – Região procurada
0 – Correspondência exata

Desta forma, o CORRESP() retorna a posição da palavra masculino, dentro da seleção, logo o valor será 2, visto que da seleção informada, ele está na segunda posição, o número **0**, informado após a região, determina que a correspondência será exata.

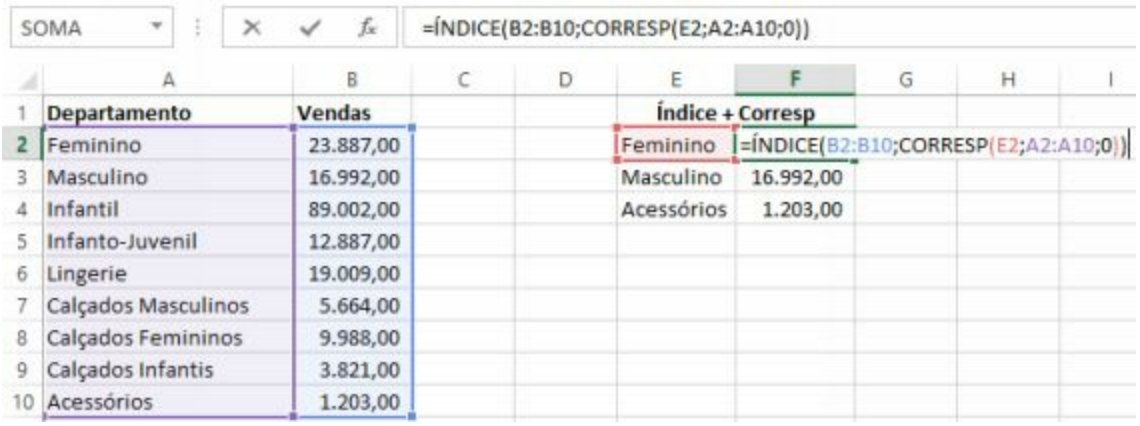
Agora que o ÍNDICE() e o CORRESP() já foram apresentados, será possível apresentar a poderosa combinação das duas funções. Uma vez que o índice retorna o valor de uma posição e o CORRESP() retorna a posição de um valor, será simples entender a utilidade desta duas funções juntas.

ÍNDICE(CORRESP())

Esta combinação, tem uma poderosa função de buscar o conteúdo de uma célula em uma região e retornar o valor de uma outra região. Como apresentado anteriormente, o ÍNDICE() tem a função de retornar o valor de uma coordenada, o CORRESP() por sua vez realiza o processo inverso, ao utiliza-los em colunas diferentes, é possível buscar qualquer valor, seja em uma direção ou em duas direções (apresentado na próxima seção).

Dica: Muitos falam sobre a importância do PROCV(), mas nem todos sabem que a fórmula índice com suas combinações tem um potencial muito maior que o PROCV(), realizando a mesma função, porém com maior eficiência e simplicidade e sem limitações indesejadas.

Exemplo:



=ÍNDICE(**B2:B10**;CORRESP(**E2**;**A2:A10**;**0**))

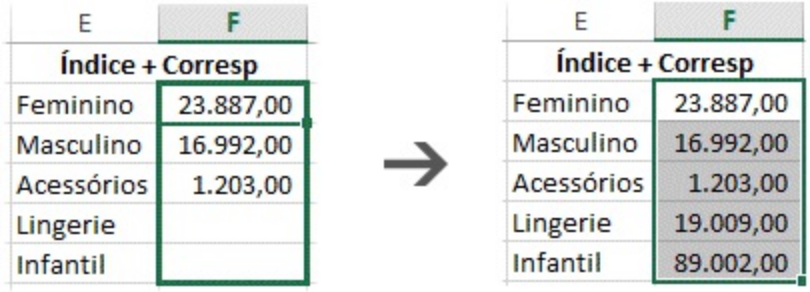
B2:B10 – Região que retorna os valores
E2 – Valor procurado
A2:A10 – Região procurada
0 – Correspondência exata

Neste exemplo, deseja-se buscar o valor de vendas do departamento "Feminino", para tanto, informa-se que a região de índice é **B2:B10**, posteriormente, é utilizado o CORRESP() buscando a célula de valor "Feminino", célula **E2**, na região **A2:A10**. Esta combinação fará o valor

correspondente ser retornado na célula onde a função está sendo aplicada.

Dica: Como muitas vezes se deseja arrastar esta formula para os demais itens buscados, é interessante “travar as fórmulas” com $, desta maneira, pode-se arrastar sem modificar a região de busca, da seguinte forma:

=ÍNDICE(**$B$2:$B$10**;CORRESP(**E2**;**$A$2:$A$10**;**0**))



E	F
Índice + Corresp	
Feminino	23.887,00
Masculino	16.992,00
Acessórios	1.203,00
Lingerie	
Infantil	

E	F
Índice + Corresp	
Feminino	23.887,00
Masculino	16.992,00
Acessórios	1.203,00
Lingerie	19.009,00
Infantil	89.002,00

INDICE(CORRESP(); CORRESP())

Como visto anteriormente, o índice retorna um valor através de uma coordenada, porém só foi apresentado o índice quando aplicado em uma coluna, porém o índice pode ser aplicado em duas direções, através de duas coordenadas. Conforme exemplo abaixo:

Exemplo:

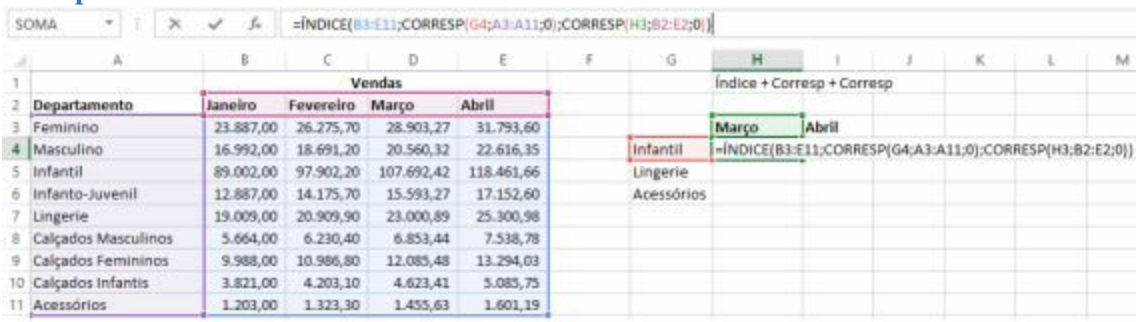


	A	B	C	D	E	F	G
1		Vendas					ÍNDICE
2	Departamento	Janeiro	Fevereiro	Março	Abril		18.691,20
3	Feminino	23.887,00	26.275,70	28.903,27	31.793,60		
4	Masculino	16.992,00	18.691,20	20.560,32	22.616,35		
5	Infantil	89.002,00	97.902,20	107.692,42	118.461,66		
6	Infanto-Juvenil	12.887,00	14.175,70	15.593,27	17.152,60		
7	Lingerie	19.009,00	20.909,90	23.000,89	25.300,98		
8	Calçados Masculinos	5.664,00	6.230,40	6.853,44	7.538,78		
9	Calçados Femininos	9.988,00	10.986,80	12.085,48	13.294,03		
10	Calçados Infantis	3.821,00	4.203,10	4.623,41	5.085,75		
11	Acessórios	1.203,00	1.323,30	1.455,63	1.601,19		

Neste exemplo, através do índice selecionando a região de todos os valores, B3:E11, deseja-se saber o valor da posição **2;2** ou seja, linha **2** da região e coluna **2**. Consequentemente o valor retornado é equivalente a Masculino para o mês de Fevereiro.

Da mesma forma como apresentado anteriormente, os valores do ÍNDICE() serão representados pela função CORRESP(), porém desta vez, será realizado de forma dupla, para retornar o departamento (linha), e o mês desejado (coluna).

Exemplo:



	A	B	C	D	E	F	G	H	I	J	K	L	M
1		Vendas						Índice + Corresp + Corresp					
2	Departamento	Janeiro	Fevereiro	Março	Abril								
3	Feminino	23.887,00	26.275,70	28.903,27	31.793,60			Março	Abril				
4	Masculino	16.992,00	18.691,20	20.560,32	22.616,35		Infantil	=ÍNDICE(B3:E11;CORRESP(G4;A3:A11;0);CORRESP(H3;B2:E2;0))					
5	Infantil	89.002,00	97.902,20	107.692,42	118.461,66		Lingerie						
6	Infanto-Juvenil	12.887,00	14.175,70	15.593,27	17.152,60		Acessórios						
7	Lingerie	19.009,00	20.909,90	23.000,89	25.300,98								
8	Calçados Masculinos	5.664,00	6.230,40	6.853,44	7.538,78								
9	Calçados Femininos	9.988,00	10.986,80	12.085,48	13.294,03								
10	Calçados Infantis	3.821,00	4.203,10	4.623,41	5.085,75								
11	Acessórios	1.203,00	1.323,30	1.455,63	1.601,19								

=ÍNDICE(**B3:E11**;CORRESP(**G4**;**A3:A11**;**0**);CORRESP(**H3**;**B2:E2**;**0**))

B3:B11 – Região que retorna os valores
G4 – Valor procurado 1
A3:A11 – Região procurada 1
0 – Correspondência exata
H3 – Valor procurado 2
B2:E2 – Região procurada 2
0 – Correspondência exata

O funcionamento é extremamente o mesmo, porém desta vez se busca duas correspondências, no caso deste exemplo, uma para a célula **G4**, relacionada aos departamentos, com região de procura **A3:A11** nos departamentos e a célula **H3**, relacionada aos meses do ano, na região de procura em **B2:E2**, de maneira semelhante, a formula do CORRESP fecha com o valor 0, para determinar que a correspondência que se busca é exata.

Dica: Da mesma maneira como apresentado anteriormente, é interessante travar as regiões de retorno e de busca, para que desta forma, se possa arrastar a formula para demais variáveis buscadas, conforme exemplo abaixo:



	A	B	C	D	E	F	G	H	I
1		Vendas						Índice + Corresp + Corresp	
2	**Departamento**	**Janeiro**	**Fevereiro**	**Março**	**Abril**				
3	Feminino	23.887,00	26.275,70	28.903,27	31.793,60			**Março**	**Abril**
4	Masculino	16.992,00	18.691,20	20.560,32	22.616,35		Infantil	107.692,42	118.461,66
5	Infantil	89.002,00	97.902,20	107.692,42	118.461,66		Lingerie	23.000,89	25.300,98
6	Infanto-Juvenil	12.887,00	14.175,70	15.593,27	17.152,60		Acessórios	1.455,63	1.601,19
7	Lingerie	19.009,00	20.909,90	23.000,89	25.300,98				
8	Calçados Masculinos	5.664,00	6.230,40	6.853,44	7.538,78				
9	Calçados Femininos	9.988,00	10.986,80	12.085,48	13.294,03				
10	Calçados Infantis	3.821,00	4.203,10	4.623,41	5.085,75				
11	Acessórios	1.203,00	1.323,30	1.455,63	1.601,19				

=ÍNDICE(**$B$3:$E$11**;CORRESP(**$G4**; **$A$3:$A$11**;**0**);CORRESP(**H$3**; **$B$2:$E$2**;**0**))

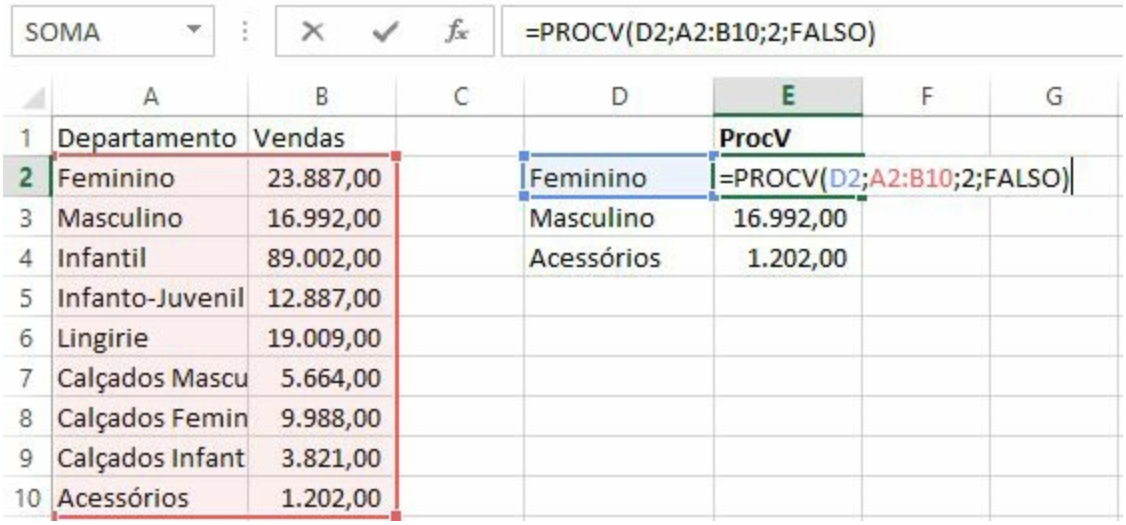
Obs1: $G4, recebeu a trava apenas na coluna, pois ainda interessa arrasta-lo

para baixo.
Obs2: H$3, recebeu a trava apenas na linha, pois ainda interessa arrasta-lo para os lados.

PROCV()

Esta é uma das funções mais conhecidas para se buscar um valor no Excel, através dela, é possível informar um valor procurado, uma matriz de busca e obter-se um valor correspondente, para facilitar a comparação, será utilizado o mesmo exemplo utilizado para o ÍNDICE(CORRESP()).

Exemplo:



	A	B	C	D	E	F	G
1	Departamento	Vendas			ProcV		
2	Feminino	23.887,00		Feminino	=PROCV(D2;A2:B10;2;FALSO)		
3	Masculino	16.992,00		Masculino	16.992,00		
4	Infantil	89.002,00		Acessórios	1.202,00		
5	Infanto-Juvenil	12.887,00					
6	Lingirie	19.009,00					
7	Calçados Mascu	5.664,00					
8	Calçados Femin	9.988,00					
9	Calçados Infant	3.821,00					
10	Acessórios	1.202,00					

=PROCV(**D2**;**A2:B10**;**2**;**FALSO**)

D2 – Célula que se deseja buscar

A2:B10 – Matriz de procura

2 – Posição da coluna de retorno

FALSO – Busca exata pelo termo procurado

A função tem por objetivo, buscar um valor, neste exemplo célula **D2**, dentro da região **A2:B10**, retornando o valor equivalente da coluna **2** da matriz informada. Utiliza-se **FALSO** para especificar que o termo buscado terá correspondência exata e não aproximada.

Obs: O PROCV() possui uma limitação, onde o valor de retorno precisa obrigatoriamente estar a direita do valor procurado, funcionando apenas para estruturas organizadas da esquerda para direita. Caso contrário será

necessário alterar a forma como os dados estão dispostos na planilha ou utilizar a combinação ÍNDICE(CORRESP()) que tem o mesmo funcionamento, sem esta limitação.

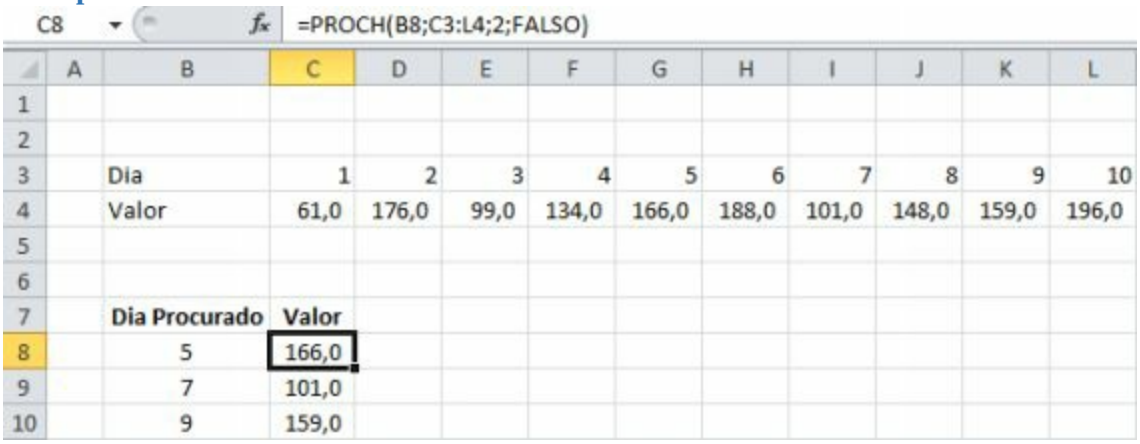
Dica: Como muitas vezes se deseja arrastar esta formula para os demais itens buscados, é interessante “travar as fórmulas” com $, desta maneira, pode-se arrastar sem modificar a região de busca, da seguinte forma:

= **PROCV(D2;$A$2:$B$10;2;FALSO)**

PROCH()

Esta função funciona de forma extremamente similar ao PROCV(), a única diferença é que o PROCH() realizar uma busca de forma horizontal ao invés de ser na vertical. A fórmula é eficaz para realizar pesquisa e retorno de dados.

Exemplo:



	A	B	C	D	E	F	G	H	I	J	K	L
1												
2												
3		Dia	1	2	3	4	5	6	7	8	9	10
4		Valor	61,0	176,0	99,0	134,0	166,0	188,0	101,0	148,0	159,0	196,0
5												
6												
7		**Dia Procurado**	**Valor**									
8		5	166,0									
9		7	101,0									
10		9	159,0									

=PROCH(**B8**;**C3:L4**;2;**FALSO**)

B8 - Valor procurado

C3:L4 - Matriz onde será o valor será encontrado

2 - Será buscado na segunda linha da matriz

FALSO - A correspondência será exata

Este exemplo demonstra de forma direta a aplicação do PROCH(), onde é necessário pesquisar um valor que está presente na célula **B8**, dentro da matriz **C3:L4**, para tanto se utiliza o índice de valor **2**, o que significa que o valor será encontrado na segunda linha desta matriz. O último argumento é dado como **FALSO**, pois se deseja buscar o exato valor e não o seu número aproximado.

Obs: É sempre interessante "travar" a região de procura para poder arrastar

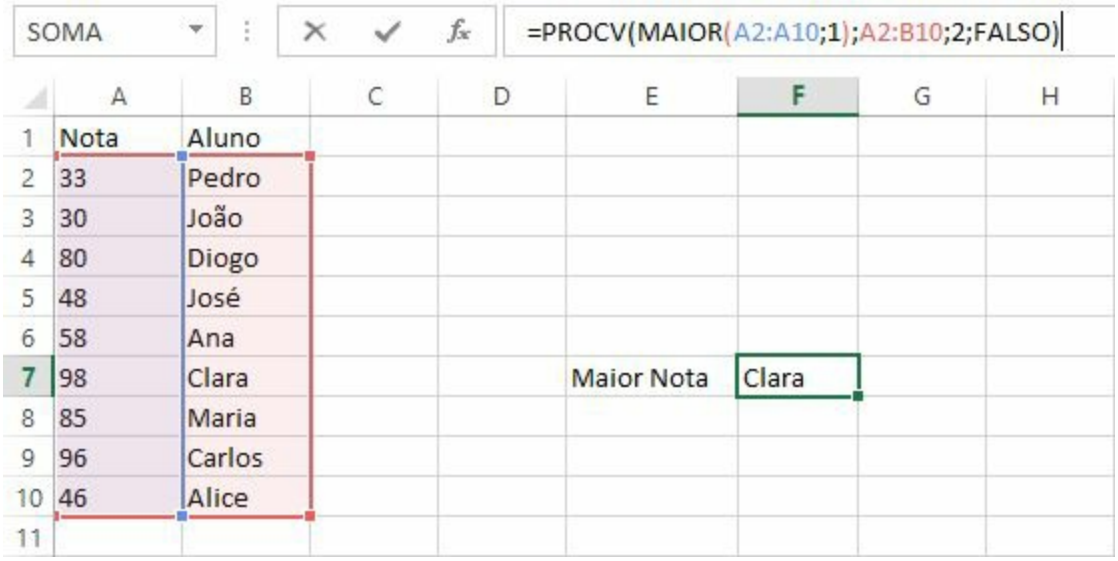
a fórmulas para os demais valores procurados, para tanto utiliza-se $, a função fica da seguinte forma: =PROCH(**B8**;**$C$3:$L$4**;2;FALSO)

PROCV(MAIOR())

Esta função tem por objetivo realizar uma busca e retornar o valor correspondente a uma célula de maior valor.

Dica: É importante ressaltar que a mesma combinação também pode ser utilizada para PROCV(MENOR())

Exemplo:



=PROCV(MAIOR(**A2:A10**;**1**);**A2:B10**;**2**;**FALSO**)

A2:A10 – Região de máximo valor desejado

1 – Valor buscado é o maior (Para segundo maior, usar 2 e assim sucessivamente)

A2:B10 – Matriz de busca

2 – Da matriz selecionada o dado será retornado da segunda coluna

FALSO – Utilizado para valor exato

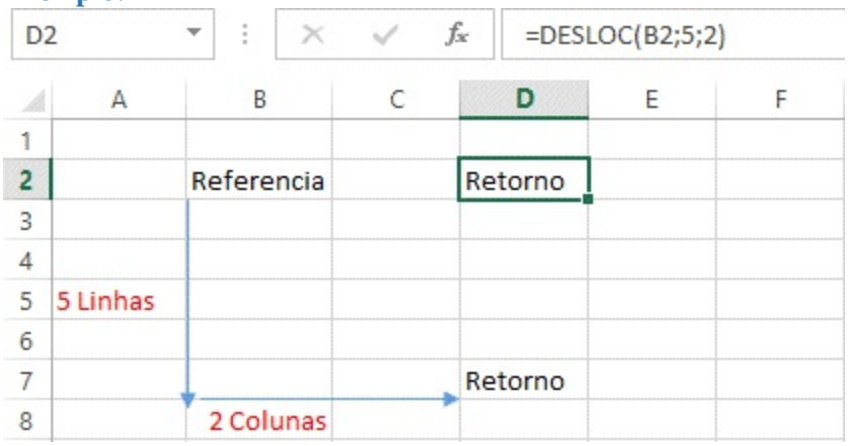
A função funciona da maneira padrão do PROCV(), porém ao invés de se definir um valor de busca, utiliza-se MAIOR(**A2:A10**;**1**). A Matriz de procura é a região onde a informação será pesquisada **A2:B10**, e o índice **2**,

indica que o valor retornado estará na segunda coluna da matriz de procura.

DESLOC()

Esta é mais uma das funções que muitos pensam não ter utilidade, porém ela é extremamente poderosa, principalmente quando combinada com outras funções. Seu funcionamento basicamente consiste em retornar o valor de células a uma distância informada a partir de uma referência inicial.

Exemplo:



=DESLOC(**B2**;**5**;**2**)

B2 – Referência de início

5 – Quantidade de linhas deslocadas

2 – Quantidade de colunas deslocadas

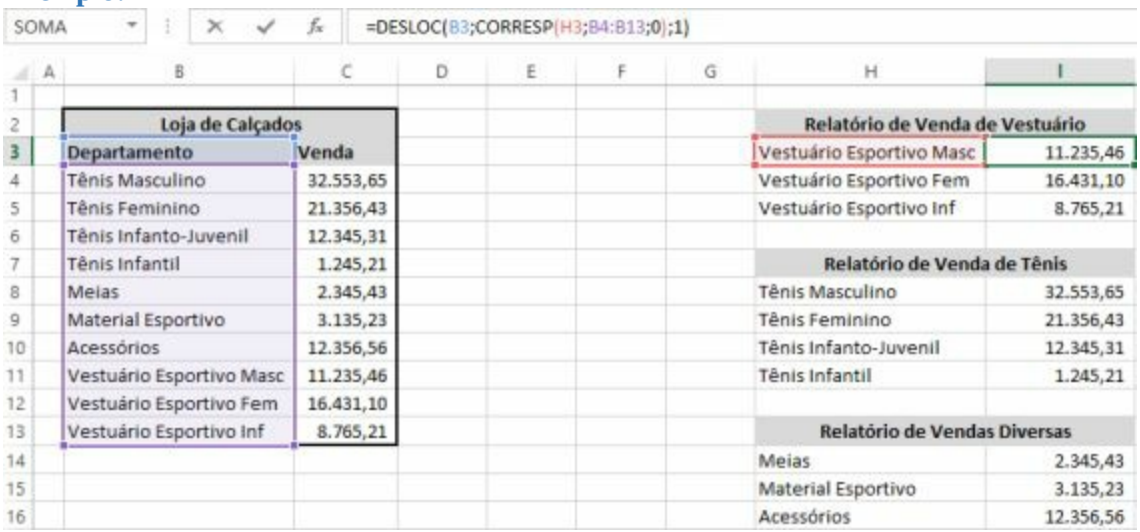
Uma vez que a fórmula seja aplicada na célula **B2** com deslocamento de **5** linhas e **2** colunas o valor encontrado é exatamente o da célula D7 que possui texto Retorno.

Obs: As setas são apenas ilustrativas neste exemplo.

DESLOC(CORRESP())

Esta é mais uma combinação que tem alto potencial para realizar busca de informações, assim como o ÍNDICE(CORRESP()), esta também é uma alternativa poderosa para o ProcV, possuindo também menores limitações e funcionando de forma mais eficaz.

Exemplo:



Loja de Calçados	
Departamento	Venda
Tênis Masculino	32.553,65
Tênis Feminino	21.356,43
Tênis Infanto-Juvenil	12.345,31
Tênis Infantil	1.245,21
Meias	2.345,43
Material Esportivo	3.135,23
Acessórios	12.356,56
Vestuário Esportivo Masc	11.235,46
Vestuário Esportivo Fem	16.431,10
Vestuário Esportivo Inf	8.765,21

Relatório de Venda de Vestuário	
Vestuário Esportivo Masc	11.235,46
Vestuário Esportivo Fem	16.431,10
Vestuário Esportivo Inf	8.765,21

Relatório de Venda de Tênis	
Tênis Masculino	32.553,65
Tênis Feminino	21.356,43
Tênis Infanto-Juvenil	12.345,31
Tênis Infantil	1.245,21

Relatório de Vendas Diversas	
Meias	2.345,43
Material Esportivo	3.135,23
Acessórios	12.356,56

=DESLOC(**B3** ;CORRESP(**H3**;**B4:B13**;**0**);**1**)

B3 – Célula referência para a função realizar o deslocamento

H3 – Célula que será buscada na região de procura

B4:B13 – Região de procura

0 – Correspondência exata

1 – A partir da célula referência (B3), será deslocado 1 coluna para direita

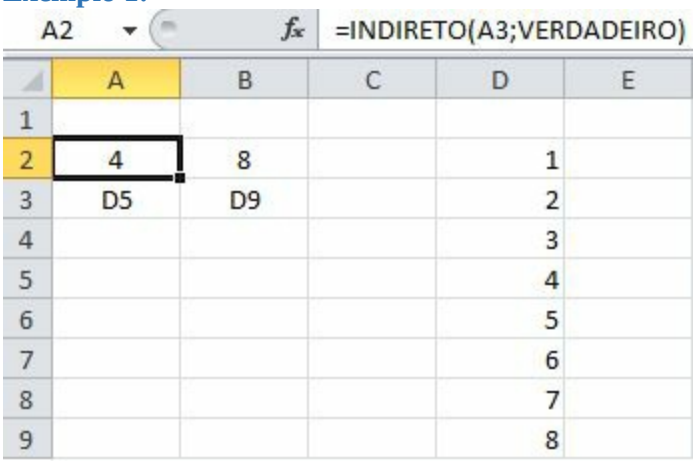
O funcionamento desta combinação se dá através da função DESLOC() que retorna um valor a partir de uma referência, a referência neste exemplo é a célula **B3** a partir da qual precisa de "coordenadas" para encontrar o valor desejado, algo que a função CORRESP() realiza, através da célula de procura **H3**, na região de busca **B4:B13**, desta forma o valor da linha desejada é encontrada, a da coluna é basicamente o número **1**, visto que a coluna dos valores fica exatamente a uma unidade para direita.

Obs: É sempre interessante “travar” as fórmulas para poder arrasta-las sem deslocar as referências corretas, no caso deste exemplo:
=DESLOC(**B$3**;CORRESP(**H3**;**$B$4:$B$13**;**0**);**1**)

INDIRETO()

A função indireto é extremamente poderosa dentro do Excel. Ela utiliza referencia textual para retornar o valor de outra célula, seja ela na mesma aba ou não. Ela tem o potencial de substituir a utilização de diversas fórmulas condicionais, simplificando a consolidação de dados. O primeiro exemplo tem por objetivo apenas demonstrar a utilização de forma simples, posteriormente outro exemplo será apresentado demonstrando sua real utilidade.

Exemplo 1:



=INDIRETO(**A3**;**VERDADEIRO**)

A3 = Referência para retornar o valor

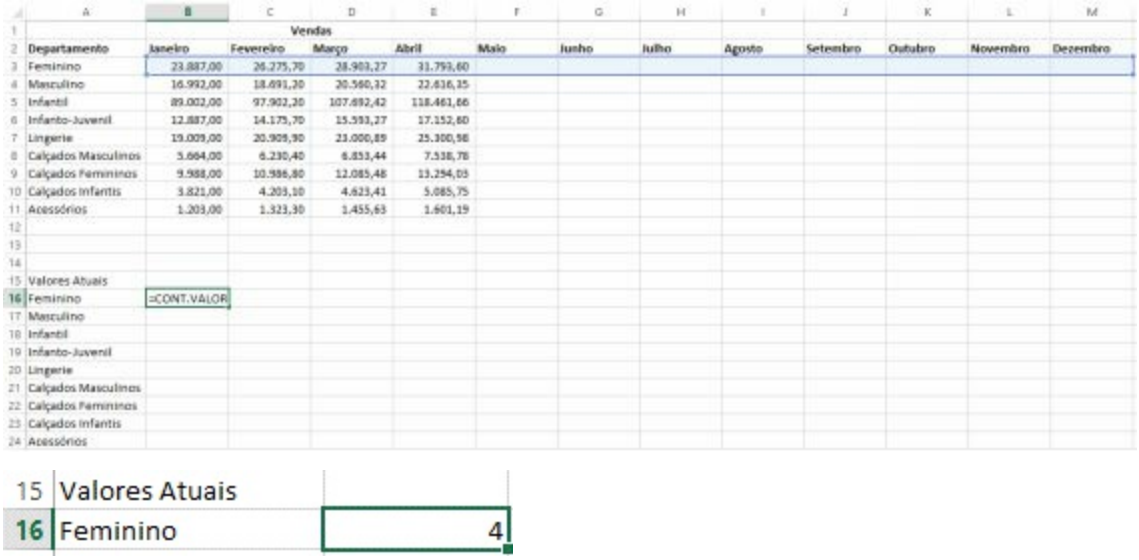
VERDADEIRO = Referência no estilo "A1", Falso retorna "L1C1"

Os argumentos da função são extremamente simples, basicamente **A3** é célula que será utilizada como referência para retornar um valor. Como a célula **A3** possui seu valor D5, o valor encontrado pela função indireto é o mesmo da célula D5. O segundo argumento é opcional, não preencher é o mesmo que optar por Verdadeiro, o que especifica a referência do tipo "A1", para alternar para referência do tipo L1C1 é preciso colocar como Falso.

CONT.VALORES()

Esta função é simples, tem o objetivo de retornar o valor de células preenchidas em uma região informada. Ela possui uma excelente funcionalidade para ser combinada com o índice, a qual será apresentada na seção a seguir.

Exemplo:



	A	B	C	D	E	F	G	H	I	J	K	L	M
1		Vendas											
2	Departamento	Janeiro	Fevereiro	Março	Abril	Maio	Junho	Julho	Agosto	Setembro	Outubro	Novembro	Dezembro
3	Feminino	23.887,00	26.275,70	28.903,27	31.793,60								
4	Masculino	16.992,00	18.691,20	20.560,32	22.616,35								
5	Infantil	89.002,00	97.902,20	107.692,42	118.461,66								
6	Infanto-Juvenil	12.887,00	14.175,70	15.593,27	17.152,60								
7	Lingerie	19.009,00	20.909,90	23.000,89	25.300,98								
8	Calçados Masculinos	5.664,00	6.230,40	6.853,44	7.538,78								
9	Calçados Femininos	9.988,00	10.986,80	12.085,48	13.294,03								
10	Calçados Infantis	3.821,00	4.203,10	4.623,41	5.085,75								
11	Acessórios	1.203,00	1.323,30	1.455,63	1.601,19								
12													
13													
14													
15	Valores Atuais												
16	Feminino	=CONT.VALOR											
17	Masculino												
18	Infantil												
19	Infanto-Juvenil												
20	Lingerie												
21	Calçados Masculinos												
22	Calçados Femininos												
23	Calçados Infantis												
24	Acessórios												

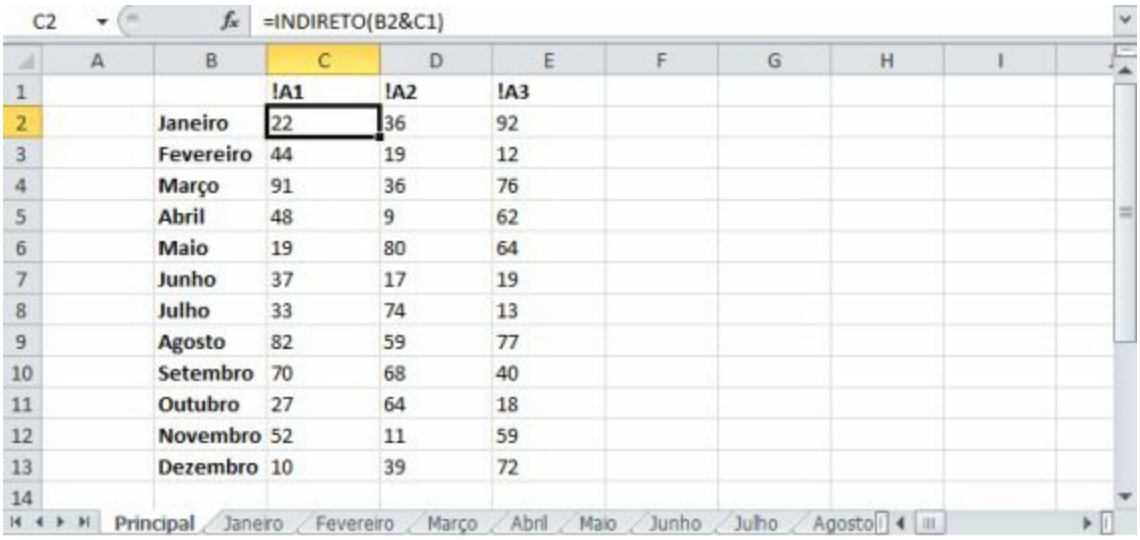
15	Valores Atuais	
16	Feminino	4

=CONT.VALORES(**B3:M3**)

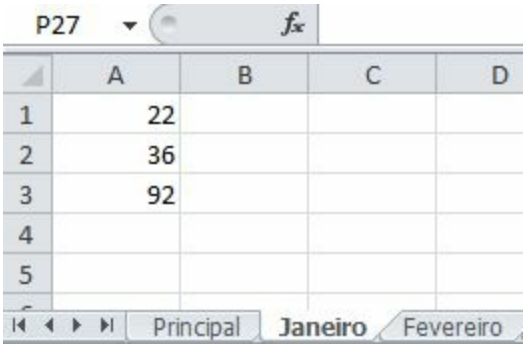
B3:M3 – Região a ser contada

Informada a região **B3:M3**, o valor retornado será a quantidade de valores preenchidos na linha especificada.

Exemplo 2:



	A	B	C	D	E
1			!A1	!A2	!A3
2		Janeiro	22	36	92
3		Fevereiro	44	19	12
4		Março	91	36	76
5		Abril	48	9	62
6		Maio	19	80	64
7		Junho	37	17	19
8		Julho	33	74	13
9		Agosto	82	59	77
10		Setembro	70	68	40
11		Outubro	27	64	18
12		Novembro	52	11	59
13		Dezembro	10	39	72
14					



	A	B	C	D
1	22			
2	36			
3	92			
4				
5				

=INDIRETO(**B2**&**C1**)
B2 – Referência para busca
& - Concatena os dois valores
C1 - Referência para busca

Este exemplo já demonstra um potencial maior da função indireto, através da mesma, pode-se consolidar dados de outras abas, utilizando a fórmula indireto com o nome da aba concatenado com a célula desejada, no caso da célula selecionada indireto está buscando o valor da célula **B2** (Janeiro), concatenado com **C1** (!A1), o valor buscado de maneira indireta será Janeiro!A1.

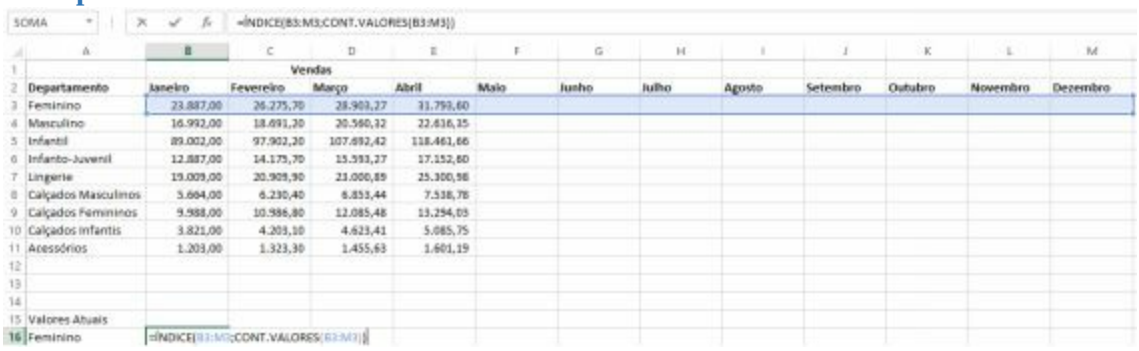
Obs: É sempre interessante "travar" a linha da região horizontal e a coluna da região vertical, para que as mesmas tenham suas posições mantidas

quando as fórmulas forem arrastadas. Neste exemplo ficaria:
=INDIRETO(**B2**&**C1**)

INDICE(CONT.VALORES())

Esta combinação é muito útil quando se deseja retornar sempre o último valor disponível em uma região desejada.

Exemplo:



15	Valores Atuais	
16	Feminino	31.793,60

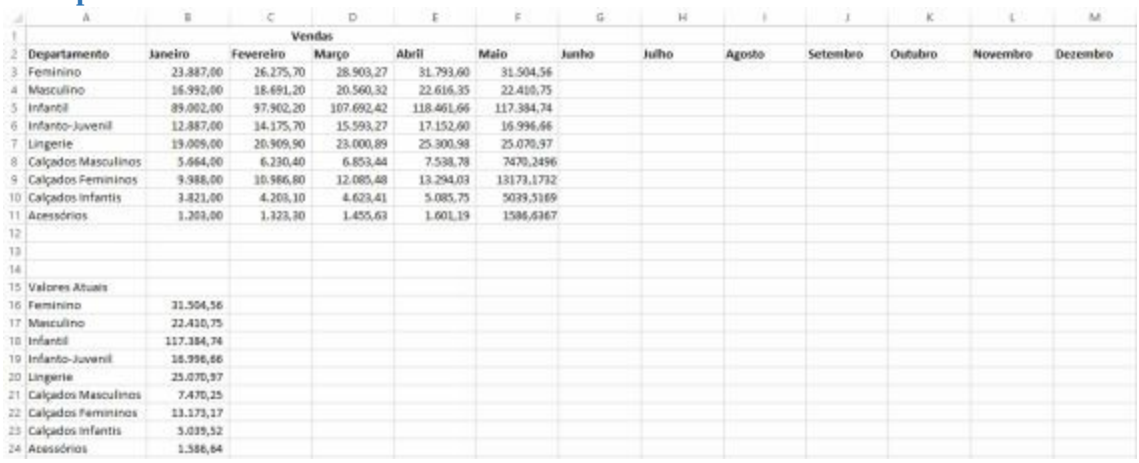
=ÍNDICE(**B3:M3**;CONT.VALORES(**B3:M3**))

B3:M3 – Região de retorno

B3:M3 – Região de contagem

Sempre que novos valores forem inseridos, a combinação atualizará os valores para os últimos inseridos, conforme na imagem abaixo, onde demonstra que uma vez que os valores de maio sejam preenchidos, a função atualizará automaticamente os valores.

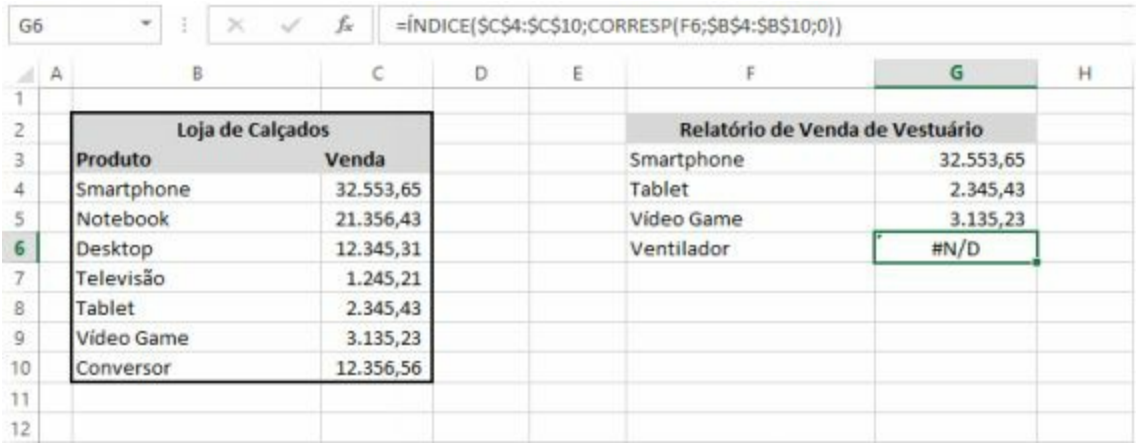
Exemplo:



SEERRO() + FUNÇÕES DE PROCURA

A função SEERRO() é comumente utilizada acompanhando funções de procura de dados, como PROCV(), ÍNDICE(CRRESP()) ou DESLOC(CORRESP()). Uma vez que é esperado que nem todos os dados sejam encontrados, desta forma, todos os que não forem encontrados apresentarão um erro de #N/D, conforme o exemplo a seguir.

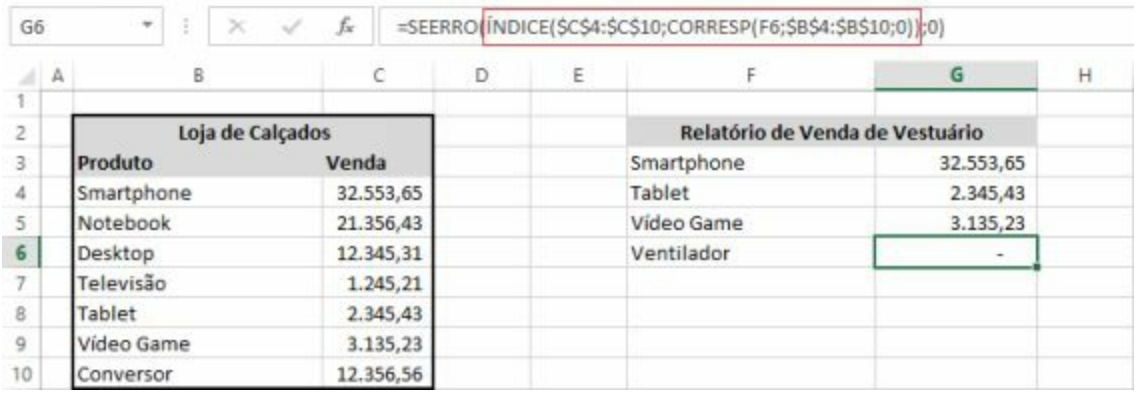
Exemplo:



	Loja de Calçados	
	Produto	**Venda**
	Smartphone	32.553,65
	Notebook	21.356,43
	Desktop	12.345,31
	Televisão	1.245,21
	Tablet	2.345,43
	Vídeo Game	3.135,23
	Conversor	12.356,56

Relatório de Venda de Vestuário	
Smartphone	32.553,65
Tablet	2.345,43
Vídeo Game	3.135,23
Ventilador	#N/D

Supondo que esta busca já seja esperada e deseja-se que ela apenas retorne o valor zero, basta utilizar a função dentro de um SEERRO() com argumento de valor zero.



	Loja de Calçados	
	Produto	**Venda**
	Smartphone	32.553,65
	Notebook	21.356,43
	Desktop	12.345,31
	Televisão	1.245,21
	Tablet	2.345,43
	Vídeo Game	3.135,23
	Conversor	12.356,56

Relatório de Venda de Vestuário	
Smartphone	32.553,65
Tablet	2.345,43
Vídeo Game	3.135,23
Ventilador	-

Desta forma, o inconveniente #N/D é substituído pelo valor zero.

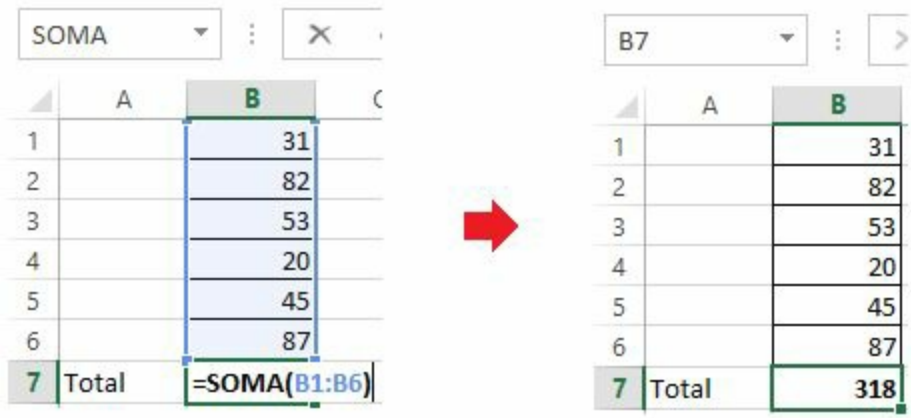
FUNÇÕES DE SOMA E MÉDIA

Existem diversas formas de somar informações no Excel, as quais vão desde fórmulas mais simples até as mais complexas, permitindo ao usuário somar dados de maneira direta ou somar com a utilização de uma condição. A seguir os diversos casos serão apresentados e exemplificados.

SOMA()

Esta é uma das funções mais básicas do Excel, consiste basicamente em escolher a região que será somada.

Exemplo:



=SOMA(**B1:B6**)

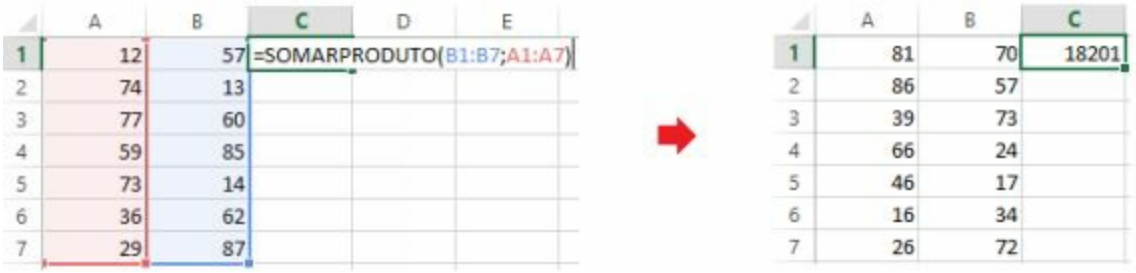
B1:B6 – Intervalo onde a soma será realizada

A função é simples e direta, ela basicamente insere a soma de uma região de valores escolhidos.

SOMARPRODUTO ()

Esta função tem por objetivo somar o produto de duas regiões, ela é extremamente simples e substitui a utilização de auxiliares para fazer o mesmo procedimento utilizando apenas Soma().

Exemplo:



=SOMARPRODUTO(**B1:B7**;**A1:A7**)

B1:B7 – Primeira Matriz

A1:A7 – Segunda Matriz

Neste exemplo, são selecionadas duas regiões que serão multiplicadas entre si e posteriormente somadas posição por posição para chegar a um resultado final.

Obs 1: Neste exemplo foram utilizadas apenas duas regiões, porém a função permite a escolha do número desejado de regiões que serão multiplicadas. É necessário que elas possuam a mesma quantidade de dados.

SUBTOTAL()

Uma característica da função soma apresentada anteriormente, é que a mesma sempre irá somar os valores de uma região, independente das células estarem filtradas ou não. Para contornar este problema, pode-se utilizar a função Subtotal(), a qual pode ser utilizada para realizar a soma apenas nas células desejadas.

Exemplo:



=SUBTOTAL(**9**;**E2:E9**)

9 – Opção escolhida para o subtotal realizar Soma

E2:E9 – Intervalo onde a soma será realizada

Neste exemplo, o subtotal é demonstrado em uma região onde um filtro é aplicado na esquerda dos valores, uma vez que um critério seja estabelecido, a função subtotal automaticamente soma apenas as informações “visíveis”.

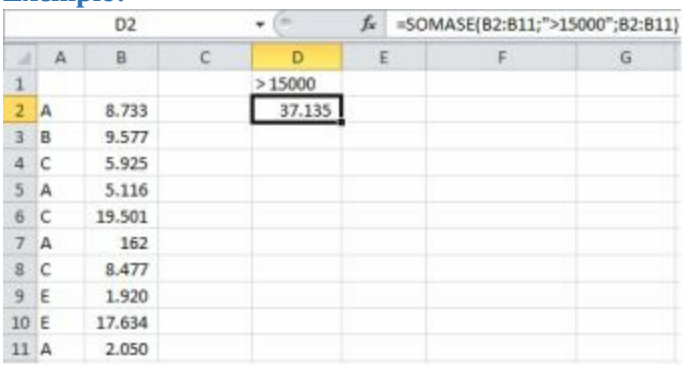
Dica: Apesar da função subtotal está sendo mostrada dentro da seção de soma, ela possui as mais variadas utilidades, no exemplo apresentado a função escolhida foi a número 9 que representa a soma, porém existem diversas outras funções para se aplicar o Subtotal(). Segue abaixo uma lista:

Código da função		Função
Inclui ocultos	Ignora ocultos	
1	101	MÉDIA
2	102	CONTA
3	103	CONT.VALORES
4	104	MÁXIMO
5	105	MÍNIMO
6	106	MULT
7	107	DESVPAD
8	108	DESVPADP
9	109	SOMA
10	110	VAR
11	111	VARP

SOMASE()

Esta é uma função de extrema importância para tratamento de dados, através dela é possível somar números seguindo um critério estabelecido.

Exemplo:



	A	B	C	D	E	F	G
1				>15000			
2	A	8.733		37.135			
3	B	9.577					
4	C	5.925					
5	A	5.116					
6	C	19.501					
7	A	162					
8	C	8.477					
9	E	1.920					
10	E	17.634					
11	A	2.050					

=SOMASE(**B2:B11**;">15000"; **B2:B11**)

B2:B11 – Intervalo onde é aplicado o critério

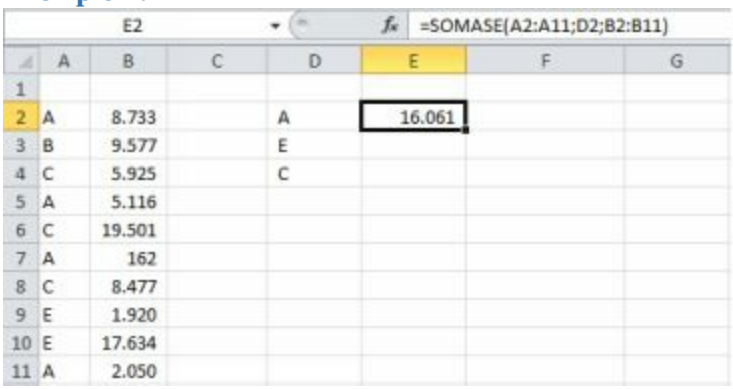
">15000" – Critério aplicado

B2:B11 – Intervalo onde a soma será realizada

Este primeiro exemplo, demonstra a aplicação mais simples do SOMASE(). Todo valor dentro do intervalo B2:B11 que atende o critério de ser maior que 15.000 (">15000"), é somado na célula D2.

Obs 1: Nos casos onde o intervalo de soma é o mesmo de critério, a terceira parte da função não é obrigatória, desta forma, colocar de maneira reduzida, também obtém o mesmo resultado: =SOMASE(**B2:B11**;">15000")

Exemplo 2:



	A	B	C	D	E	F	G
1							
2	A	8.733		A	16.061		
3	B	9.577		E			
4	C	5.925		C			
5	A	5.116					
6	C	19.501					
7	A	162					
8	C	8.477					
9	E	1.920					
10	E	17.634					
11	A	2.050					

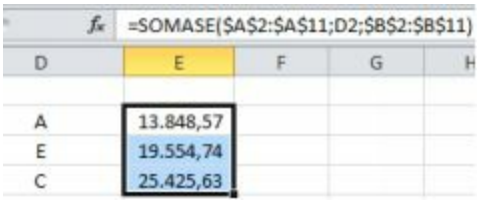
=SOMASE(**A2:A11**;**D2**;**B2:B11**)

A2:A11 – Intervalo onde é aplicado o critério

D2 – Critério aplicado

B2:B11 – Intervalo onde a soma será realizada

Neste exemplo, o SOMASE() irá somar todos os valores do intervalo de soma (**B2:B11**) que tiverem valores do intervalo de critério (**A2:A11**) iguais a célula de critério **D2**, ou seja, valores correspondentes a letra A.



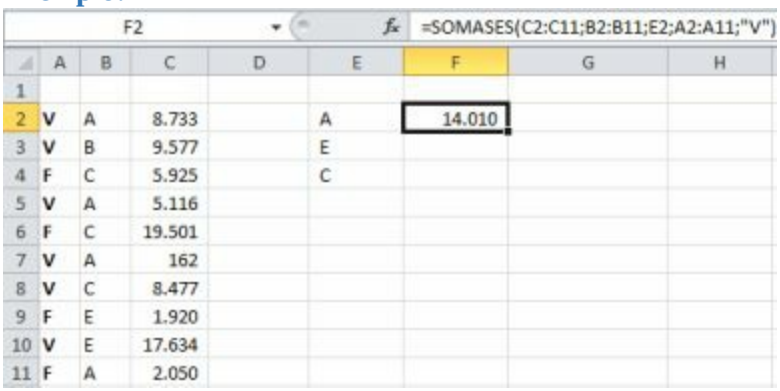
F interessante travar as posições do intervalo de soma e do intervalo de critério para poder “arrastar” a formula para os demais critérios.

=SOMASE(**$A$2:$A$11**;**D2**;**$B$2:$B$11**)

SOMASES()

O SOMASES() consiste na mesma lógica do SOMASE(), porém se aplicam mais critérios para soma. Também é trocada a ordem dos intervalos, onde é primeiro solicitado o intervalo de soma

Exemplo:



	A	B	C	D	E	F	G	H
1								
2	V	A	8.733		A	14.010		
3	V	B	9.577		E			
4	F	C	5.925		C			
5	V	A	5.116					
6	F	C	19.501					
7	V	A	162					
8	V	C	8.477					
9	F	E	1.920					
10	V	E	17.634					
11	F	A	2.050					

=SOMASES(**C2:C11**;**B2:B11**;**E2**;**A2**:**A11**;"V")

C2:C11 – Intervalo onde a soma será realizada

B2:B11 – Intervalo de critério 1

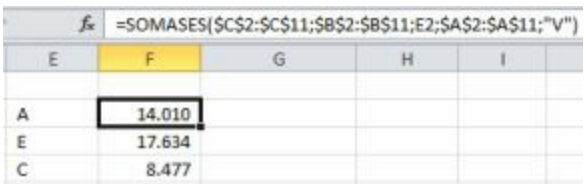
E2 – Critério 1

A2:**A11** – Intervalo de critério 2

“V” – Critério 2

Neste exemplo, o intervalo **C2:C11**, posteriormente são solicitados os critérios, sendo o primeiro **B2:B11**, com critério aplicado a célula **E2**, o segundo **A2**:**A11** com critério **“V”**.

Obs 1: Caso o intervalo de critério 2 e o critério 2 não fossem adicionados, a função se comportaria como um SOMASE() normal.



	E	F	G	H	I
A		14.010			
E		17.634			
C		8.477			

Obs 2: É interessante travar as posições do intervalo de soma e do intervalo de critério para poder “arrastar” a formula para os demais critérios.

```
=SOMASES($C$2:$C$11; $B$2:$B$11;E2;$A$2:$A$11;"V")
```

MEDIA()

A função média, consiste em selecionar uma região desejada, então o Excel realiza o cálculo do valor médio relacionado a seleção.

Exemplo:



=MÉDIA(**A1:A7**)

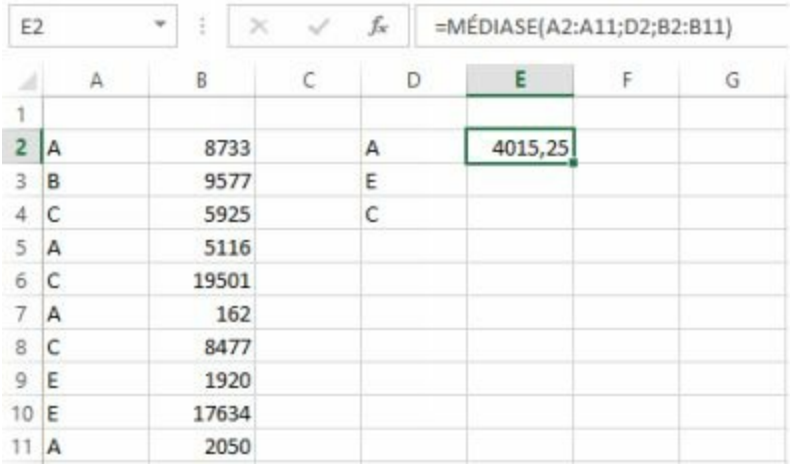
A1:A7 – Intervalo onde a média será realizada

Assim como a função SOMA(), a aplicação é direta, consiste em escolher um intervalor onde a média será calculada.

MEDIASE()

O lógica desta função é a mesma da aplicada ao SOMASE(), porém o resultado ao invés de ser uma soma através de um critério, será uma média realizada através de um critério aplicado.

Exemplo:



	A	B	C	D	E	F	G
1							
2	A	8733		A	4015,25		
3	B	9577		E			
4	C	5925		C			
5	A	5116					
6	C	19501					
7	A	162					
8	C	8477					
9	E	1920					
10	E	17634					
11	A	2050					

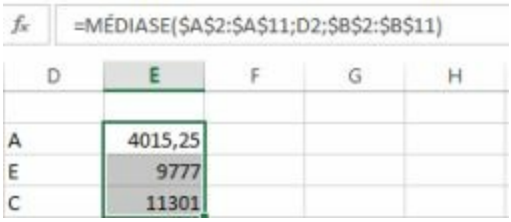
=MEDIASE(**A2:A11**;**D2**;**B2:B11**)

A2:A11 – Intervalo onde é aplicado o critério

D2 – Critério aplicado

B2:B11 – Intervalo onde a média será realizada

Neste exemplo, o MEDIASE() irá realizar a media todos os valores do intervalo (**B2:B11**) que tiverem valores do intervalo de critério (**A2:A11**) iguais a célula de critério **D2**, ou seja, valores correspondentes a letra A.



	D	E	F	G	H
A		4015,25			
E		9777			
C		11301			

Obs: É interessante travar as posições do intervalo de soma e do intervalo de critério para poder “arrastar” a formula para os demais critérios.

OUTRAS FUNÇÕES MATEMÁTICAS

As funções de soma e média são sem dúvidas as mais utilizadas das funções matemáticas dentro do Excel, porém a plataforma oferece outras funções interessantes e úteis, as quais serão apresentadas nesta seção. Para as quatro primeiras funções apresentadas nesta seção, o mesmo exemplo abaixo será utilizado.

Exemplo:

	A	B
1	Data	Vendas
2	01/01/2018	74
3	02/01/2018	65
4	03/01/2018	98
5	04/01/2018	90
6	05/01/2018	94
7	06/01/2018	59
8	07/01/2018	58
9	08/01/2018	78
10	09/01/2018	76

MÍNIMO ()

Função utilizada para retornar o menor valor de uma seleção de números.

=MÍNIMO(**B2:B10**)

B2:B10 – Região de procura

Esta função é extremamente simples, utiliza-se a região de procura **B2:B10** onde deseja-se retornar o menor valor, o qual neste exemplo será 58.

MÁXIMO ()

Função utilizada para retornar o maior valor de uma seleção de números.

=MÁXIMO(**B2:B10**)

B2:B10 – Região de procura

Esta função é extremamente simples, utiliza-se a região de procura **B2:B10** onde deseja-se retornar o maior valor, o qual neste exemplo será 98.

MENOR ()

Função utilizada para retornar o menor valor de uma seleção de números de acordo com a posição informada, por exemplo: ao informar posição 1, a função encontra o menor valor, ao informar a posição 2, a função retorna o segundo menor valor e assim sucessivamente.

=MENOR(**B2:B10**;**2**)

B2:B10 – Região de procura

2 – Segundo menor valor será retornado

Utilizando-se a região de procura **B2:B10**, é desejado retornar o segundo menor valor, o qual neste exemplo será 59.

Obs: Utilizando o valor 1 no segundo argumento, a função funciona exatamente igual a função MÍNIMO()

MAIOR()

Função utilizada para retornar o maior valor de uma seleção de números de acordo com a posição informada, por exemplo: ao informar posição 1, a função encontra o maior valor, ao informar a posição 2, a função retorna o segundo maior valor e assim sucessivamente.

=MAIOR(**B2:B10**;**3**)

B2:B10 – Região de procura

3 – Terceiro maior valor será retornado

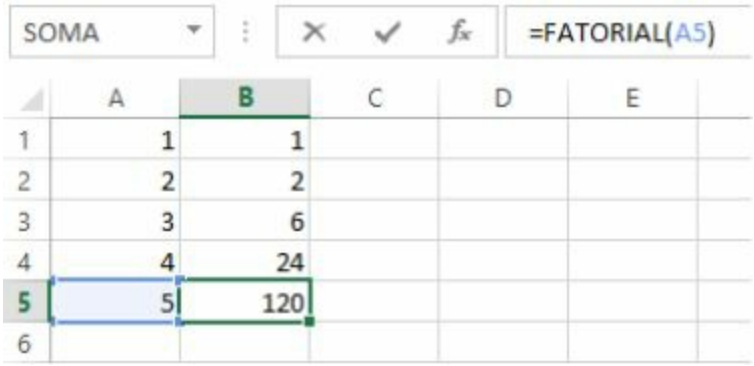
Utilizando-se a região de procura **B2:B10**, é desejado retornar o terceiro maior valor, o qual neste exemplo será 90.

Obs: Utilizando o valor 1 no segundo argumento, a função funciona exatamente igual a função MÁXIMO()

FATORIAL()

Calcula o valor fatorial de um número.

Exemplo:



=FATORIAL(A5)

A5 – Valor para calcular fatorial

Esta função é simples e direta, basicamente se escolhe uma célula para retornar seu valor em fatorial.

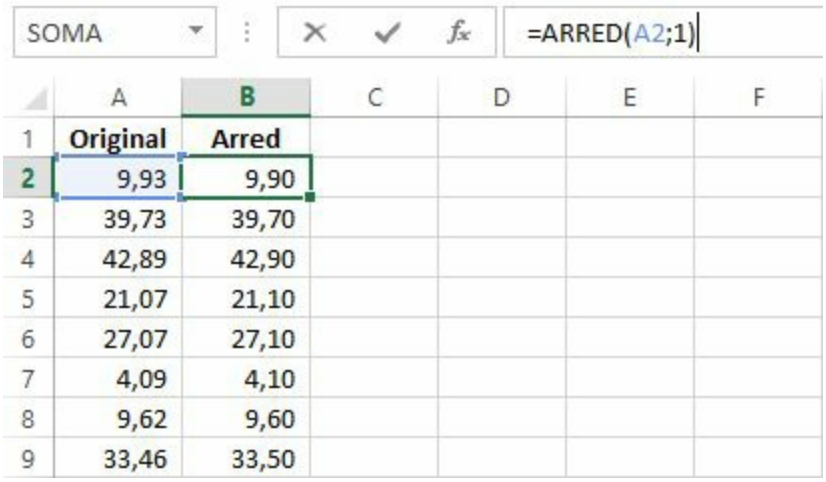
ARREDONADAMENTOS

Existem três funções para realizar o arredondamento, a primeira a ser apresentada basicamente utiliza critérios matemáticos comuns para realizar o arredondamento correto. As outras duas são arredondamentos desejados para cima ou para baixo.

ARRED()

Esta função é utilizada para arrendar os números de acordo com uma quantidade de casas decimais desejadas e seguindo critérios matemáticos.

Exemplo:



	A	B	C	D	E	F
1	Original	Arred				
2	9,93	9,90				
3	39,73	39,70				
4	42,89	42,90				
5	21,07	21,10				
6	27,07	27,10				
7	4,09	4,10				
8	9,62	9,60				
9	33,46	33,50				

=ARRED(A2;1)

A2 – Célula que será arredondada

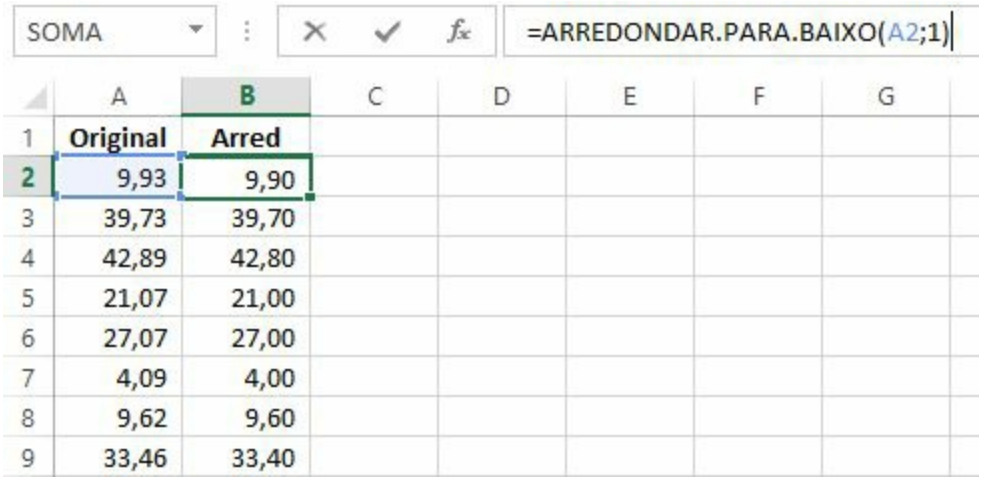
1 – Quantidade de casas decimais desejadas

A função realizará o arredondamento de acordo as casas decimais desejadas, desta forma, ao informar o número um, significa que após a aplicação da função o valor retornado será o da célula A2, porém com apenas **1** casas decimal. Caso fosse informado o argumento 0, o valor de retorno seria 10, a célula A2.

ARREDONDAR.PARA.BAIXO()

Esta função é utilizada para arrendar os números de acordo com uma quantidade de casas decimais desejadas e seguindo critério de arredondar o valor para baixo.

Exemplo:



	A	B	C	D	E	F	G
1	Original	Arred					
2	9,93	9,90					
3	39,73	39,70					
4	42,89	42,80					
5	21,07	21,00					
6	27,07	27,00					
7	4,09	4,00					
8	9,62	9,60					
9	33,46	33,40					

=ARREDONDAR.PARA.BAIXO(A2;1)

A2 – Célula que será arredondada

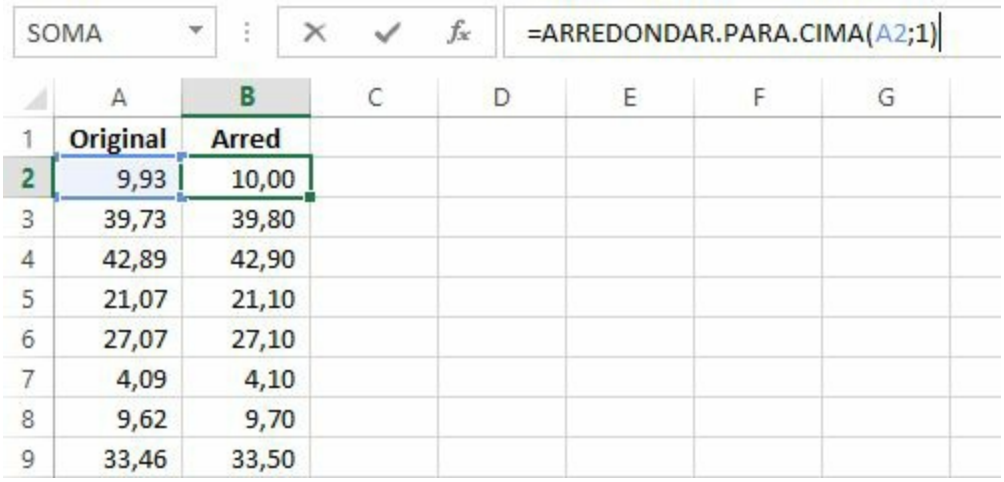
1 – Quantidade de casas decimais desejadas

A função realizará o arredondamento para baixo de acordo as casas decimais desejadas, desta forma, ao informar o número um, significa que após a aplicação da função o valor retornado será o da célula A2, porém com apenas **1** casas decimal.

ARREDONDAR.PARA.CIMA()

Esta função é utilizada para arrendar os números de acordo com uma quantidade de casas decimais desejadas e seguindo critério de arredondar o valor para cima.

Exemplo:



	A	B	C	D	E	F	G
1	Original	Arred					
2	9,93	10,00					
3	39,73	39,80					
4	42,89	42,90					
5	21,07	21,10					
6	27,07	27,10					
7	4,09	4,10					
8	9,62	9,70					
9	33,46	33,50					

=ARREDONDAR.PARA.CIMA(A2;1)

A2 – Célula que será arredondada

1 – Quantidade de casas decimais desejadas

A função realizará o arredondamento para cima de acordo as casas decimais desejadas, desta forma, ao informar o número um, significa que após a aplicação da função o valor retornado será o da célula A2, porém com apenas **1** casas decimal.

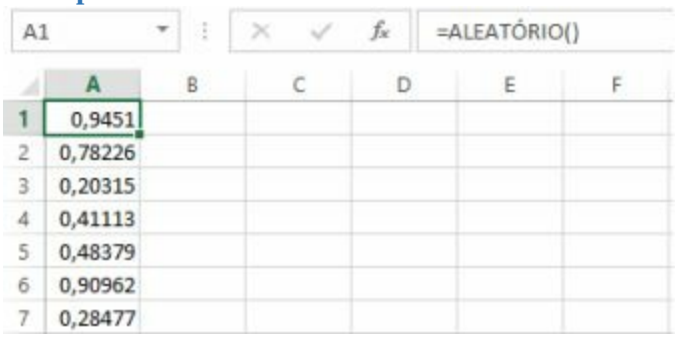
NÚMEROS ALEATÓRIOS

Números aleatórios são algoritmos complexos dentro da programação, gerar valores com completa distinção na frequência de aparições definitivamente não é uma tarefa simples, o Excel por sua vez, oferece de maneira extremamente simples a geração de valores aleatórios, para tanto basta utilizar as funções que serão apresentadas a seguir.

ALEATÓRIO()

Esta função basicamente tem por objetivo retornar valores aleatórios entre 0 e 1. Ela pode ser utilizada em combinação com outras funções, como um multiplicador, ou valor de soma, a sua utilidade depende de cada caso.

Exemplo:



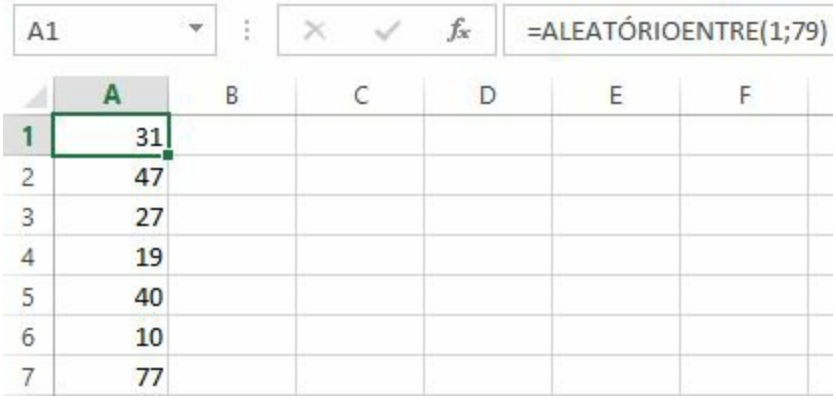
=ALEATÓRIO()

Esta função não possui argumentos, sua aplicação é direta.

ALEATÓRIOENTRE()

Esta função também tem por objetivo gerar números aleatórios, porém como seu próprio nome diz, ela gera os valores entre dois intervalos definidos pelo usuário. Um mínimo e um máximo.

Exemplo:



=ALEATÓRIOENTRE(**1;79**)

1 – Valor mínimo do intervalo

79 – Valor máximo do intervalo

Os números aleatórios são gerados entre o valor mínimo e o máximo informados pelo usuário. É importante ressaltar que os valores serão atualizados todas as vezes que o Excel realizar um cálculo. Para fixar os valores, é preciso copiar e colar como valor, para retirar a aplicação da função.

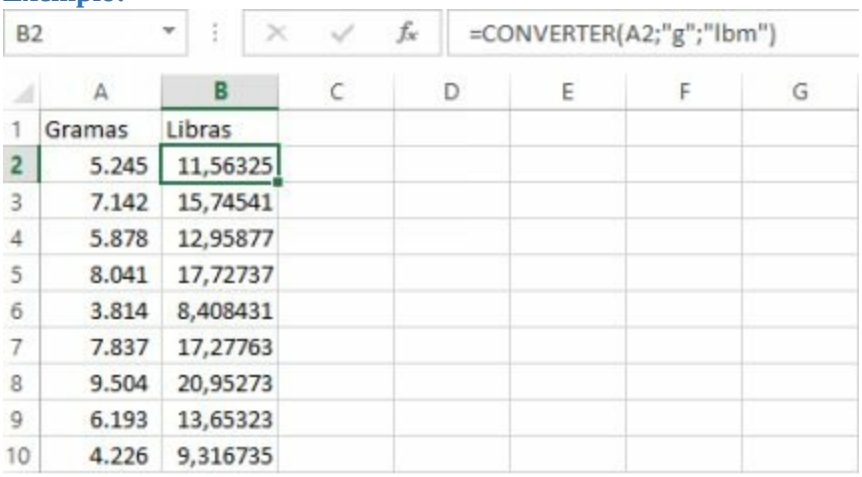
CONVERSÃO

O Excel oferece inúmeras funções que realizam a conversão de unidades ou bases numeras, as principais funções do gênero serão apresentadas a seguir.

CONVERTER()

Esta função é extremamente útil, porém pouco conhecida dentro do Excel, a plataforma oferece a conversão de diversas unidades de medida, como gramas, libras, toneladas, newton, btu e etc. Para realizar a conversão é rápido e fácil, onde a própria ajuda suspensa do Excel sugere as unidades para conversão.

Exemplo:



	A	B	C	D	E	F	G
1	Gramas	Libras					
2	5.245	11,56325					
3	7.142	15,74541					
4	5.878	12,95877					
5	8.041	17,72737					
6	3.814	8,408431					
7	7.837	17,27763					
8	9.504	20,95273					
9	6.193	13,65323					
10	4.226	9,316735					

=CONVERTER(A2;"**g**";"**lbm**")

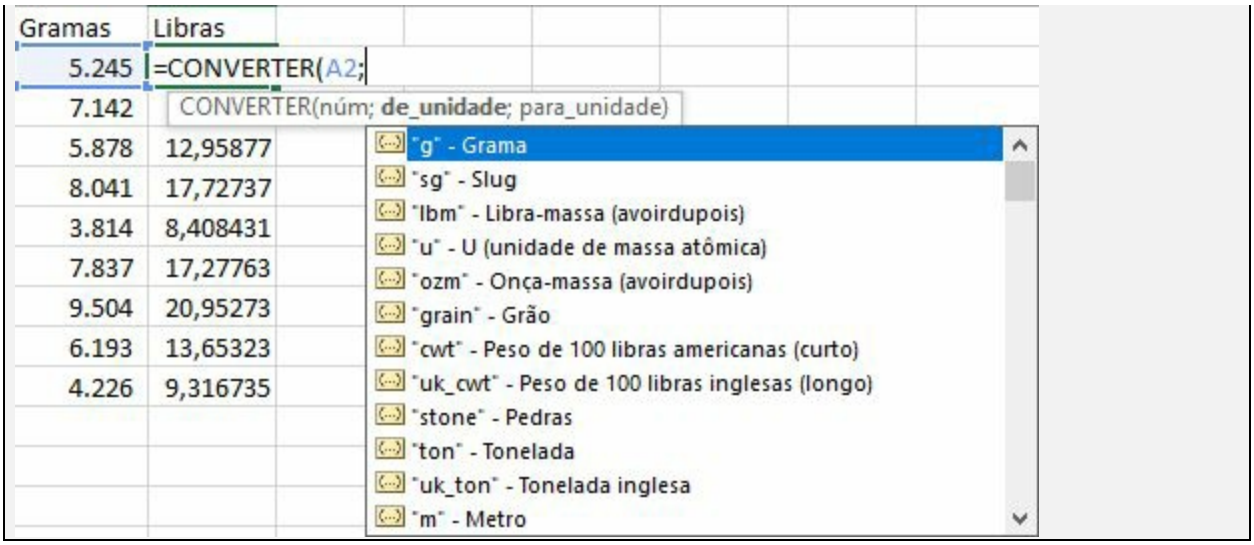
A2 – Célula que será convertida para outra unidade

"**g**" – Unidade grama origem da conversão

"**lbm**" – Unidade libra destino da conversão

Como citado anteriormente, a conversão é rápida e simples, basicamente se opta pela célula que será convertida, no exemplo A2, escolhe a sua unidade de origem ("**g**") e a sua unidade de destino após a conversão ("**lbm**").

Obs: A ajuda suspensa do Excel informa todas as unidades disponíveis para conversão, basta iniciar a digitação da fórmula.


Gramas
Libras
5.245
=CONVERTER(A2;
7.142
CONVERTER(núm; de_unidade; para_unidade)
5.878
12,95877
8.041
17,72737
3.814
8,408431
7.837
17,27763
9.504
20,95273
6.193
13,65323
4.226
9,316735
"g" - Grama
"sg" - Slug
"lbm" - Libra-massa (avoirdupois)
"u" - U (unidade de massa atômica)
"ozm" - Onça-massa (avoirdupois)
"grain" - Grão
"cwt" - Peso de 100 libras americanas (curto)
"uk_cwt" - Peso de 100 libras inglesas (longo)
"stone" - Pedras
"ton" - Tonelada
"uk_ton" - Tonelada inglesa
"m" - Metro

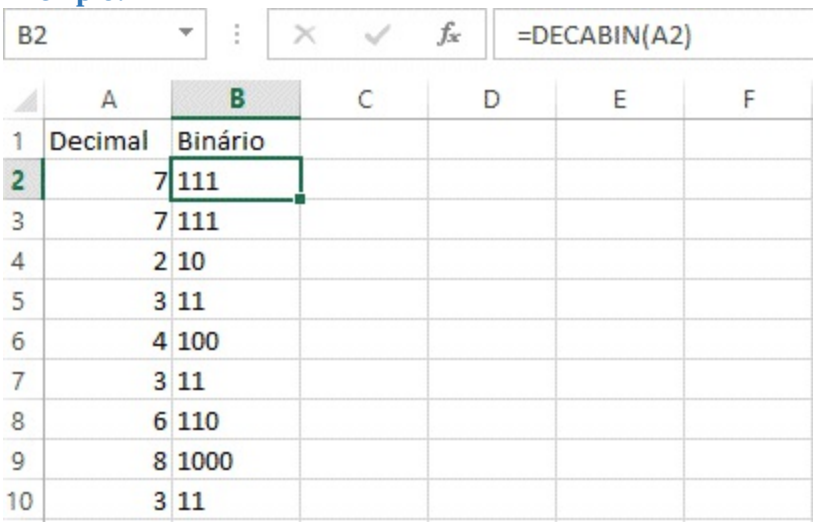
MUDAR BASE NUMÉRICA

As funções para alterar as bases numéricas no Excel são simples e diretas, o único argumento da formula é o valor desejado ou célula que será convertida. Existe uma função para cada tipo de conversão diferente.

Lista de formulas para alterar bases decimais:

=BINADEC()	Converter números binários para decimal
=BINADEC()	Converter números binários para hexadecimal
=BINAOCT()	Converter números binários para octal
=DECABIN()	Converter números decimais para binário
=DECAHEX()	Converter números decimais para hexadecimal
=DECAOCT()	Converter números decimais para octal
=HEXABIN()	Converter números hexadecimais para binário
=HEXADEC()	Converter números hexadecimais para decimal
=OCTABIN()	Converter números octais para binário
=OCTADEC()	Converter números octais para decimais
=OCTAHEX()	Converter números octais para hexadecimal

Exemplo:



	A	B	C	D	E	F
1	Decimal	Binário				
2	7	111				
3	7	111				
4	2	10				
5	3	11				
6	4	100				
7	3	11				
8	6	110				
9	8	1000				
10	3	11				

=DECABIN(A2)

A2 – Célula que será convertida para binário

A função basicamente consiste em escolher qual a célula será convertida, existe um segundo argumento opcional para escolher quantas casas decimais serão utilizadas, porém não foi aplicável neste exemplo.

Obs: Todas as demais conversões de bases funcionam da mesma maneira, basta escolher a função desejada e aplicar na célula escolhida.

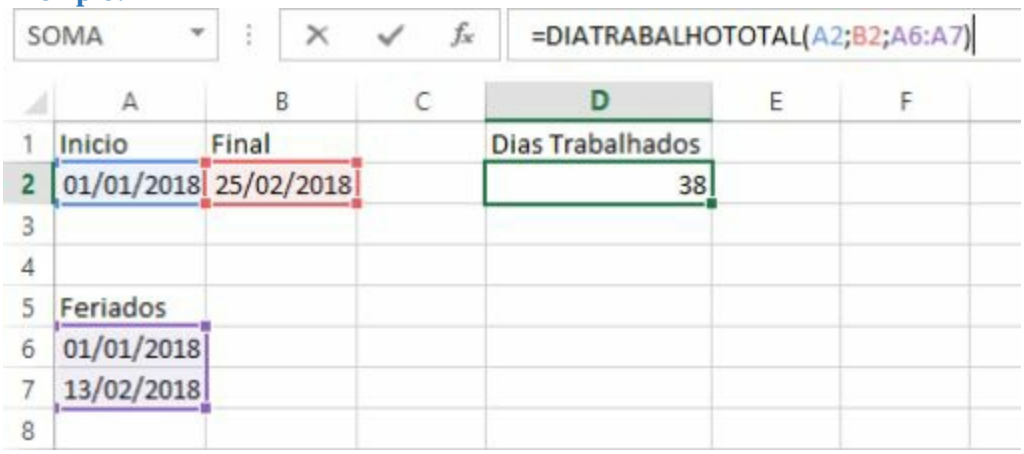
DATA E HORA

Funções relacionadas a Data e Hora, são extremamente versáteis e têm as mais variadas aplicações como por exemplo: Cálculos de banco de hora, cálculos de escala, cálculos de tempo de projeto e etc.

DIASTRABALHOTOTAL()

Esta função basicamente calcula a quantidade de dias da semana entre duas datas informadas. Existe também a possibilidade de informar feriados para serem descontados do cálculo.

Exemplo:



=DIATRABALHOTOTAL(**A2**;**B2**;**A6:A7**)

A2 - Data inicial

B2 - Data final

A6:A7 – Seleção de feriados

A fórmula é simples, consiste em escolher uma data de início, no exemplo **A2**, uma data de término que no exemplo é **B2** e o terceiro argumento é opcional, onde pode-se inserir uma seleção de células de feriados, os quais serão descontados, casos estejam dentro do intervalo solicitado, neste exemplo a região **A6:A7**.

FUNÇÕES DE HORÁRIO

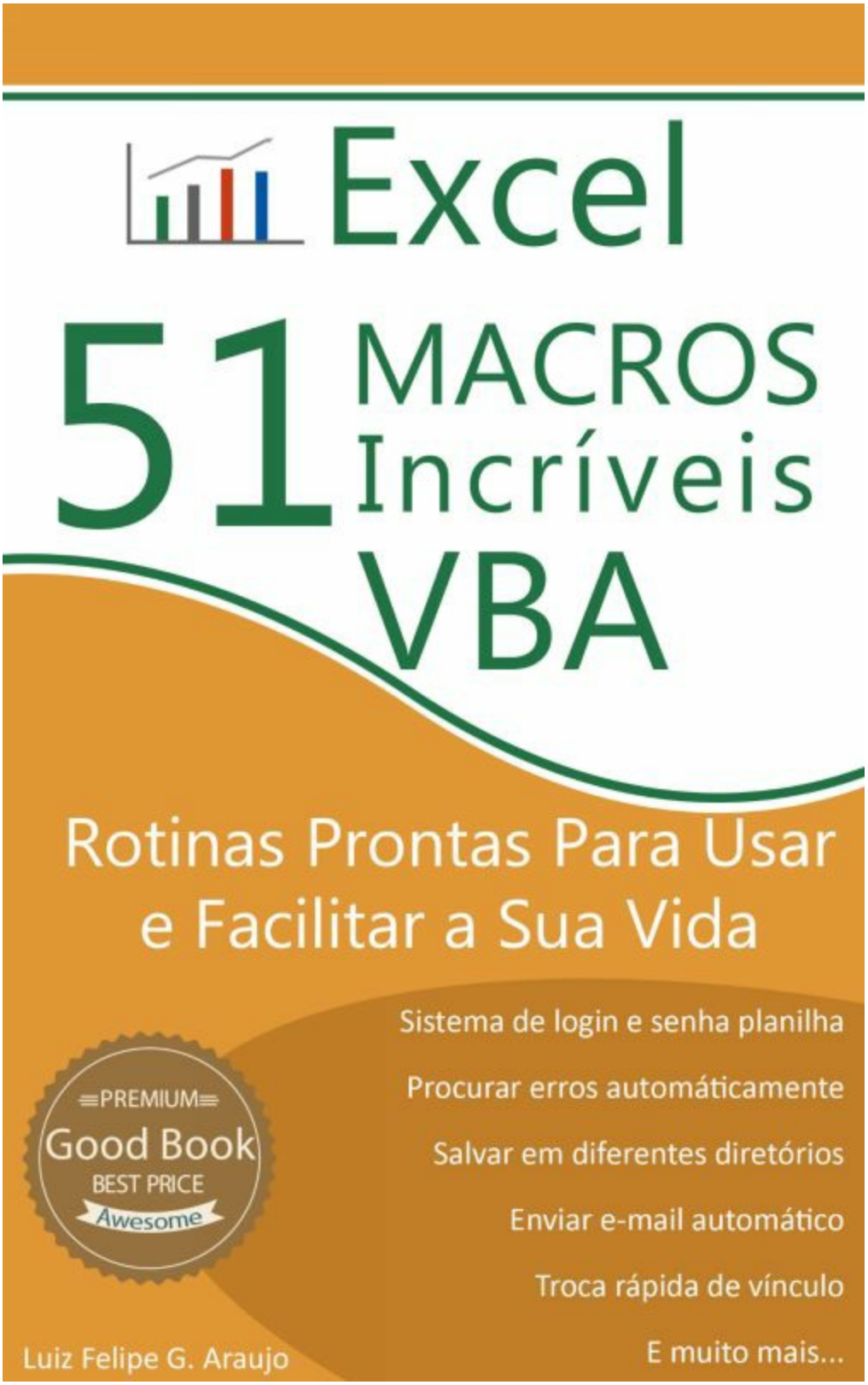
As três primeiras que serão apresentadas são extremamente simples e diretas, basicamente em uma célula que contém horário, pode-se extrair apenas o seu valor de hora, minuto ou segundo, usando suas respectivas fórmulas. A última por sua vez apenas retorna a informação DO horário atual do sistema operacional.

	A	B	C
1	Exemplo	Formula	Retorno
2	04:34:20	=HORA()	4
3	04:34:20	=MINUTO()	34
4	04:34:20	=SEGUNDOS()	20
5	-	=AGORA()	00:56:02

FUNÇÕES DE DATA

As três primeiras que serão apresentadas são extremamente simples e diretas, basicamente em uma célula que contém data, pode-se extrair apenas o seu valor do ano, mês ou dia, usando suas respectivas fórmulas. A última por sua vez apenas retorna a informação da data atual do sistema operacional.

	A	B	C
1	Exemplo	Formula	Retorno
2	05/02/2018	=ANO()	2018
3	05/02/2018	=MÊS()	2
4	05/02/2018	=ANO()	5
5	-	=HOJE()	29/12/2017


Excel
51 MACROS Incríveis VBA
Rotinas Prontas Para Usar e Facilitar a Sua Vida
Sistema de login e senha planilha
Procurar erros automáticamente
Salvar em diferentes diretórios
Enviar e-mail automático
Troca rápida de vínculo
E muito mais...
PREMIUM
Good Book
BEST PRICE
Awesome
Luiz Felipe G. Araujo

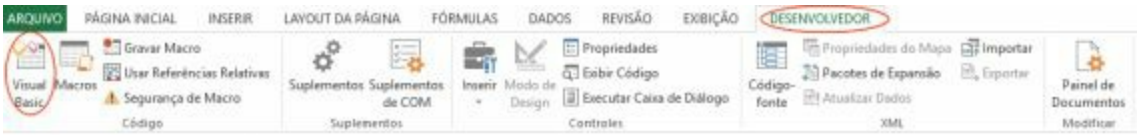
INTRODUÇÃO

Este livro é uma solução rápida para usuários que trabalham com planejamento, finanças, comercial, logística e todas as demais áreas que lidam com grandes quantidades de informações. É importante analisar as necessidades de cada projeto, processo ou sistema, para aplicar corretamente a rotinas desejadas.

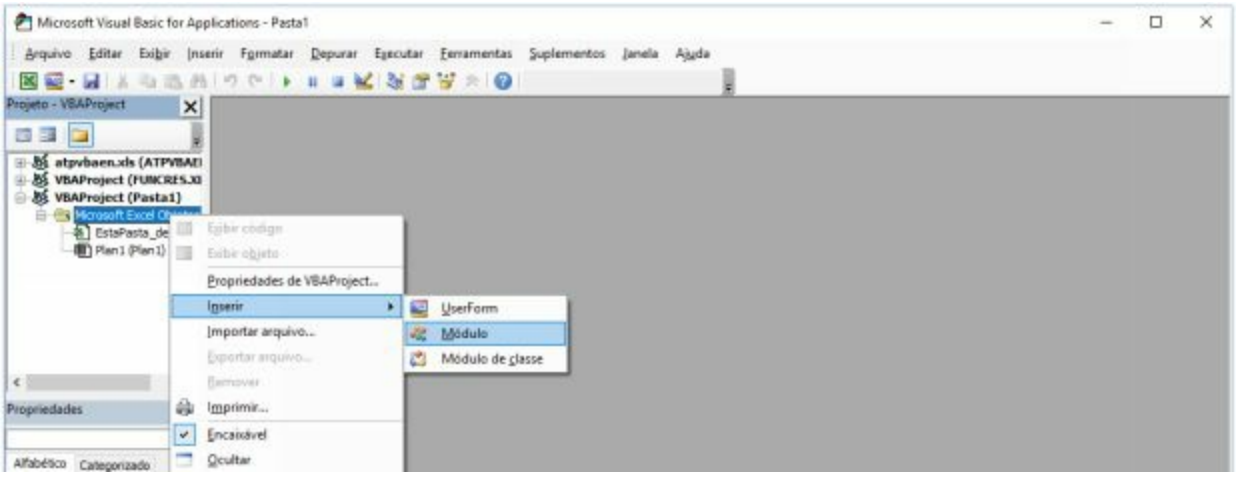
Esta breve introdução se destina aos usuários que não sabem o procedimento inicial básico de implementação de uma macro, serão instruções rápidas e diretas para demonstrar a implementação básica dos códigos.

São apenas 3 passos para adicionar suas macros:

1. Acessar o menu desenvolvedor, clicar no botão Visual Basic (Ou atalho Alt + F11).

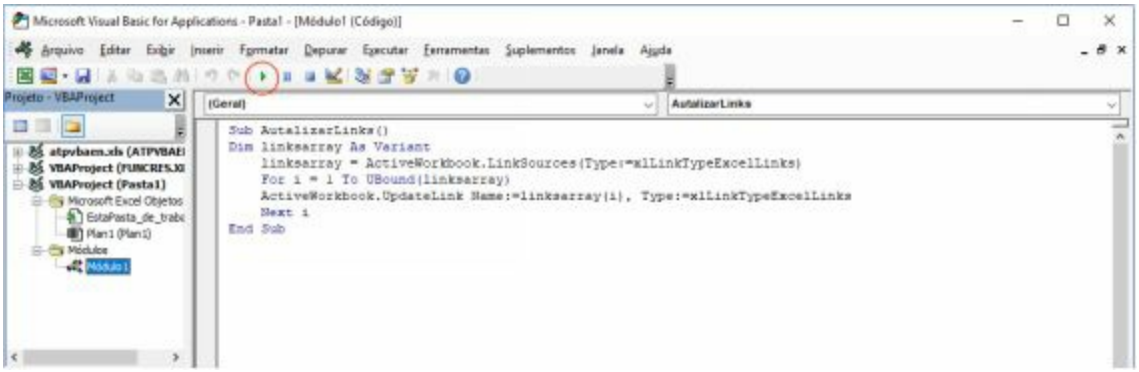


2. Escolher o objeto no qual a macro será inserida, de maneira prática, na grande maioria dos casos, poderá se adicionar as macros em um novo módulo, clicando com botão direito na pasta dos objetos, Inserir, Módulo.



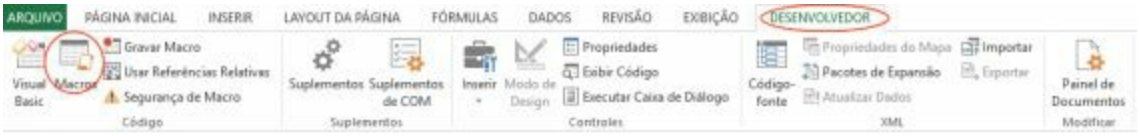
3. Inserir a macro desejada na caixa de texto e testar seu funcionamento clicando no botão “Play” ou pressionando F5. É possível também executar

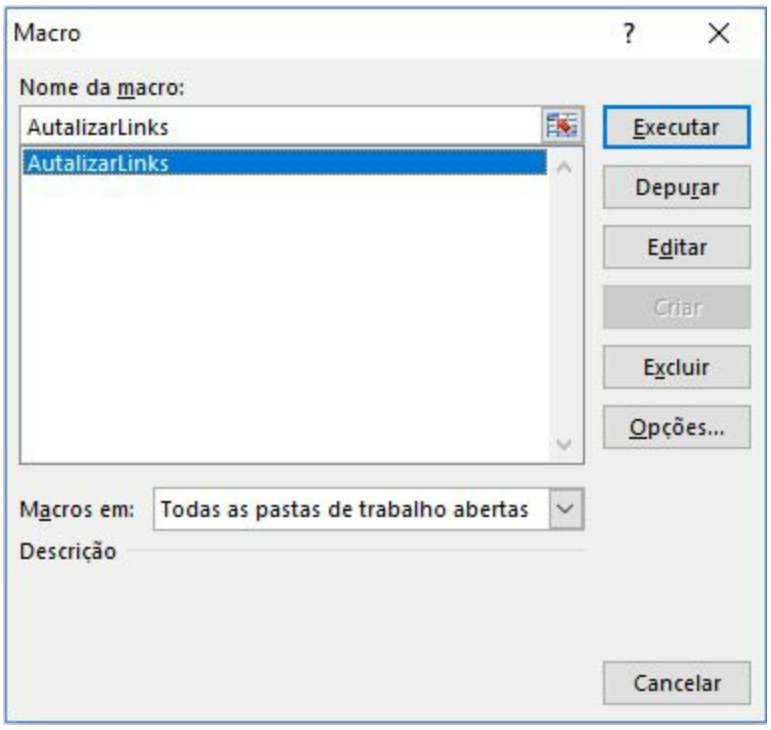
um “Passo a Passo”, da macro, para verificar o que cada linha de código executa, com o atalho F8, para depuração.



Obs 1: Para algumas macros, será necessário inserir as macros no objeto “EstaPasta” e em outros casos, na planilha desejada, por exemplo: “Plan1”.

Obs 2: Uma vez que a macro já esteja inserida, a janela do Visual Basic, pode ser fechada, a mesma poderá ser acessada de maneira mais prática, através da Guia Desenvolvedor, botão Macros.

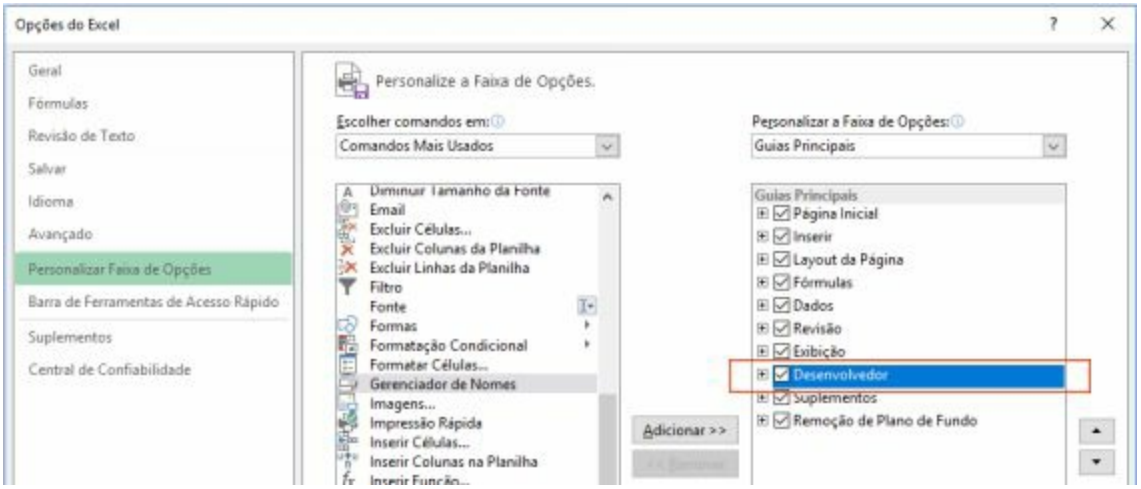




Obs 3: Caso a guia “Desenvolvedor” não esteja disponível, a mesma poderá ser adicionada da seguinte maneira:

Arquivo – Opções – Personalizar Faixa de Opções

Na seleção, basta marcar a opção desenvolvedor.



ABRIR SALVAR E FECHAR

1. Abrir um arquivo, caso já esteja aberto, maximiza-lo.

Uma macro muito utilizada para acelerar processos é a de abrir arquivos, porém por muitas vezes o arquivo já está aberto por alguma verificação manual do usuário, esta macro faz uma verificação, caso o arquivo já esteja aberto, ele é maximizado.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub AbrirMaximizar()
Dim Arquivo As String, Diretorio As String, Extensao As String
Diretorio = "C:\Users\usuarioteste\Documents"
Arquivo = "Planilha1"
Extensao = "xlsx"
Dim wkb As Workbook
On Error Resume Next
Set wkb = application.Workbooks(Arquivo)
If Err <> 0 Then
Set wkb = application.Workbooks.Open(Diretorio + "\" + Arquivo + "." &
Extensao, UpdateLinks:=0)
End If
On Error GoTo 0
Workbooks(Arquivo).Activate
End Sub
```

2. Salvar automaticamente antes de fechar

Esta macro é extremamente simples, porém muito útil, com esta rotina o Excel sempre salvará o arquivo quando for fechado, a rotina é útil para agilizar o processo de salvar e também previne o fechamento acidental sem salvar.

Obs: É necessário incluir a macro dentro do objeto WorkBook.

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
```

```
Application.DisplayAlerts = False
ActiveWorkbook.Save
Application.DisplayAlerts = True
End Sub
```

3. Copiar uma aba, converter para valor e salvar saída

Uma rotina muito comum e de grande utilidade é a de “gerar saída” de dados. Visto que muitos relatórios tem apenas uma aba de informação a ser divulgada, desta forma, esta aba será copiada, terá seus dados convertidos em valor e será salvada em uma pasta destino escolhida.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub CopiarSaida()
Dim Caminho As String, NomeArquivo As String, AbaSaida As String
AbaSaida = "Plan3"
Caminho = "C:\Users\UsuarioTeste\Documents\"
NomeArquivo = "Planilha.xls"
Worksheets("Plan3").Copy
Cells.Copy
Cells.PasteSpecial Paste:=xlPasteValues
ActiveWorkbook.SaveAs Filename:=Caminho & NomeArquivo, _
FileFormat:=xlOpenXMLWorkbookMacroEnabled
End Sub
```

4. Salvar backup rápido

Esta macro tem o objetivo de salvar rapidamente uma cópia de segurança do arquivo, na mesma pasta de origem com data e hora.

```
Sub SalvarBackUp()
ThisWorkbook.SaveCopyAs Filename:=ThisWorkbook.Path & _
"\" & Format(Date, "mm-dd-yy") & " " & _
ThisWorkbook.Name
End Sub
```

5. Salvar uma cópia backup ao fechar o arquivo

Esta macro, automaticamente salva uma cópia do arquivo antes de fechar. Esta cópia recebe o horário e data do momento em que o arquivo foi salvo.

Obs: É necessário incluir a macro dentro do objeto WorkBook.

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)
Application.DisplayAlerts = False
FileDefaultName = "TestFile"
Application.DisplayAlerts = False
ActiveWorkbook.SaveAs Filename:=Application.ActiveWorkbook.Path &
"\" & Format(Time, "hhmmss") & " " & Format(Date, "mm-dd-yy") & " " &
FileDefaultName
Application.DisplayAlerts = True
End Sub
```

6. Salvar cada planilha como arquivo Excel

Esta rotina tem por objetivo salvar cada aba como um arquivo diferente em um único destino.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub SalvarAbas()
Dim wkb, NomeArquivo As String
wkb = ActiveWorkbook.Name
Diretorio = "C:\Users\usuarioteste\Documents"
For i = 1 To Worksheets.Count
Worksheets(i).Select
NomeArquivo = ActiveSheet.Name
ActiveSheet.Copy
ActiveWorkbook.SaveAs Filename:=Diretorio + "\" + NomeArquivo + "." &
"xlsx"
ActiveWorkbook.Close
Workbooks(wkb).Activate
Next
MsgBox ("Arquivos salvados com sucesso")
```

```
End Sub
```

7. Salvar cada aba como um arquivo PDF

Esta rotina tem por objetivo salvar cada aba como um arquivo diferente em um único destino no formato PDF.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub SaveWorkshetAsPDF()
Dim ws As Worksheet
Diretorio = "C:\Users\usuarioteste\Documents"
For Each ws In Worksheets
On Error Resume Next
ws.ExportAsFixedFormat xlTypePDF, Diretorio & "\" & ws.Name & ".pdf"
Next ws
End Sub
```

8. Salvar o arquivo em diferentes pastas

É comum distribuir o arquivo na rede em diversos locais, para tanto, esta macro permite ao usuário salvar o arquivo em diversas pastas de maneira rápida e prática.

Obs1: É preciso alterar o nome do arquivo, a extensão e o diretório, marcados em negrito.
Obs2: Para alterar, apagar ou incluir novos diretórios, basta manipular os já exemplificados abaixo. Para adicionar, basta criar Diretorio(4), Diretorio(5) e assim seucessivamente.

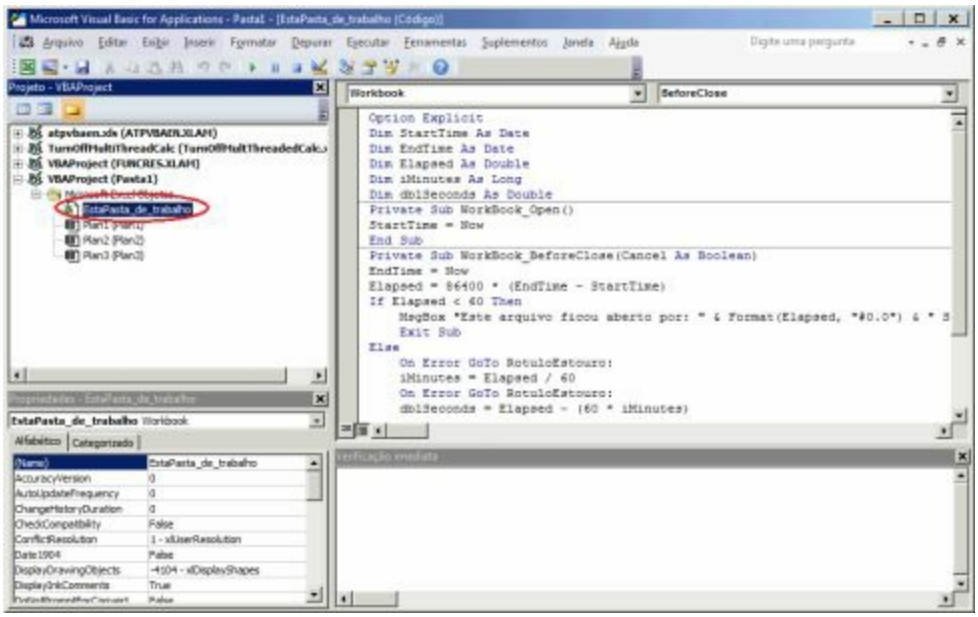
```
Sub SalvarArquivo()
Dim NomeArquivo, Extensao As String
Dim Diretorio(1 To 999) As String
NomeArquivo = "NomeDoArquivo"
Extensao = "xls"
Diretorio(1) = "C:\Users\usuarioteste\Documents"
Diretorio(2) = "C:\Users\usuarioteste\Documents"
```

```
Diretorio(3) = "C:\Users\usuarioteste\Documents"
For i = 1 To 999
    If Diretorio(i) = "" Then
        GoTo RotuloSaida:
    End If
    Perg = MsgBox("Deseja salvar com o nome: " & NomeArquivo & " no
caminho: " & Diretorio(i), vbYesNo, "Salvar Arquivo")
    If Perg = vbYes Then
        ActiveWorkbook.SaveAs Filename:=Diretorio(i) + "\" + NomeArquivo
+ "." & Extensao
    End If
Next i
RotuloSaida:
MsgBox "Arquivos salvados com sucesso"
End Sub
```

). Quando fechar o arquivo informar o tempo gasto

Esta é uma macro muito comum para o controle de rotina do usuário, onde o mesmo pode verificar o seu rendimento durante a utilização de uma planilha.

Obs1: É necessário incluir a macro dentro do objeto WorkBook.



Option Explicit

```
Dim StartTime As Date
Dim EndTime As Date
Dim Elapsed As Double
Dim iMinutes As Long
Dim dblSeconds As Double
Private Sub WorkBook_Open()
StartTime = Now
End Sub
Private Sub WorkBook_BeforeClose(Cancel As Boolean)
EndTime = Now
Elapsed = 86400 * (EndTime - StartTime)
If Elapsed < 60 Then
   MsgBox "Este arquivo ficou aberto por: " & Format(Elapsed, "#0.0") & "
Segundos", vbInformation + vbOKOnly
   Exit Sub
Else
   On Error GoTo RotuloEstouro:
   iMinutes = Elapsed / 60
   On Error GoTo RotuloEstouro:
   dblSeconds = Elapsed - (60 * iMinutes)
   MsgBox "Este arquivo ficou aberto por: " & Format(iMinutes, "#") & ":"
& Format(dblSeconds, "00") & " Minutos", vbInformation + vbOKOnly
   Exit Sub
End If
RotuloEstouro:
MsgBox "O tempo será computado a partir do próximo acesso"
End Sub
```

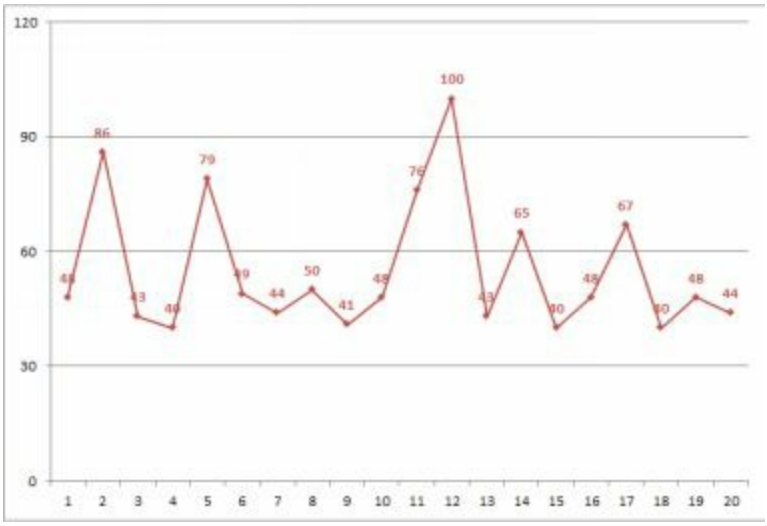
INTERAÇÃO COM GRÁFICOS

10. Automaticamente ajustar os rótulos dos gráficos

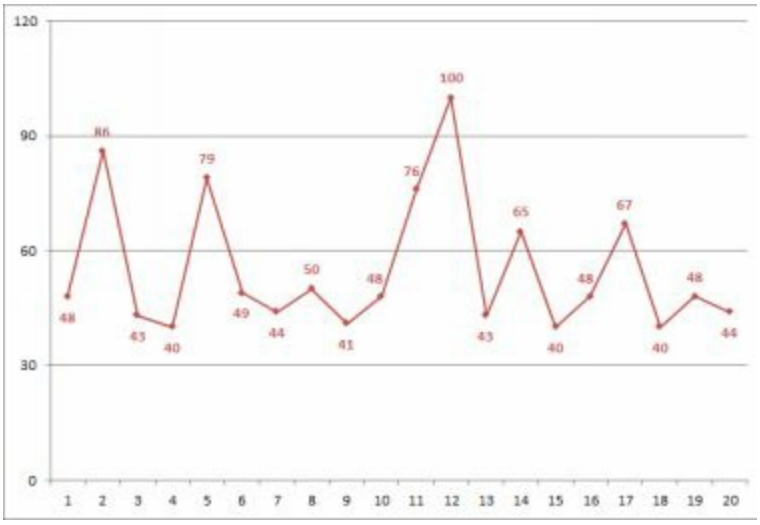
Este código automaticamente ajusta os rótulos, considerando o crescimento ou decrescimento do gráfico, evitando desta maneira a colisão do rótulo com o gráfico, conforme as imagens abaixo:

Obs: O código ajustará todos os gráficos da planilha ativa

Before:



After:



```
Sub AjustGraphic()
For i = 1 To ActiveSheet.ChartObjects.Count
ActiveSheet.ChartObjects(i).Select
Dim MaxScale, MinScale, MyPoint, DefaultPosition, AdjustedPosition As
Long
Dim MySeries As Series
Dim PointsArray As Variant
With ActiveChart
   MaxScale = .Axes(xlValue).MaximumScale
   MinScale = .Axes(xlValue).MinimumScale
   For Each MySeries In .SeriesCollection
      If MySeries.ChartType <> xlColumnClustered And _
         MySeries.ChartType <> xlLine And _
         MySeries.ChartType <> xlLineMarkers Then
         GoTo NEXTSERIES
      End If
         PointsArray = MySeries.Values
      For MyPoint = LBound(PointsArray) To UBound(PointsArray)
            If MySeries.Points(MyPoint).HasDataLabel = False Then
            GoTo NEXTDOT
            End If
            If MySeries.ChartType = xlColumnClustered Then
               MySeries.Points(MyPoint).DataLabel.Position =
xlLabelPositionOutsideEnd
               If PointsArray(MyPoint) > MaxScale * 0.9 Then
                  MySeries.Points(MyPoint).DataLabel.Position =
xlLabelPositionInsideEnd
               End If
            End If
            If MySeries.ChartType = xlLine Or MySeries.ChartType =
xlLineMarkers Then
               MySeries.Points(MyPoint).DataLabel.Position = xlBelow
               If MyPoint > 1 Then
               If PointsArray(MyPoint) > PointsArray(MyPoint - 1) Then
                  MySeries.Points(MyPoint).DataLabel.Position = xlAbove
```

```
                Else
                    MySeries.Points(MyPoint).DataLabel.Position = xlBelow
                End If
            End If
            If PointsArray(MyPoint) > MaxScale * 0.9 Or _
                PointsArray(MyPoint) < MinScale * 1.5 Then
                MySeries.Points(MyPoint).DataLabel.Position = xlRight
            End If
        End If
NEXTDOT:
    Next MyPoint
NEXTSERIES:
  Next MySeries
End With
Next
End Sub
```

11. Redimensionar todos os gráficos

Padronizar todos os gráficos de um arquivo pode ser uma tarefa árdua, esta macro facilita este processo ao estabelecer as mesmas dimensões, para tanto, basta modificar a largura e a altura destacados em negrito.

Obs 1: Substituir os valores de exemplo em negrito pelos valores desejados.
Obs 2: Widht será a largura e Height será a altura de todos os gráficos

```
Sub RedmGraf()
Dim i As Integer
For i = 1 To ActiveSheet.ChartObjects.Count
With ActiveSheet.ChartObjects(i)
.Width = 300
.Height = 200
End With
Next i
End Sub
```

VÍNCULOS COM ARQUIVOS EXTERNOS

12. Atualizar todos os links

Esta macro é essencial para usuários que lidam com grandes quantidades de informação e os vínculos não são atualizados automaticamente. Com esta rotina, todos serão atualizados automaticamente ao executa-la.

```
Sub AutalizarLinks()
Dim linksarray As Variant
   linksarray = ActiveWorkbook.LinkSources(Type:=xlLinkTypeExcelLinks)
   For i = 1 To UBound(linksarray)
   ActiveWorkbook.UpdateLink Name:=linksarray(i),
Type:=xlLinkTypeExcelLinks
   Next i
End Sub
```

13. Quebrar todos os vínculos

Esta rotina tem por objetivo quebrar todos os vínculos de um arquivo.

```
Sub QuebrarVinculos()
Dim alinksarray As Variant
   alinksarray =
ActiveWorkbook.LinkSources(Type:=xlLinkTypeExcelLinks)
   Do Until IsEmpty(alinksarray)
      ActiveWorkbook.BreakLink Name:=alinksarray(1),
Type:=xlLinkTypeExcelLinks
      alinksarray =
ActiveWorkbook.LinkSources(Type:=xlLinkTypeExcelLinks)
   Loop
End Sub
```

14. Alterar vínculo

Da mesma maneira que a macro anterior, a alteração de vínculos se dá por

um processo manual e lento, desta forma, através desta rotina, é possível executar esta tarefa de maneira rápida e automatizada.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados. Arquivo1 é o vínculo atual, Arquivo2 é o vínculo para o qual será alterado.

```
Sub TrocarVinculo()
Dim Arquivo1, Arquivo2 As String
Dim Cont As Integer
Dim wkb, wkb2 As String
wkb = ActiveWorkbook.Name
Arquivo1 = " C:\Users\usuarioteste\Documents\ Planilha1.xls"
Arquivo2 = " C:\Users\usuarioteste\Documents\ Planilha2.xls"
Application.DisplayAlerts = False
Workbooks.Open Filename:=Arquivo2, UpdateLinks:=0
wkb2 = ActiveWorkbook.Name
Workbooks(wkb).Activate
ActiveWorkbook.ChangeLink Name:=Arquivo1, NewName:=Arquivo2,
Type:=xlLinkTypeExcelLinks
Workbooks(wkb2).Activate
ActiveWorkbook.Close
Application.DisplayAlerts = False
Workbooks(wkb).Activate
End Sub
```

CONTROLE E SEGURANÇA DAS INFORMAÇÕES

15. Computar dados de usuários que acessaram a planilha

Para muitos setores, é essencial ter um controle dos usuários que acessaram e/ou modificaram os dados de uma planilha, desta forma, esta rotina computa o usuário, data e horário de acesso.

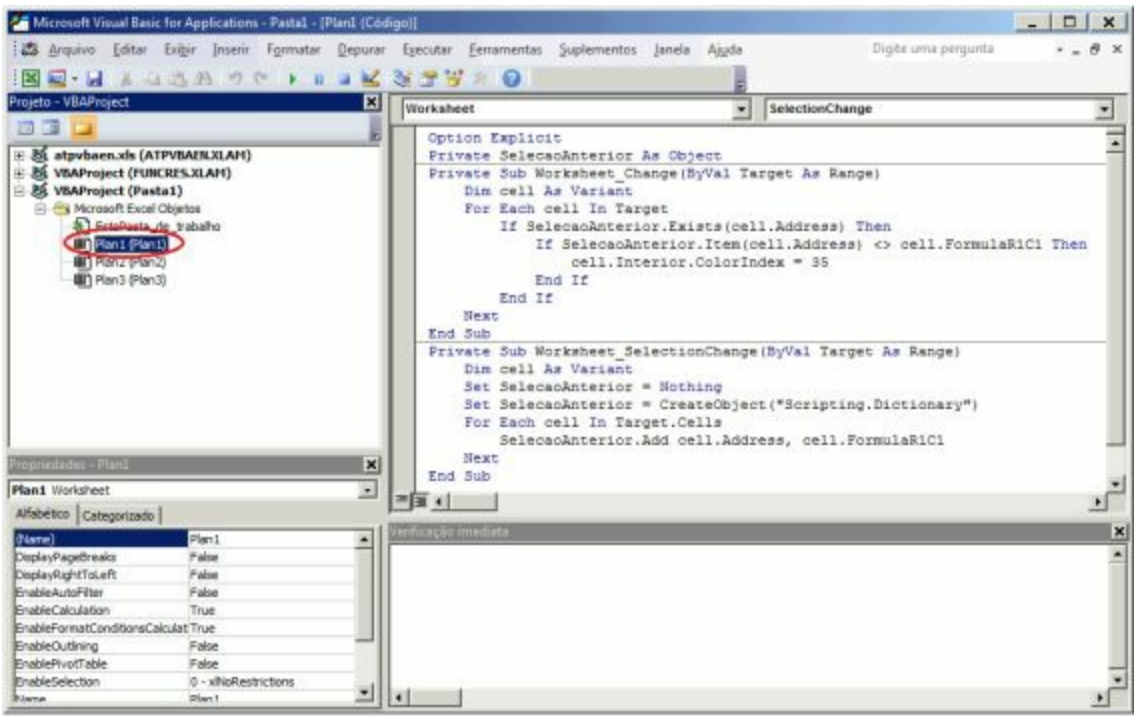
Obs: É preciso criar uma aba chamada "**Controle de Acesso**", onde os dados dos usuários serão armazenados.

```
Sub Auto_Open()
Dim linhas As Integer
Dim Data, Hora As Date
Data = Date
Hora = Time
linha = Application.WorksheetFunction.CountA(Worksheets("Controle de
Acesso").Range("A1", "A1048576")) + 1
Worksheets("Controle de Acesso").Range("A" & linha) =
Application.UserName
Worksheets("Controle de Acesso").Range("B" & linha) = Data
Worksheets("Controle de Acesso").Range("C" & linha) = Hora
Columns(1).AutoFit
Columns(2).AutoFit
Columns(3).AutoFit
End Sub
```

16. Marcar todas as células editadas pelo usuário

Esta é uma outra rotina que tem por objetivo, estabelecer um controle dos dados de uma planilha.

Obs1: É necessário incluir a macro dentro do objeto da planilha desejada, no exemplo abaixo a planilha que destacará as alterações será a "Plan1".




```
Option Explicit
Private SelecaoAnterior As Object
Private Sub Worksheet_Change(ByVal Target As Range)
   Dim cell As Variant
   For Each cell In Target
      If SelecaoAnterior.Exists(cell.Address) Then
         If SelecaoAnterior.Item(cell.Address) <> cell.FormulaR1C1 Then
            cell.Interior.ColorIndex = 35
         End If
      End If
```

17. Proteger todas as planilhas

Esta simples rotina, protege automaticamente todos os dados de todas as planilhas de possíveis alterações.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub ProtectSheets()
Dim wksht As Worksheet
For Each wksht  In ActiveWorkbook.Worksheets
```

```
wksht.Protect Password:="Senha"
Next wsheet
End Sub
```

18. Desproteger todas as planilhas

Este código é simplesmente o contrário do anterior, ele tem a função de desproteger todas as planilhas.

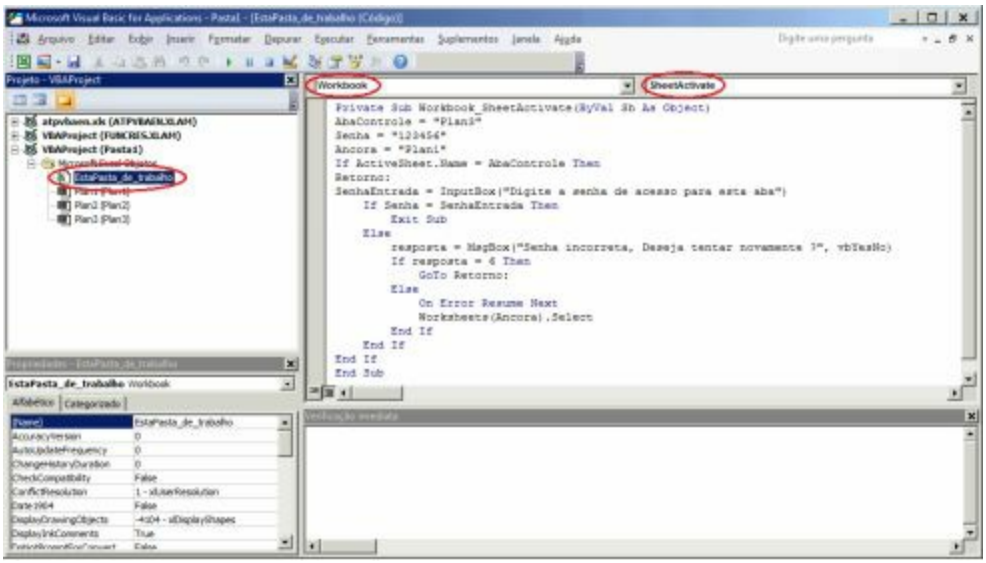
Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub UnrotectSheets()
Dim wksht As Worksheet
For Each wksht  In ActiveWorkbook.Worksheets
wksht.Unrotect Password:="Senha"
Next wsheet
End Sub
```

19. Proteger uma aba com senha

Macro de controle de acesso para acessar uma aba específica.

Obs1: A variável **AbaControle** é a que se deseja proteger com senha.
Obs2: **Senha** é o valor que deverá ser estabelecido como regra de acesso.
Obs2: **Ancora** é a aba que será selecionada, caso o usuário erre a senha.
Obs4: O código precisa ser inserido dentro do objeto Workbook para declaração “SheetActivate”, de modo a ser executado junto com a abertura do arquivo, conforme imagem abaixo:

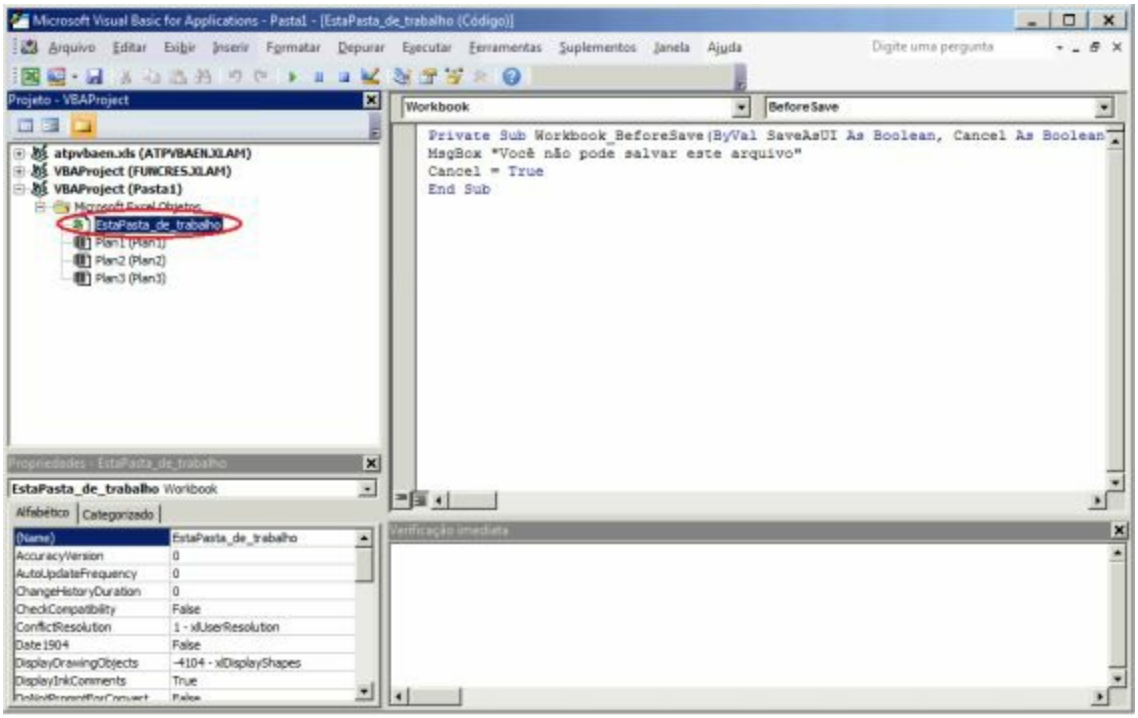



```
Private Sub Workbook_SheetActivate(ByVal Sh As Object)
AbaControle = "Plan3"
Senha = "123456"
Ancora = "Plan1"
If ActiveSheet.Name = AbaControle Then
Retorno:
SenhaEntrada = InputBox("Digite a senha de acesso para esta aba")
   If Senha = SenhaEntrada Then
      Exit Sub
   Else
      resposta = MsgBox("Senha incorreta, Deseja tentar novamente ?",
vbYesNo)
      If resposta = 6 Then
         GoTo Retorno:
      Else
         On Error Resume Next
         Worksheets(Ancora).Select
      End If
   End If
End If
End Sub
```

20. Impedir o usuário salvar o arquivo

Esta macro tem por objetivo impedir o usuário de salvar o arquivo, impedindo assim a substituição de dados.

Obs1: É necessário incluir a macro dentro do objeto WorkBook.



Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, Cancel As Boolean)
Msgbox “Você não pode salvar este arquivo”
Cancel = True
End Sub

21. Simples Sistema de login e senha para acessar o arquivo

Este é um sistema simples de Login para apenas usuários cadastrados acessarem a planilha.

Obs1: É preciso alterar o nome do arquivo, a extensão e o diretório, marcados em negrito.

Obs2: O código precisa da declaração Auto_Open() para funcionar.

Obs3: Para alterar e criar usuários com senha, basta alterar os trechos em negrito, para adicionar, basta incluir uma nova linha, seguindo a numeração, Usuario(4), Usuario(5) e assim sucessivamente. Para senha é a mesma lógica, Senha(4), Senha(5) e assim sucessivamente.

```
Sub Auto_Open()
Dim Usuario(1 To 999) As String
Dim Senha(1 To 999) As String
Dim UsuOK As Boolean
Dim SenhaOK As Boolean
'Cadastro de Usuários
'------------------
Usuario(1) = "Luiz"
Usuario(2) = "Maria"
Usuario(3) = "Clara"
'------------------
Senha(1) = "1345"
Senha(2) = "1234"
Senha(3) = "5367"
'------------------
UsuOK = False
SenhaOK = False
Retorno:
UsuarioEntrada = InputBox("Digite o seu nome de usuário: ")
For i = 1 To 999
   If Usuario(i) = UsuarioEntrada And Usuario(i) <> "" Then
   UsuOK = True
   GoTo Rotulo1:
   End If
Next
Rotulo1:
If UsuOK = True Then
   SenhaEntrada = InputBox("Bem Vindo(a): " & UsuarioEntrada & ", digite
a sua senha:")
   If Senha(i) = SenhaEntrada Then
   Exit Sub
```

```
Else
   pergunta = MsgBox("Senha incorreta, deseja tentar novamente ?",
vbYesNo)
   If pergunta = 6 Then
      GoTo Retorno:
   Else
      MsgBox "O Arquivo será fechado"
      ActiveWorkbook.Close
   End If
End If
Else
   pergunta = MsgBox("Usuário não encontrado, deseja tentar novamente ?",
vbYesNo)
   If pergunta = 6 Then
      GoTo Retorno:
   Else
      MsgBox "O Arquivo será fechado"
      ActiveWorkbook.Close
   End If
End If
End Sub
```

CONTROLE DE ERROS

22. Verificar todas as abas para encontrar erros

Esta macro tem por objetivo rodar todas as abas disponíveis de um arquivo, procurando por erros comuns de formula como #N/D, #REF!, #DIV/0! e etc. O erro será localizado e informado ao usuário por meio de mensagem, informando a Aba e a Célula onde o erro se encontra.

```
Sub VerificarErros()
Dim ws As Worksheet
Dim ra As Range
For Each ws In Worksheets
   For Each ra In ws.UsedRange
      ra.Select
      On Error Resume Next
      If IsError(ra.Value) Then
        MsgBox "Aba: " & ra.Parent.Name & Chr(13) & "Célula: " &
ra.Address
      End If
   Next
Next
End Sub
```

23. Verificar uma seleção para encontrar erros

Esta macro tem a mesma mecânica da anterior, porém ao invés de verificar todas as abas para encontrar erros, a verificação é feita nas células selecionadas pelo usuário.

```
Sub VerificarErros()
Dim ra As Range
For Each ra In Selection
   ra.Select
   If IsError(ra.Value) Then
```

```
      MsgBox "Aba: " & ra.Parent.Name & Chr(13) & "Célula: " & ra.Address
    End If
Next
End Sub
```

24. Verificar todas as abas para contabilizar erros

Esta macro tem por objetivo rodar todas as abas disponíveis de um arquivo, procurando por erros comuns de formula como #N/D, #REF!, #DIV/0! e etc. O erro será localizado e contabilizado para informar ao usuário a quantidade.

```
Sub VerificarErros()
Dim ws As Worksheet
Dim ra As Range
Dim Contador As Long
Contador = 0
For Each ws In Worksheets
    For Each ra In ws.UsedRange
        If IsError(ra.Value) Then
          Contador = Contador + 1
        End If
    Next
Next
MsgBox (Contador & " Erros encontrados")
End Sub
```

OCULTAR E MOSTRAR INFORMAÇÕES

25. Exibir todas as linhas e colunas ocultas

Ao aplicar esta macro, todas as linhas e colunas ocultas em todas as tabelas serão exibidas.

```
Sub TirarOculta()
Dim Ws As Worksheet
For Each Ws In Application.Worksheets
Ws.Select
Cells.Select
Selection.EntireRow.Hidden = False
Selection.EntireColumn.Hidden = False
Next
End Sub
```

26. Ocultar e mostrar todas as abas

Rotina utilizada para acelerar o processo de ocultar e mostrar abas. Todas as abas, exceto a primeira serão ocultas, ao rodar novamente, todas serão exibidas.

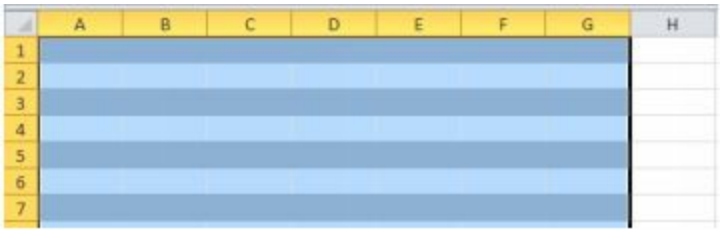
```
Sub OcultareMostrarAbas()
For i = 2 To Worksheets.Count
   If Worksheets(i).Visible = True Then
      Worksheets(i).Visible = False
   Else
      Worksheets(i).Visible = True
   End If
Next
End Sub
```

OUTROS

27. Aplicar cores alternadas em uma seleção

As cores intercaladas facilitam a leitura de dados, a única maneira de fazer isso no Excel sem macro, é aplicando a formatação de tabela, função nem sempre desejada. Esta macro faz esta função de forma simples e direta, com a cor desejada.

Obs: para trocar as cores basta trocar o código da cor destacado em negrito.




```
Sub CoresIntercaladas()
Dim Selecao As Range
Dim Linha As Range
Set Selecao = Selection
For Each Linha In Selecao.Rows
   If Linha.Row Mod 2 = 1 Then
      Linha.Interior.ColorIndex = 15
   End If
Next Linha
End Sub
```

28. Consolidar todos as abas na primeira

Um processo manual muito comum é a consolidação de dados, onde muitas vezes um arquivo bruto extraído do banco de dados, vem fragmentado em diversas abas. Esta rotina permite copiar dados de diversas abas e consolidar em uma primeira.

Obs1: A macro está configurada para agrupar os dados que começam a partir da célula A1.

Obs2: A primeira aba receberá as informações das demais.

```
Sub ConsolidarDados()
Dim total As Long
For i = 2 To Worksheets.Count
   Worksheets(i).Select
   Range("A1").Select
   If ActiveCell.Offset(0, 1) <> "" Then
      Range(Selection, Selection.End(xlToRight)).Select
   End If
   If ActiveCell.Offset(1, 0) <> "" Then
   Range(Selection, Selection.End(xlDown)).Select
   End If
Selection.Copy
Worksheets(1).Select
total =
Application.WorksheetFunction.CountA(Worksheets(1).Range("A1:A104857
Range("A" & total + 1).Select
ActiveSheet.Paste
Next
End Sub
```

29. Cronometrar o tempo de outras macros

Algumas macros tem processamento demorado, desta forma, é sempre recomendado cronometrar o tempo de execução para se fazer ajustes e melhorias. A função desta macro é cronometrar o tempo de execução de outras macros, desta forma, basta substituir o trecho em negrito destacado pelo código desejado.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub MacroCronometro()
'Início cronômetro
ti = Time
'Insira seu código
'Fim cronômetro
tf = Time
```

```
'Subtração para saber o tempo total de execução da macro
tDif = tf - ti
MsgBox "Tempo de processamento: " & WorksheetFunction.text(tDif,
"HH:MM:SS")
End Sub
```

30. Copar e colar valor em todas as abas

Macro utilizada comumente para gerar saída de dados para outros departamentos ou usuários.

```
Sub PassarValor()
pergunta = MsgBox("Esta macro converterá todas as células para valores,
deseja continuar ?", vbYesNo)
If pergunta = vbYes Then
   For i = 1 To Worksheets.Count
      Worksheets(i).Select
      If Worksheets(i).Visible = False Then
         Worksheets(i).Visible = True
      End If
      Cells.Select
      Selection.Copy
      Selection.PasteSpecial Paste:=xlPasteValues
      Range("A1").Select
   Next
End If
End Sub
```

31. Remover espaços em branco dentro das células

O objetivo desta macro é remover os espaços em branco dentro das células, para tanto, basta selecionar as células desejadas e rodar a macro.

```
Sub RemoverEspacos()
Dim Celula As Range
For Each Celula In Selection
If Not IsEmpty(Celula) Then
```

```
Celula = Trim(Celula)
End If
Next Celula
End Sub
```

32. Atualizar todas as tabelas dinâmicas

Esta macro é simples, mas eficaz, ela atualiza todas as tabelas dinâmicas de uma planilha.

```
Sub Refresh_PivotTables()
  Dim pivotTable As PivotTable
  For Each pivotTable In ActiveSheet.PivotTables
    pivotTable.RefreshTable
  Next
End Sub
```

33. Remover duplicatas em todas as abas

Esta é uma rotina para aplicar o tratamento de dados nas planilhas, permitindo ao usuário remover as duplicatas de todas as abas.

```
Sub RemoverDuplicatas()
coluna = InputBox("Em qual coluna deseja remover duplicata de todas as abas ?")
For i = 1 To Worksheets.Count
    Worksheets(i).Select
    ActiveSheet.Range("$" & coluna & "$1:$" & coluna & "$9999").RemoveDuplicates Columns:=1, Header:=xlNo
Next
End Sub
```

34. Consolidar dados na primeira aba

Esta macro tem por objetivo consolidar todas as abas dos arquivos abertos em

um novo arquivo.

```
Sub JuntarArquivos()
Dim wkbDestino As String
Dim WorkbookName(1 To 99) As String
Dim ws As Worksheet
Dim i As Integer
i = 1
For Each Workbook In Workbooks
i = 1 + 1
WorkbookName(i) = Workbook.Name
Next Workbook
Total = i
Workbooks.Add
wkbDestino = ActiveWorkbook.Name
For i = 1 To Total
   Workbooks(i).Activate
   For Each ws In Workbooks(i).Worksheets
   ws.Copy
after:=Workbooks(wkbDestino).Sheets(Workbooks(wkbDestino).Sheets.Cour
   Next ws
Next i
Application.DisplayAlerts = False
For i = 1 To 3
Sheets(1).Delete
Next i
Application.DisplayAlerts = True
End Sub
```

35. Deletar abas que estão vazias

Esta macro verifica todas as planilhas para encontrar alguma com todas as células em branco, uma vez que sejam encontradas, serão deletadas.

```
Sub DeletarAbasVazias()
Dim Ws As Worksheet
```

```
On Error Resume Next
Application.DisplayAlerts = False
For Each Ws In Application.Worksheets
If Application.WorksheetFunction.CountA(Ws.UsedRange) = 0 Then
Ws.Delete
End If
Next
Application.DisplayAlerts = True
End Sub
```

36. Ordenar as abas em ordem alfabética

Esta macro, ao ser executada ordena as abas em ordem alfabética crescente, para trocar para decrescente, basta alterar o sinal de maior (“>”), destacado em negrito por menor (“<”).

```
Sub Ordenarabas()
Dim i As Integer
Dim j As Integer
Dim Resposta As VbMsgBoxResult
Resposta = MsgBox("Deseja ordenar em ordem Crescente ?", vbYesNo +
vbQuestion + vbDefaultButton1, "Ordenar Abas")
For i = 1 To Sheets.Count
For j = 1 To Sheets.Count - 1
If Resposta = vbYes Then
If UCase$(Sheets(j).Name) > UCase$(Sheets(j + 1).Name) Then
Sheets(j).Move After:=Sheets(j + 1)
End If
End If
Next j
Next i
End Sub
```

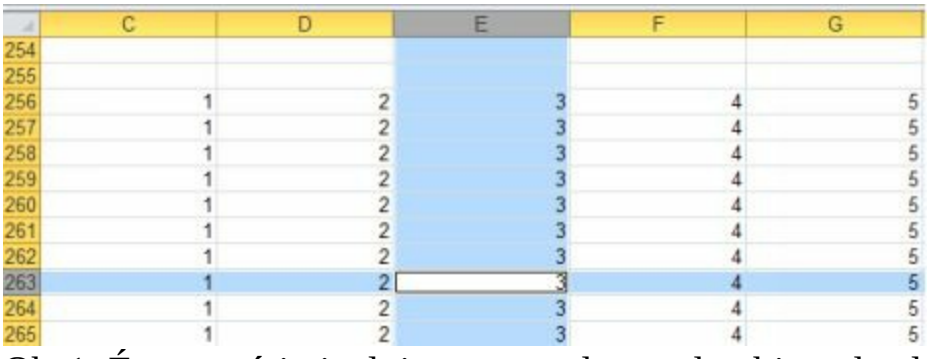
37. Trocar o nome de todas as abas

Esta macro tem o objetivo de alterar rapidamente o nome de todas as abas.

```
Sub TrocarNomeAba()
For Each Sheet In Worksheets
RotuloRetorno:
Sheet.Select
NovoNome = InputBox("Qual o novo nome para esta Aba ?")
If NovoNome = "" Then
   Exit Sub
End If
On Error GoTo RotuloRetorno:
ActiveSheet.Name = NovoNome
Next
End Sub
```

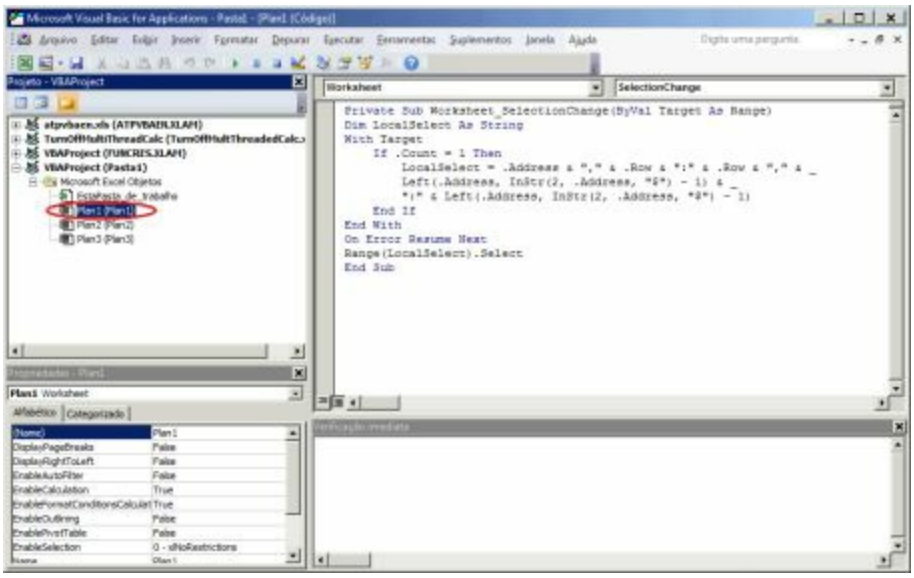
38. Destacar linha e coluna da célula selecionada

Esta macro é extremamente útil para usuários que precisam analisar grande quantidade de informações com frequência, a partir de uma célula selecionada, as colunas e linhas são destacadas, para facilitar a leitura de dados, conforme imagem abaixo:



	C	D	E	F	G
254					
255					
256	1	2	3	4	5
257	1	2	3	4	5
258	1	2	3	4	5
259	1	2	3	4	5
260	1	2	3	4	5
261	1	2	3	4	5
262	1	2	3	4	5
263	1	2	3	4	5
264	1	2	3	4	5
265	1	2	3	4	5

Obs1: É necessário incluir a macro dentro do objeto da planilha desejada, no exemplo abaixo a planilha que apresentará os destaques de linha e coluna será a “Plan1”.




```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)
Dim LocalSelect As String
With Target
   If .Count = 1 Then
      LocalSelect = .Address & "," & .row & ":" & .row & "," & _
      Left(.Address, InStr(2, .Address, "$") - 1) & _
      ":" & Left(.Address, InStr(2, .Address, "$") - 1)
   End If
End With
On Error Resume Next
Range(LocalSelect).Select
End Sub
```

INTERAÇÃO COM WINDOWS

39. Salvar uma seleção como imagem

Esta macro, salva automaticamente uma seleção de células como uma imagem, o arquivo é salvo na mesma pasta com o mesmo nome da planilha ativa.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub SelectedRangeToImage()
    Dim iFilename As String
    Dim TempObjChart As Chart
    Dim Shp As Shape
    Dim Wsht As Worksheet
    Dim fileSaveName As Variant, pic As Variant
    Set Wsht = ActiveSheet
    Selection.Copy
    Wsht.Pictures.Paste.Select
    Set Shp = Wsht.Shapes(Wsht.Shapes.Count)
    Set TempObjChart = Charts.Add
    TempObjChart.ChartArea.Clear
    TempObjChart.Name = "PicChart" & (Rnd() * 10000)
    Set TempObjChart =
TempObjChart.Location(Where:=xlLocationAsObject, Name:=Wsht.Name)
    TempObjChart.ChartArea.Width = Shp.Width
    TempObjChart.ChartArea.Height = Shp.Height
    TempObjChart.Parent.Border.LineStyle = 0
    Shp.Copy
    TempObjChart.ChartArea.Select
    TempObjChart.Paste
    iFilename = Application.ActiveWorkbook.Path & "\" & ActiveSheet.Name
& ".jpg"
    TempObjChart.Export Filename:=iFilename, FilterName:="jpg"
```

```
    Wsht.Cells(1, 1).Activate
    Wsht.ChartObjects(Wsht.ChartObjects.Count).Delete
    Shp.Delete
End Sub
```

40. Converter todos os arquivos de uma pasta para PDF

Esta rotina automaticamente converte todos os arquivos do tipo Excel em uma pasta de origem, para PDF em uma pasta destino.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub PdfConvert()
Dim iNumArq As Integer
Dim iCounter As Integer
Dim sMyFiles() As String
Dim OriginFolder As String
Dim DestinyFolder As String
 OriginFolder = "C:\Pasta_Origem"
DestinyFolder = "C:\Pasta_Destino"
 FileFound = FindFiles(OriginFolder, sMyFiles, iNumArq, "*", True)
If FileFound Then
    For iCounter = 1 To iNumArq
    Filename = sMyFiles(2, iCounter)
    ExtCount = Len(Filename) - Application.WorksheetFunction.Search(".",
Filename, 1) + 1
    Workbooks.Open (OriginFolder & "\" & Filename)
    Workbooks(Filename).Activate
    ActiveSheet.ExportAsFixedFormat Type:=xlTypePDF, Filename:= _
        DestinyFolder & "\" & Mid(Filename, 1, Len(Filename) - ExtCount) &
".pdf", Quality:= _
        xlQualityStandard, IncludeDocProperties:=True,
IgnorePrintAreas:=False, _
        OpenAfterPublish:=False
```

```
    Workbooks(Filename).Close
    Next iCounter
End If
End Sub
Function FindFiles(ByVal sPath As String, ByRef sFoundFiles() As String, _
    ByRef iArqEncontrados As Integer, _
    Optional ByVal sFileSpec As String = "*.*", _
    Optional ByVal blIncludeSubFolders As Boolean = False) As Boolean
    Dim iCounter As Integer
    Dim sFileName As String
    Dim oFileSystem As Object, oParentFolder As Object, oFolder As Object
    Set oFileSystem = CreateObject("Scripting.FileSystemObject")
    On Error Resume Next
    Set oParentFolder = oFileSystem.GetFolder(sPath)
    If oParentFolder Is Nothing Then
        FindFiles = False
        On Error GoTo 0
        Set oParentFolder = Nothing
        Set oFileSystem = Nothing
        Exit Function
    End If
    sPath = IIf(Right(sPath, 1) = "\", sPath, sPath & "\")
    sFileName = Dir(sPath & sFileSpec, vbNormal)
    Do While sFileName <> ""
        iCounter = UBound(sFoundFiles, 2)
        iCounter = iCounter + 1
        ReDim Preserve sFoundFiles(1 To 2, 1 To iCounter)
        sFoundFiles(1, iCounter) = sPath
        sFoundFiles(2, iCounter) = sFileName
        sFileName = Dir()
    Loop
    If blIncludeSubFolders Then
        For Each oFolder In oParentFolder.SubFolders
            FindFiles oFolder.Path, sFoundFiles, iArqEncontrados, sFileSpec,
blIncludeSubFolders
        Next
```

```
    End If
    FindFiles = UBound(sFoundFiles, 2) > 0
    iArqEncontrados = UBound(sFoundFiles, 2)
    On Error GoTo 0
    Set oFolder = Nothing
    Set oParentFolder = Nothing
    Set oFileSystem = Nothing
End Function
```

41. Listar todos os arquivos de uma pasta

Esta macro é essencial para usuários que desejam listar todos os arquivos disponíveis em uma pasta dentro de um arquivo do Excel, seja para relatórios ou para verificações.

Obs1: Os arquivos serão listados na coluna A, iniciando pela primeira linha.
Obs2: Trocar a pasta origem em negrito.

```
Sub ListarArquivos()
Dim iNumArq As Integer
Dim iContador As Integer
Dim sMyFiles() As String
Dim pastaorigem As String
Dim pastadestino As String
pastaorigem = "C:\Users\UsuarioTeste\Documents"
ArqEncontrado = ProcurarArquivos(pastaorigem, sMyFiles, iNumArq, "*",
True)
If ArqEncontrado Then
    For iContador = 1 To iNumArq
    Filename = sMyFiles(2, iContador)
    ActiveSheet.Range("A" & iContador) = Filename
    Next iContador
End If
Columns(1).AutoFit
End Sub
```

```
Function ProcurarArquivos(ByVal sPath As String, ByRef sFoundFiles() As
String, _
    ByRef iArqEncontrados As Integer, _
    Optional ByVal sFileSpec As String = "*.*", _
    Optional ByVal blIncludeSubFolders As Boolean = False) As Boolean
    Dim iContador As Integer
    Dim sFileName As String
    Dim oFileSystem As Object, oParentFolder As Object, oFolder As Object
    Set oFileSystem = CreateObject("Scripting.FileSystemObject")
    On Error Resume Next
    Set oParentFolder = oFileSystem.GetFolder(sPath)
    If oParentFolder Is Nothing Then
        ProcurarArquivos = False
        On Error GoTo 0
        Set oParentFolder = Nothing
        Set oFileSystem = Nothing
        Exit Function
    End If
    sPath = IIf(Right(sPath, 1) = "\", sPath, sPath & "\")
    sFileName = Dir(sPath & sFileSpec, vbNormal)
    Do While sFileName <> ""
        iContador = UBound(sFoundFiles, 2)
        iContador = iContador + 1
        ReDim Preserve sFoundFiles(1 To 2, 1 To iContador)
        sFoundFiles(1, iContador) = sPath
        sFoundFiles(2, iContador) = sFileName
        sFileName = Dir()
    Loop
    If blIncludeSubFolders Then
        For Each oFolder In oParentFolder.SubFolders
            ProcurarArquivos oFolder.Path, sFoundFiles, iArqEncontrados,
sFileSpec, blIncludeSubFolders
        Next
    End If
    ProcurarArquivos = UBound(sFoundFiles, 2) > 0
    iArqEncontrados = UBound(sFoundFiles, 2)
```

```
    On Error GoTo 0
    Set oFolder = Nothing
    Set oParentFolder = Nothing
    Set oFileSystem = Nothing
End Function
```

42. Copiar os arquivos de uma pasta para outra

Esta macro é muito utilizada para empresas e usuários que tenham acesso a uma rede, onde muitas vezes existe a rotina de realizar cópia de arquivos da rede para armazenar em uma outra pasta. Esta rotina é incrivelmente mais fácil com uma macro de Excel.

Obs: Substituir os valores de exemplo em negrito pelos valores desejados.

```
Sub CopiarColarArquivo()
Dim iNumArq As Integer
Dim iContador As Integer
Dim sMyFiles() As String
Dim pastaorigem As String
Dim pastadestino As String
pastaorigem = "C:\Users\usuarioteste\Documents\Macro origem"
pastadestino = "C:\Users\usuarioteste\Documents\Macro destino"
ArqEncontrado = ProcurarArquivos(pastaorigem, sMyFiles, iNumArq, "*",
True)
If ArqEncontrado Then
    For iContador = 1 To iNumArq
    FileCopy pastaorigem & "\" & sMyFiles(2, iContador), pastadestino & "\"
& sMyFiles(2, iContador)
    Next iContador
End If
End Sub
Function ProcurarArquivos(ByVal sPath As String, ByRef sFoundFiles() As
String, _
    ByRef iArqEncontrados As Integer, _
    Optional ByVal sFileSpec As String = "*.*", _
    Optional ByVal blIncludeSubFolders As Boolean = False) As Boolean
```

```
    Dim iContador As Integer
    Dim sFileName As String
    Dim oFileSystem As Object, oParentFolder As Object, oFolder As Object
    Set oFileSystem = CreateObject("Scripting.FileSystemObject")
    On Error Resume Next
    Set oParentFolder = oFileSystem.GetFolder(sPath)
    If oParentFolder Is Nothing Then
        ProcurarArquivos = False
        On Error GoTo 0
        Set oParentFolder = Nothing
        Set oFileSystem = Nothing
        Exit Function
    End If
    sPath = IIf(Right(sPath, 1) = "\", sPath, sPath & "\")
    sFileName = Dir(sPath & sFileSpec, vbNormal)
    Do While sFileName <> ""
        iContador = UBound(sFoundFiles, 2)
        iContador = iContador + 1
        ReDim Preserve sFoundFiles(1 To 2, 1 To iContador)
        sFoundFiles(1, iContador) = sPath
        sFoundFiles(2, iContador) = sFileName
        sFileName = Dir()
    Loop
    If blIncludeSubFolders Then
        For Each oFolder In oParentFolder.SubFolders
            ProcurarArquivos oFolder.Path, sFoundFiles, iArqEncontrados,
sFileSpec, blIncludeSubFolders
        Next
    End If
    ProcurarArquivos = UBound(sFoundFiles, 2) > 0
    iArqEncontrados = UBound(sFoundFiles, 2)
    On Error GoTo 0
    Set oFolder = Nothing
    Set oParentFolder = Nothing
    Set oFileSystem = Nothing
End Function
```

INTERAÇÕES COM OUTOLOOK

43. Enviar um e-mail simples com VBA

Esta macro tem por objetivo enviar um e-mail através de macro.

Obs 1: Alterar o assunto e a mensagem marcados em negrito
Obs 2: .CC e .BCC são opcionais, utilize apenas para os casos enviar cópias e cópias ocultas respectivamente.
Obs 3: O commando .**Send** marcado em negrito faz o e-mail ser enviado automaticamente, se trocar por .Display, o e-mail será apenas criado e deixado pronto, porém não será enviado.

```
Sub Send_Mail()
    Dim OutApp As Object
    Dim OutMail As Object
    Dim bMessage As String
    Set OutApp = CreateObject("Outlook.Application")
    Set OutMail = OutApp.CreateItem(0)
    bMessage = "Type the content line 1" & vbNewLine & _
            "Type the content line 2" & vbNewLine & _
            "Type the content line 3"
    On Error Resume Next
    With OutMail
        .to = "example@email.com"
        .CC = "copia@email.com"
        .BCC = "copiacega@email.com"
        .Subject = "Assunto do e-mail"
        .Body = "Texto do e-mail"
        .Send
    End With
    On Error GoTo 0
    Set OutMail = Nothing
    Set OutApp = Nothing
End Sub
```

44. Enviar arquivo como anexo por E-mail

Esta macro envia um e-mail incluindo o arquivo ativo como anexo deste e-mail.

Obs 1: Alterar o assunto e a mensagem marcados em negrito
Obs 2: .CC e .BCC são opcionais, utilize apenas para os casos enviar cópias e cópias ocultas respectivamente.
Obs 3: O commando **.Send** marcado em negrito faz o e-mail ser enviado automaticamente, se trocar por **.Display**, o e-mail será apenas criado e deixado pronto, porém não será enviado.

```
Sub SendWorkbookEmail()
    Dim OutApp As Object
    Dim OutMail As Object
    Set OutApp = CreateObject("Outlook.Application")
    Set OutMail = OutApp.CreateItem(0)
    On Error Resume Next
    With OutMail
        .to = "example@email.com"
        .CC = "copia@email.com"
        .BCC = "copiacega@email.com"
        .Subject = "Assunto do e-mail"
        .Body = "Texto do e-mail"
        .Attachments.Add ("C:\Arquivo_Exemplo.txt")
        .Send
    End With
    On Error GoTo 0
    Set OutMail = Nothing
    Set OutApp = Nothing
End Sub
```

45. Enviar planilha ativa como anexo por E-mail

Esta macro envia um e-mail incluindo a planilha ativa como anexo deste e-

mail.

Obs 1: Alterar o assunto e a mensagem marcados em negrito
Obs 2: .CC e .BCC são opcionais, utilize apenas para os casos enviar cópias e cópias ocultas respectivamente.
Obs 3: O commando .**Send** marcado em negrito faz o e-mail ser enviado automaticamente, se trocar por **.Display**, o e-mail será apenas criado e deixado pronto, porém não será enviado.

```
Sub SendActiveSheetEmail ()
    Dim Exten As String
    Dim FormtN As Long
    Dim OutApp As Object
    Dim OutMail As Object
    Dim OriginWKB As Workbook
    Dim DestWKB As Workbook
    Dim TempFilePath As String
    Dim TempFileFolder As String
    Application.ScreenUpdating = False
    Application.EnableEvents = False
    Set OriginWKB = ActiveWorkbook
    ActiveSheet.Copy
    Set DestWKB = ActiveWorkbook
    With DestWKB
        If Val(Application.Version) < 12 Then
            Exten = ".xls": FormtN = -4143
        Else
            Select Case OriginWKB.FileFormat
            Case 51: Exten = ".xlsx": FormtN = 51
            Case 52:
                If .HasVBProject Then
                    Exten = ".xlsm": FormtN = 52
                Else
                    Exten = ".xlsx": FormtN = 51
                End If
            Case 56: Exten = ".xls": FormtN = 56
```

```
            Case Else: Exten = ".xlsb": FormtN = 50
            End Select
        End If
    End With
    TempFilePath = Environ$("temp") & "\"
    TempFileFolder = "Part of " & OriginWKB.Name & " " & Format(Now,
"dd-mmm-yy h-mm-ss")
    Set OutApp = CreateObject("Outlook.Application")
    Set OutMail = OutApp.CreateItem(0)
    With DestWKB
        .SaveAs TempFilePath & TempFileFolder & Exten,
FileFormat:=FormtN
        On Error Resume Next
            With OutMail
                .to = "example@email.com"
                .CC = "copia@email.com"
                .BCC = "copiacega@email.com"
                .Subject = "Assunto do e-mail"
                .Body = "Texto do e-mail"
                .Attachments.Add ("C:\Arquivo_Exemplo.txt")
                .Send
            End With
        On Error GoTo 0
        .Close savechanges:=False
    End With
    Kill TempFilePath & TempFileFolder & Exten
    Set OutMail = Nothing
    Set OutApp = Nothing
    Application.ScreenUpdating = True
    Application.EnableEvents = True
End Sub
```

46. Enviar e-mail com uma seleção como anexo

Esta rotina cria um arquivo a partir de uma seleção de células, este arquivo será enviado por e-mail através do Outolook.

Obs 1: Alterar o assunto e a mensagem marcados em negrito
Obs 2: .CC e .BCC são opcionais, utilize apenas para os casos enviar cópias e cópias ocultas respectivamente.
Obs 3: O commando **.Send** marcado em negrito faz o e-mail ser enviado automaticamente, se trocar por .Display, o e-mail será apenas criado e deixado pronto, porém não será enviado.

```
Sub Mail_Range()
    Dim wb As Workbook
    Dim iTempFolder As String
    Dim iTempFile As String
    Dim Source As Range
    Dim Dest As Workbook
    Dim iFormatNum As Long
    Dim iExt As String
    Dim OutApp As Object
    Dim OutMail As Object
    Set Source = Nothing
    On Error Resume Next
    Set Source = Selection.SpecialCells(xlCellTypeVisible)
    On Error GoTo 0
    If Source Is Nothing Then
        MsgBox "The source is out of range, please try again.", vbOKOnly
        Exit Sub
    End If
    With Application
        .ScreenUpdating = False
        .EnableEvents = False
    End With
    Set wb = ActiveWorkbook
    Set Dest = Workbooks.Add(xlWBATWorksheet)
    Source.Copy
    Dest.Sheets(1).Cells(1).PasteSpecial Paste:=8
    Dest.Sheets(1).Cells(1).PasteSpecial Paste:=xlPasteValues
    Dest.Sheets(1).Cells(1).PasteSpecial Paste:=xlPasteFormats
```

```
    Dest.Sheets(1).Cells(1).Select
    Application.CutCopyMode = False
    iTempFolder = Environ$("temp") & "\"
    iTempFile = "Selection of " & wb.Name & " " & Format(Now, "dd-mmm-
yy h-mm-ss")
    If Val(Application.Version) < 12 Then
       iExt = ".xls": iFormatNum = -4143
    Else
       iExt = ".xlsx": iFormatNum = 51
    End If
    Set OutApp = CreateObject("Outlook.Application")
    Set OutMail = OutApp.CreateItem(0)
    With Dest
       .SaveAs iTempFolder & iTempFile & iExt, FileFormat:=iFormatNum
       On Error Resume Next
            With OutMail
               .to = "example@email.com"
               .CC = "copia@email.com"
               .BCC = "copiacega@email.com"
               .Subject = "Assunto do e-mail"
               .Body = "Texto do e-mail"
               .Attachments.Add ("C:\Arquivo_Exemplo.txt")
               .Send
            End With
       On Error GoTo 0
       .Close savechanges:=False
    End With
    Kill iTempFolder & iTempFile & iExt
    Set OutMail = Nothing
    Set OutApp = Nothing
    With Application
       .ScreenUpdating = True
       .EnableEvents = True
    End With
End Sub
```

47. Enviar e-mail com arquivo externo como anexo

Esta macro envia automaticamente um e-mail com um arquivo anexado, é possível também combinar esta macro com as que foram apresentadas anteriormente, enviando desta maneira a planilha ou seleção juntamente com um outro arquivo anexado.

Obs 1: Alterar o assunto e a mensagem marcados em negrito
Obs 2: .CC e .BCC são opcionais, utilize apenas para os casos enviar cópias e cópias ocultas respectivamente.
Obs 3: O commando .**Send** marcado em negrito faz o e-mail ser enviado automaticamente, se trocar por .Display, o e-mail será apenas criado e deixado pronto, porém não será enviado.

```
Sub SendWorkbookEmail()
    Dim OutApp As Object
    Dim OutMail As Object
    Set OutApp = CreateObject("Outlook.Application")
    Set OutMail = OutApp.CreateItem(0)
    On Error Resume Next
    With OutMail
        .to = "example@email.com"
        .CC = "copia@email.com"
        .BCC = "copiacega@email.com"
        .Subject = "Assunto do e-mail"
        .Body = "Texto do e-mail"
        .Attachments.Add ("C:\Arquivo_Exemplo.txt")
        .Send
    End With
    On Error GoTo 0
    Set OutMail = Nothing
    Set OutApp = Nothing
End Sub
```

Interações com PowerPoint

48. Exportar gráficos para Microsoft PowerPoint

Esta macro automaticamente cria uma apresentação em PowerPoint com todos os gráficos dentro de uma planilha ativa.

Obs 1: É possível ajustar a ajustar a posição do gráfico no slide alterando o valor destaco em negrito
Obs 2: Dentro do editor VBA, clicar em ferramentas, referencias e então procurar por Microsoft PowerPoint Object Library, marque a opção na caixa de seleção e então pressione OK.




```
Sub Excel_chart_to_PPT ()
    Dim PptApp As PowerPoint.Application
    Dim iSlide As PowerPoint.Slide
    Dim ChartObj As Excel.ChartObject
    On Error Resume Next
    Set PptApp = GetObject(, "PowerPoint.Application")
    On Error GoTo 0
    If PptApp Is Nothing Then
       Set PptApp = New PowerPoint.Application
    End If
    If PptApp.Presentations.Count = 0 Then
       PptApp.Presentations.Add
    End If
    PptApp.Visible = True
```

```
        For Each ChartObj In ActiveSheet.ChartObjects
            PptApp.ActivePresentation.Slides.Add
PptApp.ActivePresentation.Slides.Count + 1, ppLayoutText
            PptApp.ActiveWindow.View.GotoSlide
PptApp.ActivePresentation.Slides.Count
            Set iSlide =
PptApp.ActivePresentation.Slides(PptApp.ActivePresentation.Slides.Count)
            ChartObj.Select
            ActiveChart.ChartArea.Copy
            On Error Resume Next
            iSlide.Shapes.PasteSpecial(DataType:=ppPasteMetafilePicture).Select
            iSlide.Shapes(1).TextFrame.TextRange.Text =
ChartObj.Chart.ChartTitle.Text
            PptApp.ActiveWindow.Selection.ShapeRange.Left = 25
            PptApp.ActiveWindow.Selection.ShapeRange.Top = 150
            iSlide.Shapes(2).Width = 300
            iSlide.Shapes(2).Left = 600
        Next
    AppActivate ("Microsoft PowerPoint")
    Set iSlide = Nothing
    Set PptApp = Nothing
End Sub
```

49. Exportar seleção para Microsoft PowerPoint

Esta macro automaticamente exporta uma seleção para uma apresentação de PowerPoint

Obs 1: É possível ajustar a ajustar a posição da tabela que será exportada no slide alterando o valor destaco em negrito
Obs 2: Dentro do editor VBA, clicar em ferramentas, referencias e então procurar por Microsoft PowerPoint Object Library, marque a opção na caixa de seleção e então pressione OK.




```
Sub Selection_to_PowerPoint()
Dim iRange As Range
Dim PptObj As Object
Dim iPresent As Object
Dim iSlide As Object
Dim iShape As Object
  Set iRange = Selection
  On Error Resume Next
     Set PptObj = GetObject(class:="PowerPoint.Application")
     Err.Clear
     If PptObj Is Nothing Then Set PptObj =
CreateObject(class:="PowerPoint.Application")
     If Err.Number = 429 Then
       MsgBox "PowerPoint could not be found, aborting."
       Exit Sub
     End If
  On Error GoTo 0
  Application.ScreenUpdating = False
  Set iPresent = PptObj.Presentations.Add
  Set iSlide = iPresent.Slides.Add(1, 11)
  iRange.Copy
  iSlide.Shapes.PasteSpecial DataType:=2
  Set iShape = iSlide.Shapes(iSlide.Shapes.Count)
     iShape.Left = 100
     iShape.Top = 160
  PptObj.Visible = True
  PptObj.Activate
```

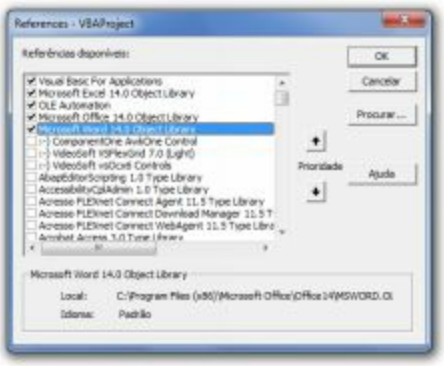
```
  Application.CutCopyMode = False
  Application.ScreenUpdating = True
End Sub
```

INTERAÇÕES COM WORD

50. Exportar seleção para o Microsoft Word

Esta macro exporta automaticamente a seleção atual do Excel para o arquivo em Word aberto.

Obs: Dentro do editor VBA, clicar em ferramentas, referências e então procurar por Microsoft Word Object Library, marcar a caixa de seleção e pressionar OK.

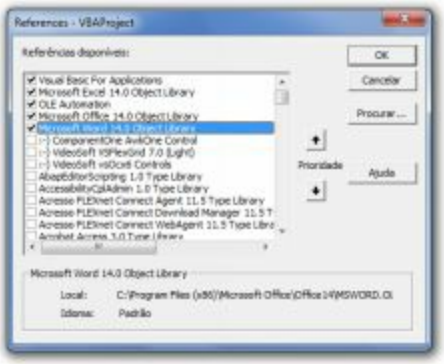



```
Sub CopyRangetoWord()
Dim WodAPP As Word.Application
Dim WordDOC As Word.Document
If Not TypeName(Selection) = "Range" Then
    MsgBox "Erro de seleção, tente novamente.", vbExclamation, "Range
Error"
Else
    Set WodAPP = GetObject(, "Word.Application")
    Set WordDOC = WodAPP.ActiveDocument
    Selection.Copy
    WodAPP.Selection.PasteSpecial Link:=False, DataType:=wdPasteRTF, _
        Placement:=wdInLine, DisplayAsIcon:=False
    Set WordDOC = Nothing
    Set WodAPP = Nothing
End If
End Sub
```

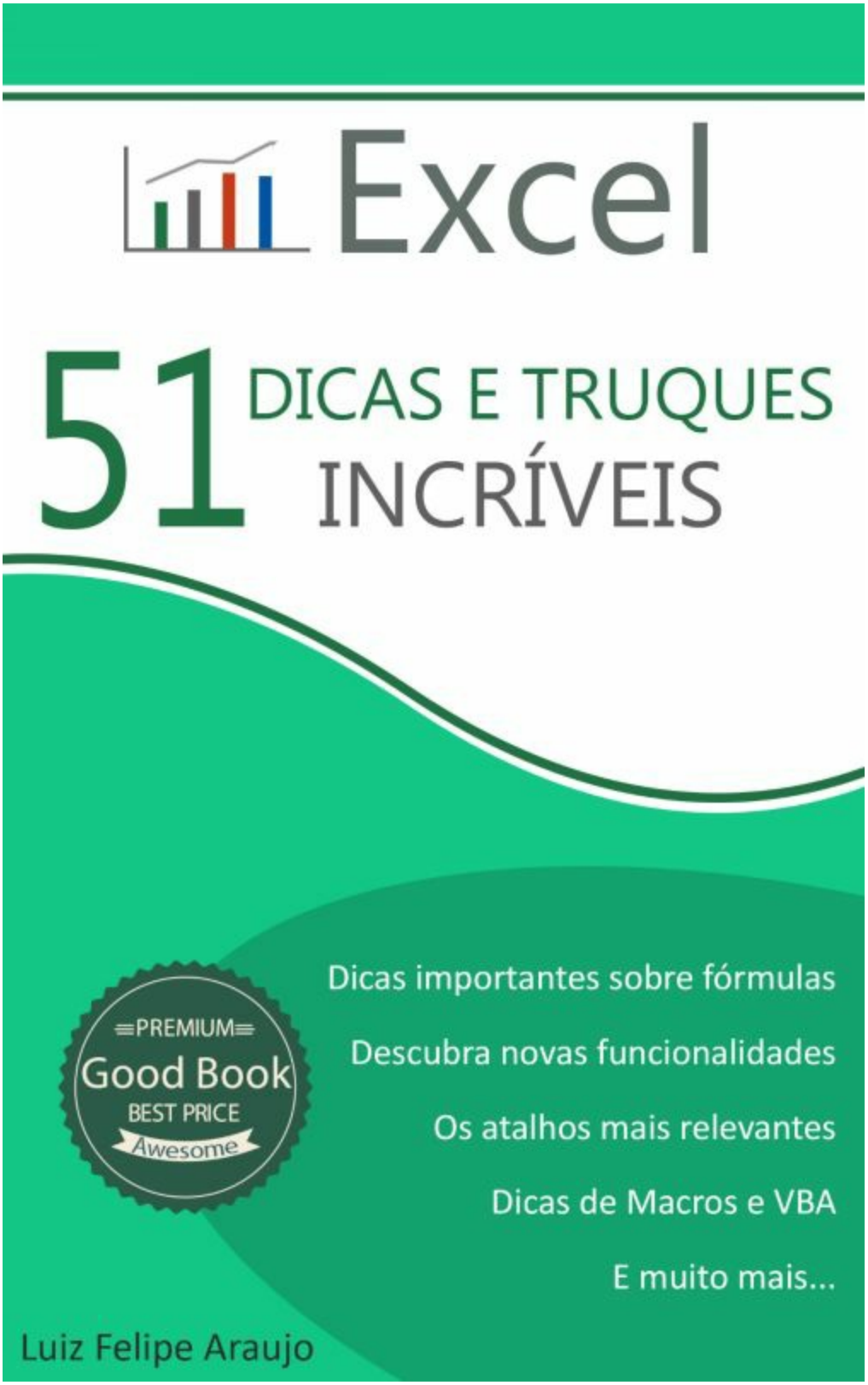
51. Exportar dados para o Microsoft Word

Esta rotina exporta as informações de uma seleção para um novo arquivo do Word, salva o arquivo com o mesmo nome do Excel e também no mesmo diretório.

Obs: Dentro do editor VBA, clicar em ferramentas, referências e então procurar por Microsoft Word Object Library, marcar a caixa de seleção e pressionar OK.




```
Sub Worksheet_to_Word()
    Dim Question As Integer
    Question = MsgBox("Esta macro vai exportar toda seleção para word, deseja continuar ?", vbYesNo)
    If question <> 6 Then
    Exit Sub
    End If
    Set Object = CreateObject("Word.Application")
    Object.Visible = True
    Set newObject = Object.Documents.Add
    ActiveSheet.UsedRange.Copy
    newObject.Range.Paste
    Application.CutCopyMode = False
    Object.Activate
    On Error Resume Next
    newObject.SaveAs Filename:=Application.ActiveWorkbook.Path & "\" & ActiveSheet.Name
End Sub
```


Excel
51 DICAS E TRUQUES INCRÍVEIS
PREMIUM
Good Book
BEST PRICE
Awesome
Dicas importantes sobre fórmulas
Descubra novas funcionalidades
Os atalhos mais relevantes
Dicas de Macros e VBA
E muito mais...
Luiz Felipe Araujo

INTRODUÇÃO

Excel é uma plataforma extremamente completa, permite ao usuário criar soluções para os mais diversos tipos de problemas. A ferramenta como um todo é simples de aprender, porém existem diversos segredos e truques para dominá-la por completo, tanto em relação as suas possibilidades, como em relação a produtividade. Neste aspecto que este livro se destaca, serão apresentadas 51 dicas e truques para o usuário dominar a plataforma de uma maneira mais consistente e eficaz, cada dica será explorada de uma maneira completa a apresentar ao usuário a sua utilidade, aplicação e quando necessário sua aplicação em exemplos práticos.

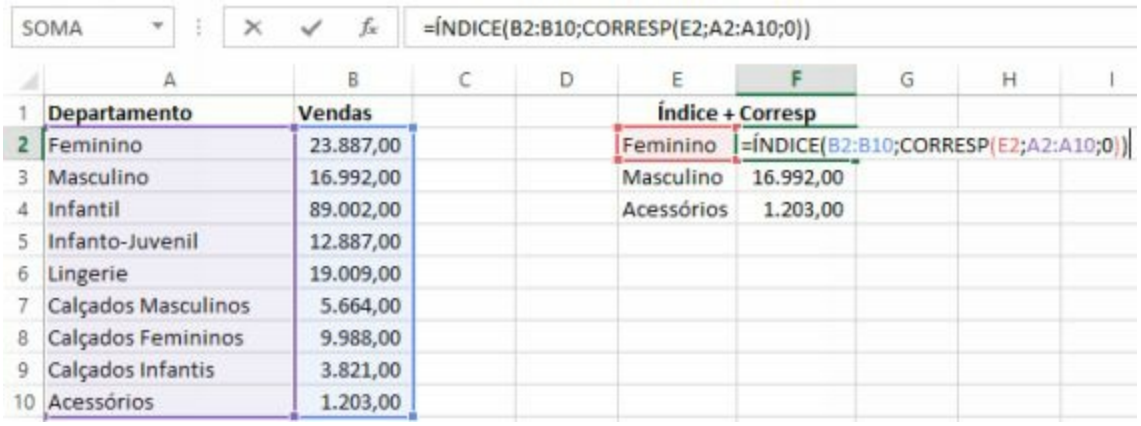
DICAS DE FÓRMULAS

1. Utilize o Índice(corresp()) ao invés do ProcV()

ProcV é uma das funções mais famosas no Excel, o seu objetivo é simplesmente procurar um valor em uma matriz e retornar um valor relativo a um índice informado. Entretanto, a função tem limitações em relação a ordem dos termos procurados e os valores de retorno, caso esta ordem não esteja adequada para o ProcV, é necessário reorganizar a estrutura da planilha para poder aplicar a fórmula.

O Índice+Corresp por sua vez não tem esta limitação, funcionando em qualquer estrutura de planilha, para aplicar o índice+corresp, basta seguir as dicas abaixo:

Exemplo:



	A	B	C	D	E	F	G	H	I
1	**Departamento**	**Vendas**				**Índice + Corresp**			
2	Feminino	23.887,00			Feminino	=ÍNDICE(B2:B10;CORRESP(E2;A2:A10;0))			
3	Masculino	16.992,00			Masculino	16.992,00			
4	Infantil	89.002,00			Acessórios	1.203,00			
5	Infanto-Juvenil	12.887,00							
6	Lingerie	19.009,00							
7	Calçados Masculinos	5.664,00							
8	Calçados Femininos	9.988,00							
9	Calçados Infantis	3.821,00							
10	Acessórios	1.203,00							

=ÍNDICE(**B2:B10**;CORRESP(**E2**;**A2:A10**;**0**))

B2:B10 – Região que retorna os valores
E2 – Valor procurado
A2:A10 – Região procurada
0 – Correspondência exata

Neste exemplo, deseja-se buscar o valor de vendas do departamento “Feminino”, para tanto, informa-se que a região de índice é **B2:B10**, posteriormente, é utilizado o CORRESP() buscando a célula de valor “Feminino”, célula **E2**, na região **A2:A10**. Esta combinação fará o valor

correspondente ser retornado na célula onde a função está sendo aplicada.

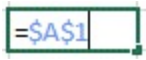
2. Trave as suas fórmulas

Congelar intervalos, consiste em inserir uma trava representada pelo símbolo de cifrão $, onde a coluna ou linha que sucede o símbolo estará travada no momento de “arrastar uma fórmulas” (alça de preenchimento).
Ainda utilizando o exemplo anterior, onde se deseja arrastar a fórmula para as demais células, é preciso travar a região de retorno e a região de procura, a única célula neste exemplo que fica totalmente livre é a E2, visto que se deseja que ela seja alterada quando copiada para as linha subsequentes, o exemplo com as devidas travas aplicadas fica:

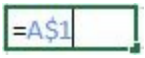
=ÍNDICE(**$B$2:$B$10**;CORRESP(**E2**;**$A$2:$A$10**;**0**))

Para inserir as travas pode-se colocar manualmente o símbolo de cifrão $, ou apenas pressionar F4.

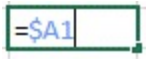
Pressionar uma vez F4 – Trava linha e coluna



Pressionar duas vezes F4 – Trava apenas linha

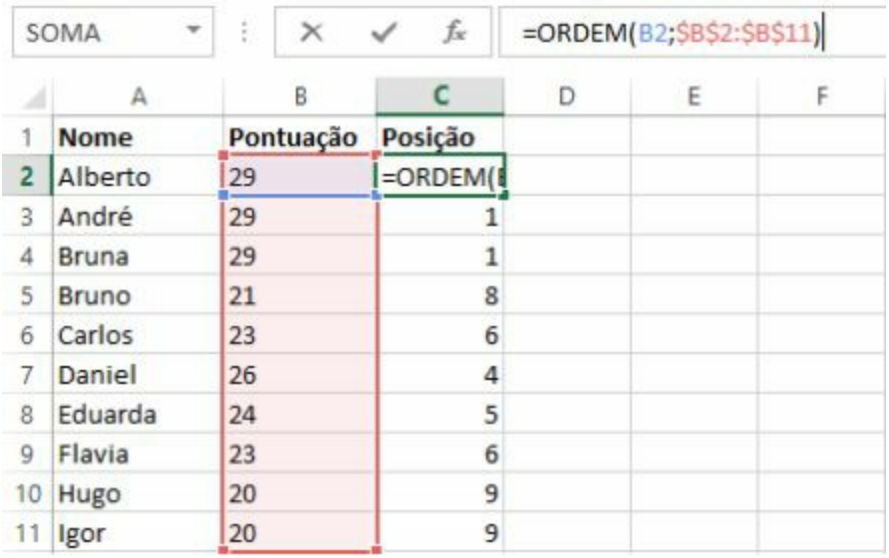


Pressionar três vezes F4 – Trava apenas coluna



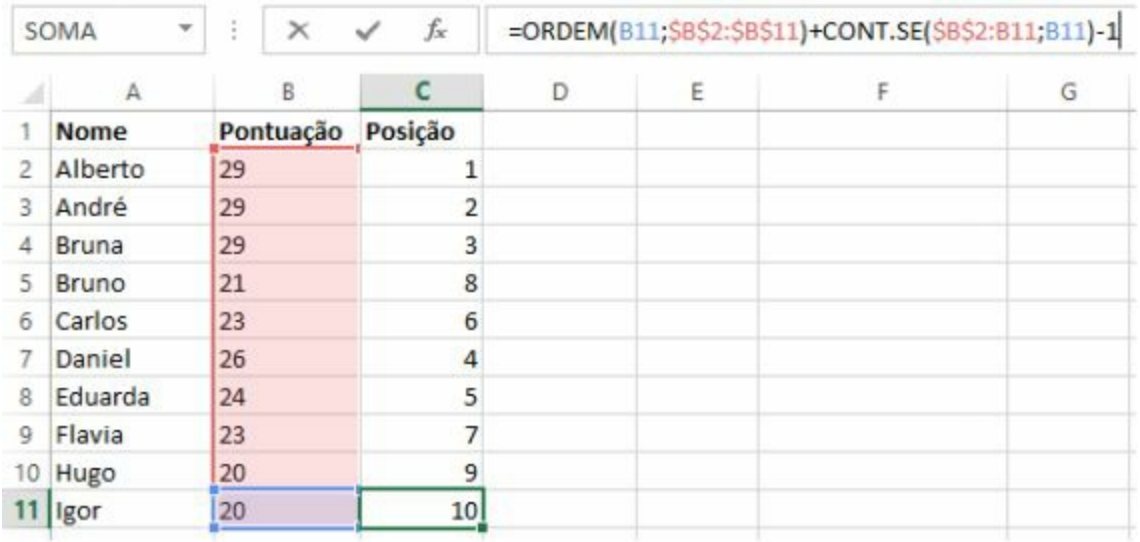
3. Fórmula ordem com desempate

Por padrão, quando se aplica a fórmula ordem em uma região, caso existam valores iguais, o resultado serão posições repetidas, como por exemplo na imagem abaixo, onde existem retornos repetidos das posições 1 e 9.



	A	B	C	D	E	F
1	**Nome**	**Pontuação**	**Posição**			
2	Alberto	29	=ORDEM(			
3	André	29	1			
4	Bruna	29	1			
5	Bruno	21	8			
6	Carlos	23	6			
7	Daniel	26	4			
8	Eduarda	24	5			
9	Flavia	23	6			
10	Hugo	20	9			
11	Igor	20	9			

Uma maneira de contornar este problema é somar um CONT.SE subtraindo 1. Este artificio basicamente conta quantas repetições o número possui para desempatar o a ordenação. A subtração do número 1 se dá pela correção de que o número sempre aparecerá pelo menos uma vez no CONT.SE.



	A	B	C	D	E	F	G
1	**Nome**	**Pontuação**	**Posição**				
2	Alberto	29	1				
3	André	29	2				
4	Bruna	29	3				
5	Bruno	21	8				
6	Carlos	23	6				
7	Daniel	26	4				
8	Eduarda	24	5				
9	Flavia	23	7				
10	Hugo	20	9				
11	Igor	20	10				

4. Auxiliar de fórmula

O Excel oferece uma ferramenta que facilita o preenchimento de uma fórmula, para isso basta começar a digita-la, e depois pressionar Ctrl + A, para então abrir a caixa de argumentos de função, ela não apenas facilita o preenchimento da fórmula como também explica o seu funcionamento.

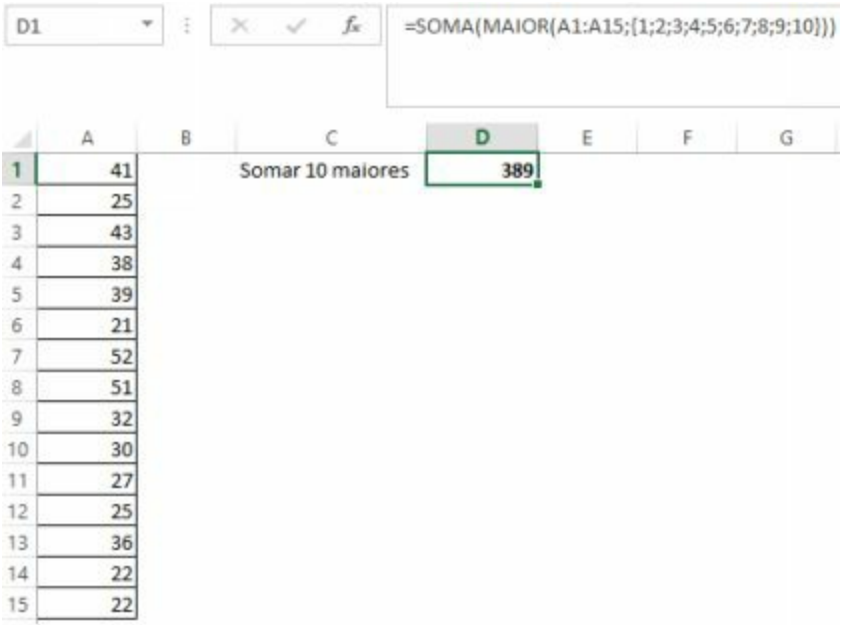


5. Somar maiores ou menores valores

Para somar os maiores ou menores valores de uma região, basta combinar a função soma, com as funções maior ou menor, utilizando uma matriz para representar a quantidade dos maiores valores que serão somados. A matriz é representada por {}, conforme exemplo abaixo onde se soma os 10 maiores números de uma região A1:A15.

Desta forma se utiliza a função:

=SOMA(MAIOR(A1:A15;{1;2;3;4;5;6;7;8;9;10}))



6. Formula para aplicar letra maiúscula

Embora pouco utilizada, existe uma forma de garantir que uma célula sempre possua a letra maiúscula desejada, para tanto se utiliza as funções abaixo:

Original	Modificado	Fórmula Utilizada
planilha	PLANILHA	=MAIÚSCULA(A2)
PLANILHA	planilha	=MINÚSCULA(A3)
planilha	Planilha	=PRI.MAIÚSCULA(A4)

7. Formula para diferentes formatações de data

É possível transformar datas em textos de diversas maneiras diferentes, para tanto basta utilizar a fórmula texto e utilizar os códigos de formatação, segue abaixo alguns exemplos de aplicações da função texto para datas.

Data	Texto	Fórmula Utilizada
05/01/2018	05/01/18	=TEXTO(B3;"dd/mm/aa")
05/01/2018	05 jan 2018	=TEXTO(A3;"dd mmm aaa")
05/01/2018	sex 05 jan	=TEXTO(A4;"ddd dd mmm")
05/01/2018	janeiro 2018	=TEXTO(A5;"mmmm aaaa")
05/01/2018	05 janeiro 2018	=TEXTO(A6;"dd mmmm aaaa")

DICAS DE ATALHOS

3. Atalhos de Menu

Atalhos de menu são pouco utilizados pela maioria dos usuários pelo fato de exigirem uma série de combinações, algumas vezes não tão intuitivas, por outro lado usuários que treinam a sua utilização, em pouco tempo conseguem aumentar drasticamente a produtividade não apenas no Excel mas também em outros softwares do pacote Office.
Para visualizar os atalhos basta pressionar Alt.



Uma vez que o **Alt** é pressionado a dica dos próximos atalhos são mostradas no menu superior.
Por exemplo:
Pressionar **Alt → S** leva até o menu de dados



Posteriormente os atalhos dos demais menus são apresentados. Desta forma é possível navegar por praticamente todas as opções do Excel apenas pelos atalhos relacionados ao **Alt**, uma vez que se ganhar prática, é possível decorar diversos caminhos de maneiras rápidas, como por exemplo:

Alt → S → F
Aplica filtros

Alt → S → RE
Remove duplicatas

Caso em algum momento seja desejado sair do modo atalho de menu, pode-se pressionar ESC e então os atalhos superiores serão ocultados.

9. Colar especial

Para abrir a caixa de colar especial rapidamente, utilize o atalho abaixo.



10. Aplicar filtro rapidamente

Para aplicar filtro rapidamente em uma região, basta selecioná-la e pressionar:



11. Navegar no filtro rapidamente

Para rapidamente expandir o filtro, utilize:



Ou



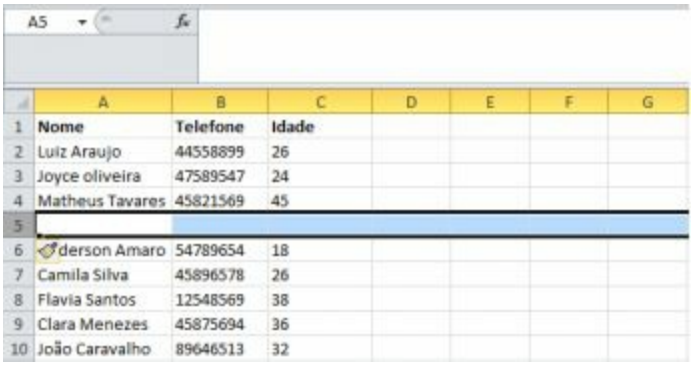


Use as setas do teclado para navegar entre as opções, use a barra de espaço para marcar e desmarcar as caixas de seleção.

12. Adicionar e remover linhas/colunas rapidamente

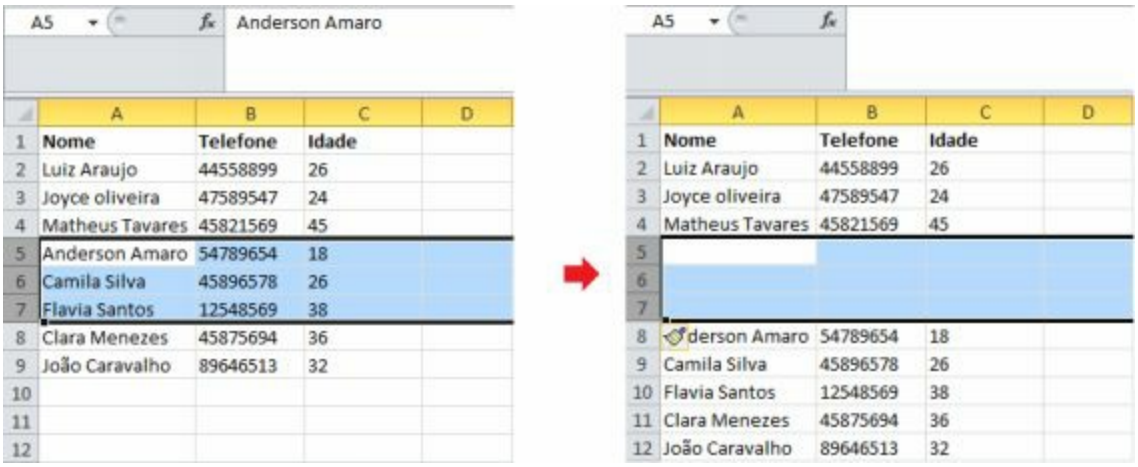
Para adicionar linhas ou colunas de forma extremamente rápida, basta selecionar a linha toda e utilizar o atalho Ctrl +, a linhas será adicionada acima da seleção

Para remover, basta selecionar a linha desejada e utilizar Ctrl –



	Nome	Telefone	Idade
2	Luiz Araujo	44558899	26
3	Joyce oliveira	47589547	24
4	Matheus Tavares	45821569	45
5			
6	derson Amaro	54789654	18
7	Camila Silva	45896578	26
8	Flavia Santos	12548569	38
9	Clara Menezes	45875694	36
10	João Caravalho	89646513	32

Obs: O mesmo procedimento pode ser executado para adicionar mais de uma linha ou coluna, basta selecionar a quantidade desejada e pressionar o atalho Ctrl +, ou para remover Ctrl -



13. Atalhos de Seleção

Dentre as dicas de atalho deste livro, as de seleção são sem dúvida as mais fundamentais par ganhar produtividade, através delas pode-se selecionar e editar células de maneira extremamente rápida.

Seleciona células para baixo



Seleciona células para cima



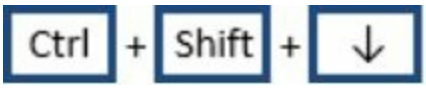
Seleciona células para direita



Seleciona células para esquerda



Selecionar até o fim para baixo



Selecionar até o fim para cima



Selecionar até o fim para direita



Selecionar até o fim para esquerda



14. Selecionar uma região de células

Este atalho seleciona todas as células em uma região próxima.

Seleciona todas as células de uma região



15. Selecionar linhas e colunas

Estes dois atalhos selecionam linhas e colunas de maneira extremamente rápida, acostume-se a usa-los.

Selecionar coluna inteira



Selecionar a linha inteira



6. Atalhos para formatos de números

Para usuários que lidam com muitos números, os atalhos de formatos são fundamentais para aumentar a produtividade. Desta forma, segue abaixo os principais atalhos para alterar os formatos dos números no Excel.

Formato de número contábil



Formato de número geral



Formato de número percentual



Formato de número cientifico



Formato de data



Formato de hora e minuto



Formato com duas casas decimais



Separar Milhares



Aumentar Casas decimais



Diminuir Casas decimais



17. Criar gráficos de forma instantânea

Para criar gráficos de forma extremamente rápida, basta selecionar a região dos dados utilizar o atalho:



18. Inserir Data e Hora

Para inserir hora e data de forma rápida pode-se utilizar respectivamente os atalhos a seguir





Fórmula	Exemplo	Descrição (Resultado)
=Hoje()	11/12/2017	A data atual (variável)
=Agora()	11/12/2017 23:29	A data e hora atual (variável)

TRUQUES VARIADOS

19. Procurar termo com caractere desconhecido

Quando se busca um termo ou frase no Excel através do Ctrl + F, é possível colocar um termo variado no meio da busca, com auxílio da interrogação (?) ou asterisco (*). O til (~) funciona para dar a função real do símbolo quando posicionado antes do mesmo.

Interrogação (?) – Substitui um único caractere. Por exemplo, ao se buscar a palavra Cl?ra, pode-se encontrar Clara, Cloro, Clero e etc.

Asterisco(*) - Substitui caractere mas pode assumir a função de diversas posições. Por exemplo, ao se buscar a palavra A*do, pode-se encontrar Aldo, Atrasado, Alambrado, ou seja, qualquer termo que se inicia por “A” e termina com “do”.

Til (~) – A função do til é apenas para que os termos especiais acima sejam buscados como eles próprios, por exemplo, quando se deseja buscar: “Porque?”, deve-se escrever “Porque~?”, para que a interrogação seja buscada como ela mesma, ao invés do termo duvidoso.

20. Cálculo automático e manual

Esta funcionalidade do Excel pode passar muitas vezes despercebida, até o usuário precisar lidar com uma grande quantidade de informações e enfrentar uma série de problemas ligados a travamentos, lentidões e perda de informações. Uma solução pode ser a utilização do cálculo manual para evitar processamento desnecessário. Uma vez que o cálculo esteja em modo manual, é preciso informar ao Excel quando se deseja calcular e o que se deseja calcular. Para optar entre cálculo automático e manual, basta acessar o menu Fórmulas e posteriormente clicar no botão Opções de Cálculo, então poderá se optar entre automático e manual.



O cálculo manual pode ser realizado em até três níveis diferentes de objetos:

a) A nível de célula, onde basta edita-la e pressionar Enter, então o resultado será calculado
b) A nível de planilha, pode ser realizado clicando no menu superior Fórmulas > Calcular Planilha ou através do atalho Sift + F9
c) A nível de arquivo, pode ser acessado clicando no menu superior Fórmulas > Calcular Agora ou através do atalho F9

21. Verificar Erros

Este se trata de outro recursos subutilizado no Excel, antes de enviar um relatório para terceiros é sempre interessante verificar os erros de uma planilha, para tanto, o Excel oferece recursos para rastrear e identificar possíveis erros de informação na planilha. Para acessar os recursos de verificação do Excel, basta acessar no menu superior Fórmulas e posteriormente clicar em Verificação de erros.



22. Rastrear precedentes e dependentes

Uma vez que se lida com muitas informações dentro das planilhas, usuários podem se deparar com a necessidade de entender as informações precedentes e dependentes de uma fórmula, para tanto o Excel oferece a ferramenta de rastreamento. Para acessar este recurso basta clicar no menu superior



Utilize este recurso sempre que precisar entender de maneira rápida a origem ou dependência de uma função, como no exemplo abaixo:



Neste exemplo, a fórmula célula F2 depende das informações da coluna A e C, quando se ativa o rastrear precedentes para na célula F2 as setas indicam sua dependência com as duas outras colunas. Porém quando se ativa o rastrear dependentes, uma seta para direita, indica a dependência da célula I2, a qual possui fórmula que referência F2.

23. Valores iniciados por zero

Valores iniciados por zero sofrem supressão do Excel, existem duas maneira de mostrar os números iniciados por zero, uma delas é transformar a célula em texto em página inicial e posteriormente alterar o formato da seleção para texto, conforme imagem abaixo:



Outra forma de inserir números iniciados por zero é inserir apóstrofo no início da célula, conforme exemplo abaixo:

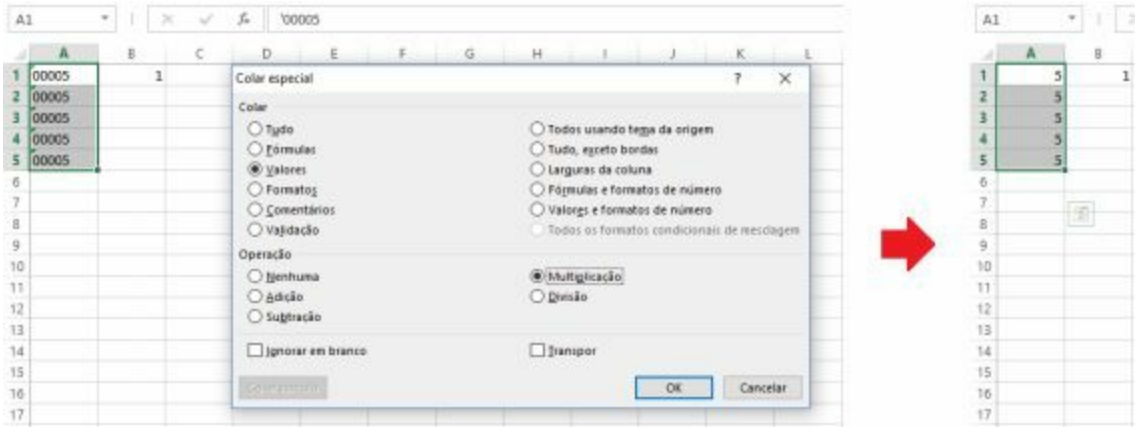


Isso faz com que o número seja armazenado como texto, porém sem precisar alterar a formatação da célula.

Transformar texto em número de forma rápida

Para transformar textos em números rapidamente, basta copiar uma célula com valor equivalente a 1 e colar especial em cima da região desejada, escolher a opção valor e operação de multiplicação, desta forma, todo texto será transformado em número.

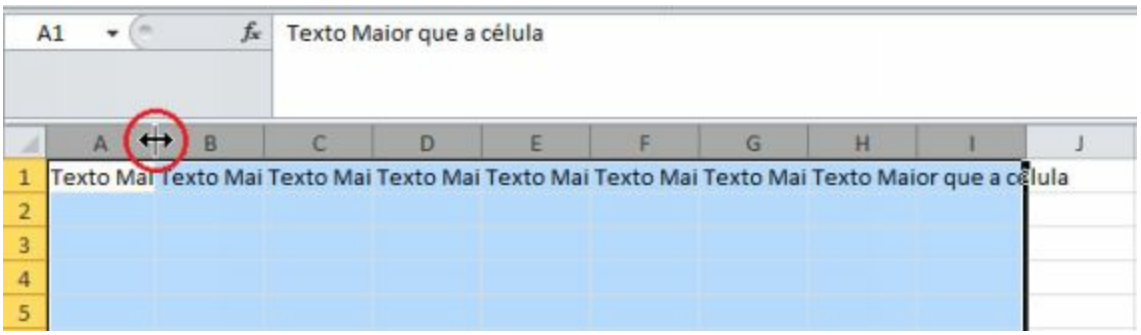
Obs: Colar especial pode ser acessado rapidamente com Ctrl + Shift + V



24. Redimensionar simultaneamente diversas colunas ou linhas

Para alterar a largura das colunas, basta posicionar o mouse entre duas colunas clicar e arrastar, para realizar o mesmo procedimento diversas colunas, basta selecionar a região desejada, posicionar o mouse conforme indicado na imagem, segurar a tecla Shift e arrastar. Todas as colunas da região serão redimensionadas igualmente.

Obs: o procedimento para linhas é o mesmo.

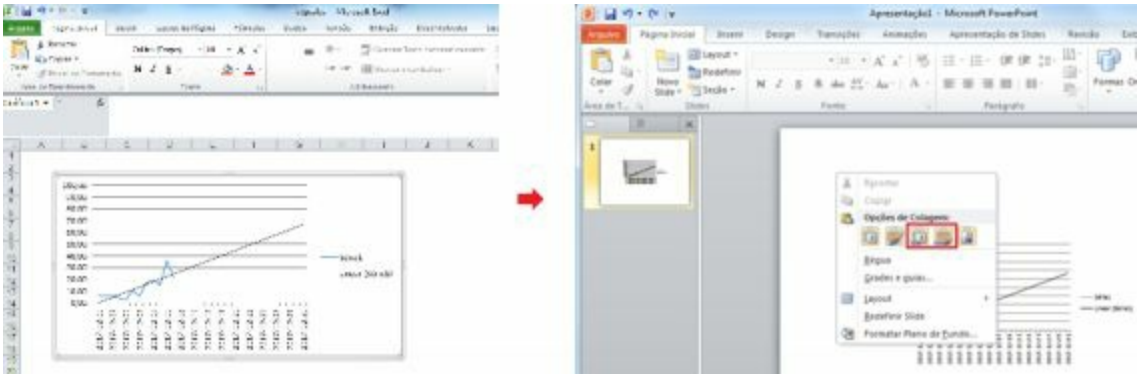


25. Copiar seleção ou gráfico como vínculo no PowerPoint ou Word

Um recurso extremamente pouco utilizado e desconhecido pela maioria dos usuários é a opção de vincular os arquivos entre o Excel e o PowerPoint e Word. Criar vínculos é fundamental para todos que trabalham com relatórios, desta maneira tudo que for alterado no Excel, poderá facilmente ser

atualizado nos respectivos objetos vinculados, seja no PowerPoint ou no Word.

Para criar vínculo, basta copiar a região de células ou gráfico no Excel, ir até o PowerPoint colar especial, optar pela opção de colar com vínculo, conforme imagem abaixo:

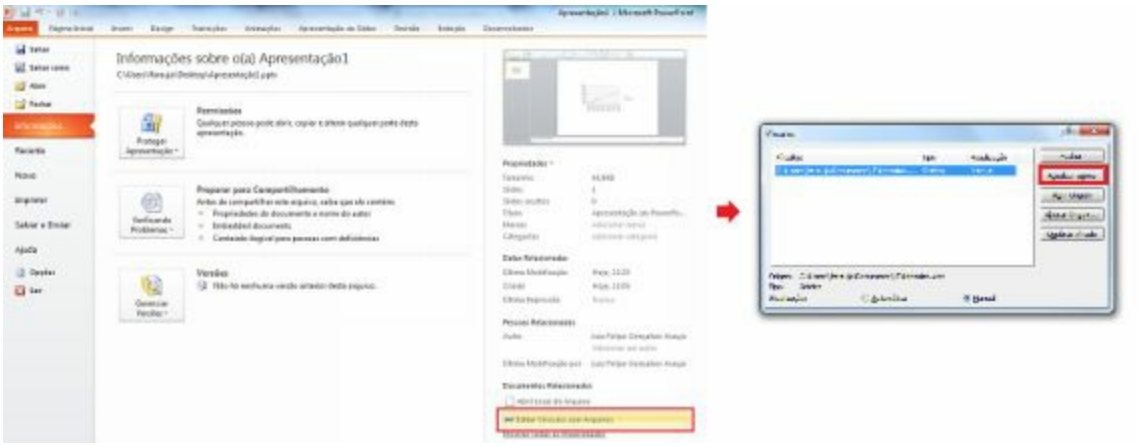


Obs 1: A única diferença entre as duas opções de colar especial é manter a formatação original ou não.

Obs 2: O procedimento para colar seleção de células é o mesmo.

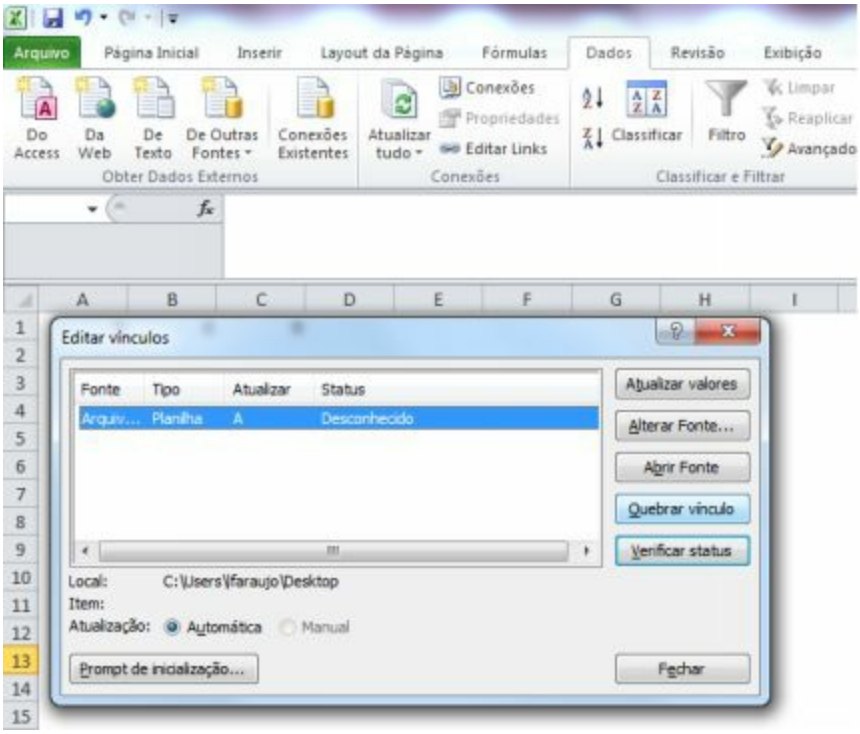
Obs 3: O procedimento para o Word também é o mesmo.

Obs 4: No caso de colar muitas informações, é necessário atualizar os dados manualmente no PowerPoint, para tanto, basta acessar Arquivo, Informações, Editar Vínculos com Arquivos, Selecionar os vínculos desejados e clicar no botão "Atualizar Agora".



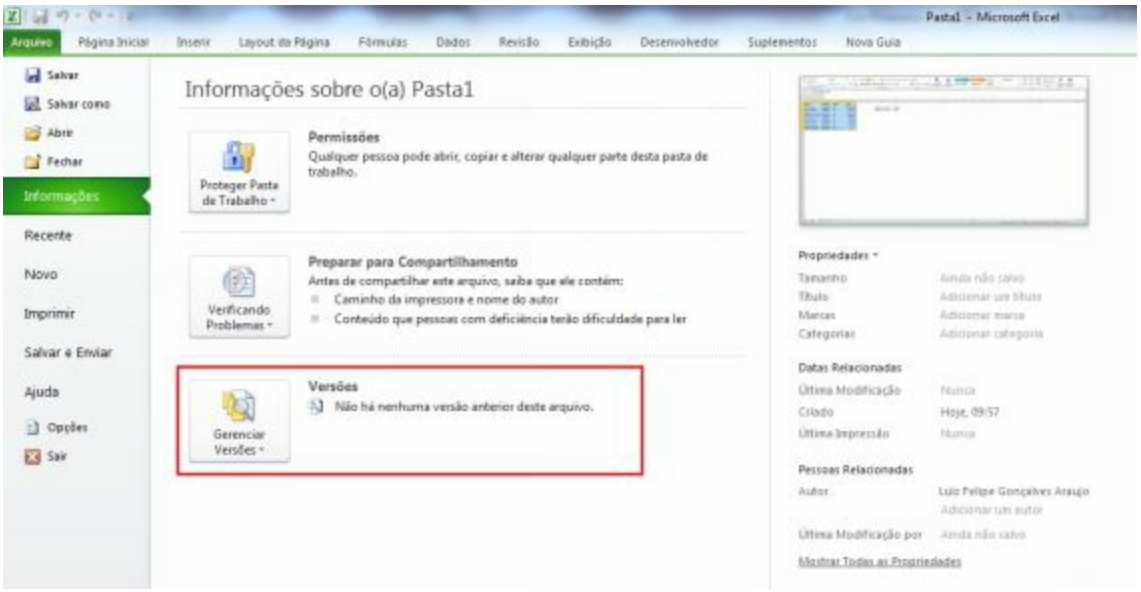
26. Quebre os vínculos antes de enviar um relatório

Sempre que uma fórmula inserida realizar referência com outro arquivo, um vínculo entre arquivos é automaticamente criado. Porém, é necessário tomar cuidado com os vínculos quando se deseja enviar um relatório por e-mail ou através de uma rede, visto que o destinatário precisaria ter acesso ao mesmo arquivo de vínculo, no mesmo diretório. Para evitar este tipo de problema, existe a opção de quebrar vínculo, o qual pode ser acessado pelo menu superior, Dados, Editar Links, Quebrar vínculo. Segue exemplo abaixo:



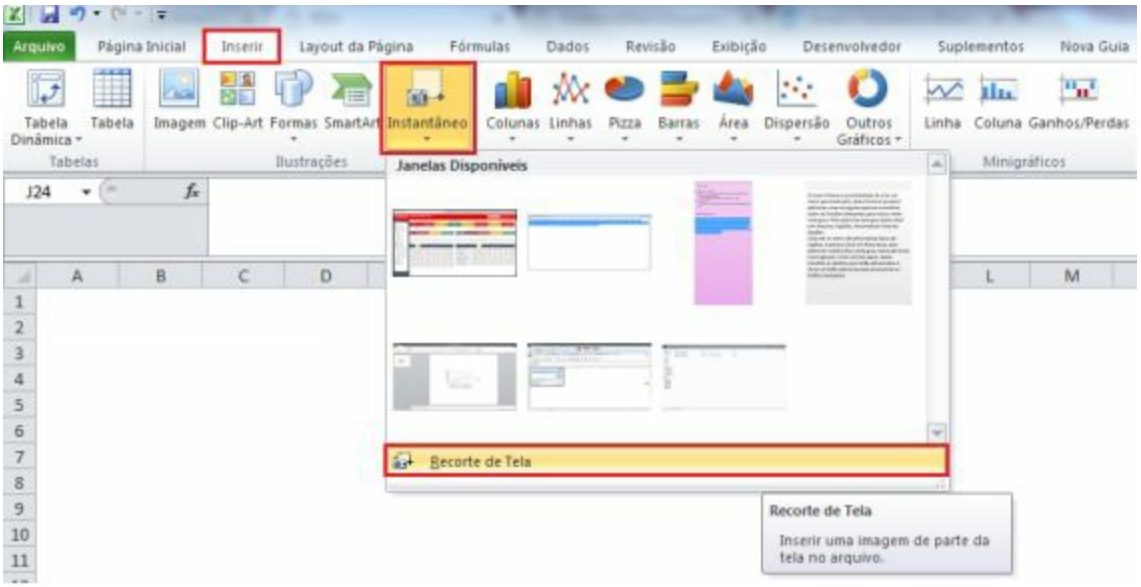
27. Recuperar arquivo após travamento

Por padrão o Excel realizar salvamentos automáticos de arquivos temporários, para que nos casos de travamento, exista um backup para restaurar os dados. É possível verificar estes arquivos temporários salvos pelo Excel no menu Arquivo, Informações e Versões. Todas as versões salvas automaticamente (caso existam) pela plataforma estarão disponíveis neste local.



28. Ferramenta do Excel para capturar tela

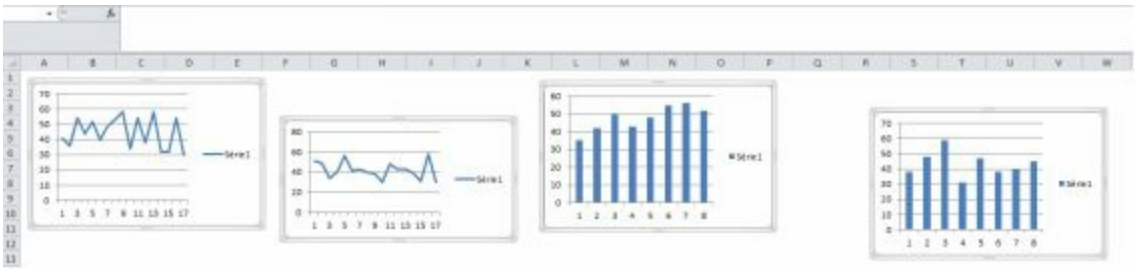
Poucos sabem, mas o Excel possui uma própria ferramenta de captura de tela, semelhante a ferramenta do windows 7 ou superior. Através dela pode-se rapidamente capturar uma região de um "print screen". Basta acessar no menu superior**: Inserir > Instantâneo> Recorte de Tela**, conforme na imagem a seguir.



29. Padronize e organize os gráficos e demais objetos

Alinhar, padronizar tamanhos e organizar os objetos é primordial para um relatório profissional, para realizar esta tarefa o Excel oferece algumas ferramentas para em poucos cliques alinhar e padronizar todos os objetos. Para demonstrar como este procedimento é feito, segue abaixo quatro gráficos fora de padrão e alinhamento.

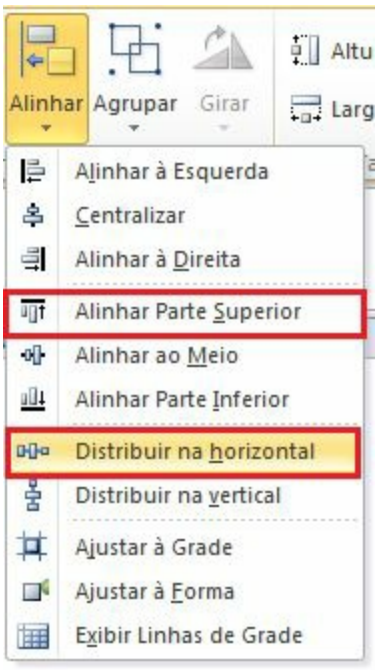
a) Selecione todos os objetos que se deseja alinhar segurando Ctrl e clicando em cada um deles.



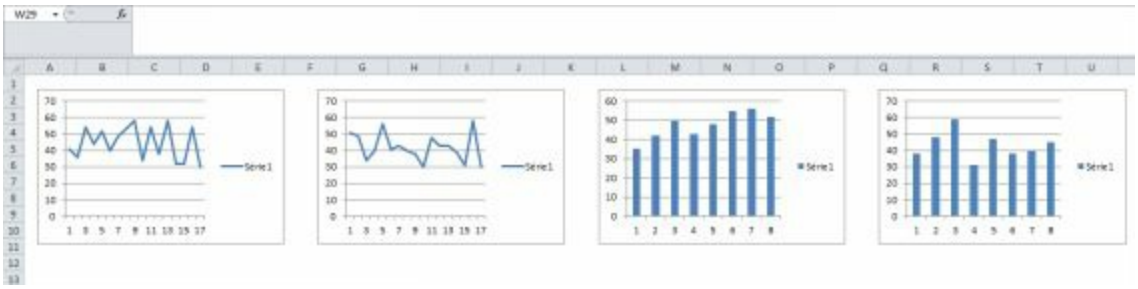
b) Selecione o menu Formatar e escolha a dimensão que será atribuída a todos eles.



c) Ainda dentro do menu superior Formatar, clicar em alinhar e então escolher as opções Alinhar Parte Superior e Distribuir na Horizontal



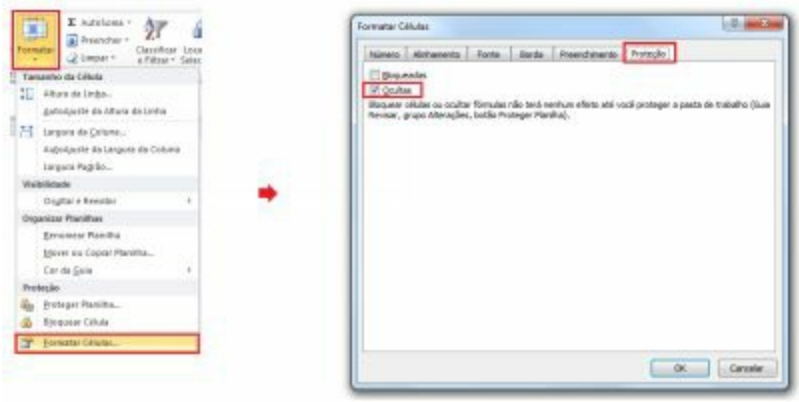
d) Resultado final dos gráficos devidamente alinhados e padronizados



Obs: As opções “Distribuir na horizontal” e “Alinhar Parte Superior” foram escolhidas pela disposição horizontal dos objetos, caso estivessem dispostos na vertical, bastaria escolher as opções relacionadas a “Distribuir na Vertical” e “Alinhar a Esquerda”.

30. Ocultar formulas

O Excel possibilita ocultar as formulas mostrando apenas o seu resultado, para tanto basta acessar: **Página inicial > Formatar > Formatar células > Proteção > Ocultas**



A opção “bloqueadas” impede o usuário de editar as células e Ocultas impede o usuário de visualizar as formulas. Agora basta utilizar a opção de proteção de planilha para que as células não mostrem mais as formulas.

Para proteger a planilha: **Página inicial > Formatar > Proteger Planilha...**

31. Extensões de arquivo Excel

As extensões no Excel possuem diferentes propósitos, abaixo será possível conferir as principais extensões e as suas respectivas utilidades e ocasiões para uso.

FORMATO	EXTENSÃO	DESCRIÇÃO
Pasta de trabalho do Excel	*.xlsx*	O formato de arquivo padrão com base em XML do Office Excel 2007.Não armazena macros.
Pasta de trabalho do Excel (código)	*.xlsm*	O formato de arquivo do Office Excel 2007 baseado em XML e habilitado por macro.
Pasta de Trabalho Binária do Excel	*.xlsb*	O formato de arquivo binário do Office Excel 2007. Formato compactado, reduz o tamanho do arquivo.
Pasta de Trabalho do Excel 97-Excel 2003	*.xls*	O formato de arquivo binário do Excel 97 - Excel 2003 (BIFF8).

Obs: a única recomendação para a extensão xls é para utilizar para compatibilidade com versões do Excel 2003 ou inferior.

DICAS DE MACROS E VBA

32. Traduzir fórmulas de português para inglês usando VBA

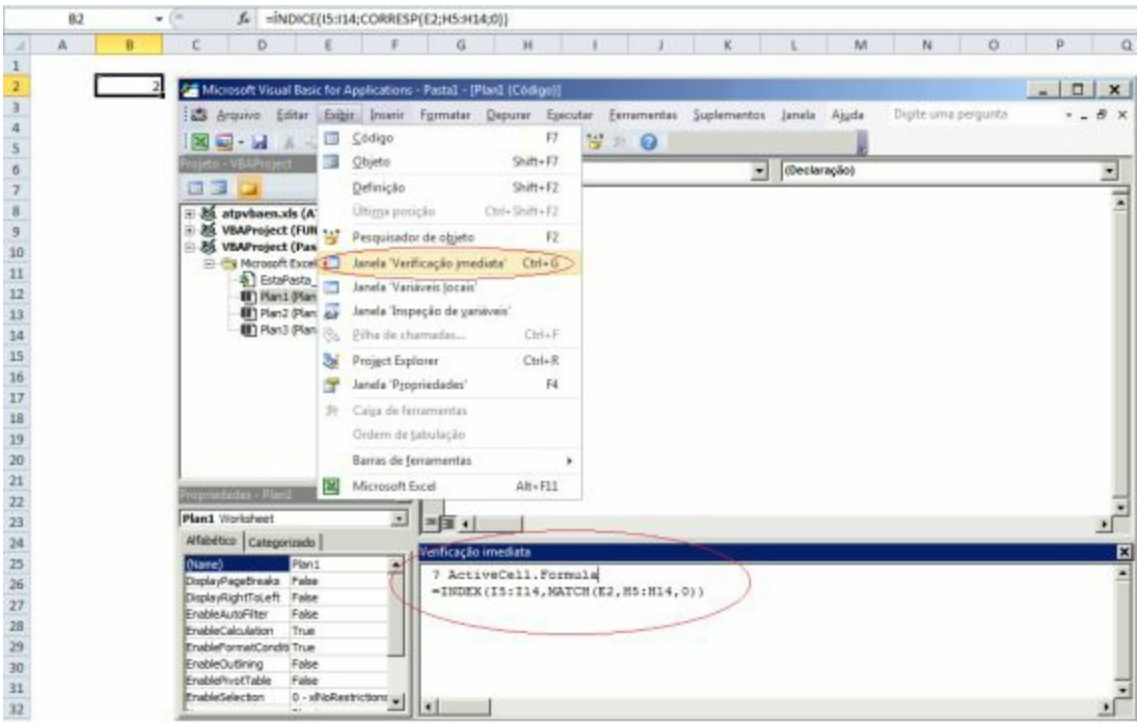
Um dos problemas que os usuários do Excel em português enfrentam ao utilizar o VBA é a questão das funções em inglês. Uma vez que se deseja utilizar funções dentro do código é necessário digita-las no idioma de origem da linguagem, para tanto fórmulas como por exemplo:

=ÍNDICE(H5:H14;CORRESP(E2;I5:I14;0))

Precisam ser escritas dentro do código como:

=INDEX(H5:H14,MATCH(E2,I5:I14,0))

Para usuários habituados com o Excel em português, isso pode ser um problema, porém existe um truque para traduzir as funções utilizando a janela de verificação imediata do editor VBA. Para tanto, basta acessar o editor com o atalho Alt + F11 e então exibir a janela de verificação imediata:



Para “traduzir” as funções basta seguir os passos abaixo:

- Selecionar a célula que contém a função desejada
- Digitar na janela de verificação imediata: “? ActiveCell.Formula”

- Pressionar Enter

Desta maneira, o Excel retorna como a fórmula é originalmente escrita em inglês.

33. Utilização do “With Statement” para simplificar o código

É muito comum pessoas utilizarem o VBA no Excel e desconhecerem a função do With, o qual é uma excelente ferramenta para simplificar o código, para tanto basta seguir o exemplo abaixo:

Sem aplicação do With:

```
Sub Aprendizado()
Range("B2").Interior.Color = vbBlack
Range("B2").Font.Color = vbWhite
Range("B2").Font.Name = "Arial"
Range("B2").Font.Size = 12
End Sub
```

Com aplicação do With:

```
Sub Aprendizado_With()
With Range("B2")
   .Interior.Color = vbBlack
   .Font.Color = vbWhite
   .Font.Name = "Arial"
   .Font.Size = 12
End With
End Sub
```

Sem a utilização do With o objeto precisa ser repetido toda as vezes que um método é aplicado, com o With, o objeto é citado uma única vez, depois se aplica apenas os métodos nas linhas subsequentes.

34. Desativar atualização de tela (Melhorar desempenho)

Para acelerar o processamento das macros é recomendado desativar a atualização de tela, para tanto basta desativar no início do código e reativar no final, da seguinte maneira:

```
Application.ScreenUpdating = False
‘Insira o código desejado
Application.ScreenUpdating = True
```

35. Calculo manual (Melhorar desempenho)

Outra maneira de agilizar as macros é desativando o cálculo automático no início e reativando no final.

Obs: Utilizando o cálculo manual, Sempre que um cálculo precisar ser realizado em uma planilha, deve-se utilizar o comando: “ActiveSheet.Calculate”

```
Application.ScreenUpdating = False
‘Insira o código desejado
Application.ScreenUpdating = True
```

36. Copiar e colar corretamente (Melhorar desempenho)

Copiar e colar através de VBA é extremamente simples, porém a maioria dos usuários utilizam mais memória do que o necessário para realizar este método, realizando da seguinte maneira:

```
Range(“B1”,”B3”).Copy
Range(“C1”).Select
ActiveSheet.Paste
```

Entretanto existe uma maneira direta de copiar e colar sem utilizar a “Área de transferência”, para isso basta utilizar o código a seguir com o mesmo efeito das três linhas anteriores.

```
Range(“B1”,”B3”).Copy Destination:=Range("C1")
```

37. Desabilitar eventos (Melhorar desempenho)

Esta é mais uma dica com ênfase em melhorar o desempenho das macros, ela basicamente desativa eventos como por exemplo o “AutoSave”, responsável por realizar backups automáticos. Para tanto basta utilizar o código a seguir:

```
Application.EnableEvents = False

‘Insira o código desejado

Application.EnableEvents = True
```

38. Impedir notificações

Algumas notificações interrompem e atrapalham o funcionamento das macros, para impedi-las, basta utilizar os códigos abaixo:

```
Application.DisplayAlerts = False

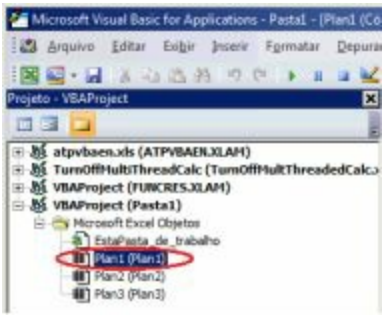
 ‘Insira o código desejado

Application.DisplayAlerts = True
```

39. Chamar uma macro quando uma célula for alterada

Este simples código, permite que uma macro seja chamada quando uma célula sofrer alteração.

Obs: É necessário adicionar o código no objeto da planilha desejada, no exemplo abaixo, a macro “ExemploMacro”, será chamada quando a célula A1 for alterada na planilha “Plan1”



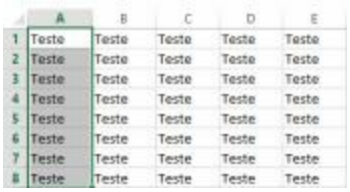

```
Private Sub worksheet_change(ByVal target As Range)
If target.Address = "$A$1" Then
Call ExemploMacro
End If
End Sub
```

```
Sub ExemploMacro()
‘Seu código
End Sub
```

40. Selecionar informações do início ao fim em uma direção

Existem diversas formas de selecionar informações do início ao final, porém os códigos mais eficientes são:

```
Range("A1",Range("A1").End(xlDown)).Select
```



Para duas direções:

Neste exemplo se seleciona da célula A1 até o final, considerando as direções para baixo e no segundo caso para direita também, para trocar por outros direções, como por exemplo:

```
Range("A8",Range("A1").End(xlUp)).Select
```

```
Range("A1",Range("A1").End(xlToRight)).Select
```

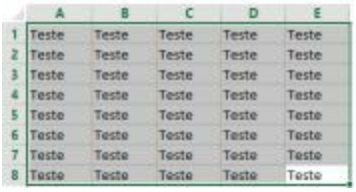
```
Range("A1",Range("E1").End(xlToLeft)).Select
```

Obs: Pode-se realizar diversas combinações, basta avaliar a necessidade de cada caso.

41. Selecionar informações do início ao fim em duas direções

Aplica-se exatamente o mesmo conceito, porém se aplica o método “End” duas vezes, considerando as duas direções desejadas, no exemplo abaixo, considera-se a partir da célula “A1”, primeiro o “End” é aplicado para baixo e depois para direita, a ordem poderia ser invertida sem alterar o resultado da seleção.

```
Range("A1", Range("A1").End(xlDown).End(xlToRight)).Select
```



	A	B	C	D	E
1	Teste	Teste	Teste	Teste	Teste
2	Teste	Teste	Teste	Teste	Teste
3	Teste	Teste	Teste	Teste	Teste
4	Teste	Teste	Teste	Teste	Teste
5	Teste	Teste	Teste	Teste	Teste
6	Teste	Teste	Teste	Teste	Teste
7	Teste	Teste	Teste	Teste	Teste
8	Teste	Teste	Teste	Teste	Teste

42. Abrir planilhas na web com PHP

Acessar uma planilha em uma rede através de PHP é extremamente simples basta usar o comando normal de abrir arquivo utilizando o link como endereço desejado.

Exemplo:

```
Sub openphp ()
Workbooks.Open "http://endercoteste/gerarExcel.php"
End sub
```

43. Controlar mouse e teclado com SendKeys

O VBA permite controlar o mouse e teclado através de comandos simples, segue abaixo o exemplo de uma macro que controla as funções básicas de mouse e teclado.

Obs: Os comentários da macro demonstram o funcionamento de cada linha.

```
Private Sub Macro_Mouse_Teclado()
  'Determina a posição X e Y do ponteiro do mouse
  SetCursorPos 150, 850
  'Pressiona o botão esquerdo do mouse
  mouse_event MOUSEEVENTF_LEFTDOWN, 0, 0, 0, 0
  'Solta o botão esquerdo do mouse
  mouse_event MOUSEEVENTF_LEFTUP, 0, 0, 0, 0

  'Pressiona o botão direito do mouse
  mouse_event MOUSEEVENTF_RIGHTDOWN, 0, 0, 0, 0
  'Solta o botão esquerdo do mouse
  mouse_event MOUSEEVENTF_RIGHTUP, 0, 0, 0, 0
  'Espera 1 segundo
  Application.Wait Now + #12:00:01 AM#
  'Digita texto
  SendKeys ("Texto")
```

```
  'Pressiona a tecla Enter
  SendKeys ("{Enter}")
  'Pressiona a tecla Backspace
  SendKeys ("{BS}")
End Sub
```

44. Esperar X segundos

Este é um método muito simples, porém desconhecido por muitos usuários. Ao utilizar VBA, existe um comando simples que faz com que o código entre em um modo de espera por um tempo determinado, este tempo de espera pode ter diversas finalidades. Para aplicar o tempo de espera basta utilizar a seguinte linha de código:

Excel 2007 ou inferior:

```
Application.Wait Now + #0:00:05#
```

Excel 2010 ou superior:

```
Application.Wait Now + #12:00:05 AM#
```

O código conforme citado espera 5 segundos para dar continuidade na macro.

45. Melhorar a visualização do código

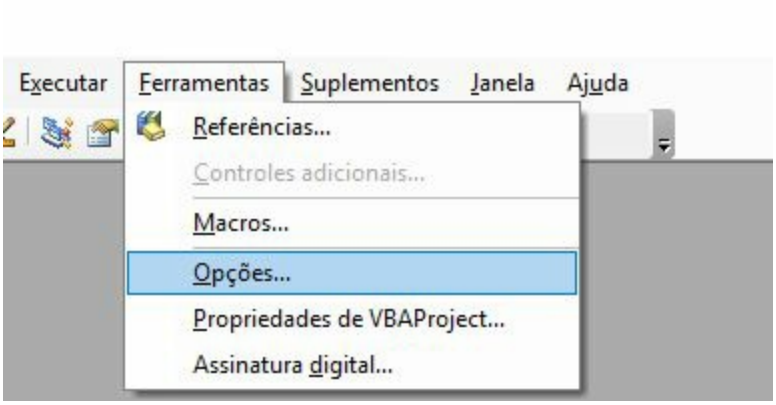
Esta dica envolve a configuração de como as cores dos códigos são exibidas no editor. O padrão é o fundo branco, texto padrão preto e etc. Entretanto, utilizar por muito tempo a tela branca para programação pode ser algo desconfortável, desta forma muitos usuários optam por alterar o esquema de cores do Editor, alguns preferindo fundo preto ou cinza. Desta forma, segue abaixo as dicas relacionadas a alteração dos esquemas de cores.

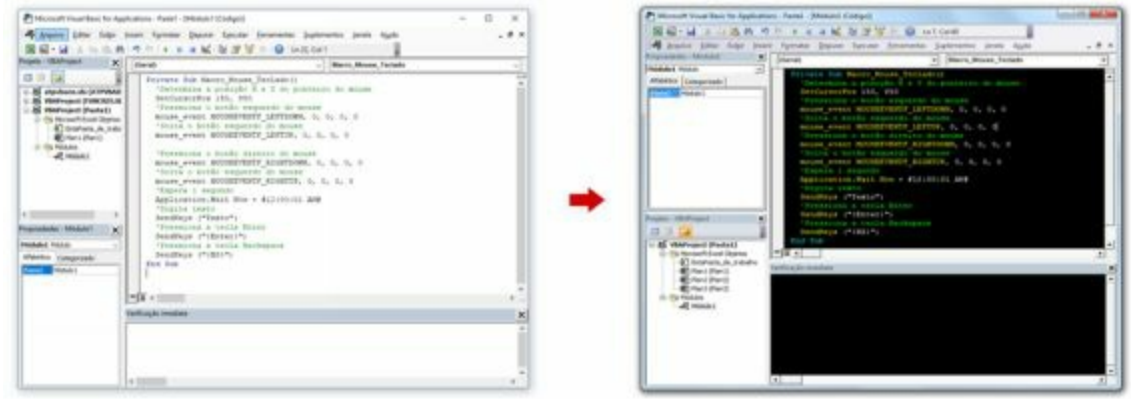
Antes e depois das alterações:

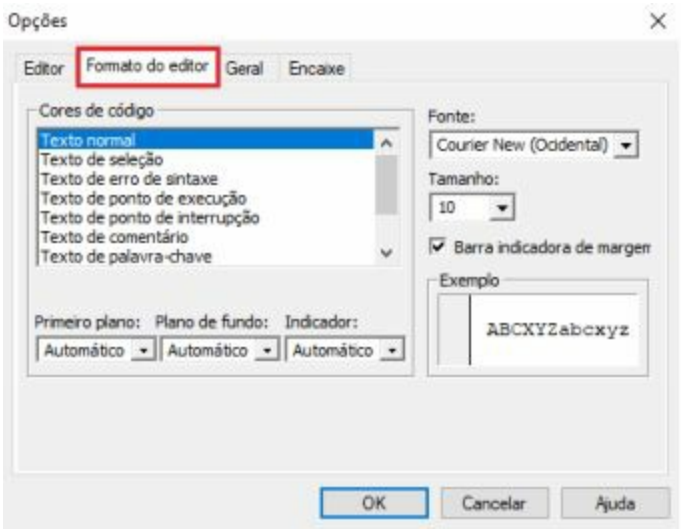
Obs: Antes de prosseguir, é preciso alertar que o Excel não possui uma opção de “Restaurar Padrão”, desta forma, para retornar ao esquema original de cores, é preciso realizar esta tarefa de maneira manual.

Para alterar as cores, basta acessar o editor VBA, clicar no menu superior Ferramentas, Opções, na nova janela que irá abrir, basta acessar o Formato do

editor, conforme as imagens abaixo:







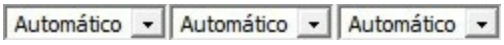
Desta forma será possível alterar todas as configurações de cores do Editor.

Sugestão de cores:

Texto normal:



Texto de seleção:



Texto de erro de sintaxe:



Texto de ponto de execução:



Texto de ponto de interrupção:



Texto de comentário:



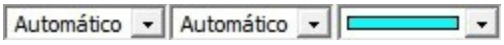
Texto de palavra-chave:



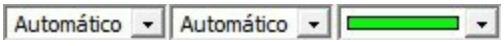
Texto de identificador:



Texto de indicador:



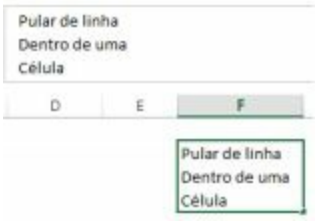
Texto de retorno de chamada:



DICAS RÁPIDAS

46. Para pular de linha dentro de uma célula

Para pular de linha dentro do texto de uma célula basta usar Alt + Enter



47. Ao invés de “concatenar ()” utilize “&”

A função concatenar funciona combinar o valor de duas células, porém pode-se realizar a mesma função utilizando &.

48. Auto completar formula

Ao começar a digitar uma função no Excel ele sugere opções em uma caixa abaixo da região de digitação, para auto completar, basta pressionar a tecla “Tab”.

49. Somar linhas com atalho

Para rapidamente somar os valores distribuídos em linhas, selecione-os e pressione o atalho **Alt =**

50. Gerar valores aleatórios

Existe uma função para gerar valores aleatórios, para tanto utilize:

=ALEATÓRIOENTRE()

=ALEATÓRIO()

Na primeira escolhe-se um valor máximo e um mínimo e na segunda o valor aleatório é gerado entre 0 e 1.

51. Utilizar uma célula pra cálculo rápido

Uma vez que seja necessário realizar uma conta rápida, insira a mesma dentro de uma célula como se estivesse escrevendo uma função, porém ao final utilize **Ctrl =** então o resultado será mostrado na célula sem a "fórmula" digitada.