

Capítulo 10 - Funções e Procedimentos

Neste capítulo vamos aprender uma forma de melhorar a sua programação. Utilizando **funções** e **procedimentos** nós podemos reaproveitar código, melhorar a leitura dos algoritmos e criar códigos mais limpos e legíveis.

Neste capítulo vamos ver um pouquinho de geometria básica. Só pra lembrar um pouquinho a escola. Mas não se assuste, vai ser fácil.

Vamos lá?

O que são Funções e Procedimentos

A primeira coisa que você tem que entender é, afinal, que raios são funções e procedimentos?

Bom, já adianto que você já usou procedimentos e nem percebeu!

Lembra quando você quis mostrar algum texto na tela? Você usou o procedimento **ESCREVA** e passou um texto como parâmetro, justamente o texto que você queria que aparecesse na tela.

```
ESCREVA("Olá mundo!")
```

Você saberia mostrar um texto na tela sem usar esse procedimento? Não né.

Outra pergunta: Você saberia fazer um algoritmo para calcular a raiz quadrada de um número? Reflita um pouquinho sobre a complexidade de tal algoritmo. E um algoritmo para gerar um número aleatório? Você saberia fazer?

Imprimir um texto na tela, raiz quadrada, geração de número aleatório, entre outros, são funções e procedimentos clássicos que um programador usa, mas não precisa implementar na unha. Pra quê re-inventar a roda??? Alguém já fez esses algoritmos e a gente apenas usa. O que precisamos é apenas solicitar a execução desses algoritmos dentro do nosso algoritmo.

Qual a diferença entre função e procedimento?

A única diferença entre uma função (*function*) e um procedimento (*procedure*) é que a função retorna um valor (por exemplo uma função que calcula raiz quadrada retorna um número) e o procedimento não retorna nada (por exemplo o procedimento 'escreva' que já falei).

A figura abaixo exemplifica como acontece a utilização de uma função, o procedimento é a mesma coisa, menos na atribuição do resultado à variável "a".



Funções (e **procedimentos**) podem ou não receber parâmetros. No caso da função de raiz quadrada, é necessário passar como parâmetro o número que se

deseja calcular a raiz, o procedimento **ESCREVA**, requer um texto como parâmetro para apresentar na tela do usuário.

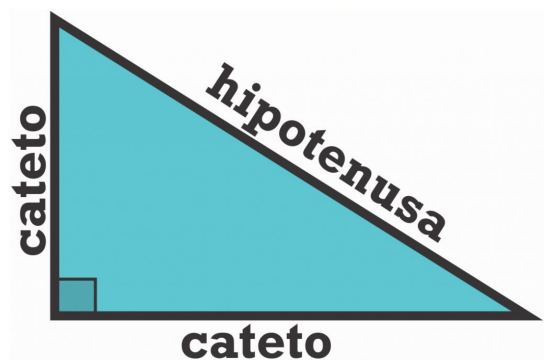
Agora que já sabemos o que são e pra quê servem. Vamos para a prática!

Hora de praticar: Utilizando funções e procedimentos

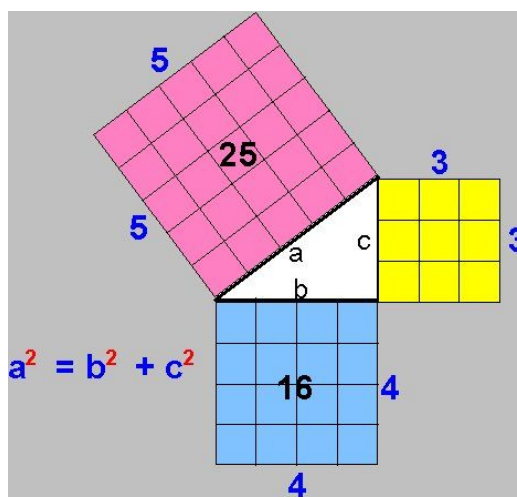
Você lembra como calcular a hipotenusa de um triângulo retângulo?

Primeiro, vou te relembrar o que é um triângulo-retângulo. Um triângulo em que um dos ângulos tem 90° . Ou seja, dois lados do triângulo são perpendiculares entre si. Esses

lados que formam o ângulo de 90° (ou ângulo reto) são chamados de "catetos". E o lado oposto ao ângulo reto é a hipotenusa.



Quando conhecemos o tamanho dos catetos nós conseguimos calcular o tamanho da hipotenusa. Este é o famoso **teorema de Pitágoras** que diz: **A soma dos quadrados dos catetos equivale ao quadrado da hipotenusa**. A imagem abaixo ilustra bem isso.



Então para descobrir o valor da hipotenusa, temos que encontrar a raiz quadrada de $(b^2 + c^2)$.

Com base neste cálculo, vamos fazer um algoritmo que solicita ao usuário o valor dos dois catetos, calcula e apresenta na tela o valor da hipotenusa do triângulo retângulo. Para isso precisaremos usar a função RAIZQ do Visualg para calcular a raiz quadrada pra gente.

```
algoritmo "Hipotenusa"
var
    a, b, c : REAL
inicio

    ESCREVA ("Digite o valor do primeiro cateto do triângulo retângulo: ")
    LEIA (b)
    ESCREVA ("Digite o valor do segundo cateto do triângulo retângulo: ")
    LEIA (c)

    a := RAIZQ ( b*b + c*c )//Cálculo da hipotenusa utilizando a FUNÇÃO RAIZQ,

    ESCREVA ("O valor da hipotenusa é: ", a)

fimalgoritmo
```

Observe que utilizamos a função RAIZQ para calcular a raiz quadrada do valor que passamos como parâmetro (valor entre parênteses) " $b*b + c*c$ ", o valor retornado por essa função armazenamos na variável "a".

Como criar as suas próprias funções e procedimentos

Você também pode criar as suas próprias funções e procedimentos. Entre as vantagens de criar as próprias funções e procedimentos cito duas, melhora a

legibilidade do código, tirando complexidades de dentro do fluxo principal do seu algoritmo e remove repetição de código.

Abaixo a sintaxe para criação das suas próprias funções e procedimentos no Visualg.

```
funcao <nome-de-função> [(<seqüência-de-declarações-de-parâmetros>)] : <tipo-de-dado>
// Seção de Declarações Internas
inicio
// Seção de Comandos
fimfuncao
```

```
procedimento <nome-de-procedimento> [(<seqüência-de-declarações-de-parâmetros>)]
// Seção de Declarações Internas
inicio
// Seção de Comandos
fimprocedimento
```

Vamos criar e usar uma função pra praticar. Vamos criar uma função que recebe um número inteiro e retorna o fatorial deste número.

Fatorial é a multiplicação de todos os números entre 1 e o número especificado.

Exemplo: Fatorial de 5 (ou 5!) corresponde a: $1 * 2 * 3 * 4 * 5 = 120$

Então vamos ver como ficaria esta função.

```
funcao calculaFatorial(numero: inteiro): inteiro
var
    fatorial: inteiro
    contador: inteiro
inicio
    fatorial <- 1
    ENQUANTO numero > 1 FACA
        fatorial <- fatorial * numero
        numero <- numero - 1
```

```
FIMENQUANTO
    retorne fatorial
fimfuncao
```

O fluxo principal do nosso Algoritmo poderia ser assim.

```
ESCREVA("Informe o número para o cálculo do Fatorial: ")
LEIA(numeroParaFatorial)
ESCREVA("O fatorial de ", numeroParaFatorial, " é: ",
calculaFatorial(numeroParaFatorial))
```

Esse é o algoritmo completo, com a função e o fluxo principal.

```
algoritmo "Cacula Fatorial"
var

    numeroParaFatorial: inteiro

funcao calculaFatorial(numero: inteiro): inteiro
var
    fatorial: inteiro
    contador: inteiro
inicio
    fatorial <- 1
    ENQUANTO numero > 1 FACA
        fatorial <- fatorial * numero
        numero <- numero - 1
    FIMENQUANTO
    retorne fatorial
fimfuncao

inicio

    ESCREVA("Informe o número para o cálculo do Fatorial: ")
    LEIA(numeroParaFatorial)
    ESCREVA("O fatorial de ", numeroParaFatorial, " é: ",
calculaFatorial(numeroParaFatorial))
```

fimalgoritmo

Resumindo

Vimos neste capítulo que **Funções** e **procedimentos** são "subalgoritmos" que podem ser chamados dentro de outros algoritmos.

São utilizados com muita frequência em desenvolvimento de softwares. Existem vários benefícios como: evita duplicação de código quando precisamos executar a mesma operação várias vezes, deixa o entendimento do algoritmo mais intuitivo, pois tiramos a parte complexa do código do fluxo principal do algoritmo, etc.

Importante: em linguagens orientada a objeto como java, C++ e C#, funções e procedimentos são chamados de **MÉTODOS**. Mais por uma questão de conceito de Orientação a Objetos, mas no fundo é a mesma coisa, podem receber parâmetros e retornam ou não um resultado.

Solução do exercício do capítulo 2

Se não conseguiu fazê-lo, não tem problema. Eu pedi para você solicitar as 4 notas do usuário, calcular a média e apresentar na tela. Neste capítulo você aprendeu sobre variáveis e os tipos de dados. Para resolver este exercício você precisará criar 5 variáveis do tipo real, 4 variáveis para armazenar as 4 notas e uma para armazenar a média.

Em seguida nós devemos solicitar ao usuário que digite as notas e armazená-las nas respectivas variáveis.

O passo seguinte é o cálculo da média, ou seja, a soma das 4 notas dividido por 4. Repare que precisamos colocar as somas entre parênteses, pois os operadores de multiplicação e divisão têm precedência quanto aos operadores de soma e subtração. Você vai aprender um pouco mais sobre os operadores no próximo capítulo.

O resultado do cálculo é armazenado na variável "media". Por fim, apresentamos a média na tela para o usuário.

Aqui está o meu algoritmo:

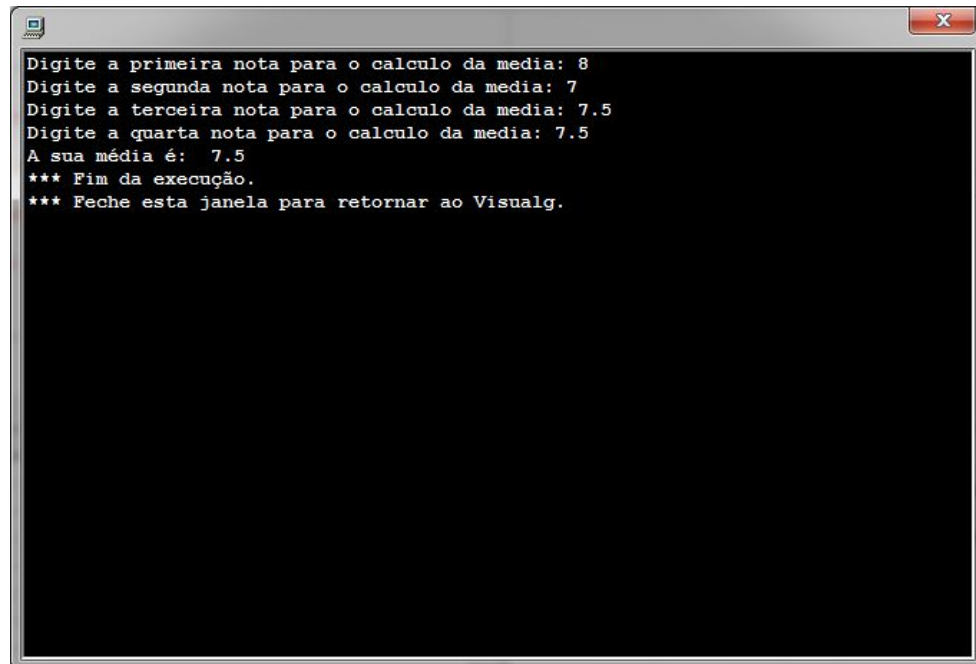
```
algoritmo "MédiaAnoLetivo"
var
    nota1, nota2, nota3, nota4, media : real
inicio
    escreva("Digite a primeira nota para o calculo da media: ")
    leia(nota1)
    escreva("Digite a segunda nota para o calculo da media: ")
    leia(nota2)
    escreva("Digite a terceira nota para o calculo da media: ")
    leia(nota3)
    escreva("Digite a quarta nota para o calculo da media: ")
    leia(nota4)

    media <- ( nota1 + nota2 + nota3 + nota4 ) / 4

    escreva("A sua média é: ", media)
```


fimalgoritmo

Apresento abaixo o resultado da execução deste algoritmo.



```
Digite a primeira nota para o calculo da media: 8
Digite a segunda nota para o calculo da media: 7
Digite a terceira nota para o calculo da media: 7.5
Digite a quarta nota para o calculo da media: 7.5
A sua média é: 7.5
*** Fim da execução.
*** Feche esta janela para retornar ao Visualg.
```

Solução do exercício do capítulo 5

No final do capítulo 5 eu pedi pra você tentar resolver um exercício de lógica para verificar se um aluno foi aprovado ou reprovado no final do ano.

Você fez? Espero que sim! Teve alguma dificuldade? Bom, abaixo eu mostro como eu escrevi um algoritmo para resolver esse exercício. Compare com o que você fez. Se o seu não deu certo, continue lendo que eu explico cada parte do algoritmo.

Esse é o algoritmo:

```
algoritmo "AprovacaoFinalDeAno"
var
    nota1, nota2, nota3, nota4, media: real
inicio

    escreva("Informe a nota (de 0 a 10) do primeiro bimestre: ")
    leia(nota1)
    escreva("Informe a nota (de 0 a 10) do segundo bimestre: ")
    leia(nota2)
    escreva("Informe a nota (de 0 a 10) do terceiro bimestre: ")
    leia(nota3)
    escreva("Informe a nota (de 0 a 10) do quarto bimestre: ")
    leia(nota4)

    media := (nota1 + nota2 + nota3 + nota4) / 4

    escreval("Sua média foi: ", media)

    se media >= 6 entao
        escreva("Você foi APROVADO!")
    senao
        escreva("Você foi REPROVADO!")
    fimse

fimalgoritmo
```

Entendendo o algoritmo. Primeiro eu declarei 5 variáveis do tipo REAL. Elas têm que ser do tipo REAL porque as notas podem ter valores decimais, por exemplo 5.5.

Depois eu escrevi na tela "Digite a nota (de 0 a 10) do primeiro bimestre: " e armazenei na variável nota1 o valor que o usuário digitou. Fiz o mesmo para as outras 3 notas.

Na sequência e calculei a média das 4 notas e armazenei o resultado na variável "media". Importante colocar o parênteses para somar as notas ANTES de dividir por 4.

Agora que vem a parte da decisão, o SE-ENTÃO-SENÃO.

Eu verifiquei se a média é MAIOR OU IGUAL a 6. Se SIM ENTÃO imprimi na tela a mensagem informando que o aluno foi aprovado. SENÃO imprimi a mensagem informando que o aluno foi reprovado.

Veja abaixo o resultado da execução do algoritmo no Visualg, quando a média era menor que 6 e quando foi maior.

Viu como foi simples? Se você teve dificuldades para resolver, não se preocupe. No início parece difícil mesmo. Mas como sempre digo, é preciso praticar!

Se conseguiu resolver sem dificuldades ótimo, mas continue praticando.

Solução do exercício do capítulo 6

Espero que você tenha tentado fazer esse exercício sozinho. Afinal, treinar é muito importante.

Eu escrevi um algoritmo com a solução deste exercício usando a estrutura ESCOLHA-CASO. Dê uma olhada:

```
algoritmo "Posição da letra no alfabeto"
var
    letra : CARACTERE
    posicao : INTEIRO
inicio

    ESCREVA("Digite uma letra: ")
    LEIA(letra)

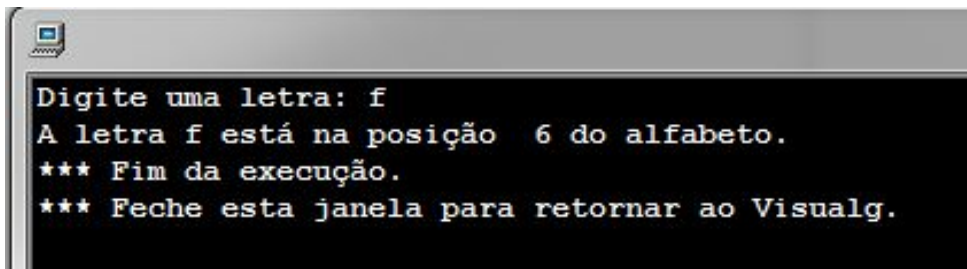
    ESCOLHA letra
        CASO "a"
            posicao := 1
        CASO "b"
            posicao := 2
        CASO "c"
            posicao := 3
        CASO "d"
            posicao := 4
        CASO "e"
            posicao := 5
        CASO "f"
            posicao := 6
        CASO "g"
            posicao := 7
        CASO "h"
            posicao := 8
        CASO "i"
            posicao := 9
        CASO "j"
            posicao := 10
        CASO "k"
            posicao := 11
        CASO "l"
```

```
        posicao := 12
CASO "m"
        posicao := 13
CASO "n"
        posicao := 14
CASO "o"
        posicao := 15
CASO "p"
        posicao := 16
CASO "q"
        posicao := 17
CASO "r"
        posicao := 18
CASO "s"
        posicao := 19
CASO "t"
        posicao := 20
CASO "u"
        posicao := 21
CASO "v"
        posicao := 22
CASO "w"
        posicao := 23
CASO "x"
        posicao := 24
CASO "y"
        posicao := 25
CASO "z"
        posicao := 26
FIMESCOLHA
```

```
        ESCREVA("A letra ", letra, " está na posição ", posicao, " do alfabeto.")
```

```
fimalgoritmo
```

Olha um resultado da execução deste algoritmo:



É possível implementar um algoritmo com a estrutura SE-ENTÃO-SENÃO, mas ficaria bem maior. Veja só o início deste algoritmo:

```
algoritmo "Posição da letra no alfabeto com SE"
var
    letra : CARACTERE
    posicao : INTEIRO
inicio

    ESCREVA("Digite uma letra: ")
    LEIA(letra)

    SE letra = "a" ENTÃO
        posicao := 1
    SENÃO
        SE letra = "b" ENTÃO
            posicao := 2
        SENÃO
            SE letra = "c" ENTÃO
                posicao := 3
            SENÃO
                SE letra = "d" ENTÃO
                    posicao := 4
                SENÃO
                    SE letra = "e" ENTÃO
                        posicao := 5
                    SENÃO
                        SE ....
                        .....
                    FIMSE
                FIMSE
            FIMSE
        FIMSE
    FIMSE
FIMSE
```

```
    ESCREVA("A letra ", letra, " está na posição ", posicao, " do  
alfabeto.")  
  
finalgoritmo
```

Bom agora eu tenho uma surpresa pra você! Se você acompanhou esse exercício até aqui e está gostando do e-book, eu vou te ensinar como fazer todo o trabalho dessa estrutura ESCOLHA-CASO com apenas UMA LINHA de código e sem usar nenhuma estrutura de controle de fluxo!

Isso mesmo, um algoritmo que diz a ordem da letra no alfabeto sem usar nenhuma estrutura de controle de fluxo como você aprendeu nas duas últimas lições. Ficou curioso? A malandragem é a seguinte...

Na computação, todos os caracteres tem um correspondente numérico para que este caractere possa ser armazenado na forma de bits.

Existe uma tabela chamada **Tabela ASCII** para sabermos qual o número de uma letra. E as letras do alfabeto estão em sequência nesta tabela.

Veja abaixo uma parte da tabela ASCII e identifique o valor numérico do caractere "a".

Número	Caractere	Número	Caractere	Número	Caractere	Número	Caractere	Número	Caractere	Número	Caractere
32	ESPAÇO	48	0	64	@	80	P	96	·	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	\$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(56	8	72	H	88	X	104	h	120	x
41)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	
45	-	61	=	77	M	93]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	96	_	111	o	127	DELETE

Viu que o valor do caractere "a" é 97 e que as outras letras estão na sequência? b = 98, c = 99, d = 100, ...

Agora ficou fácil, só precisamos descobrir o valor da letra que o usuário digitou e subtrair 96. Certo?

Para descobrir o valor ASCII de um caractere no Visualg, podemos utilizar a *função ASC*, passando como parâmetro a letra que o usuário digitou.

Importante! Vamos falar mais sobre **funções** e **procedimento** no capítulo 10, não se preocupe. Por hora só saiba que uma função executa uma tarefa pra gente. (Talvez a função ASC use a estrutura ESCOLHA-CASO internamente para retornar o número ASCII da letra.)

A função `ASC(caracter)` retorna o número da tabela ASCII da letra que passamos como parâmetro.

Logo, o nosso algoritmo ficaria assim:

```
algoritmo "Posição da letra no alfabeto"
var
```



```
letra : CARACTERE
posicao : INTEIRO
inicio

    ESCREVA("Digite uma letra: ")
    LEIA(letra)

    posicao := ASC(letra) - 96

    ESCREVA("A letra ", letra, " está na posição ", posicao, " do
alfabeto.")

finalgoritmo
```

O resultado é o mesmo do algoritmo que usa a estrutura ESCOLHA-CASO ou SE-ENTÃO-SENÃO.

Solução do exercício do capítulo 7

No final do capítulo 7 eu pedi pra você resolver um exercício criando um algoritmo capaz de fazer multiplicação de dois números positivo.

Espero que você tenha tentado fazer sozinho heim! Se não fez, tente fazer primeiro pra depois olhar a resposta que apresento abaixo.

Eu mostrei esse algoritmo no primeiro capítulo deste e-book. Lembra? Talvez naquele momento você não tenha compreendido direito, mas agora você já tem o conhecimento mínimo para fazer um algoritmo de multiplicação. Aqui está o meu algoritmo de multiplicação entre números positivos:

```
algoritmo "Multiplicação"
var
    numero1, numero2, resultado, contador: INTEIRO
inicio
    ESCREVA("Informe o primeiro número: ")
    LEIA(numero1)
    ESCREVA("Informe o segundo número: ")
    LEIA(numero2)

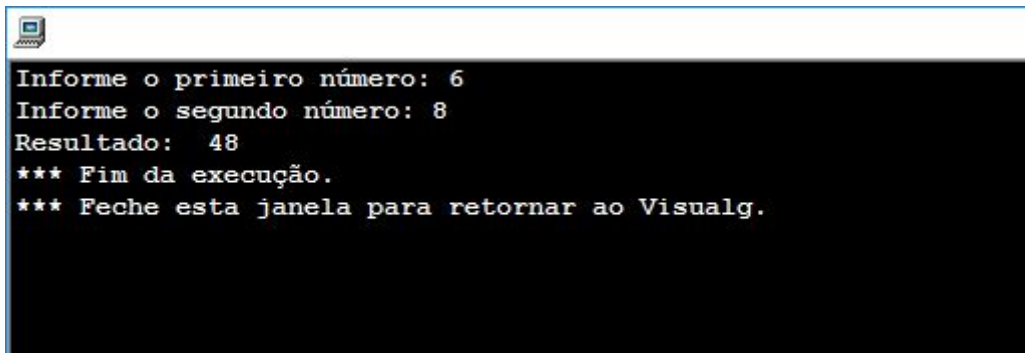
    contador <- 0
    resultado <- 0

    ENQUANTO ( contador < numero2 ) FACA
        resultado <- resultado + numero1
        contador <- contador + 1
    FIMENQUANTO

    ESCREVA("Resultado: ", resultado)

fimalgoritmo
```

Aqui um resultado da execução deste algoritmo.



```
Informe o primeiro número: 6
Informe o segundo número: 8
Resultado: 48
*** Fim da execução.
*** Feche esta janela para retornar ao Visualg.
```

Nesse algoritmo nós definimos como realizar uma multiplicação somando o *número1* a quantidade de vezes do *número2*. Exatamente como aprendemos na escola! A gente controla a execução do loop com a variável *contador*. Nós também podemos implementar esse algoritmo utilizando a estrutura PARA-FAÇA, que aprendemos neste capítulo.

Com a estrutura PARA-FAÇA, esse algoritmo ficaria assim:

```
algoritmo "MultiplicaçãoComParaFaca"
var
    numero1, numero2, resultado, contador: INTEIRO
inicio
    ESCREVA("Informe o primeiro número: ")
    LEIA(numero1)
    ESCREVA("Informe o segundo número: ")
    LEIA(numero2)

    resultado <- 0

    PARA contador DE 1 ATE numero2 FAÇA
        resultado <- resultado + numero1
    FIMPARA

    ESCREVA("Resultado: ", resultado)
```

fimalgoritmo

Solução do exercício do capítulo 8

Algoritmo de identificação de números primos

No final do capítulo 8 deste e-book de lógica de programação, eu pedi pra você resolver um exercício.

Fazer um algoritmo para dizer se um determinado número é primo ou não.

E aí, conseguiu fazer? Espero que você tenha tentado e conseguido fazer sozinho!

Se não conseguiu, tudo bem, com a prática você vai ficando craque na lógica de programação.

O problema é simples, como eu disse no capítulo anterior, um número primo só pode ser divisível (resto = 0) por 1 e por ele mesmo, ou seja, se ele for divisível por qualquer outro número entre 2 e ele mesmo menos 1, ele não é primo. Sacou?

Abaixo você vai ver o algoritmo que eu fiz para este problema.

```
Algoritmo "NumeroPrimo"
Var
    contador : INTEIRO
    numero : INTEIRO
    eprimo : LOGICO
Inicio
    ESCREVA("Informe um número para verificar se ele é primo: ")
    LEIA(numero)
    eprimo := VERDADEIRO
    PARA contador DE 2 ATÉ numero-1 FAÇA
        SE (numero MOD contador) = 0 ENTAO
```

```
        eprimo := FALSO
    FIMSE
FIMPARA
SE eprimo = VERDADEIRO ENTAO
    ESCREVA("O número ", numero, " é primo!")
SENAO
    ESCREVA("O número ", numero, " NÃO é primo!")
FIMSE
Fimalgoritmo
```

Neste algoritmo eu faço um *loop* de 2 até o número imediatamente anterior ao número que estou verificando se é primo. Por quê?

Bom, eu já expliquei que um número primo só é divisível (resto 0) por 1 e por ele mesmo. Ele não deve ser divisível por qualquer outro número.

Então, caso algum número entre 2 e "numero-1" seja capaz de dividir o número verificado com resto zero (SE (numero MOD contador) = 0 ENTÃO ...), significa que ele **não** é primo (eprimo = FALSO).

Aqui um resultado da execução deste algoritmo.

```
Início da execução
Informe um número para verificar se ele é primo: 53
O número 53 é primo!
Fim da execução.
```

O que você achou? Gostou da minha resolução? O seu algoritmo pode ter sido diferente. Não tem problema. Há várias formas de se fazer algoritmos. Não precisa estar igual o meu. Só precisa funcionar corretamente. ;)

Aliás essa é a beleza da lógica de programação.

Solução do exercício do capítulo 9

Algoritmo do jogo da velha

No capítulo 9 te desafiei a fazer um algoritmo do jogo da velha.

Não é um algoritmo facinho, mas já estamos chegando ao final deste e-book e sei que você tem capacidade de criar este algoritmo sozinho.

Mas você tinha que quebrar a cabeça um pouquinho. Neste exercício, que passei como um desafio para você, é necessário utilizar várias estruturas, operadores, variáveis, etc. Tudo que já aprendemos nas lições anteriores.

Então vamos ao meu algoritmo do jogo da velha, mas antes vamos lembrar as regras:

1 - As jogadas do jogo da velha deverão ser armazenadas numa matriz (3x3) de caractere, chamada "tabuleiro", cada posição desta matriz armazenará um dos valores: " ", "_", "X" ou "O". Abaixo uma representação gráfica desta matriz.

	1	2	3		
1	__		__		__
2	__		__		__
3					

2 - A cada jogada o programa deverá mostrar na tela a situação atual do "tabuleiro". Por exemplo:

```
  1   2   3
1  _ _ | _ _ | _ _
2  _ _ | _X_ | _ _
3  0  |   | 0
```

3 - Terão dois jogadores no jogo. O programa deve solicitar o nome dos dois jogadores antes de começar o jogo. A cada jogada o programa deverá perguntar separadamente as posições horizontal e vertical da jogada, nesta ordem.

4 - Quando um jogador vencer o jogo, o programa deve apresentar imediatamente o vencedor e a situação do "tabuleiro".

Abaixo você encontra o meu algoritmo do jogo, você pode copiar e colocar o algoritmo no VisuAlg.

Para baixar o Visualg Acesse:

<https://dicasdeprogramacao.com.br/download-visualg/>

Veja o algoritmo, entenda, [execute-o](#), observe porque usei cada estrutura PARA-FAÇA, REPIRA-ATÉ, SE-ENTÃO-SENÃO, cada variável, operadores lógicos (E e OU) etc.

```
algoritmo "JogoDaVelha"
var
    tabuleiro: vetor[1..3,1..3] de caractere
    nomeJogador1, nomeJogador2, jogadorAtual, vencedor : caractere
    linhaJogada, colunaJogada, i, j : inteiro
inicio
    escreval("###Jogo da velha###")

    //Inicialização do tabuleiro com "_" nas linhas 1 e 2 e " " na terceira linha
    //i é a linha e j é a coluna.
    para j de 1 ate 3 faca
        para i de 1 ate 2 faca
            tabuleiro[i,j] := "_"
```



```
fimpara
    tabuleiro[3,j] := " "
fimpara

escreva("Informe o nome do(a) primeiro(a) jogador(a): ")
leia(nomeJogador1)
escreva("Informe o nome do(a) segundo(a) jogador(a): ")
leia(nomeJogador2)

escreval("Vamos começar o jogo.")

jogadorAtual := nomeJogador1
vencedor := ""

repita
    //Apresenta a situação atual do tabuleiro
    escreval("Neste momento o tabuleiro está assim:")
    escreval("  1   2   3")
    escreval("1 _", tabuleiro[1,1], "_|_", tabuleiro[1,2], "_|_", tabuleiro[1,3],
" _")
    escreval("2 _", tabuleiro[2,1], "_|_", tabuleiro[2,2], "_|_", tabuleiro[2,3],
" _")
    escreval("3  ", tabuleiro[3,1], " | ", tabuleiro[3,2], " | ", tabuleiro[3,3], "
")

    escreval("É a vez do(a) jogador(a): ", jogadorAtual)

repita
    repita
        escreva("Informe o número da linha da sua jogada: ")
        leia(linhaJogada)
        //Valida se o usuário digitou um valor válido
        se (linhaJogada < 1) ou (linhaJogada > 3) entao
            escreval("A linha deve ser entre 1 e 3")
        fimse
    ate (linhaJogada >= 1) e (linhaJogada <= 3)
    repita
        escreva("Informe o número da coluna da sua jogada: ")
```

```
        leia(colunaJogada)
        //Valida se o usuário digitou um valor válido
        se ((colunaJogada < 1) ou (colunaJogada > 3)) entao
            escreval("A coluna deve ser entre 1 e 3")
        fimse
    ate ((colunaJogada >= 1) e (colunaJogada <= 3))

    //Valida se a posição jogada á está preenchida
        se (tabuleiro[linhaJogada,colunaJogada] <> "_") e
(tabuleiro[linhaJogada,colunaJogada] <> " ") entao
            escreval("A posição ", linhaJogada, ", ", colunaJogada, " já está
preenchida.")
        fimse
    ate (tabuleiro[linhaJogada,colunaJogada] = "_") ou
(tabuleiro[linhaJogada,colunaJogada] = " ")

    se jogadorAtual = nomeJogador1 entao
        tabuleiro[linhaJogada,colunaJogada] := "X"
        jogadorAtual := nomeJogador2
    senao
        tabuleiro[linhaJogada,colunaJogada] := "O"
        jogadorAtual := nomeJogador1
    fimse

    //Valida se alguém ganhou o jogo
    para j de 1 ate 3 faca
        //Verifica as colunas
        //_X_|_O_|_X_
        //_X_|_O_|_X_
        //_X_|_O_|_X_
        se ((tabuleiro[1,j] = "X") ou (tabuleiro[1,j] = "O")) e (tabuleiro[1,j] =
tabuleiro[2,j]) e (tabuleiro[2,j] = tabuleiro[3,j]) entao
            vencedor := tabuleiro[1,j]
        fimse
    fimpara

    para i de 1 ate 3 faca
        //Verifica as linhas
        //_X_|_X_|_X_
```

```
//_o_|_o_|_o_
// x | x | x
    se ((tabuleiro[i,1] = "X") ou (tabuleiro[i,1] = "O")) e (tabuleiro[i,1] =
tabuleiro[i,2]) e (tabuleiro[i,2] = tabuleiro[i,3]) entao
        vencedor := tabuleiro[i,1]
    fimse
fimpara

//Verifica as diagonais
//_X_|__|_X_
//__|_X_|__
// x |   | x
    se (((tabuleiro[2,2] = "X") ou (tabuleiro[2,2] = "O")) e ((tabuleiro[1,1] =
tabuleiro[2,2]) e (tabuleiro[2,2] = tabuleiro[3,3])) ou ((tabuleiro[3,1] =
tabuleiro[2,2]) e (tabuleiro[2,2] = tabuleiro[1,3]))) entao
        vencedor := tabuleiro[2,2]
    fimse

//Verifica se deu velha
    se ((vencedor <> "") e (tabuleiro[1,1] <> "_") e (tabuleiro[1,2] <> "_") e
(tabuleiro[1,3] <> "_") e (tabuleiro[2,1] <> "_") e (tabuleiro[2,2] <> "_") e
(tabuleiro[2,3] <> "_") e (tabuleiro[3,1] <> " ") e (tabuleiro[3,2] <> " ") e
(tabuleiro[3,3] <> " ")) entao
        vencedor := "V"
    fimse

ate vencedor <> ""

//Apresenta a situação atual do tabuleiro
escreval("Neste momento o tabuleiro está assim:")
escreval("  1   2   3")
escreval("1 _", tabuleiro[1,1], "_|_", tabuleiro[1,2], "_|_", tabuleiro[1,3], "_")
escreval("2 _", tabuleiro[2,1], "_|_", tabuleiro[2,2], "_|_", tabuleiro[2,3], "_")
escreval("3  ", tabuleiro[3,1], " | ", tabuleiro[3,2], " | ", tabuleiro[3,3], " ")

se vencedor = "X" entao
    escreva("O vencedor do jogo foi: ", nomeJogador1)
```

```
fimse
se vencedor = "O" entao
    escreva("O vencedor do jogo foi: ", nomeJogador2)
fimse
se vencedor = "V" entao
    escreva("O jogo deu Velha!")
fimse

finalgoritmo
```

Se tiver dúvida, acesse as lições anteriores, onde falo sobre cada um desses assuntos.

Meu agradecimento

Não queria terminar este e-book sem agradecer a você por ter acompanhado este texto e por ter saído da sua zona de conforto para se dedicar a aprender uma coisa nova. Saiba que o mundo precisa de mais pessoas como você!

Espero que você utilize este conhecimento para o bem e contribua com o mundo criando tecnologia que melhore a vida das pessoas.

Obrigado!

Gustavo