Sumário

APRESENTAÇÃO

Este livro tem como objetivo ser utilizado em cursos de construção de algoritmos básico. O público-alvo são os professores e alunos de graduação dos cursos de computação. Foi elaborado para ser o primeiro contato dos alunos com a lógica de programação.A estratégia deste livro é introduzir os conhecimentos básicos de programação passo a passo. Supõe que o estudante nunca teve contato com programação. O foco é na lógica de programação e nas regras que estão presentes na maioria das linguagens de programação.

A fim de focar o aprendizado apenas na lógica de programação e não em alguma linguagem ou ferramentas, este livro foi concebido de tal maneira que todos os exercícios sejam resolvidos sem o uso de computadores. Sugere-se que apenas após a resolução dos exercícios deste livro ou durante a parte final deste livro, o professor introduza alguma linguagem de programação, como C ou Pascal. Assim, os alunos que tem maior facilidade se sentem mais estimulados. É importante que o professor explique para os alunos que arte de construir algoritmos tem como maior dificuldade a lógica de programação, a conversão do português estruturado para uma determinada linguagem é mais fácil. Claro que sempre os alunos devem ser estimulados a procurar material complementar a fim de desenvolver os algoritmos em uma determinada linguagem.

Espera-se que no final do livro os alunos estejam com uma base sólida em linguagem de programação, facilitando na seqüência o aprendizado da construção dos mesmos problemas em uma linguagem de programação específica e o conteúdo de problemas mais complexos. Assim, neste segundo momento o professor terá como foco a linguagem de programação, ferramentas e tópicos mais complexos, e não mais lógica de programação básica.

Para o professor

Ensinar a construir algoritmos é um grande desafio para os professores de computação. Esta matéria exige muito do professor, porém também gera grande satisfação. Principalmente, quando os alunos começam a vibrar por terem conseguido a resolver os problemas e até a sonhar com estes. É notório que alguns alunos têm maior dificuldade do que outros, porém a determinação de cada um é o fator que faz com que o aluno atinja o objetivo. Nestes anos de ensino desta matéria notei que os alunos que se dedicam na matéria e que nunca desistem, atingem sempre os objetivos. Além disso, é muito importante que o professor sempre esteja atento a cada aluno e que esclareça todas as dúvidas. Em algumas situações, exige-se que o professor resolva o mesmo problema de várias formas a fim de comparar a soluções para os alunos.

A experiência do autor mostra que todos os algoritmos devem ser resolvidos e testados com alunos. Deve-se conscientizar os alunos da importância de resolver os exercícios antes da resolução dos mesmos em classe. Nestes anos o autor notou que uma das grandes dificuldades do aluno é entender o problema, por isso, todos os exercícios apresentados são explicados detalhadamente antes da resolução. Esta dificuldade dos alunos é originada pela falta de experiência na área, então no decorre do livro espera-se que haja um amadurecimento dos alunos, pois este será necessário para a resolução de problemas nos curso de programação seguinte.

Para o aluno

Aprender a construir algoritmos é fundamental para ser um profissional da área de Computação. Por isso, exige-se que muita dedicação. Sugiro que o aluno resolva todos os exercícios do livro de maneira seqüencial, pois o grau de dificuldade aumentará a cada novo algoritmo. Então, para resolver um determinado algoritmo, deve-se, geralmente, saber resolver todos os anteriores.

Antes de começar a resolver o problema, deve-se entendê-lo, somente após comece a resolver. Após e durante a resolução, utilize sempre o teste de mesa (ferramenta que ajuda a verificar se o algoritmo está correto – será apresentado posteriormente). O segredo para aprender a algoritmos é construir muitos.

Inicialmente, pode-se pensar que o raciocínio necessário para resolver os algoritmos deste livro são novos e difíceis, mas não são. A qualidade principal para o sucesso neste conteúdo é a persistência. A minha sugestão é a seguinte: caso não consiga solucionar o problema na primeira tentativa, não desista, pense mais um pouco; certifique-se de que entendeu o enunciado do problema. As soluções são muito semelhantes umas as outras. Além disso, execute os testes que serão ensinados no decorrer deste livro. Os testes são capazes de guiar você até a solução final. Por fim, lembre-se que quando você acordar no meio da noite com as idéias para resolver os problemas, então estará colocando a quantidade apropriada do esforço nesta disciplina. Além disso, recomendo que você venha para aula com os algoritmos que serão explicados pelo professor já resolvidos. Assim, será possível discutir melhorias e outras soluções.

Organização do livro

Este livro está organizado em vários capítulos. Vários exercícios de fixação são apresentados em cada capítulo. No início de cada assunto explica-se em detalhes o assunto e introduz um exemplo. Logo após, apresenta-se diversos problemas de fixação. Explica-se para cada problema o seu objetivo e realiza-se os comentários.

O Capítulo 1 apresenta os fundamentos da construção dos algoritmos. Ele faz a revisão de alguns conceitos de lógica e de matemática que são utilizados em computação. Além disso, apresenta alguns conceitos básicos da programação, como o conceito de variáveis e atribuição de valores.

O Capítulo 2 mostra o primeiro algoritmo, os comandos, as principais regra e o teste de mesa. Além disso, algumas características de um bom algoritmo.

O Capítulo 3.

O Capítulo 4 apresenta uma importante ferramenta computacional (estrutura de dados) conhecida como vetores e o capítulo 6 outra, que é denominada Matrizes. Por fim, o capítulo 6 apresenta a resolução de todos os problemas.

Termos definidos são apresentados em **negrito.** Definições importantes são apresentadas da seguinte forma:

Algoritmo é a seqüência de procedimentos que levam à resolução de um problema.

---

1. FUNDAMENTOS

Este livro tem como objetivo demonstrar técnicas para resolução de problemas computacionais. O aprendizado de algoritmos é essencial para a formação de um bom programador, servindo como base para o aprendizado de todas as linguagens de programação. A maneira de se aprender algoritmos é **construindo** e **testando** algoritmos. A seguir alguns termos básicos da área.

> **Programação** é a arte de construir algoritmos, representando-os através de uma linguagem de programação pré-existente, visando a sua execução no computador.

> **Algoritmo** é a seqüência de procedimentos que levam à resolução de um problema.

> Um **procedimento** é um conjunto de um ou mais comandos, com objetivo definido. Os comandos podem ser do tipo elementar ou de controle, podendo constituir-se de palavras-chave, argumentos ou expressões.

A Computação tem como base a Matemática. Por isso, a Matemática faz parte do cotidiano dos profissionais da área. A seção seguinte faz uma revisão de alguns fundamentos matemáticos e introduz alguns conceitos básicos de programação que serão utilizados na construção dos algoritmos.

1.1. Operadores matemáticos

Os operadores aritméticos permitem a realização das operações matemáticas básicas, usadas no cálculo de expressões aritméticas. A Figura 1 mostra uma tabela com operadores. A primeira coluna está a ordem na qual as operações devem ser realizadas. Por exemplo, na expressão 4 + 2 * 3, primeiro a multiplicação deve ser realizada (2 * 3) e, em seguida, a somatória (4+6), o que resultará no valor 10.

Ordem	Operador	Significado	Exemplo
1°	**	Exponenciação	2 ** 3 = 8
2°	*	Multiplicação	4 * 3 = 12
	/	Divisão	10 / 2 = 5
3°	+	Adição	7+ 1 = 8
	-	Subtração	9 – 2 = 7

Figura 1 – Operadores matemáticos

Somente o uso de parênteses pode quebrar esta ordem de prioridade. Se numa mesma expressão existir parênteses aninhados (um dentro do outro), a prioridade será do parêntese mais

interno. Então, na expressão (4 + 2)* 3, primeiro será realizada a soma (4+2) e depois a multiplicação (6 * 3), que resultará 18.

Um fato importante em relações às expressões aritméticas na construção de algoritmos é que estas devem ser linearizadas, ou seja, colocadas em linhas.

Por exemplo, a expressão $\frac{2+4}{2+1}$, deverá ser escrita da seguinte forma : (2+4) / 2 +1

As operações aritméticas são sempre executadas com números. De maneira geral, os tipos de números que são utilizados pela maioria das linguagens de programação são os inteiros e os reais. Porém, é comum que cada linguagem de programação defina os tipos de dados que ela suporta. Cada tipo de dado tem suas características: a faixa de números suportada – por exemplo, de -10000 até 10000 – ; se suporta a parte fracionária ou não; e a quantidade de memória exigida. Geralmente, os algoritmos estão restritos por dois tipos de números: os inteiros e os reais.

Os números **inteiros** são dados numéricos positivos ou negativos, excluindo-se destes qualquer número fracionário.

São exemplos de números inteiros: 10; -21; 23; -12; 8. Observe que os números negativos também fazem parte dos números inteiros.

Os números **reais** são dados numéricos positivos, negativos e números fracionários.

São exemplos de números reais: 12,3; 234,4; -45,5 ; 324,0. Pela definição nota-se que os números inteiros fazem partes dos reais. Como o programador busca sempre desenvolver algoritmos eficientes, então eles sempre declaram o tipo de dado a ser utilizado conforme o problema. Por exemplo, se o algoritmo tem como objetivo realizar o controle da quantidade de cadernos, então o programa utilizará números inteiros para realizar o armazenamento da quantidade. Porém, se for um controle de salários de funcionários, o algoritmo utilizará números reais.

É importante ressaltar que as variáveis dos algoritmos não trabalham apenas com números. Elas podem também trabalhar com outros tipos de dados, por exemplo, os alfanuméricos. Quando o dado é do tipo alfanumérico, ele pode ter como valor qualquer caracter (letra / número/ caracter especial). Por exemplo: “a/c-d”, “caio”,”b3”, “5”, “d2”, “;”, ”/”. Com este tipo de dado não é possível realizar operações, como: multiplicação, soma, divisão ou subtração. Porém, este tipo de dado suporta a operação concatenação, que é simbolizada pelo sinal de mais (+). Se for realizada a concatenação de “bo” + “la”, o resultado será “bola”; se for realizada a concatenação de “5”+”3”, o resultado será “53”. Note que os dados alfanuméricos sempre estarão entre aspas duplas. As aspas os diferenciam dos números.

Os dados **alfanuméricos** podem conter letras, números e caracteres especiais.

1.2. Operadores relacionais

Os operadores relacionais são utilizados para a comparação entre dois objetos do mesmo tipo. O resultado de uma operação relacional será sempre um valor do tipo lógico (falso ou verdadeiro).

Operador	Significado	Exemplo
>	Maior	8 > 2
<	Menor	12 < 43
>=	Maior ou igual que	11 >= 11
<=	Menor ou igual que	1 <= 23
!=	Diferente	90 != 23
==	Igual	7 == 7

Figura 2 – Operadores relacionais

Estes operadores serão utilizados em situações, por exemplo, quando deseja-saber "se o usuário tem idade maior ou igual a 18 anos".

1.3. Operadores lógicos

A lógica proposicional teve como fundador Aristóteles, que teve como objetivo modelizar o raciocínio humano. A lógica tem como base frases declarativas (proposições), que podem ser verdadeiras ou falsas. Por exemplo, a sentença "Noventa é menor do que cinco". Note que esta sentença pode ser verdadeira ou falsa, no caso é fala, portanto é uma proposição. Para a construção de algoritmos básicos será utilizado os conectivos ou (ou), e o e (e) e o não(!).

Por intermédio destes conectivos pode-se formar sentenças compostas. O valor lógico de uma proposição composta depende dos valores lógicos de seus componentes e dos conectivos usados. Nos algoritmos estes operadores serão utilizados para informar aos algoritmos se proposições são verdadeiras ou falsas. Exemplo de situações:

- "o usuário é do sexo feminino"

-"o usuário é do sexo feminino" e " usuário têm filhos"

-"o usuário é diretor" ou "usuário é presidente da empresa"

A Figura 3 mostra três tabelas, conhecidas como tabelas verdade, que tem como objetivo demonstrar a avaliação das preposições A e B, onde V significa que a sentença é verdadeira e F é falsa, então:

A		B	Resultado
V	E	V	V
V	E	F	F
F	E	V	F
F	E	F	F

A		B	Resultado
V	ou	V	V
V	ou	F	V
F	ou	V	V
F	ou	F	F

A	!A
V	F
F	V

Figura 3- Tabela verdade

Conforme mostra a Figura 4 existe entre os operadores lógicos uma ordem de prioridade de cálculo em uma expressão.

Ordem	Operador
1º	!
2º	e
3º	ou

Figura 4 – Prioridade dos operadores lógicos

1.4. Variáveis

Um computador precisa armazenar as informações para que consiga realizar as operações. Por exemplo, quando o usuário está utilizando a calculadora (que é um programa), acontece o seguinte (Figura 5):

	Usuário	**Computador**
1º	O usuário pressiona os números no teclado	Recebe os números do teclado e armazena (**guarda o dado na memória)**
2º	O usuário pressiona o operador	Recebe o operador do teclado e armazena (**guarda o operador na memória**)
3º	O usuário pressiona os números no teclado	Recebe os números do teclado e armazena (**guarda o dado na memória**)
4º	O usuário pressiona o sinal de igual (=) no teclado	Recebe o sinal de igual do teclado, então processa o cálculo e apresenta o resultado para o usuário

Figura 5 – Armazenamento na memória

Note que o computador armazena (guarda) os dados na memória. Este espaço de armazenamento de dados é denominado variável. O programa (algoritmo) que realizou estas operações utilizou uma variável para armazenar o primeiro número, outra para o segundo e outra para o resultado.

Variável é um termo usado para se referir a um único valor, que pode a qualquer momento ser substituído por outro. No português estruturado podem ser inteiras, reais ou alfanuméricas. Importante: para cada variável é definido o seu nome e tipo.

Em português estruturado não importa se os comandos forem escritos em **maiúsculo ou minúsculo**. Então, será a mesma variável a denominada IDADE e a idade. Algumas linguagens de programação não seguem esta regra, com C, e outras seguem, como Pascal.

Em português estruturado pode-se acentuar os comandos, porém quando se escreve programas nas linguagens de programação, como C, Pascal e Java não se deve acentuar as palavras. Isto ocorre porque os comandos das linguagens são em inglês, então não suporta acentos.

Uma variável deve ser criada sempre que for necessário armazenar valores. Assim, um programa que visa armazenar o nome e a idade do usuário deverá ter uma variável para armazenar o nome e outra para a idade. Quando o primeiro usuário informar os seus dados, estes serão armazenados nas respectivas variáveis, porém quando um segundo usuário informar os seus dados, estes sobrescreverão os dados do primeiro usuário. Para que os dados do primeiro usuário não sejam perdidos, deve-se criar outras variáveis ou antes de sobrescrever, deve-se armazenar os dados em outro locais, como, por exemplo, no disco.

Exemplificação:

Quando o programa deste problema é iniciado, ele solicita para o sistema operacional a criação de duas variáveis (Figura 6). A primeira delas será nomeada como *Nome* e será do tipo *alfanumérica*, a segunda será nomeada como *Idade* é será do tipo *Inteira*. Então, como a memória do computador é dividida em quadros, cada quadro é associado a uma das variáveis. Observe que na memória existem outros quadros (espaços para outras variáveis), porém este programa tem acesso somente às suas variáveis. Como o programa acabou de ser inicializado, então as variáveis não possuem valor.

Nome da variável: Nome Tipo: alfanumérica Valor: (vazio)	Nome da variável: Idade Tipo: inteira Valor: (vazio)	

Figura 6- Estado da memória quando o programa é inicializado

Quando o primeiro usuário informar os seus dados, os valores são atribuídos às respectivas variáveis.

Nome da variável: Nome Tipo: alfanumérica Valor: “Pedro”	Nome da variável: Idade Tipo: inteira Valor: 18	

Figura 7- Os dados do primeiro usuário são informados

Quando o segundo usuário informar os seus dados, os valores subscrevem os dados do primeiro usuário.

Nome da variável: Nome Tipo: alfanuméricas Valor: “Maria”	Nome da variável: Idade Tipo: inteiras Valor: 22	

Figura 8- Os dados do segundo usuário são informados

Regras criação de variáveis:

1. Toda variável tem um nome
    a. O nome da variável é único, ou seja, o nome já atribuído a uma variável não pode ser atribuída à outra;

b. O primeiro caracter do nome da variável deve ser sempre ser uma letra, ou seja, não pode ser um número. Então, o nome ABC para uma variável é válido, porém o nome 1ABC não é válido;

c. Os caracteres válidos para nomear as variáveis são: A…Z,0…9, e o caractere _. Então, uma variável pode ser nomeada como *Qtd_Tipo_4*. Porém, não é um nome válido *Qtd_/4* (o caractere barra não é aceito); outra variável que seria inválida seria a denominada *QTD Alunos* (não pode ter espaço em branco no nome da variável.

d. As variáveis não podem ser nomeadas com palavras chaves. As palavras chaves são aquelas reservadas pela linguagem de programação. Na linguagem de programação utilizada neste livro, português estruturado, não permite, por exemplo, que uma variável seja nomeada como inteiras, alfanuméricas ou reais. As outras palavras chaves serão apresentadas no decorrer do livro.

e. A fim de desenvolver um algoritmo de boa qualidade sugere-se (não é obrigatório) associar o nome da variável com problema. Por exemplo, se a variável tem como objetivo armazenar a idade do aluno, então se deve nomeá-la como *idade* e não como *carro*.

2. Todas variáveis têm um tipo: reais, inteiras ou alfanuméricas.
3. Cada variável armazena somente um valor a cada momento. Entretanto, este valor pode ser alterado no decorrer da execução do algoritmo.

1.5. Exercícios de fixação

1. Calcule o resultado de cada expressão

a. (13) 5 + 2 * 4

b. (-17) -2 – 5 * 3

c. (16) 4 ** 2

d. (2,5) 3 – 1 / 2

e. (13) 10 / 2 + 4 * 2

f. (-8) -4 * 3 / 2 – 2

g. (8) 2 ** 2 * 2

h. (-15) -2 -3 -5 *2

i. (16) +2*2**3

2. Coloque I para os números inteiros, R para os números reais e A para os alfanuméricos. Desconsidere que os números reais fazem parte dos inteiros.

a. (R) 8,51

b. (A) “32f”

c. (I) 5

d. (R) -4,42
e. (A)” -4,42”
f. (I) -32
g. (R) 9,00
h. (A) “manga”
i. (R) 3,10
j. (A) “32452,233”

3. Indique se a expressão é verdadeira (V) ou falso (F)

a. (V) 3 < 4
b. (F) 5 = 6
c. (F) 7 >= 9
d. (F) 8 <= 8,0
e. (V) 9,00==9

4. Dado que a proposição M é verdadeira e N é falsa, indique se é o é verdadeira (V) ou (F)

a. (F) M e N
b. (V) M ou N
c. (V) N e N
d. (V) M ou M
e. (V) (M e N) ou M
f. (V) M e (N ou M)
g. (F) N e (M ou N)

5. Assuma que todas as variáveis são do tipo inteiras, encontre o valor de cada uma delas:

a. (9) x ← (2+1)*3;
b. (2) M ← 2+1/2;
c. (9) y ← (5+1)/2*3;
d. (11) I ← 2+(3*3);
e. (11) H← 8+7/2;

Observação: A seta (←) indica que o valor da expressão está sendo atribuído à variável

6. Quais são os valores de cada variável nas seguintes expressões?Inteiras a=1, b=2, c =3, d =4;

a. (3) a ← a +b;
b. (7) b ← c *2+a;
c. (13) d ← d /a + 3 * c;
d. (-2) c ← d + a – d + a – d;

7. Qual é o valor das seguintes expressões? (Verdadeiro – V ou Falso - F)

a. (F) 1 >2
b. (F) 5==6
c. (V) 8!= 9
d. (V) 4 < (3+2)
e. (V) 7>2

8. Quais são os valores de cada variável resultante nas seguintes seqüências de comandos?

a) Seqüência de comandos

a ←1;
b←2;
c←3;
d←4;
b ← b –c;
a ← a +b;

Resultado: a = -1 b = 1 c = 3 d = 4

b) Seqüência de comandos

a ←1;
b←2;
c←3;
d←4;
a ← a *c;
a ← a +b;
b ← b –c;
c ← b-c;

Resultado: a = 6 b = -1 c = -4 d = 4

c) Seqüência de comandos

a ←1;
b←2;
c←3;
d←4;
c ← b-c;
a ← a *c;
a ← a +b;
a ← a +b;

Resultado: a = 3 b = 2 c = -1 d = 4

d) Seqüência de comandos

a ←1;
b←2;
c←3;
d←4;
c ← b-c;
a ← a *c;
c ← b+a;
a ← a +b;

Resultado: a = 1 b = 2 c = 1 d = 4

9. Dadas as variáveis x=1, a=3, b=5, c=8 e d = 7, assinale a coluna (Verdadeiro ou Falso) conforme a condição apresentada.

Condição	Verdadeiro	Falso
(b<5)		
(x>3)		
(x<1) e (b>d)		
(d<0) e (c>5)		
(x>3) ou (c<7)		
(a>b) ou (c>b)		
(x>=2)		
(x<1) e (b>=d)		
(a<>b)		
(c==d)		
(c<=7)		
(c<a) e (b>c) e (b==5)		
(x==1) e (d==7) e (a !=3)		
(B<>6)		
(d>2) ou (b!=7)		

10. Dadas as variáveis alfanuméricas A,B e C com os seguintes valores: A=”carro” , B= “bicicleta” e C=”moto”. Qual o valor da variável alfanumérica x ?

a. x ← A+B;

b. x ←A+B+C;

c. x ← A+” ”+C;

d. x ← C+B+A;

11. Quais dos seguintes nomes são válidos para variáveis (V- válido I – inválido)

a. ( ) 3ab
b. ( ) n_a_o
c. ( ) Inteiras
d. ( ) z*x
e. ( ) —x
f. ( ) y-2
g. ( ) kkkMMM
h. ( ) kjh

12. Resolva as seguintes equações
    a. M ← G + 2 G← 4 *2 — O valor de M é ( )
    b. Z ← L + F F ← 3 + K K←8 L←1 — O valor de Z é ( )
    c. A ← B + C C← 2 + W W← N +O N←2 +F
       F←1 O←2 B←1 — O valor de A é ( )
    d. P ← H + B + V V ← 2
       B←V+8 H ← V *2 — O valor de P é ( )

13. Transforme as seguintes equações para que estejam no formato de um comando.

    a. $\frac{2 * 8}{2 + 1} + 4$ Resposta : _____

    b. $\frac{8 + (3-1)}{2 + 9}$ Resposta : _____

    c. $\frac{2 * 8}{2 + 1} + 4^3$ Resposta : _____

    d. $4 + 2 * 3^2$ Resposta : _____

    e. $3 - 8 + \frac{3 * 2}{5}$ Resposta : _____

2. Algoritmo: introdução

Um algoritmo é formado por um conjunto de comandos (palavras-chaves) seqüências. Cada comando tem um objetivo definido e está contido em uma linha do programa. Os comandos elementares são os seguintes:

a) Comando de entrada Leia
Sintaxe: Leia variável;
Finalidade: o programa pára e aguarda o usuário digitar. O valor digitado é atribuído para a variável especificada
Importante:
- A variável pode ser de qualquer tipo. Com o objetivo didático, assumi-se que o usuário sempre digitará valores conforme o tipo de variável. Então, se a variável é inteira, o usuário, com certeza, digitará um valor inteiro; se a variável for do real, o usuário digitará um valor do tipo real.
Exemplo: (v é uma variável)
Leia v; .

b) Comando de saída Imprima
Finalidade: o programa mostra o conteúdo de uma variável ou texto na tela do computador.
Sintaxe: Imprima [texto | variável]
Exemplo: (v é uma variável e "Olá mundo" é um texto)
Imprima v ;
Imprima "Olá mundo";

c) Comando de atribuição
Finalidade: um determinado valor ou conteúdo de uma variável é atribuído a uma variável
Sintaxe: variável ← [variável | texto | número];
Exemplo: (v, x e cidade são variáveis)
v ← 5;
v ← x;
cidade ← "São Paulo";

d) Comando de Programa
Sintaxe: Programa texto;
Finalidade: nomeia um programa. As mesmas regras para nomear uma variável são válidas para nomear um programa
Exemplo:
Programa calculadora;

e) Comando Variáveis
Sintaxe: Variáveis
Finalidade: indica que iniciará a seção de declaração de variáveis que serão utilizadas no programa
Exemplo:
Variáveis

f) Comando Início
Sintaxe: Início
Finalidade: indica o início da seção principal do programa. Esta seção contém os comandos que serão executados quando o programa iniciar
Exemplo:
Início

g) Comando Fim.
Sintaxe: Fim.
Finalidade: indica o fim da seção principal do programa.
Exemplo:
Fim.

Os comandos Leia, Imprima, Início e Fim devem sempre estar grifados.

Cada linha do programa pode conter somente uma vez este comando.

O fim do comando é indicado pelo ponto-e-vírgula.

2.1. Primeiro programa

Cada programa tem um objetivo. Então, antes de resolvê-lo, o programador deve entender o objetivo. Um algoritmo é formado pela seguinte estrutura:

● Cabeçalho : comando de definição o nome do algoritmo;

● Definição das variáveis: comandos de declaração das variáveis (nome e tipo de cada uma);

● Corpo do programa: comandos realmente fazem o processamento.

Enunciado:

Elabore um algoritmo que leia dois números inteiros. Mostre o valor da soma destes dois números.

Objetivo:

Toda vez que o programa for executado, este deve realizar a soma de dois números inteiros e apresentar o resultado.

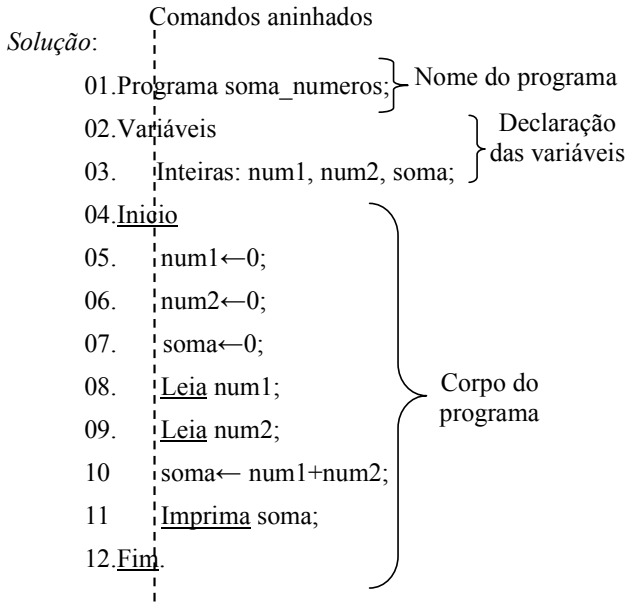
Funcionamento:

1º O programa deve obter o primeiro valor do usuário

2º O programa deve obter o segundo valor do usuário

3º O programa deve realizar a soma dos dois números

4º O programa deve mostrar o resultado da soma



Comentários:

- Cada linha deste programa foi numerada a fim de facilitar a explicação, então não é necessário que se numere as linhas. Todos os comandos de uma seção são aninhados a direita. Observe que isto aconteceu com a seção variáveis e com a início.

Linha 01: Programa soma_número;

- Objetivo: definir o nome do programa

- Sintaxe: Programa Nome_do_programa;

-onde Programa é um comando (palavra reservada) que tem como objetivo definir o nome do programa; Nome_do_programa é nome que será dado ao programa. Os caracteres válidos para formar o nome de um programa são os mesmos para se nomear as variáveis;

- o fim do comando é indicado pelo ponto-e-vírgula.

Linha 02.Variáveis

- Objetivo: indicar o início da seção de declaração das variáveis.

- Sintaxe: Variáveis

Linha 03. Inteiras: num1, num2, soma;

- Objetivo: declaração das variáveis que serão utilizadas no programa. A num1 tem como objetivo armazenar o primeiro número; a num2 armazenará o segundo número; e a variável soma armazenará o resultado da soma.

- Sintaxe : Tipo da variável: nome das variáveis;

A quantidade de variáveis em um programa é determinada pelo problema. Uma variável deve ser criada sempre que exista a necessidade de se armazenar um dado.

Este comando está alinhado (deslocado) a direita. Isto ocorre para facilitar a visualização do programa. Então, o programador visualizar um comando ou vários comandos alinhados a direita, automaticamente ele sabe que estes comandos fazem parte de uma seção que foi iniciada anteriormente. Neste caso, sabe-se que as declarações de todas as variáveis iniciaram-se logo apos co comando Variáveis.

Linha 04.Inicio

-Objetivo: este comando indica o início do corpo do programa.

-Sintaxe: Inicio.

Este comando deve ser sempre grifado. Como indica o início de uma seção, então dos os comandos posteriores são alinhados a direita.

Linha 05. num1←0;

Linha 06. num2←0;

Linha 07. soma←0;

- Objetivo: inicializar as variáveis com algum valor;

- Sintaxe: variável ← valor;

Neste caso as variáveis estão sendo inicializadas com valor zero(0). Porém, se estas variáveis fossem alfanuméricas, elas deveriam ser inicializadas com valores alfanuméricos. Inicializa-se as variáveis para certificar que os valores das variáveis são aqueles atribuídos.

Linha 08. Leia num1;

- Objetivo: obter o primeiro número do usuário

Linha 09. Leia num2;

- Objetivo: obter o segundo número do usuário

Linha 10. soma← num1+num2;

- Objetivo: realizar a soma do valor das duas variáveis.

Linha 11. Imprima soma;

- Objetivo: mostrar na tela o valor da variável soma.

Linha 12. Fim.

- Objetivo: indicar o fim do corpo do programa.

- Sintaxe: Fim.

Este comando deve ser sempre grifado. Como indica o fim de uma seção, então o seu alinhamento volta para a posição referente ao início da seção.

2.1. Teste de mesa

A fim de certificar-se que o programa está correto, deve-se realizar duas verificações antes de finalizar a resolução:

1) Verificação da sintaxe: nesta verificação deve-se realizar a checagem linha a linha do programa a fim de descobrir os possíveis erros de sintaxe, como, por exemplo:

   - Falta de ponto-e-vírgula;

   - Falta de ponto final;

   - Comando na alinhado de maneira apropriada;

   - Nome ou tipo de variável incorreto;

2) Verificação de funcionalidade: para este teste utiliza-se a ferramenta denominada "Teste de Mesa". Ela consiste na simulação da execução do programa. Sempre que um comando realizar alguma alteração na memória do computador (variáveis), o teste simula a alteração. Então, inicializa a execução:

   a. Programa soma_numeros;

      Comentários: somente nomeia o programa

   b. Variáveis

      Comentários: informa que a declaração das variáveis será inicializada a seguir

   c. Inteiras: num1, num2, soma;

      Comentários: Este comando declara as variáveis. Ele solicita espaços na memória do computador, ou seja, altera a memória. Então, neste momento cria-se a tabela que será utilizada para o teste de mesa. A quantidade de colunas desta tabela corresponde a quantidade de variáveis no programa. A quantidade de linhas é criada conforme a necessidade, inicialmente pode-se criar com 2 ou 3.A tabela para o Teste de mesa do programa anterior seria a seguinte:

num1	num2	Soma

Figura 9- Teste de mesa : criação das variáveis

   d. Inicio

      Comentários: não altera a memória, apenas indica o começo da seção

   e. num1←0;

      Comentários: altera a memória, pois atribui o valor zero para a variável num1. Então, atualiza-se a tabela:

num1	num2	soma
0		

Figura 10- Teste de mesa : alteração da variável num1

f. num1←0;

Comentários: altera a memória, pois atribui o valor zero para a variável num2. Então, atualiza-se a tabela:

num1	num2	soma
0	0	

Figura 11- Teste de mesa : alteração da variável num2

Note que este é o estado atual do programa. A variáveis num1 e num2 estão com valor zero.

g. soma←0;

Comentários: altera a memória, pois atribui o valor zero para a variável soma. Então, atualiza-se a tabela:

num1	num2	soma
0	0	0

Figura 12- Teste de mesa : alteração da variável soma

Note que este é o estado atual do programa. A variáveis num1 e num2 estão com valor zero.

h. Leia num1;

Comentários: neste momento o programa pára e aguarda a digitação de algum valor. Para isso, simula-se que o usuário digitou um valor qualquer válido. Neste exemplo será simulado que o usuário digitou o valor 4 para num1. Então, o estado atual do teste de mesa será o seguinte:

num1	num2	soma
~~0~~	0	0
4		

Figura 13- Teste de mesa : alteração da variável num1

Observe que a variável num1 tem como valor neste momento 4. O valor anterior foi substituído (passa-se um traço no valor anterior e coloca-se o nome valor na linha abaixo). O valor que estiver na última linha da coluna será sempre o atual.

i.Leia num2;

O mesmo que ocorreu para a variável num1, agora ocorrerá para a num2. Será simulado que o usuário digitou o valor 5. Deve-se atualizar o teste de mesa.

num1	num2	soma
~~0~~	~~0~~	0
4	5	

Figura 14- Teste de mesa : alteração da variável num1

j.soma← num1+num2;

Comentários: este comando altera o valor da memória, pois atribui para a variável soma o resultado da adição de num1 e num2. O estado atual do teste de mesa será o seguinte:

num1	num2	soma
~~0~~	~~0~~	~~0~~
4	5	9

Figura 15- Teste de mesa : alteração da variável soma

k. Imprima soma;

Comentários: este comando não altera o conteúdo da memória, mas mostra um resultado na tela do computador do usuário. Então, destaca-se o que ocorreu, para isso circula-se o valor.

num1	num2	soma
~~0~~	~~0~~	~~0~~
4	5	(9)

Figura 16- Teste de mesa : apresentação de resultado na tela do usuário

a. Fim.

Comentários: não altera o conteúdo da memória.

Conclusões:

Este programa tinha como objetivo realizar a soma de dois números fornecidos pelo usuários. O Teste de mesa demonstrou que o objetivo foi alcançado, portanto o algoritmo está correto.

Observação

Se durante o Teste de mesa for notado que o programa não está funcionando de maneira correta, deve-se realizar a alteração do programa e realizar o teste de mesa desde o início novamente.

A Figura 17 mostra um diagrama de como os algoritmos são construídos. Inicialmente, deve-se entender o problema. Nesta fase o objetivo do algoritmo deve estar claro para o desenvolvedor. Em seguida, propõe-se uma solução, mesmo incompleta. Em seguida, realizam-se os testes (sintaxe e de mesa). O teste de mesa tem como objetivo além de verificar a funcionalidade do algoritmo, auxiliar o seu desenvolvimento. Pois, ele possibilita que a cada passo o desenvolvedor verifique se o programa está funcionando de maneira correta. Se for percebido algum problema, então o desenvolvedor deve fazer a alteração e inicializar os testes novamente. As alterações de sintaxe ou de lógica (Teste de mesa) detectadas devem quantas vezes forem necessárias. Ao final, tem-se o algoritmo desejado.

Geralmente, a primeira solução proposta não está totalmente correta. Sempre existem erros. Então, mesmo os desenvolvedores experientes fazem vários testes para ter certeza que o programa está funcionando.

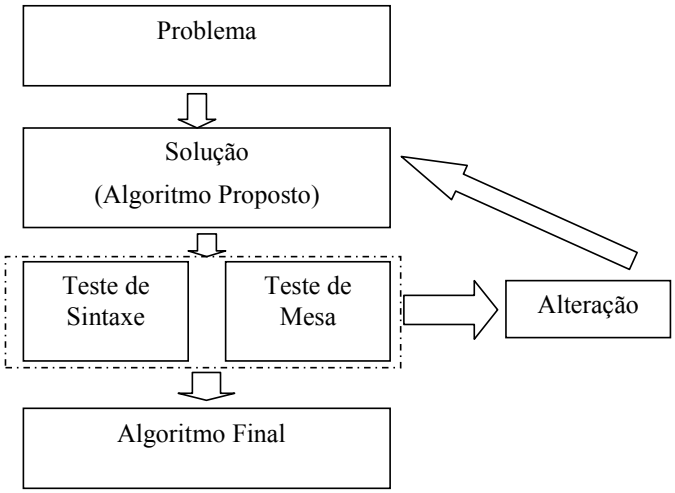


Figura 17- Como construir um algoritmo ?

2.2. Regras para elaboração de algoritmos

1)Ler a definição do problema até entender o que é pretendido;

2)Decidir por um método a ser utilizado;

3)Determinar as variáveis de entrada, de trabalho, e de saída necessárias a resolução do problema, especificando o seu tipo;

4)Sempre que possível, detalhar o problema em partes e sub-partes, cada uma das quais com objetivos específicos, até chegar ao procedimento mais simples;

5)Os procedimentos devem ser escritos na forma de comando, de maneira simples, clara, e eficiente, observando-se que:

a)Cada linha deve conter, preferencialmente, apenas um comando;

b)Deve-se deixar espaços à esquerda das linhas visando ressaltar a estrutura, utilizando chaves, colchetes, etc.

c)Um ponto e vírgula (;) separa comandos consecutivos;

d)Um ponto final (.) encerra o algoritmo;

e)Testar o algoritmo com o teste de mesa, com todos os tipos de entradas possíveis.

2.3. Um bom algoritmo

3. Exercícios: Algoritmos básicos

A seguir são apresentados vários exercícios de fixação dos conceitos básicos de programação

3.1. Introdução

1. Elabore um algoritmo que leia três valores inteiros. Imprima a soma destes valores.

   Objetivo: realizar a soma de três números inteiros e mostrar o resultado para o usuário.

   Comentário: como existem três números, cada um deles deve estar armazenado em um local da memória, então devem ser criadas três variáveis para armazenar cada um deles. Além disso, deve-se criar uma variável para armazenar o resultado.

   Funcionamento geral: obtêm-se os três números do usuário; realiza-se a soma; mostra o resultado para o usuário.

2. Construa um algoritmo que leia 3 números reais. Imprima a soma destes dividido por 2.

   Objetivo: realizar a soma de três números reais, dividir por dois e apresentar o resultado.

   Comentário: como existem três números, cada um deles deve estar armazenado em um local da memória, então devem ser criadas três variáveis para armazenar cada um deles. Além disso, deve-se criar uma variável para armazenar o resultado.

   Funcionamento geral: obtêm-se os três números do usuário; realiza-se a divisão; mostra o resultado para o usuário.

3. Faça um algoritmo que leia 3 notas (reais) de provas realizadas por um aluno. Imprima a sua média final.

   Objetivo: realizar a soma de três números reais, achar a média dos três valores (dividir por 3) e apresentar o resultado.

   Comentário: como existem três números, cada um deve estar armazenado em um local da memória, então devem ser criadas três variáveis para armazenar cada um deles. Além disso, deve-se criar uma variável para armazenar o resultado.

   Funcionamento geral: obtêm as três notas do usuário; realiza-se a divisão; mostra o resultado para o usuário.

4. Elabore um algoritmo que leia um número inteiro. Imprima o quadrado deste número. Por exemplo, se o usuário digitar o número 4, o programa deverá imprimir como resultado o valor 16.

   Objetivo: calcular o quadrado (número x número) de um número informado pelo usuário e apresentar o resultado.

   Comentário: o usuário informará um número, então é necessário uma variável para armazenar este número. Além disso, será feito um cálculo com este número, então é preciso de uma outra variável para armazenar o resultado.

   Funcionamento geral: solicita-se um número para o usuário; calcula-se o quadrado dele; mostra-se o resultado para o usuário.

5. Construa um algoritmo que leia 4 valores negativos. Imprima o valor absoluto da soma.

   Objetivo: somar quatro números negativos e mostrar para o usuário o valor absoluto da soma.

   Comentário: como existem quatro números, cada um deve estar armazenado em um local da memória, então devem ser criadas quatro variáveis para armazenar cada um deles. Além disso, deve-se criar uma variável para armazenar o resultado. O valor absoluto de um número consiste em transformá-lo em positivo, se este for negativo. Para isto, basta multiplicá-lo por (-1). Como neste algoritmo a soma resultará em um valor negativo, então esta deve ser multiplicada por (-1). É importante lembrar que se o enunciado do problema está informa que o usuário digitará números negativos, então com certeza absoluta isto acontecerá.

   Funcionamento geral: obtêm-se quatro números; calcula-se o valor absoluto; mostra o resultado para o usuário.

6. Faça um algoritmo que leia um número inteiro. Imprima o valor deste numero mais 20. Por exemplo, se o usuário digitar o número 50, o programa deverá imprimir o valor 70.

   Objetivo: obter um número do usuário, adicionar o valor de 50 e mostrar o resultado.

   Comentário: como será lido apenas um número e realizado um cálculo com este, então será necessária uma variável para armazenar o número lido e outra para o resultado.

   Funcionamento geral: obtém o valor do usuário; adiciona-se 50 a este valor; mostra o resultado para o usuário.

7. Escreva um algoritmo que leia três números inteiros e positivos (A, B, C). Calcule a seguinte expressão:

   D = (R+S)/4

onde

R = (A+B)*2

S = (B+C)*A

PS: O resultado deve mostrar as casas decimais do valor calculado.

   Objetivo: obter três números do usuário (A,B,C), calcular o valor de D, conforme a equação, e mostrar o resultado para o usuário.

   Comentário: o objetivo final é calcular o valor de D. Porém, ao analisar a equação é possível notar que primeiro deve-se encontrar o valor de R e S. Então, o algoritmo deve primeiro resolver R e S, e somente depois D. Será necessário uma variável para cada valor que será necessário armazenar. Como o enunciado deixou claro que o resultado deve mostrar as casas decimais, então as variáveis deverão ser do tipo Reais.

   Funcionamento geral: obtêm os três valores do usuário (A,B,C); calcula-se R e S; calcula-se D; mostra-se o resultado para o usuário.

3.2. Algoritmos com estrutura de repetição: Enquanto

Laços de condicionais (laços de repetição) são aqueles cujo conjunto de comandos em seu interior são executados enquanto determinada condição é satisfeita. Este comando é utilizado sempre que existe a necessidade de repetição dos mesmos comandos por várias vezes. Por exemplo, quando se deseja obter o nome e idade de 50 usuários.

- Sua estrutura é a seguinte: inicialmente a condição é testada. Se seu resultado for falso, então os comandos que estão dentro da chave não são executados. Se for verdadeiro, os comandos são executados.
- Enquanto a condição for verdadeira, os comandos que estão dentro da chave são executados. Após a execução testa-se a condição novamente.
- Se no teste a condição for falsa, o fluxo de execução prosseguirá normalmente pelas instruções posteriores ao laço condicional.

Estrutura do "enquanto… faça":

Enquanto (condição)

Faça
{
comando1;
comando2;
. . .
comando n;
}

Observações:

- Os comandos enquanto e faça são grifados;
- Condição pode ser qualquer operação relacional. Por exemplo, $a > 5$, onde a é uma variável qualquer;
- Todos os comandos que estão dentro da chave serão repetidos enquanto a condição for verdadeira; quando a condição for falsa, a instrução seguinte ao "enquanto … faça" é executada.
- Dentro da chave pode ser colocado qualquer dos comandos que fazem parte do corpo do programa, como o leia, imprima e outros; inclusive pode conter outros laços condicionais (enquanto… faça).
- Não existe limite de quantidade de comandos dentro da chave.

8. Elabore um algoritmo que leia 3 notas (reais) de provas realizadas por 5 alunos. Imprima a média final de cada aluno.

   Objetivo: calcular a média final de 5 alunos. Cada aluno realizou 3 provas.

   Comentário: cada aluno possui três notas, então serão necessárias três variáveis para armazenar as notas (n1, n2 e n3), além delas, é necessária uma variável para armazenar o cálculo da média (media). Também será necessária uma variável que tem como objetivo contar a quantidade de alunos que já tiveram a média calculada, esta variável foi denominada *cont* no exemplo. A variável *cont* será inicializada com o valor zero e sempre que uma nova média for calculada será adicionada uma unidade a ela (cont←cont+1;).

Funcionamento geral: solicitam-se as três notas no primeiro aluno; calcula-se a média do primeiro aluno; sinaliza-se que a média do primeiro aluno foi calculada (cont←cont+1;); verifica-se se já realizou a média de todos os alunos; como ainda não, então se solicita as três notas do segundo aluno; calcula-se a média do segundo aluno; sinaliza que média do segundo aluno foi calculada; verifica-se se já realizou o cálculo da média de todos os alunos; como ainda não, então segue-se para o próximo aluno até que todas a médias estejam calculadas.

Solução:

```
Programa oito;
Variáveis
        Reais: n1,n2,n3,media;
        Inteiras: cont;
Inicio
        cont←0;
        Enquanto (cont<5)
                Faça  Leia n1;
                      Leia n2;
                      Leia n3;
                      media ← (n1+n2+n3)/3;
                      Imprima media;
                      cont←cont+1;
Fim.
```

9. Escreva um algoritmo que imprima a Tabuada do 7.

7 X 0 = 0

7 X 1 =7

7 X 2 =7

….

7 X 10 =70

Objetivo: calcular a tabuada do 7.

Comentário: este programa exige a utilização de um laço condicional, pois ele deve multiplicar o valor 7 por 1, depois por 7 por 2, até 7 por 10. Note que o próprio contador do laço condicional, que deverá variar de 0 até 10, pode ser utilizado para o cálculo. Será necessária para armazenar o resultado da tabuada para toda vez que ela for calculada. Além observe que não será necessário solicitar nenhuma informação para os usuários, pois sabe-se que o objetivo é de calcular exatamente a tabuada do 7.

Funcionamento geral: inicializa-se o contador com zero; multiplica-se o contador por 7 e mostra o resultado; adiciona 1 no contador (cont←cont+1;); faz a verificação se toda a

tabuada foi calculada; como não foi, calcula-se a tabuada novamente (o valor atual do será 1); adiciona 1 no contador; faz a verificação se toda a tabuada foi calculada; como não foi, calcula-se a tabuada novamente (o valor atual do será 2); esta repetição deve ocorrer enquanto o contador, que foi denominado *cont*, for menor ou igual a 10.

10. Elabore um algoritmo que imprima o resultado da seguinte equação N vezes. Toda vez que a equação for calculada, o usuário irá informar um novo valor para M e R. Os valores são todos inteiros.

X=M+3+R.

Objetivo: Calcular o valor de X.

Comentário: o objetivo final é calcular o valor de X. Esta equação será calculada N vezes, ou seja, quando o programa iniciar-se deve-se solicitar para o usuário o valor de N. Logo após a equação deverá ficar sendo calculada até que o contador atinja a quantidade N. Para cada N deve-se solicitar ao usuário um novo M e um novo R.

Funcionamento geral: obtêm o valor de N; enquanto o contador não atingir o valor de N deve-se solicitar o valor de M e de R; calcular o valor de X e mostrar o resultado.

11. Elabore um algoritmo que imprima o resultado da seguinte equação N vezes. Toda vez que a equação for calculada, o usuário irá informar um novo valor para X e Y. Os valores são todos reais.

M=X +3 * Y+4+(X/2)+Y.

Comentário: semelhante ao algoritmo número 10.

12. Converter uma temperatura de Fahrenheit (F) para Centigrados (c) N vezes. Toda vez que a equação for calculada, o usuário irá informar um novo valor para F.

C = (F - 32) * (5/9)

Comentário: semelhante ao algoritmo número 10.

13. Elabore um algoritmo que leia N números inteiros. Calcule a soma de todos. Mostre o resultado da soma no final da execução do algoritmo.

Objetivo: Calcular a soma de N números.

Comentário: este algoritmo exigirá seja o valor solicitado do usuário seja armazenado em uma variável específica. Então, será necessária uma variável para o armazenamento da soma e outra para verificar se N números já foram lidos.

Funcionamento geral: obtêm o valor de N; enquanto o contador não atingir o valor de N deve-se: solicita-se um número para o usuário; acumula-se este número com a somatória anterior. Quando o laço de condicional for finalizado, deve-se mostrar o valor acumulado da soma.

14. Elabore um algoritmo que leia N números. Calcule a soma e a média de todos os números lidos. Mostre os resultados no final.

Comentário: semelhante ao algoritmo número 13. A média deve ser calculada somente após o fim do laço condicional (soma/N).

15. Elabore um algoritmo que leia as notas de 3 provas feitas por N alunos. Imprima a média final de cada aluno e a geral dos N alunos.

Comentário: semelhante ao algoritmo número 13 e 14.

16. Leia duas variáveis inteiras A e B. Troque os valores entre elas.

Ex: Entrada: A=6 e B=8 Saída: A=8 e B=6

Comentário: este programa exige apenas o comando de atribuição (←) no corpo do programa.

17. Faça um algoritmo que leia o nome e a idade de N alunos. A seguir, mostre a seguinte mensagem para cada aluno "O nome do usuário é XXX e sua idade é ZZZ". Substitua XXX e ZZZ pelo respectivo nome e idade do aluno.

Comentário: semelhante ao algoritmo número 13.

3.3. Algoritmos com condicional: se … então...senão

Existem situações onde existem alternativas de comandos a serem executados conforme determinadas situações. Por exemplo, um programa pode calcular o aumento de 10% no salário do empregado se ele for gerente. Se ele não for gerente, o seu aumento será de 20%. Note que neste exemplo o programa terá que verificar qual é o cargo do funcionário. Se for gerente então o aumento é de 10%, caso contrário será de 20%.

Estruturas do "se…então"

Alternativa 1:

se (condição)

então
- comando 1;
- comando 2;
- …..
- comando n;

Funcionamento: se a condição for verdadeira, então executa-se todos os comandos que estão dentro da chave. Não existe limite de quantidade de comandos.

Estrutura exemplo: deseja–se contar a quantidade de mulheres. Assim, toda vez que o usuário for mulher (sexo="feminino"), o comando que adiciona mais 1 no contador de mulheres (mulheres←mulheres +1;).

```
se (sexo="feminino")
    então   mulheres ← mulheres+ 1;
```

Alternativa 2:

se (condição)

então
- comando 1;
- comando 2;
- …..
- comando n;

senão
- comando 1;
- comando 2;
- …..
- comando n;

Funcionamento: se a condição for verdadeira, então se executa todos os comandos que estão dentro da chave do "então", caso contrário, executa-se os comandos que estão dentro da chave do "senão". Não existe limite de quantidade de comandos.

Problema exemplo: quando o programa deseja contar a quantidade de mulheres e de homens. Assim, existem duas situações bem definidas, os usuários podem ser mulheres ou homens. Então, toda vez que o usuário for mulher (sexo="feminino"), o comando, que está dentro da chave do "então" e que adiciona mais 1 no contador de mulheres (dentro da chave) é executado (mulheres←mulheres +1;). Porém, se a condição for falsa, então o comando, que está dentro da chave do "senão" e que adiciona mais 1 no contador de homens (dentro da chave) é executado (homens←homens +1;).

Estrutura do exemplo:

se (sexo="feminino")

então
- mulheres ← mulheres+ 1;

senão
- homens ← homens +1;

Observações:

- Os comandos "se.. então … senão" são grifados;
- Um programa pode ter vários "se... então" ou "se… então… senão";
- Após a execução dos comandos que estão dentro das chaves do "então" ou do "senão", os próximos são os que estiverem a seguir, fora da chave;
- Dentro das chaves pode ser colocado qualquer dos comandos que fazem parte do corpo do programa, como o leia, imprima e outros; inclusive pode conter outros "se... então", "se… então… senão" e "enquanto… faça".

18. Construa um algoritmo que leia dois valores reais (A e B). Imprima a mensagem "A é maior ou igual do que B" ou "A é menor do que B"

Comentários: observe que existem duas situações bem definidas "A é maior ou igual do que B" ou "A é menor do que B", então com certeza pode-se utilizar a estrutura "se… então… senão".

19. Elabore um algoritmo que leia o nome e a idade dos N habitantes de uma cidade. Imprima o nome dos habitantes que tenham mais de 20 anos.

*Comentário*s: este programa está interessado apenas nos habitantes com mais de 20 anos, ou seja, uma situação apenas. Assim, a estrutura a ser utilizada será "se… então". Além disso, é importante notar que precisará do laço de repetição ("enquanto ..faça").

Observação: a maioria dos algoritmos seguintes será necessário utilizar laço(s) de repetição e estrutura(s) condicional(is) ("se… então".)

20. Elabore um algoritmo que leia o nome e a idade dos N habitantes de uma cidade. Imprima o nome dos habitantes que tenham mais de 20 anos e a idade dos que tem menos de 15 anos.

*Comentário*s: este programa deseja imprimir o nome dos habitantes com mais de 20 anos e a idade do que tem menos de 15 anos. São duas situações, porém observe que se a primeira situação for falsa, não existe certeza que a segunda é verdadeira. Se a idade do habitante for 18 anos o programa não deverá fazer nada. Assim, a estrutura "se… então .. senão" não pode ser utilizada neste caso. Deve-se utilizar dois "se… então"

21. Elabore um algoritmo que leia o nome e a idade dos N habitantes de uma cidade. Imprima o nome dos habitantes que tenham mais de 20 anos. No final do algoritmo, imprima a quantidade de habitantes com mais de 40 anos.

Comentários: algoritmo semelhante ao algoritmo número 20. Porém, após o laço condicional (final do algoritmo), deve imprimir a quantidade de habitantes com mais de 40 anos.

22. Elabore um algoritmo que leia um número inteiro. Se o número for par, imprima dobro dele. Se o número for ímpar, imprima o triplo dele.

Dica : 8/2 = 4 9/2 = 4 (o resto é ignorado na divisão de números inteiros). Quando o resto da divisão é 0, então o número é par, caso contrário é impar.

Comentários: algoritmo semelhante ao algoritmo número 20.

23. Elabore um algoritmo que leia o nome e a idade dos habitantes de uma cidade. Imprima o nome e a idade dos com idade entre 25 e 40 anos. O algoritmo deverá ser encerrado quando o décimo habitante com idade entre 25 e 40 anos for encontrado.

Comentários: a condição de saída do laço condicional é ter encontrado do décimo habitante com idade entre 25 e 40.

24. Elabore um algoritmo que leia N números reais. Imprima o valor absoluto de cada número.

Comentários: o valor absoluto de um número é o valor positivo dele. Então, deve ler o número e verificar se ele é negativo ou não. Se for negativo, multiplica-se por menos um e realização a impressão do resultado.Caso contrário, apenas imprima o número.

25. Elabore um algoritmo que leia N números reais. Imprima o valor absoluto de cada número. Imprima a média de números lidos entre –8 e 10. Imprima a quantidade lida de números negativos.

Comentários: este programa é a junção do conhecimento utilizado nos problemas número 20 e 24.

26. Faça um algoritmo que leia um número e verifique se ele é um dos 30 primeiros termos da seqüência de Fibonacci: 1,1,2,3,5,8,13,21,...

Comentários: observe que a seqüência inicia-se com o número 1 e é seguida por outro 1. Logo após, aparece o número 2, que é a soma dos dois números anteriores (1+1). O número 3 é a soma dos dois números anteriores (1+2). Isto é um laço condicional. Assim, para cada repetição deve-se somar os dois anteriores e mostrar o novo número gerado. Então, para cada repetição, o segundo número anterior deve-se tornar o primeiro anterior, e o número gerado o segundo anterior, pois na próxima iteração basta somar estes números para gerar um novo. Isto deve ser repetido até que os 50 primeiros termos tenham sido gerados (será necessário um contador). No final, deve-se imprimir uma mensagem " N pertence" ou "N não pertence", onde N é o número lido.

27. A empresa XYZ contratou você para calcular o salário mensal de cada um de seus vendedores. O salário de cada vendedor é composto pelo salário fixo acrescido de um determinado percentual sobre a venda mensal, que varia para cada vendedor.

Comentários: deve-se solicitar o salário, as vendas e o percentual. Então, adiciona-se este percentual sobre as vendas no salário do funcionário.

28. Escrever um algoritmo que leia um número X (n vezes) e calcule:

a) O resto da divisão de $X / 5$
b) $X^2 + 20$
c) X^2 - X + 2

Comentários: deve-se solicitar o valor de X e mostrar o resultado das 3 expressões.

29. Faça um algoritmo que imprima os 20 primeiros termos da seqüência: 4/1, – 12/5 , 36/9, -108/13, ...

Comentários: note que para cada número multiplica-se numerador por -3, enquanto que para denominador soma-se mais 4. Assim, para cada interação, o numerador e o denominador devem ser gerados separadamente, pois seguem regras distintas. Para cada iteração no laço condicional, executa-se as respectivas regras de geração.

30. Faça um algoritmo para imprimir a seguinte figura. Deve-se utilizar laços condicionais para construir as figuras abaixo. Cada figura, um algoritmo. O comando Imprima deve imprimir um caracter por vez (exemplo: Imprima "*";).

Figura 1:

```
**************************************************
**************************************************
++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++
**************************************************
**************************************************
```

Figura 2:

```
**********++++++++++++++++++**********
**********++++++++++++++++++**********
**********++++++++++++++++++**********
**********++++++++++++++++++**********
**********++++++++++++++++++**********
```

Figura 3:

```
*++++++++++++++++++
**++++++++++++++++++
***+++++++++++++++++
****++++++++++++++++
*****+++++++++++++++
******++++++++++++++
*******+++++++++++++
```

Figura 4

```
**********
*****++*****
****++++****
***+++++++***
**+++++++++**
*++++++++++*
++++++++++++
```

31. Elabore um algoritmo que calcule a quantidade de litros de combustível gastos em N viagens.

Outras informações:

- Considerar que um automóvel faz 12 Km por litro.
- Distância = tempo * velocidade.
- Qtd litros = distância / 12.

Observação: considere que tempo, velocidade e distância são números inteiros.

Comentários: para cada viagem solicita-se o tempo e a velocidade. Logo após, usa-se a fórmula dada para calcular a quantidade de combustível gasta.

32. Com o objetivo de ir para olimpíadas um determinado atleta participou de várias corridas. Obtenha deste atleta o seu nome e idade. Além disso, solicite o nome e o tempo de corrida de cada competição que ele participou. Apresente a pontuação total do atleta segundo a tabela abaixo:

Tempo	**Pontuação ganha**
T < 10 segundos	100
T >= 10 e *T <=13* segundos	70
T > 13 segundos	40

Onde T é o tempo gasto em cada prova

Comentários: Como é apenas um atleta, os comandos referentes a leitura do seu nome e idade deverão estar antes no início da estrutura de repetição. Dentro da estrutura de iteração deve-se solicita os dados de cada corrida e calcular a pontuação do atleta.

33. Construa o algoritmo anterior. Porém, considere que existem N atletas.

Comentários: O algoritmo anterior calculava a pontuação de apenas um atleta, neste deve-se calcular para N atletas. Para isto, baste adicionar uma estrutura de repetição que englobará o algoritmo anterior.

34. Leia *n* números inteiros. Descubra o menor e a média dos números que estao entre 4 e 20.

35. Leia N números inteiros. Imprima se cada um é par ou ímpar.

Comentários: Semelhante ao algoritmo 22. Porém, este algoritmo é para n números.

36. Leia *m* e *n* (assuma que o valor digitado pelo usuário para m sempre será menor do que n). Calcule a soma de todos os números ímpares entre *m* e *n*.

37. Elabore um programa que calcule N! (fatorial de N). O valor inteiro de N é fornecido pelo usuário (maior do que zero).

Sabendo que:

N! = 1 * 2 * 3 * ... * (N-1) * N;

0! = 1, por definição.

Comentários: Por exemplo, o fatorial do número 5 é 120, que é o resultado da operação (5*4*3*2*1).

38. Faça um algoritmo que calcule e imprima a seguinte soma: $2/50 + 2^2/49 + 2^3/48 + \ldots 2^{50}/1$.

39. Faça um algoritmo que calcule a multa de N pescadores. Segundo o regulamento do estado de São Paulo os pescadores devem pagar uma multa de R$ 4,00 por quilo excedente a 50 quilos (valores inteiros). Caso o pescador deva ser multado, imprima o valor da multa e a mensagem "Siga o regulamento de pesca". No final, imprima a soma total de multas arrecadadas e a peso total de todos os peixes pescados.

40. Elabore um algoritmo que leia o código e o número de horas (inteiras) trabalhadas de N operários. Calcule o salário sabendo que cada um ganha R$ 10,50 por hora. Quando o número de horas exceder a 50, calcule o excesso de pagamento. A hora excedente de trabalho vale R$ 12,00. Imprima o salário total de cada operário. No final, imprima a soma total dos salários e apresente a quantidade de funcionários que trabalharam mais de 40 horas.

41. Elabore um algoritmo que leia a idade de N nadadores. Informe para o nadador a categoria que ele pertence conforme a tabela abaixo. No final, imprima a quantidade de nadadores de cada categoria.

Categoria	**Idade**
Infantil A	5 a 7 anos
Infantil B	8 a 11 anos
Juvenil A	12 a 13 anos
Juvenil B	14 a 17 anos
Adultos	Maiores de 18 anos

Observação: este clube não aceita alunos com idade inferior a 5 anos.

42. Faça um algoritmo que imprima o maior número entre **N** números. A condição de parada é a entrada de um valor 0, ou seja, o algoritmo deve ficar pedindo números ao

usuário até que a entrada seja igual a 0 (ZERO). O número zero deve ser desconsiderado, ou seja, se o usuário digitar somente números negativos e ao final o número zero (para sair), o maior número deverá ser o maior número negativo digitado.

43. Construa um algoritmo que leia 60 valores inteiros e positivos e:

- Encontre o maior valor
- Encontre o menor valor
- Calcule a média dos números lidos

44. Foi feita uma pesquisa entre N habitantes de uma região. Foram coletados os seguintes dados: idade, sexo (M/F) e salário. Faça um algoritmo que informe:

- A média de salário do grupo
- Maior e menor idade do grupo
- Quantidade de mulheres com salário até R$ 100,00
- Quantidade de homens

45. Elabore um algoritmo leia as seguintes informações de N produtos: Código do produto (CODPROD), Quantidade Mínima (QTDMIN), Quantidade Máxima (QTDMAX) e a quantidade em estoque (QTDEST) de cada produto. Informe se o produto deve ser comprado. Se sim, informe a quantidade. Um produto somente deverá ser comprado quando a quantidade em estoque for menor ou igual a quantidade mínima. Para determinar a quantidade a ser comprada deve-se utilizar a seguinte fórmula QTCOMPRAR = (QTDMAX – QTDEST).

46. Faça um algoritmo que leia vários números inteiros e calcule o somatório dos números negativos. O fim da leitura será indicado pelo número 0.

47. Elabore um algoritmo que apresente o valor de uma potência de uma base qualquer elevada a um expoente qualquer, ou seja, de N^m. Exemplo: $5^3 = 5 \times 5 \times 5$. Neste algoritmo o operador ** (exponenciação) não deve ser utilizado.

48. Faça um algoritmo que implemente uma enquete de futebol (50 pessoas). O programa deve ler o nome do torcedor e a seleção que ele torce. No final, imprima a quantidade de torcedores de cada seleção e a porcentagem que estes representam do total. As únicas opções de seleção são: Brasil, Argentina, Alemanha e Portugal.

49. Faça um algoritmo que leia os seguintes itens de uma nota fiscal: código do produto, quantidade do produto, descrição do produto e o valor unitário. Os dados devem ser aceitos até que se digite 0 no código do produto. Mostre no final as seguintes informações para o

usuário: a quantidade de itens da Nota Fiscal, o valor total da nota, e o valor total dos impostos (5% sobre a nota)

50. Uma pesquisa sobre características físicas de uma região coletou os seguintes dados de cada habitante: idade, sexo (masculino, feminino), cor dos olhos (pretos, castanhos, azuis, verdes), cor dos cabelos (preto, castanho, ruivo) e a renda mensal. Faça um algoritmo que mostre no final a quantidade de homens com olhos pretos e a quantidade de mulheres com olhos azuis ou verdes e com cabelos castanhos. Além disso, informe a quantidade de mulheres com cabelos ruivos e que ganham mais de R$1.000,00.

51. Faça um algoritmo que leia os dados necessários para a realização da pesquisa abaixo. O algoritmo deve encerrar quando for lida qualquer idade menor que zero. Apresentar no final do algoritmo:

- A quantidade de habitantes da região
- A quantidade de pessoas com mais de 30 anos
- A porcentagem de indivíduos do sexo feminino com olhos pretos ou castanhos e que tenham renda mensal maior que 300,00.
- A quantidade de indivíduos do sexo masculino com cabelos ruivos e com renda menor que 1.000,00.

52. Faça um algoritmo que leia a 3 notas de N alunos. Calcule a média final de cada aluno. Considere que a média é ponderada e que o peso das notas é: 2,3 e 5 respectivamente.

53. O custo ao consumidor de um carro novo é a soma do custo de fábrica com a porcentagem do distribuidor e dos impostos (aplicados ao custo de fábrica). Supondo que a porcentagem do distribuidor seja de 28% e os impostos de 45%, escreva um algoritmo que leia o custo de fábrica de N carros e apresente o custo de fábrica, o valor que o distribuidor receberá, o valor dos impostos e o valor final de cada carro.

54. Tendo como dados de entrada a altura e o sexo de uma pessoa, construa um algoritmo que calcule o peso ideal de N pessoas. Utilize as seguintes fórmulas:

- para homens: (72.7*h)-58
- para mulheres: (62.1*h)-44.7

55. Um banco concederá um crédito especial aos seus clientes, variável com o saldo médio no último ano. Faça um algoritmo que leia o saldo médio dos clientes e que calcule o valor do crédito de cada um conforme a tabela abaixo. Mostre uma mensagem informando o saldo médio e o valor do crédito.

Saldo médio	Percentual
de 0 a 200	nenhum crédito
de 201 a 400	20% do valor do saldo médio
de 401 a 600	30% do valor do saldo médio
acima de 601	40% do valor do saldo médio

56. Uma empresa concederá um aumento de salário aos seus funcionários, variável de acordo com o cargo, conforme a tabela abaixo. Faça um algoritmo que leia o salário e o código de cada funcionário. Calcule o novo salário. Se o cargo do funcionário não estiver na tabela, ele deverá, então, receber 40% de aumento. Mostre o salário antigo, o novo salário e a diferença.

Código	Cargo	Percentual
101	Gerente	10%
102	Engenheiro	20%
103	Técnico	30%

57. Chico tem 1,50 metros e cresce dois centímetros por ano, enquanto Zé tem 1,10 metros e cresce três centímetros por ano. Construa um algoritmo que calcule e imprima quantos anos serão necessários para que Zé seja maior que Chico.

58. Uma empresa deseja aumentar seus preços em 20% (N produtos). Faça um algoritmo que leia o código e o preço de custo de cada produto. Calcule o preço novo com o respectivo aumento de 20%. Calcule também a média dos preços com e sem aumento. Mostre o código e o preço novo de cada produto. Imprima as médias no final.

59. Escreva um algoritmo que leia os seguintes itens de N peças: o código da peça, o valor unitário da peça, quantidade e a porcentagem do IPI a ser acrescido no valor de cada peça. O algoritmo deve calcular o valor total a ser pago e apresentar o resultado. Utilize a seguinte fórmula para o cálculo do valor do IPI de cada peça: (valor *quantidade)*(IPI/100 + 1).

4. Vetores

4.1. Introdução

No início deste livro explicou-se que os computadores precisam armazenar os dados na memória e denominou-se estes locais de variáveis. É importante lembrar que uma variável refere-se a um único valor, que pode a qualquer momento ser substituído por outro.

Os vetores também são variáveis, porém eles possibilitam o armazenamento de mais de um valor do mesmo tipo. Por exemplo, a Figura 18 mostra como é memória do computador em um problema que precisa armazenar três nomes em duas situações. Utilizando variáveis, serão necessárias 3 locais de armazenamento, as variáveis A,B e C ; porém, se for utilizado um vetor (V), será necessário apenas um vetor com 3 posições. As posições de um vetor são denominadas índices. Para a impressão do nome João utilizando variáveis é necessário o comando "Imprima A;", entretanto, no caso dos vetores é preciso utilizar o índice para indicar qual a posição do vetor que será mostrada, então o comando será "Imprima V[0];". Observe que a referência a um determinado elemento é feita através da seguinte sintaxe: nome[índice], onde nome é o nome do vetor e o índice é a posição do vetor. Neste caso o índice do conteúdo "João|" é 0 (zero).

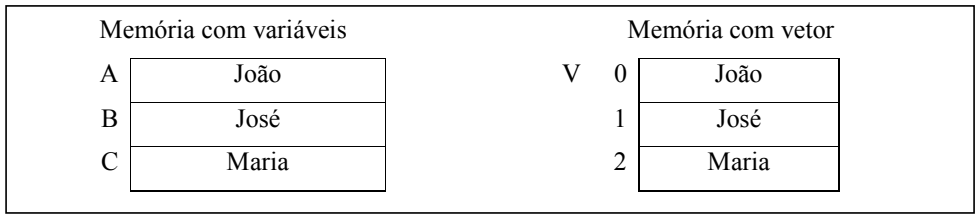


Figura 18- Memória do Computador – Variáveis X Vetor

O índice sempre começa pelo valor 0, então, no exemplo mostrado, que contém 3 elementos, o índice pode variar somente entre 0 e 2. Se o programa referenciar um índice que não esteja no intervalo, isto gerará um erro. O índice sempre será um número inteiro, que pode estar armazenado em uma variável ou não. Se por exemplo a variviável n ter como valor o número 1, então o comando "Imprima V[n];" irá mostrar para o usuário o nome "José".

Apesar de um vetor ser visto como uma variável que contem n valores, é importante notar que cada posição deve ser tratada de maneira individual. Então, quando não é possivel atribuir um valor para todas as posições do vetor com um único comando. Cada posição deve ser tratada

de maneira indiviual. O comando Leia V[2] atribuirá o valor digitado pelo usuário somente na a posição. Assi como o comando Imprima V[1] irá mostrar o conteúdo da posição 1. Portanto, **não é permitido** comandos como Leia V[ ] ou Imprima Vetor[ ]. As mesmas regras para nomeação das variáves são também aplicadas ao nome dos vetores.

Observação Importante: geralmente, os exercícios propostos utilizam vetores com poucos elementos (a fim de facilitar o teste de mesa), por exemplo, 10 elementos, porém as soluções desenvolvidas devem ser feitas de tal forma que se o vetor(es) tivessem 1000 ou mais elementos também funcionaria. Sempre que for realizar o teste de mesa, desenhe um vetor com a quantidade de elementos especificado no problema.

4.2. Exercícios

60. Faça um algoritmo que leia 10 elementos inteiros e armazene-os em um vetor. Por fim, imprima todos os elementos do vetor.

Objetivo: ler 10 elementos inteiros; armazená-los em um vetor e mostrá-los para o usuário.

Comentário: se este programa fosse resolvido apenas com variáveis, seria necessária a utilização de 10 variáveis, uma para cada elemento. Lembre-se que o controle de um vetor é exercido pelo índice. Então, o índice deverá variar entre 0 e 9 neste problema.

Funcionamento geral: solicita-se um elemento; armazena-o na primeira posição do vetor; solicita-se outro elemento; armazena-o na segunda posição do vetor; este processo deve ser repetido até que o décimo elemento seja armazenado no vetor. Em seguida, deve-se imprimir todos os elementos do vetor. Na solução apresentada utilizou-se a variável denominada *ind* para referenciar a posição do elemento no vetor. Logo após a leitura de um elemento, o valor de *ind* é acrescentado de um (*ind*←*ind+1*;) para que o próximo elemento seja referenciado ou para que seja concluída a operação sobre o último elemento. Observe que ela vai variar entre 0 e 9, pois existem 10 elementos. Utilizou-se duas estruturas de repetição, a primeira realiza somente a leitura dos elementos e a segunda faz a impressão dos elementos.

Solução:

```
Programa exemplo;
Variáveis
    Inteiras: V[10], ind;
Inicio
    ind←0;
    Enquanto (ind<10)
            Faça  Leia  V[ind];
                  ind ← ind +1;
    ind←0;
    Enquanto (ind<10)
            Faça  Imprima  V[ind];
                  ind ← ind +1;
Fim.
```

61. Faça um algoritmo que leia um vetor de 10 elementos reais e imprima a média destes elementos.

62. Faça um algoritmo que leia um vetor com 10 elementos alfanuméricos. Imprima o vetor em sua ordem inversa.

63. Faça um algoritmo que leia um vetor com 12 elementos reais. Imprima os 5 últimos elementos.

64. Faça um algoritmo que leia as notas (reais) de uma prova feita por 5 alunos. Imprima as notas que ficaram acima da média da classe.

65. Faça um algoritmo que leia um vetor com 20 elementos inteiros. Imprima o maior, o menor e a média dos elementos.

66. Faça um algoritmo que leia o conjunto A (5 elementos inteiros), o conjunto B (5 elementos reais) e o elemento N. Apresente uma mensagem ao usuário que informe que N está em A ou em B; ou que N está em nenhum dos dois conjuntos; ou que N está nos dois conjuntos.

67. Leia o vetor A (8 elementos) e construa o vetor B com os elementos do vetor A multiplicados por 3. Apresente o vetor B. Assim, o elemento B[1] deverá ser multiplicado pelo elemento A[1]*3, o elemento B[2] pelo elemento A[2] *3 e assim por diante, até 8.

68. Leia os vetores A (20 elementos) e B (20 elementos) e construa o vetor C. Cada elemento de C é a subtração do elemento correspondente de A–B. Imprima os vetores A,B e C.

69. Dados 3 vetores A, B e C de mesma dimensão, com 15 elementos inteiros, obtenha um quarto vetor D que seja a multiplicação de A por B diminuído de C. Portanto, o elemento D[i] = A[i]*B[i] – C[i].

70. Leia o vetor A (15 elementos). Construa o vetor B de mesmo tipo, observando a seguinte lei de formação: ”Todo elemento de B deverá ser o quadrado do elemento de A correspondente”. Imprima o vetor B.

71. Leia o vetor A (20 elementos), o vetor B (30 elementos) e construa o vetor C. O vetor C é a junção seqüencial dos dois outros vetores. Desta forma, C deverá ter a capacidade de armazenamento de 50 elementos. Imprima o vetor C.

72. Leia um vetor A (15 elementos) e construa o vetor B. Cada elemento do vetor B é o fatorial do elemento correspondente do vetor A. Apresente o vetor B.

73. Faça um algoritmo que leia 20 números inteiros e armazene-os em um vetor, e, a seguir, leia 5 números inteiros e armazene-os em outro vetor. O programa os índices dos elementos com mesmo conteúdo nos dois vetores.

74. Leia o vetor A (5 elementos reais) e B (5 elementos inteiros). Construa o vetor C. O vetor C é composto pelos elementos do vetor A e B intercalados.

75. Dados dois vetores (20 elementos inteiros). Imprima o maior e o menor elemento de cada vetor; e o maior e o menor elemento dos dois vetores.

76. Leia a seqüência de 5 caracteres, armazene-os em um vetor e verifique se a palavra armazenada é um palíndromo, ou seja, verifique se a palavra quando lida da esquerda para direita é a mesma quando lida da direita para esquerda. Não deve ser utilizado outro vetor. Se o vetor tiver o tamanho 5000, a solução também deverá funcionar.

Por exemplo: ARARA pode ser lida a partir da esquerda ou da direita, obtemos a mesma palavra.

77. Dado um vetor de 20 elementos inteiros. Determinar a soma dos elementos pares do vetor e a soma dos elementos ímpares. Imprima os resultados.

78. Dado um vetor de 10 caracteres. Faça um algoritmo que remova o caracter "a". Um novo vetor deverá ser criado (sem o caracter "a"). Em cada posição do vetor existe apenas um caracter.

79. Dado os vetores A e B com 20 elementos inteiros. Determine quantas vezes cada elemento de A aparece em B.

80. Dado um vetor de tamanho 20 (inteiro), com os 10 primeiros elementos já informados. Estes 10 elementos estão ordenados por ordem crescente. Leia os outros 10 elementos e insira-os na posição correta, ou seja, mantendo-os ordenados.

81. Dado um vetor de 10 elementos, inverta os elementos do vetor. O primeiro elemento deverá ser o último e vice-versa; o segundo deve ser o penúltimo e vice-versa; até o término da inversão. Observar que essa inversão deve ser feita usando o próprio vetor e não utilizando um vetor auxiliar, entretanto é permitido o uso de variáveis auxiliares.

82. Leia um vetor com 5 elementos (número binário). Cada elemento pode ser 0 ou 1. Imprima o número inteiro representado por esse vetor binário.

    Por exemplo: Suponha o vetor v :

    O valor inteiro de 01101 será v[0]*2**4 + v[1]*2**3 + v[2]*2**2 + v[3]*2**1 + v[4]*2**0

    Resultado deste exemplo: 13

5. Matrizes

5.1. Introdução

Como visto no capítulo anterior, os vetores são variáveis que possibilitam o armazenamento de mais de um valor do mesmo tipo. Uma importante característica deles é que são unidimensionais, então possuem apenas um índice que se referentes somente às linhas. Por outro lado, existem as matrizes, que são multidimensionais. Então, por exemplo, elas poder ter índices que se referem às linhas e às colunas, como mostra a Figura 19. Esta figura mostra a matriz M com duas dimensões. Então, para se referenciar a qualquer posição sempre será necessário informar as duas posições. Por exemplo, na posição M[1][2] está o numero 245; na posição M[0][1] está o número 947. Observe que os valores armazenados são todos do mesmo tipo (números).

Todos os princípios utilizados em vetores são válidos para as matrizes, com o uso de índices para referenciar as posições e armazenamentos de dados do mesmo tipo. A diferença é na quantidade de dimensões. Vetores apenas uma dimensão e matrizes pode ter n dimensões.

	0	1
0	34	947
1	346	6
2	125	245

Figura 19- Matriz M com duas dimensões

É importante notar que as matrizes não estão restritas a duas dimensões. Estas poderiam ter N dimensões. Poderia, por exemplo, ter uma matriz com 3 dimensões a fim de armazenar as coordenadas de uma imagem x, y e z, sendo que cada coordenada em uma dimensão.

Os exercícios a seguir, por questão de simplificação usam sempre matrizes com duas dimensões (linha,coluna). Sempre que for realizar o teste de mesa, desenhe uma matriz com a quantidade de linhas e colunas especificadas no problema.

5.2. Exercícios

83. Leia uma matriz 4X5 de elementos inteiros e imprima cada elemento da matriz.

Por exemplo:

Linha 0	4	6	7	1	2
Linha 1	0	3	4	4	7
Linha 2	1	5	6	6	1
Linha 3	9	2	3	2	0

Objetivo: ler 20 elementos inteiros; armazená-los em uma matriz [4][5] e mostrá-los para o usuário.

Comentário: se este programa fosse resolvido apenas com variáveis, seria necessária a utilização de 20 variáveis, uma para cada elemento. Lembre-se que o controle de uma matriz é exercido pelo índice. Então, o índice da linha deverá variar entre 0 e 3; por outro lado, o índice das colunas deverá variar entre 0 e 4.

Funcionamento geral: solicita-se um elemento; armazena-o na linha 0 e coluna 0 da matriz; solicita-se outro elemento; armazena-o na linha 0 e coluna 1 da matriz; solicita-se outro elemento; armazena-o na linha 0 e coluna 2 da matriz; este processo deve ser repetido até que todos os elementos da linha 0 sejam lidos. Em seguida, deve-se começar fazer a leitura dos elementos da linha 1, desde a coluna 0 até a coluna4. Sendo assim, é possível perceber que em todas as vezes que lemos um elemento, adicionar 1 no índice de coluna. Porém, quando chega-se ao último elemento da coluna, adiciona-se 1 no índice da linha e atribui-se 0 para a coluna. Em seguida, deve-se imprimir todos os elementos do vetor utilizando o mesmo controle de índices que foi utilizado na leitura.

Solução:

```
Programa exercicio83;
Variáveis
    Inteiras: M[4,5],linha,coluna;
Inicio
    linha←0;
    Enquanto (linha<4)
        Faça    coluna←0;
                Enquanto (coluna<5)
                    Faça  Leia  M[linha][coluna];
                          coluna← coluna +1;
                linha← linha +1;
    linha←0;
    Enquanto (linha<4)
        Faça    coluna←0;
                Enquanto (coluna<5)
                    Faça  Imprima  M[linha][coluna];
                          coluna← coluna +1;
                linha← linha +1;
Fim.
```

84. Seja a seguinte matriz A:

A =

0	1	2	3	4	5
175	25	10	90	3,7	4
9,8	100	3	42	156	18
40	301	30	1	1	0
4	42	72	992	44	1
21	3	2	1	90	3

a) Quantos elementos fazem parte do conjunto;
b) Qual o conteúdo do elemento indicado por A[4][5]?
c) Qual o conteúdo de X após a execução do comando X ← A[3][2] + A[4][1]?
d) O que aconteceria caso fosse referenciado o elemento A[6][2] no algoritmo ou programa.
e) Faça um algoritmo para somar os elementos da coluna 4.

Qual o valor da soma dos elementos da linha 3 (não precisa fazer algoritmo).

85. Faça um algoritmo que calcule a quantidade de números negativos de uma matriz real A[4][4].

86. Faça um algoritmo que leia 40 números inteiros. Gere uma matriz real M[5][8]. Cada elemento desta matriz será o dobro de cada elemento lido.

87. Faça um algoritmo que leia duas matrizes A[3][4] e B[3][4]. Gere a matriz C[3][4]. A matriz C será a soma de A e B.

88. Leia o vetor A com 10 elementos. Construa uma matriz C com três colunas. Cada elemento da primeira coluna da matriz C é formado pelos elementos do vetor A somados com mais 5. Cada elemento da segunda coluna é formado pelo triplo de cada elemento correspondente do vetor A. Cada elemento da terceira coluna será o quadrado do elemento correspondentes do vetor A. Apresente a matriz C.

89. Leia os vetores A(7 elementos) e B (7 elementos). Construa a matriz C. Cada elemento da a primeira coluna de C será o elemento do correspondente de A. Cada elemento da a primeira coluna de C será o elemento do correspondente de B. Apresentar a matriz C.

90. Faça um algoritmo que leia a matriz A [3][6]. Some as suas colunas e armazene os resultados no vetor soma de tamanho 6, que é o número de colunas da matriz A.

91. Em uma sala de aula tem 50 alunos, sabe-se que cada aluno cursa 8 disciplinas. Elabore um algoritmo que armazene as notas semestrais de cada disciplina cursada por cada aluno em uma matriz Notas 50x8. Após a leitura, o algoritmo deve calcular, a partir da matriz lida, a média geral de cada aluno.

92. Elabore um algoritmo que leia uma matriz P[5][5]. Informe se a mesma é simétrica ou não.

OBS: Para uma matriz ser simétrica os elementos A[I][J] = A[J][I].

93. Elabore um algoritmo que leia uma matriz VALOR[4][4]. Calcule a soma da cada uma das linhas da matriz. Os valores das somas calculadas devem ser armazenados em um vetor RESULTADO de tamanho 4. Ao final do algoritmo o vetor deve ser impresso.