

# ALGORITMOS

APOSTILA por Lucília Ribeiro

---



*Qualquer tempo é tempo.  
A hora mesma da morte  
é hora de nascer.*

*Nenhum tempo é tempo  
bastante para a ciência  
de ver, rever.*

*Tempo, contratempo  
anulam-se, mas o sonho  
resta, de viver.*

(Mario Quintana, "Qualquer")



# Índice

*"Quantas vezes a gente, em busca da ventura,  
Procede tal e qual o avozinho infeliz:  
Em vão, por toda parte, os óculos procura,  
Tendo-os na ponta do nariz!" (Mário Quintana)*

<b>ÍNDICE</b> .....	<b>3</b>
<b>VISÃO GERAL</b> .....	<b>6</b>
1.1 DADOS E INFORMAÇÕES .....	6
1.2 CONVERSÃO DE DADOS EM INFORMAÇÕES .....	6
1.3 DIVISÃO DE TAREFAS – SERES HUMANOS X SISTEMAS DE COMPUTAÇÃO.....	6
1.4 INFORMÁTICA.....	7
1.5 TIPOS DE DADOS .....	7
1.6 O QUE É UM COMPUTADOR? .....	7
1.6.1 UNIDADE CENTRAL DE PROCESSAMENTO.....	10
1.6.2 MEMÓRIA.....	10
1.6.3 DISPOSITIVOS DE ENTRADA.....	10
1.6.4 DISPOSITIVOS DE SAÍDA.....	10
1.7 CICLO DE PROCESSAMENTO.....	10
1.7.1 ENTRADA / SAÍDA DE DADOS .....	11
1.7.2 PROGRAMA.....	11
1.7.3 PROCESSAMENTO.....	11
1.8 REPRESENTAÇÃO DA INFORMAÇÃO.....	11
1.9 BYTES .....	12
1.10 EXERCÍCIOS.....	12
<b>NOÇÕES DE LÓGICA</b> .....	<b>14</b>
2.1 LÓGICA.....	14
2.2 LÓGICA DE PROGRAMAÇÃO.....	14
2.3 ALGORITMO .....	15
2.4 PROGRAMAÇÃO.....	15
2.5 IMPORTÂNCIA DE UM ALGORITMO.....	15
2.5.1 LINGUAGEM NATURAL .....	15
2.5.2 FLUXOGRAMA.....	16
2.5.3 LINGUAGEM ESTRUTURADA.....	16
2.6 EXEMPLOS .....	16
2.7 EXERCÍCIOS.....	18
<b>ITENS FUNDAMENTAIS</b> .....	<b>20</b>
3.1 INTRODUÇÃO .....	20
3.2 TIPOS DE DADOS .....	20
3.2.1 INTEIRO.....	20
3.2.2 REAL.....	20
3.2.3 CHARACTER.....	20
3.2.4 LÓGICO.....	21
3.3 FORMAÇÃO DE IDENTIFICADORES .....	21
3.4 CONSTANTES.....	21
3.5 VARIÁVEIS.....	21
3.6 COMENTÁRIOS.....	22
3.7 EXERCÍCIOS.....	22
<b>EXPRESSÕES E OPERADORES</b> .....	<b>24</b>
4.1 INTRODUÇÃO .....	24
4.2 EXPRESSÕES ARITMÉTICAS.....	24
4.3 EXPRESSÕES LÓGICAS.....	25
4.4 EXERCÍCIOS.....	26
<b>ESTRUTURA SEQUENCIAL</b> .....	<b>28</b>

5.1	COMANDO DE ATRIBUIÇÃO.....	28
5.2	COMANDOS DE ENTRADA E SAIDA .....	28
5.2.1	COMANDO DE ENTRADA .....	28
5.2.2	COMANDO DE SAÍDA .....	29
5.3	ESTRUTURA SEQUENCIAL .....	29
5.4	TESTE DE MESA.....	29
5.5	EXERCÍCIOS.....	30
<b>ESTRUTURAS CONDICIONAIS .....</b>		<b>32</b>
6.1	ESTRUTURA CONDICIONAL SIMPLES .....	32
6.2	ESTRUTURA CONDICIONAL COMPOSTA .....	32
6.3	ESTRUTURA CONDICIONAL ENCADEADA.....	33
6.3.1	SELEÇÃO ENCADEADA HETEROGÊNEA.....	34
6.3.2	SELEÇÃO ENCADEADA HOMOGÊNEA.....	34
6.5	EXERCÍCIOS.....	36
<b>SELEÇÃO DE MÚLTIPLA ESCOLHA .....</b>		<b>40</b>
7.1	SELEÇÃO DE MÚLTIPLA ESCOLHA .....	40
7.2	EXERCÍCIOS.....	42
<b>ESTRUTURAS DE REPETIÇÃO .....</b>		<b>44</b>
8.1	ESTRUTURAS DE REPETIÇÃO .....	44
8.2	ESTRUTURA DE REPETIÇÃO COM VARIÁVEL DE CONTROLE .....	44
8.3	ESTRUTURA DE REPETIÇÃO COM TESTE NO INÍCIO .....	45
8.4	ESTRUTURA DE REPETIÇÃO COM TESTE NO FINAL.....	46
8.5	EXERCÍCIOS.....	46
<b>VETORES E MATRIZES.....</b>		<b>48</b>
9.1	ESTRUTURA DE DADOS .....	48
9.2	VARIÁVEIS COMPOSTAS HOMOGÊNEAS.....	48
9.3	VETORES - VARIÁVEIS COMPOSTAS UNIDIMENSIONAIS .....	48
9.3.1	MANIPULAÇÃO DE VETORES .....	48
9.4	MATRIZES - VARIÁVEIS COMPOSTAS MULTIDIMENSIONAIS .....	50
9.4.1	DECLARAÇÃO DE MATRIZES.....	51
9.4.2	MANIPULAÇÃO DE MATRIZES.....	51
9.4.3	PERCORRENDO UMA MATRIZ BIDIMENSIONAL.....	51
9.5	EXERCÍCIOS.....	53
<b>REGISTROS .....</b>		<b>54</b>
10.1	ESTRUTURA DE DADOS .....	54
10.2	REGISTROS.....	54
10.2.1	DECLARAÇÃO DE UM REGISTRO .....	54
10.2.2	MANIPULAÇÃO DE UM REGISTRO .....	54
10.2.3	REGISTRO DE CONJUNTOS.....	55
10.2.4	MANIPULAÇÃO DE REGISTRO DE CONJUNTOS.....	55
10.2.5	CONJUNTO DE REGISTROS.....	56
10.2.6	MANIPULAÇÃO DE CONJUNTO DE REGISTROS.....	56
10.3	EXERCÍCIOS.....	57
<b>MODULARIZAÇÃO .....</b>		<b>58</b>
11.1	DECOMPOSIÇÃO .....	58
11.2	MÓDULOS .....	58
11.3	SUB-ROTINA .....	61
11.3.1	PASSAGEM DE PARÂMETROS POR VALOR .....	63
11.3.2	PASSAGEM DE PARÂMETROS POR REFERÊNCIA .....	63
11.4	FUNÇÃO.....	64
11.5	EXERCÍCIOS.....	66
<b>LISTAS DE EXERCÍCIOS .....</b>		<b>68</b>
	LISTA 1: LÓGICA .....	69
	LISTA 2: ESTRUTURA SEQUENCIAL .....	70
	LISTA 3: ESTRUTURA CONDICIONAL.....	72

<i>LISTA 4: SELEÇÃO DE MÚLTIPLA ESCOLHA</i> .....	75
<i>LISTA 5: REVISÃO</i> .....	76
<i>LISTA 6: ESTRUTURAS DE REPETIÇÃO</i> .....	79
<i>LISTA 7: VETORES E MATRIZES</i> .....	82
<i>LISTA 8: REGISTROS</i> .....	87
<i>LISTA 9: MODULARIZAÇÃO</i> .....	88
<b>BIBLIOGRAFIA</b> .....	<b>90</b>

## 1

**Visão Geral**

*“As coisas são sempre melhores no começo”*  
(Blaise Pascal)

## 1.1 DADOS E INFORMAÇÕES

“**Dados**” são conjuntos de fatos distintos e objetivos, relativos a eventos. Os dados, por si só, tem pouca relevância ou propósito. Por exemplo, se for dito que a temperatura ambiente é de 32°C (celsius), provavelmente todos compreenderão, mas se for dito que a temperatura é de 82°F (fahrenheit), a compreensão dependerá do conhecimento do ouvinte sobre essa unidade de medida. O dado 32°C é rapidamente convertido em sensação térmica, portanto ele tem algum significado ou importância. Nesse caso, pode-se dizer que ele é uma informação. **Informações** são dados com algum significado ou relevância.

Se o ouvinte não tem nenhum conhecimento sobre a unidade de medida F, ela não fornece a exata sensação de frio ou calor.

A informação é a compreensão dos dados, a matéria-prima para o processamento mental. Sem dados e um mecanismo (processo) de compreensão desses dados existe o processamento mental e, se não houver esse processamento mental, os dados não se transformam em informações, continuam sendo apenas dados.

DADO x INFORMAÇÃO	
DADO	INFORMAÇÃO
Data de Nascimento: 16/07/21	Idade: 3 anos (bebê)
Soma de Preço Unitário x Quantidade	Valor Total da Fatura: R\$ 2500,00
Medição x Métrica de Temperatura = 38° C	Quente
Medição x Métrica de Distância = 100 Km	Longe

## 1.2 CONVERSÃO DE DADOS EM INFORMAÇÕES

Os sistemas de computação trabalham somente com dados. Eles permitem a coleta, processamento, armazenamento e distribuição de enormes quantidades de dados. A conversão de dados em informações é uma tarefa do ser humano, mas os sistemas de computação podem auxiliar alguns processos que ajudam nessa conversão:

<b>Contextualização</b>	Relacionar os dados coletados com outros existentes
<b>Categorização</b>	Separar os dados em categorias
<b>Cálculo</b>	Analisar matematicamente ou estatisticamente os dados
<b>Condensação</b>	Resumir os dados para uma forma concisa

## 1.3 DIVISÃO DE TAREFAS – Seres Humanos x Sistemas de Computação

Quando o ser humano trabalha com informações, existem determinadas tarefas que podem ser realizadas:

<b>Pensar / Criar</b>	Absorver e combinar conhecimentos e informações de modo não programado para criar novas informações e conhecimentos. É o processo criativo propriamente dito.
<b>Tomar Decisões</b>	Usar informações para definir, avaliar e selecionar entre possíveis ações a serem tomadas.
<b>Realizar Ações Físicas</b>	Qualquer combinação de movimentos físicos, com algum propósito.
<b>Comunicar-se</b>	Apresentar conhecimentos e informações para outras pessoas, de modo que elas entendam. Daí, a diferença entre comunicar e transmitir.
<b>Processar Dados</b>	Capturar, transmitir, armazenar, recuperar, manipular ou apresentar dados.

Os equipamentos são criados para facilitar e agilizar as tarefas realizadas pelos seres humanos. Isso inclui os sistemas de computação, que têm como uma de suas finalidades até substituir o ser humano em uma ou mais tarefas ligadas à informação. Das tarefas

apresentadas, a mais adequada para os sistemas de computação realizarem é o **Processamento de Dados**. As outras apresentam características humanas difíceis de serem “imitadas”.

Dentro do processamento de dados, algumas tarefas são básicas:

<b>Capturar</b>	Buscar os dados onde eles existem e trazê-los
<b>Manipular</b>	Tratar os dados de forma que possam ser organizados e ganhar sentido (transformando-se em informação)
<b>Armazenar</b>	Guardar os dados de maneira organizada
<b>Recuperar</b>	Buscar os dados que foram armazenados de forma organizada
<b>Apresentar</b>	Mostrar os dados de forma compreensível
<b>Transmitir</b>	Enviar e receber dados de outros locais

A Tecnologia da Informação é formada por dispositivos que processam dados de forma precisa e rápida, facilitando alguma tarefa para o ser humano. O equipamento mais importante dessa tecnologia é o computador, e a informática estuda essa tecnologia.

## 1.4 INFORMÁTICA

**Informática** é o estudo de tudo o que se relaciona à tecnologia da informação. É uma união de trechos de duas outras palavras e foi criada pelos franceses.

INFORMÁTICA = INFORMação + AutoMÁTICA

Outra definição possível para informática é: “O estudo do tratamento da informação, utilizando-se, como ferramenta básica, recursos dos sistemas de computação.”

Este conceito amplia bastante a idéia inicial. Primeiro, porque a informática é tratada como um estudo, por isso, é dotada de conceitos próprios e distintos. Como estudo, seu objetivo é o tratamento da informação, usando como ferramenta os recursos de sistemas de computação, ou seja, o computador e outros recursos ligados a ele.

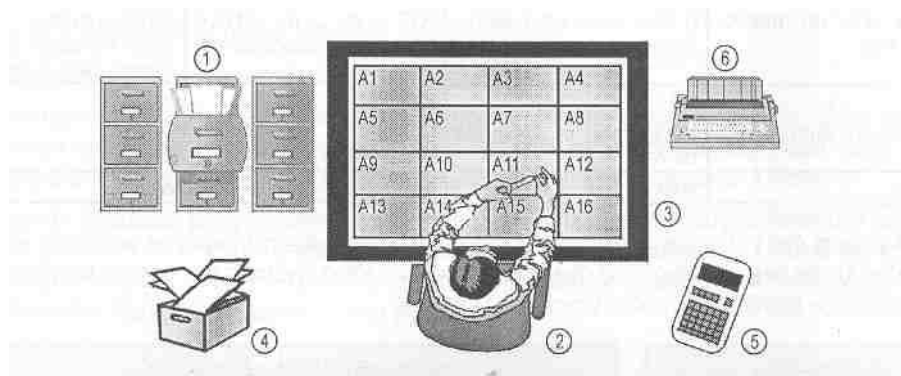
## 1.5 TIPOS DE DADOS

Os sistemas de computação, atualmente, manipulam vários tipos diferentes de dados:

<b>Números</b>	Podem ser organizados, alterados, calculados e armazenados.
<b>Textos</b>	Podem ser escritos, corrigidos, alterados na forma e cor, armazenados e impressos.
<b>Imagens</b>	Podem ser estáticas (em duas ou três dimensões) ou em movimento (animações e vídeos). Podem ser criadas, alteradas, armazenadas e reproduzidas.
<b>Sons</b>	Podem ser gerados eletronicamente (sintetizados) ou gravados diretamente da realidade. Podem ser alterados, armazenados e reproduzidos.

## 1.6 O QUE É UM COMPUTADOR?

Para facilitar a compreensão do funcionamento e dos componentes de um computador, é apresentada, a seguir, uma analogia entre o funcionamento de um computador e o local de trabalho de um operador, formado basicamente pelos utensílios comuns de um escritório (obviamente, sem um computador). Layout e funcionamento desse local de trabalho:



Regras para realizar as tarefas:

1) No arquivo de aço (1), estão armazenadas as instruções para realização de cada tarefa. Essas instruções apresentam uma seqüência de passos a serem seguidos.

2) Quando o operador (2) receber as instruções, ele deve copiar cada uma delas no quadro-negro (3), que possui 16 áreas para isso (A1 -A16). Cada instrução deve ser escrita em uma das áreas livres do quadro-negro, sempre iniciando pela área A1.

3) Após copiar as instruções, o operador deve começar a realizar cada uma delas, respeitando a seqüência. Caso alguma indique ao operador para escrever em uma área já ocupada do quadro, ele deve apagar o conteúdo anterior e escrever o novo conteúdo.

4) Os dados que serão usados para realizar as tarefas encontram-se escritos em fichas empilhadas ao lado do operador, no escaninho (4). As fichas deverão ser usadas na seqüência em que se encontram e, ao ser usada, a ficha deve ser descartada.

5) O operador possui uma calculadora (5) para realizar todos os cálculos matemáticos necessários para a realização da sua tarefa (dependendo das instruções).

6) Para apresentar os resultados da tarefa realizada, o operador possui uma máquina de escrever (6), utilizada para escrever os resultados.

Suponha-se que o operador receba a seguinte seqüência de instruções que estavam armazenadas no arquivo de aço:

- 1) Pegue uma ficha e copie o seu valor no quadro - área A16
- 2) Pegue uma ficha e copie o seu valor no quadro - área A15
- 3) Some o conteúdo de A15 com o de A16 e coloque o resultado em A16
- 4) Se não houver mais fichas, avance para a área A6; caso contrário, avance para a área A5
- 5) Volte para a área A2
- 6) Datilografe o conteúdo de A16
- 7) Pare

O operador copiava as *instruções*, uma a uma, nas primeiras áreas do quadro-negro. O quadro ficava com a seguinte aparência:

<b>A1</b> Pegue uma ficha e copie o seu valor no quadro, área A16	<b>A2</b> Pegue uma ficha e copie o seu valor no quadro, área A15	<b>A3</b> Some o conteúdo de A15 com o de A16 e coloque o resultado em A16	<b>A4</b> Se não houver mais fichas, avance para a área A6; caso contrário, avance para a área A5
<b>A5</b> Volte para a área A2	<b>A6</b> Datilografe o conteúdo de A16	<b>A7</b> Pare	<b>A8</b>
<b>A9</b>	<b>A10</b>	<b>A11</b>	<b>A12</b>

Terminada a cópia das instruções, o operador começa a realizar cada uma delas, na seqüência em que foram apresentadas. Como exemplo, supõe-se que existam, no *escaninho*, quatro fichas com os seguintes valores: 7, 1, 4 e 2. Veja o que acontece no quadro e nas áreas afetadas:

Início	Pegue uma ficha e copie o seu valor na área A16	Pegue uma ficha e copie seu valor na área A15
A15 A16	A15 A16 7	A15 A16 1 7
Some o conteúdo de A15 com o de A16 e coloque o resultado em A16	Se não houver mais fichas, avance para a área A6; caso contrário, avance para a área A5	Volte para a área A2
A15 A16 1 8	A15 A16 1 8	A15 A16 1 8

Pegue uma ficha e copie seu valor na área A15		Some o conteúdo de A15 com o de A16 e coloque o resultado em A16		Se não houver mais fichas, avance para a área A6; caso contrário, avance para a área A5	
A15 4	A16 8	A15 4	A16 12	A15 4	A16 12
Volte para a área A2		Pegue uma ficha e copie seu valor na área A15		Some o conteúdo de A15 com o de A16 e coloque o resultado em A16	
A15 4	A16 12	A15 2	A16 12	A15 2	A16 14
Se não houver mais fichas, avance para a área A6; caso contrário, avance para a área A5		Datilografe o conteúdo de A16		Pare	
A15 2	A16 14	A15 2	A16 14	A15 2	A16 14

A palavra computador vem da palavra latina “*computare*”, que significa calcular. Pode até parecer estranho, mas essa idéia não está de toda errada, mesmo assim, é muito pouco para se ter uma idéia do que seja um computador, então, eis mais uma definição:

**Computador** é uma máquina que recebe e trabalha os dados de maneira a obter um resultado. Para realizar isso, ele é programável, ou seja, responde a um grupo de comandos específicos (instruções) de uma maneira bem definida e pode executar uma lista pré-gravada desses comandos. Essa lista é chamada de programa.

A partir dessa definição, podem ser retiradas algumas conclusões importantes:

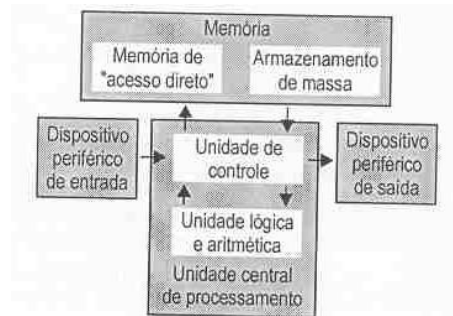
- O computador é uma máquina.
- Realiza um trabalho com os dados para obter resultados.
- O trabalho realizado pelo computador chama-se **Processamento**.
- O computador é programável. Pode realizar somente tarefas bem definidas, e cada uma delas corresponde a uma única instrução, que sempre é realizada da mesma maneira. Além disso, ele pode responder a uma lista de instruções pré-gravadas, realizando uma instrução após a outra.
- Essa lista de instruções pré-gravadas é chamada de **Programa**. Existem computadores que apresentam programas fixos e invariáveis – o computador realiza sempre as mesmas tarefas - que já acompanham o computador. Também existem computadores cujos programas instalados são diferentes, portanto realizam tarefas diferentes de acordo com os programas.

Outra definição para computador: "É um sistema integrado, composto de hardware e de software."

Concluindo:

- O computador é um sistema formado por determinados componentes que, atuando em conjunto, permitem que ele realize as tarefas que foram determinadas. Esse sistema é composto, basicamente, de dois elementos, **Hardware** e **Software**.

- **Hardware** é a parte física do computador, ou seja, o próprio computador e todos os dispositivos ligados a ele (periféricos). O hardware é composto por "dispositivos eletrônicos que fornecem capacidade de computação, dispositivos de interconectividade (por exemplo, switches de rede, dispositivos de telecomunicação) que permitem o fluxo dos dados e dispositivos eletromecânicos (por exemplo, sensores, motores, bombas) que fornecem funções do mundo exterior". Normalmente, o hardware de um sistema de computação apresenta a seguinte estrutura geral:



### 1.6.1 UNIDADE CENTRAL DE PROCESSAMENTO

A Unidade Central de Processamento UCP (CPU - Central Processing Unit), é o cérebro do computador, o componente de hardware que realmente executa as instruções apresentadas pelo programa. A Unidade Central de Processamento possui dois componentes principais:

- **Unidade de Controle** (UC): responsável pelo controle do fluxo dos dados entre as partes do computador e por sua interpretação (se são dados ou instruções). Na simulação de computador, é o operador.
- **Unidade Lógica e Aritmética** (ULA): responsável pelos cálculos e pela manipulação dos dados. Na simulação de computador, é a calculadora.

### 1.6.2 MEMÓRIA

Possibilita ao computador armazenar dados e instruções durante o processamento. Podem existir dois tipos de memória em um computador:

- Memória de "Acesso Direto" ou Memória Principal, também conhecida por RAM (Memória de Acesso Randômico): o armazenamento dos dados e programas é temporário. Na simulação de computador, é o quadro-negro.
- Dispositivos de Armazenamento Secundários: dispositivos que permitem ao computador armazenar permanentemente grandes quantidades de dados ou programas. Na simulação de computador, é o arquivo de aço.

### 1.6.3 DISPOSITIVOS DE ENTRADA

Dispositivos através dos quais os dados e instruções entram no sistema de computação para o processamento. Traduz essas entradas para um código que a Unidade Central de Processamento entende. Na simulação de computador, é o escaninho com as fichas.

### 1.6.4 DISPOSITIVOS DE SAÍDA

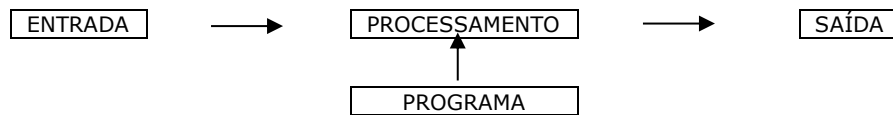
Dispositivos que permitem a visualização dos resultados do processamento dos dados. Na simulação de computador, é a máquina de escrever.

- **-Software** é o conjunto de instruções (programas de computador) que, quando executadas, produzem o desempenho desejado e dados que permitem que os programas manipulem adequadamente a informação. É a parte lógica do computador, aquela com a qual não existe contato físico. Na simulação de computador, são as instruções realizadas pelo operador e o conteúdo das fichas utilizadas por ele.

Observando a simulação de computador, apresentada anteriormente, percebe-se que as operações de um computador dependem da lógica das instruções que ele realiza. Essas instruções são criadas pelo homem e alimentadas no computador, que apenas as executa, de acordo com os seus componentes internos.

## 1.7 CICLO DE PROCESSAMENTO

O computador, de maneira simplificada, realiza uma determinada seqüência ("ciclo") para processar os dados. É chamado de Ciclo de Processamento, e é representado graficamente da seguinte maneira:



### 1.7.1 ENTRADA / SAÍDA DE DADOS

O computador lê os dados a serem processados (a partir de periféricos de entrada) e os coloca na memória principal. Após o processamento, o computador envia uma cópia dos resultados, a partir da memória principal, para os periféricos de saída. Essa saída pode ser composta por dados de entrada modificados ou por novos dados gerados pelo processamento.

### 1.7.2 PROGRAMA

Lista de instruções que o computador deve seguir, ou seja, é a seqüência das operações necessárias para que os dados sejam processados. Normalmente esse programa está gravado num dispositivo de armazenamento secundário e é copiado para a memória principal do computador durante o processamento.

### 1.7.3 PROCESSAMENTO

É o trabalho realizado pela CPU do computador. O que ela faz depende do programa, mas quem processa os dados é o hardware do computador. Para que o processamento aconteça, os dados devem estar na memória principal.

## 1.8 REPRESENTAÇÃO DA INFORMAÇÃO

A informação e os dados necessitam de meios para que sejam exibidos. Normalmente são utilizados modelos que imitam a realidade.

O sistema de computação funciona, basicamente, da mesma maneira, pois **imita** a informação real criando um modelo eletrônico para trabalhar. Esse modelo é numérico e aritmético. Alguns pontos em comum entre os equipamentos de computação e a matemática permitem essa imitação da realidade. Esses pontos em comum são a **Numeração** e a **Aritmética Binária**.

Os sistemas de computação trabalham com o sistema de numeração binário. Cada símbolo 0 ou 1 da numeração binária é chamado de dígito binário. Em inglês, **Binary Digit**, que resulta: **Binary digit = BIT**

Como esse bit é usado para modelar (representar a informação), pode-se definir que "Bit é a menor unidade da informação".

Os bits servem muito bem para a representação de números, mas o sistema de computação não trabalha apenas com informações numéricas, então, como representar letras e símbolos?

Para entender como o sistema faz isso, imagine a existência de duas lâmpadas e a necessidade de criar uma maneira (um modelo) para indicar o estado do movimento de um carro em determinado momento.

As lâmpadas apresentam dois "estados" possíveis: acesa ou apagada. Com isso foi criada uma tabela que associa os estados das lâmpadas aos estados do movimento do carro.

Se em vez de lâmpadas houvesse bits – 0 (apagada) ou 1 (acesa), a tabela ficaria com os valores apresentados à direita.



0 0	Carro andando em frente
0 1	Carro virando à direita
1 0	Carro virando à esquerda
1 1	Carro parado

Com isso é possível criar tabelas de equivalência entre as combinações possíveis dos bits e as informações que devem ser representadas. Quando são usados dois bits, o número de combinações possíveis é quatro, pois na numeração binária existe a seguinte relação: Número de combinações =  $2^n$  sendo  $n$  = número de bits.

O sistema de computação utiliza uma tabela de equivalência entre combinações de bits e caracteres (números, letras e símbolos). É claro que, se o sistema utilizasse apenas dois bits, só conseguiria representar quatro caracteres, o que não é o caso, pois ele pode utilizar qualquer quantidade de bits para representar os dados.

Normalmente, utilizam-se grupos de oito bits. Usando a fórmula anterior: número de combinações =  $2^8 = 256$ . Portanto, o sistema de computação utilizando oito bits consegue representar até 256 caracteres diferentes (256 combinações diferentes).

## 1.9 BYTES

Cada um desses grupos de oito bits é chamado de **byte**. Pode-se considerar cada byte representando um caractere, portanto o byte é utilizado para medir o tamanho dos trabalhos realizados no sistema de computação, principalmente se for levado em consideração que sistemas antigos utilizavam somente textos em seus trabalhos. Por exemplo: um livro com 250 páginas tem, aproximadamente, 1.000.000 de caracteres (contando-se espaços, que também são caracteres). Caso fosse usado um computador para editar esse mesmo texto, ele continuaria tendo o mesmo número de caracteres que o livro real, mas esses caracteres seriam modelados em bytes. Esse texto seria representado, então, por 1.000.000 de bytes, ou melhor, o tamanho desse texto para o computador seria de 1.000.000 de bytes.

Como em outras unidades de medida, no caso de bytes, são usados múltiplos para representar grandes quantidades (por exemplo, 1000 m = 1 km). Estes símbolos servirão para fazer um arredondamento de valores, o que facilitará a operação:

Quantidade de Bytes	Valor	Nome
$2^{10} = 1024$ bytes	1024 bytes	1 Kb – Kilobyte
$2^{20} = 1.048.576$ bytes	1024 Kb	1 Mb – Megabyte
$2^{30} = 1.073.741.824$ bytes	1024 Mb	1 Gb – Gigabyte
$2^{40} = 1.099.511.627.776$ bytes	1024 Gb	1 Tb – Terabyte

## 1.10 EXERCÍCIOS

1) Verifique a configuração de algum computador que você tenha acesso. Verifique tipo do processador, capacidade de memória RAM, capacidade do HD e sistema operacional utilizado.

2) Selecione dois anúncios de propagandas de lojas de computadores. Um para computadores de marca (Itautec, Dell, HP) e outro para computadores sem marca. Compare configurações e preços.

3) Explique a diferença entre dados e informações.

4) Baseado na resposta anterior, comente porque o computador processa dados e não informações.

5) Entre os componentes de um sistema de computação, qual é o responsável pelo processamento dos dados?

6) Defina programa e a sua função em um sistema de computação.

7) Utilizando as suas palavras e os conceitos apresentados neste capítulo, defina computador.

8) Quais são os principais componentes de um sistema de computação?

9) Quando se afirma que um computador é programável, o que isso significa?

10) Utilize a simulação de computador apresentada para "processar" as instruções a seguir e apresente o resultado final dessa tarefa. No escaninho, existem quatro fichas com os seguintes valores: 5, 3, 2 e 2:

- a) Escreva 1 em A14;
- b) Pegue uma ficha e copie o seu valor no quadro - área A 16;
- c) Pegue uma ficha e copie o seu valor no quadro - área A 15;
- d) Some o conteúdo de A15 com o de A16 e coloque o resultado em A16;
- e) Some 1 ao valor de A14 e coloque o resultado em A14;
- f) Se não houver mais fichas, avance para a área A8; caso contrário, avance para a área A7;
- g) Volte para a área A3;
- h) Divida o valor de A16 pelo valor de A14 e coloque o resultado em A16;
- i) Datilografe o conteúdo de A16;
- j) Pare.

-X-

## 2

**Noções de Lógica**

*“Cada ferramenta carrega consigo o espírito com o qual foi criada.”*  
(Werner Karl Heisenberg)

## 2.1 LÓGICA

Coerência e racionalidade. "Arte do bem pensar", "ciência das formas do pensamento". Visto que nossa razão pode funcionar desordenadamente, a lógica tem em vista a "correção do raciocínio", colocando "ordem no pensamento".

Exemplos:

- Todo mamífero é um animal
- Todo cavalo é um mamífero
- Portanto, todo cavalo é um animal
  
- Kaiton é país do planeta Stix
- Todos os Xinpins são de Kaiton
- Logo, todos os Xinpins são Stixianos

A lógica objetiva a criação de uma representação mais FORMAL. Utilizamos sempre a lógica para pensar, falar, escrever ou agir corretamente. Precisamos, para todas essas atividades, colocar ordem no pensamento.

- A gaveta está fechada
- A caneta está dentro da gaveta
- Precisamos primeiro abrir a gaveta para depois pegar a caneta
  
- Anacleto é mais velho que Felisberto
- Felisberto é mais velho que Marivaldo
- Portanto, Anacleto é mais velho que Marivaldo

## 2.2 LÓGICA DE PROGRAMAÇÃO

Significa o uso correto das leis do pensamento, da "ordem da razão" e de processos de raciocínio e simbolização formais na programação de computadores, objetivando racionalidade no desenvolvimento de técnicas que produzam soluções logicamente válidas e coerentes que resolvam os problemas que se deseja programar.

Um mesmo raciocínio pode ser expresso em qualquer um dos inúmeros idiomas existentes, mas continuará representando o mesmo raciocínio, usando apenas outra convenção.

Acontece exatamente a mesma coisa na Lógica de Programação. Pode ser representada em qualquer uma das inúmeras linguagens de programação existentes. Só que, no raciocínio geral, uma grande diversidade de detalhes computacionais não tem nada a ver com o mesmo. Portanto, para escapar dessa "Torre de Babel" e, ao mesmo tempo, representar mais fielmente o raciocínio da Lógica de Programação, utilizamos os Algoritmos.

O objetivo principal da Lógica de Programação é a construção de algoritmos coerentes e válidos.

## 2.3 ALGORITMO

Algoritmo é um conjunto de procedimentos a ser seguido para que um determinado problema seja resolvido. Para trazer os algoritmos para a memória do computador é necessário que o mesmo seja programado, ou seja, convertido em uma linguagem que possa ser entendida pela máquina.

É uma seqüência de passos que visam atingir um objetivo bem definido. Cada passo deve ser simples e sem ambigüidade.

Apesar do nome pouco usual, algoritmos são comuns em nosso cotidiano, como, por exemplo, uma receita de bolo. Na receita existe uma série de ingredientes necessários e uma seqüência de diversos passos (ações) que devem ser fielmente executados para que se consiga o alimento desejado, conforme se esperava antes do início da atividade (objetivo bem definido).

Portanto, ao elaborar um algoritmo devemos: Especificar ações claras e precisas → Partindo de um estado inicial → Após um período de tempo finito → Produzem um estado final previsível e bem definido.

Um algoritmo deve garantir que sempre que seja executado, sob as mesmas condições, produza o mesmo resultado.

Obs.:

Observe a seqüência: 1, 6, 11, 16, 21, 26,...

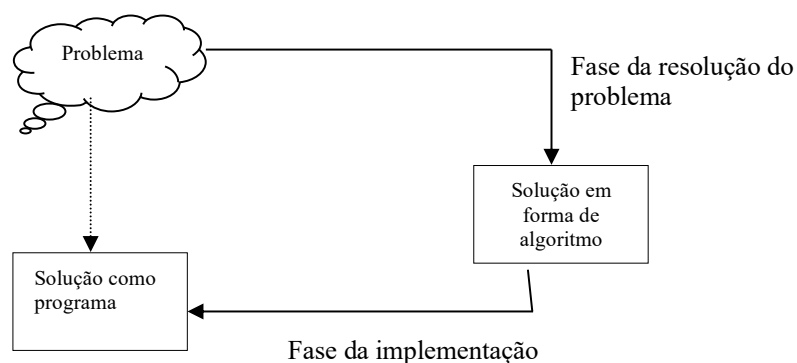
Qual o sétimo elemento?

Qual o padrão de comportamento?

## 2.4 PROGRAMAÇÃO

Programar consiste em elaborar um conjunto finito de instruções reconhecidas pela máquina, de forma que o computador as execute.

É a implementação de um algoritmo em determinada linguagem de programação.



## 2.5 IMPORTÂNCIA DE UM ALGORITMO

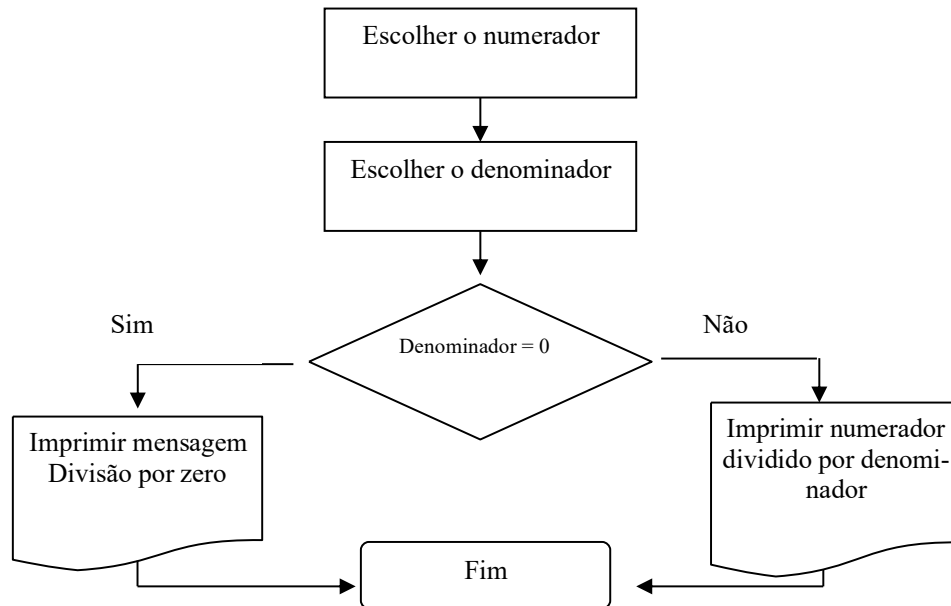
Representar mais fielmente o raciocínio envolvido na Lógica de Programação bem como codificar para qualquer linguagem de programação. Existem várias formas de representar um algoritmo: Linguagem Natural, Fluxograma e Linguagem Estruturada (Algoritmo - Português) que é a mais utilizada. Para exemplificar utilizaremos o seguinte problema: "Dividir dois números".

### 2.5.1 LINGUAGEM NATURAL

- Escolher o numerador
- Escolher o denominador

- Se o denominador for igual a zero, então Escreva que não existe divisão por zero
- Senão, divida o numerador pelo denominador

### 2.5.2 FLUXOGRAMA



### 2.5.3 LINGUAGEM ESTRUTURADA

```

Algoritmo PrimeiroExemplo {
  real numerador, denominador, resultado;
  escreva ("Digite o numerador: ");
  leia (numerador);
  escreva ("Digite o denominador: ");
  leia (denominador);
  se (denominador = 0) {
    escreva ("Divisão por Zero");
  }
  senão {
    resultado = numerador / denominador;
    escreva ("A divisão = ", resultado);
  }
}
  
```

## 2.6 EXEMPLOS

Muitas vezes realizamos tarefas aparentemente óbvias demais como, por exemplo, trocar uma lâmpada, sem percebermos seus pequenos detalhes. Vejamos esse primeiro algoritmo:

1. pegar uma escada;
2. posicionar a escada embaixo da lâmpada;
3. buscar uma lâmpada nova;
4. subir na escada;
5. retirar a lâmpada velha;
6. colocar a lâmpada nova;

Neste primeiro algoritmo, seguimos uma determinada seqüência de ações que fazem com que ele seja seguido naturalmente por qualquer pessoa, estabelecendo um padrão de comportamento, pois qualquer pessoa agiria da mesma maneira.

MAS, e se a lâmpada não estivesse queimada? A execução das ações conduziria a uma troca, independentemente de a lâmpada estar ou não queimada, pois essa possibilidade não foi prevista. Uma solução seria:

1. pegar uma escada;
2. posicionar a escada embaixo da lâmpada;
3. buscar uma lâmpada nova;
4. acionar o interruptor;
5. se a lâmpada não acender, então
  - 5.1. subir na escada;
  - 5.2. retirar a lâmpada velha;
  - 5.3. colocar a lâmpada nova;

Este algoritmo funciona, mas pode ser melhorado visto que buscamos uma lâmpada e a escada sem saber se serão necessárias. Mudemos o teste condicional para o início da seqüência de ações:

1. acionar o interruptor;
2. se a lâmpada não acender, então
  - 2.1. pegar uma escada;
  - 2.2. posicionar a escada embaixo da lâmpada;
  - 2.3. buscar uma lâmpada nova;
  - 2.4. subir na escada;
  - 2.5. retirar a lâmpada velha;
  - 2.6. colocar a lâmpada nova;

Há muitas formas de resolver um problema, afinal cada pessoa pensa e age de maneira diferente. Isto significa que, para este mesmo problema de trocar lâmpadas, poderíamos ter diversas soluções diferentes e corretas (se atingissem o resultado desejado de efetuar a troca), portanto, o bom senso e a prática de lógica de programação é que indicarão a solução mais adequada, que com menos esforço e menor objetividade produz o resultado almejado.

O algoritmo acima não prevê ainda o fato de a lâmpada nova não funcionar e, portanto, não atingir o objetivo nesta situação específica. Melhorando o algoritmo:

1. acionar o interruptor;
  2. se a lâmpada não acender, então
    - 2.1. pegar uma escada;
    - 2.2. posicionar a escada embaixo da lâmpada;
    - 2.3. buscar uma lâmpada nova;
    - 2.4. subir na escada;
    - 2.5. retirar a lâmpada velha;
    - 2.6. colocar a lâmpada nova;
    - 2.7. se a lâmpada nova não acender, então
      - 2.7.1. buscar uma lâmpada nova;
      - 2.7.2. subir na escada;
      - 2.7.3. retirar a lâmpada queimada;
      - 2.7.4. colocar outra lâmpada nova;
      - 2.7.5. se a lâmpada nova não acender, então
        - 2.7.5.1. buscar uma lâmpada nova;
        - 2.7.5.2. subir na escada;
        - 2.7.5.3. retirar a lâmpada queimada;
        - 2.7.5.4. colocar outra lâmpada nova;
- (repite os quatro últimos passos até quando?)

Percebemos que as ações cessarão quando conseguirmos colocar uma lâmpada que acenda; caso contrário, ficaremos testando indefinidamente (note que o interruptor continua acionado!) Temos que expressar a repetição da ação sem repetir o texto que representa a ação, assim como determinar um limite para tal repetição, para garantir uma condição de parada. Observemos que o número de repetições é indefinido, mas finito.

1. acionar o interruptor;
2. se a lâmpada não acender, então

- 2.1. pegar uma escada;
- 2.2. posicionar a escada embaixo da lâmpada;
- 2.3. buscar uma lâmpada nova;
- 2.4. subir na escada;
- 2.5. retirar a lâmpada velha;
- 2.6. colocar a lâmpada nova;
- 2.7. enquanto a lâmpada nova não acender, faça
  - 2.7.1. buscar uma lâmpada nova;
  - 2.7.2. subir na escada;
  - 2.7.3. retirar a lâmpada queimada;
  - 2.7.4. colocar outra lâmpada nova;

Até agora estamos efetuando a troca de uma única lâmpada. Na verdade, estamos testando um soquete (acionado por um interruptor), trocando tantas lâmpadas quantas forem necessárias para assegurar que o conjunto funcione. O que faríamos se tivéssemos mais soquetes a testar, por exemplo, 10 soquetes? A solução aparentemente mais óbvia seria repetir o algoritmo de uma única lâmpada para os 10 soquetes existentes. Entretanto, isso não é uma maneira eficiente de se resolver esse problema. O conjunto de ações desenvolvidas é exatamente igual para os 10 soquetes. Então, podemos fazer com que o algoritmo volte a executar o conjunto de ações relativas a um único soquete tantas vezes quantas forem desejadas (no nosso caso, 10 vezes). O algoritmo ficaria assim:

1. ir até o interruptor do **primeiro** soquete;
2. enquanto a quantidade de soquetes testados for menor do que dez, faça
  - 2.1. acionar o interruptor;
  - 2.2. se a lâmpada não acender, então
    - 2.2.1. pegar uma escada;
    - 2.2.2. posicionar a escada embaixo da lâmpada;
    - 2.2.3. buscar uma lâmpada nova;
    - 2.2.4. subir na escada;
    - 2.2.5. retirar a lâmpada velha;
    - 2.2.6. colocar a lâmpada nova;
  - 2.3. enquanto a lâmpada nova não acender, faça
    - 2.3.1. buscar uma lâmpada nova;
    - 2.3.2. subir na escada;
    - 2.3.3. retirar a lâmpada queimada;
    - 2.3.4. colocar outra lâmpada nova;
  - 2.4. ir até o interruptor do **próximo** soquete;

Quando a condição “quantidade de soquetes testados for menor do que dez” for verdadeira, as ações responsáveis pela troca ou não de um único soquete serão executadas. Caso a condição de parada seja falsa, ou seja, todos os dez soquetes já tiverem sido testados, nada mais será executado.

## 2.7 EXERCÍCIOS

11)Elabore algoritmo passo-a-passo para trocar um pneu furado. Admita que estão disponíveis no porta malas do carro todos os materiais necessários.

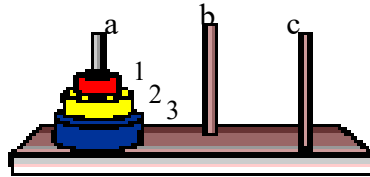
12)Elabore algoritmo passo-a-passo para fazer um bolo. Admita que estão disponíveis todos os ingredientes necessários.

13)Elabore algoritmo passo-a-passo para pegar uma coca-cola em uma máquina de refrigerantes. Admita que você esteja em frente à máquina e a ficha esteja em sua carteira/bolsa.

14)Elabore algoritmo passo-a-passo representando seus atos de um dia da semana (meio de semana) desde o momento que você abre o olho de manhã até o momento que você os fecha para dormir

15) Três jesuítas e três canibais precisam atravessar um rio; para tal dispõem de um barco com capacidade para duas pessoas. Por medidas de segurança, não se deve permitir que em alguma margem a quantidade de jesuítas seja inferior à de canibais. Qual a solução para efetuar a travessia com segurança? Elabore um algoritmo mostrando a resposta, indicando as ações que concretizam a solução deste problema.

16) Elabore um algoritmo que mova três discos de uma Torre de Hanói, que consiste em três hastes (a, b, c), uma das quais serve de suporte para três discos de tamanhos diferentes (1, 2, 3), os menores sobre os maiores. Pode-se mover um disco de cada vez para qualquer haste, contanto que nunca seja colocado um disco maior sobre um menor. O objetivo é transferir os três discos para outra haste



-X-

## 3

## Itens Fundamentais

*“Quantas maçãs caíram na cabeça de Newton até ele ter inspiração!”*  
(Robert Frost)

## 3.1 INTRODUÇÃO

Todo o trabalho realizado por um computador é baseado na manipulação de dados contidos em sua memória.

A memória do computador pode ser comparada a um conjunto de caixas numeradas. Em cada caixa podemos guardar uma informação. O número da caixa serve para localizá-la e possui o nome de endereço.

1	2	3	4	Guarde o valor João na posição de memória 3
		João	25	
5	6	7	8	Guarde o valor 25 na posição de memória 4

Para armazenar uma informação na memória, não precisamos referenciar o endereço da posição. Isso pode ser feito pela associação de um nome a cada posição. A associação é feita por um mecanismo interno das linguagens de programação que associa nomes que criamos às posições de memória. Este mecanismo utiliza uma tabela de símbolos.

Tabela de Símbolos	
NOME	POSIÇÃO DA MEMÓRIA
Idade	1
Cpf	2

## 3.2 TIPOS DE DADOS

Inicialmente vamos diferenciar **dado de informação**. Se falarmos: “A escada possui 8 degraus”. O valor 8 é um dado, e informação é a associação de 8 com o número de degraus da escada.

Iremos trabalhar com quatro tipos primitivos. Usaremos estes tipos básicos na construção de algoritmos:

## 3.2.1 INTEIRO

Podem ser positivos, negativos ou nulos e não possuem parte decimal.

- Eu tenho 2 filhos.
- A escada possui 9 degraus.
- Meu vizinho ganhou 1 carro novo.

## 3.2.2 REAL

Podem ser positivos, negativos ou nulos e possuem parte decimal.

- Ele tem 1,65 m de altura.
- Meu saldo no banco é de R\$ 500,30.
- No momento Geraldo está pesando 73,5 kg.

## 3.2.3 CARACTER

Podem ser as letras maiúsculas (A...Z), as letras minúsculas (a...z), os números (0...9) e os caracteres especiais (ex.: &, #, @, ?, +). São sempre representados entre aspas no algoritmo.

- Constava na prova: “Use somente caneta preta!”.
- O parque municipal estava cheio de placas: “Não pise na grama!”.

### 3.2.4 LÓGICO

Possuem apenas duas possibilidades de representação, ou seja, podem assumir apenas duas situações (Verdadeiro ou Falso, Verdade ou Falsidade, V ou F). Convencionaremos que dados lógicos poderão assumir um dos seguintes valores: Verdade ou Falsidade (V ou F)

- A porta pode estar aberta ou fechada.
- A lâmpada pode estar acesa ou apagada.

### 3.3 FORMAÇÃO DE IDENTIFICADORES

Os identificadores são os nomes das variáveis, das constantes, das rotinas etc. Existem regras básicas para a formação de identificadores:

1. Os caracteres que você pode utilizar na formação dos identificadores são: números, letras maiúsculas e minúsculas e o caracter sublinhado ( \_ );
2. Devem começar por um caracter alfabético ou pelo caracter sublinhado;
3. Podem ser seguidos por mais caracteres alfabéticos ou numéricos;
4. Não são permitidos espaços em branco e caracteres especiais (@, \$, #, +, !, %...);
5. Não é permitido usar palavras reservadas nos identificadores, ou seja, palavras que pertençam à linguagem de programação.

Alguns exemplos válidos: Lambda, Y, CK32, K9, Nota, CBC, SALARIO, CONTADOR. Alguns exemplos inválidos: 5Y, F(15), B:C, Nota1/2; WXZ\*, B&CC,@143, a-b, Q:c.

É importante observar que no desenvolvimento de algoritmos, o nome de identificador deve ser o mais significativo possível, pois desta maneira, o entendimento do mesmo se torna mais fácil.

### 3.4 CONSTANTES

Um dado constante é o que não sofre nenhuma variação no decorrer do tempo. Seu valor é constante do início ao fim da execução do algoritmo, assim como é constante para execuções diferentes no tempo. As definições das constantes são feitas antes das declarações das variáveis. No formato apresentado logo a seguir, *identificador* é o nome da constante que está sendo criada.

**Formato:** `Constante identificador = valor;`

**Exemplos:**

```
Constante FRASE = "Não Fume";
Constante NOME = "Bruna";
Constante FATOR = 1,24;
```

**Observação:** É importante observar que dados entre aspas caracterizam caracteres, mesmo que possuam valores numéricos. Veja os exemplos a seguir:

```
Constante SORTE = "13";
Constante TMP = "E%R#Y*";
Constante LIXO = "Abc34F19";
Constante NUMCHAR = "9,16";
```

### 3.5 VARIÁVEIS

Um dado é classificado como variável quando pode ter seu valor alterado em algum instante durante a execução do algoritmo. Exemplos: cotação do dólar, peso de uma pessoa, índice da inflação.

Embora uma variável possa assumir diferentes valores, ela só pode armazenar um valor a cada instante. Toda variável é identificada por um nome ou identificador.

Exemplo para ilustrar a diferença entre valores constantes e variáveis: Construção de um algoritmo para calcular o valor da área de uma circunferência. A fórmula que expressa a área da circunferência é  $\pi R^2$ , onde  $\pi$  tem valor **constante** de 3,1416... independente de qual seja a circunferência (vale para todas as ocasiões em que calculamos a área). Já o valor de R (raio) é dependente da circunferência que estamos calculando. Portanto, é **variável** a cada execução do algoritmo.

**Formato:** Tipo de Dado lista\_de\_identificadores;

**Exemplos:**

```
inteiro numero, a;
real xyz, valor;
caracter nome, endereco;
lógico resposta;
```

**Observações:** Como visto anteriormente, os tipos primitivos que iremos trabalhar são: inteiro, real, caracter e lógico. No exemplo anterior, a variável RESPOSTA é o nome de um local de memória que só pode conter valores do tipo lógico, ou seja, verdadeiro (V) ou falso (F), assim como o identificador NUM é o nome de um local de memória que só pode conter valores do tipo inteiro e assim por diante.

Não é permitido que mais de uma variável possua o mesmo identificador, visto que se isso acontecer não será possível saber que variável usar. Só podemos guardar dados em variáveis do mesmo material, ou seja, uma variável do tipo primitivo inteiro só poder armazenar números inteiros; uma variável do tipo primitivo lógico, somente verdade (V) ou falsidade (F) e assim por diante.

### 3.6 COMENTÁRIOS

Todo algoritmo deve conter comentários, a fim de que as pessoas possam entendê-lo mais facilmente. Os comentários são representados por um texto, ou simplesmente uma frase, que aparece sempre após o símbolo //. Os comentários podem ser colocados em qualquer ponto do algoritmo onde se façam necessários.

**Formato:** // Comentário de uma linha só

```
/* comentário de mais de uma linha
Quantas forem necessárias */
```

**Exemplos:**

```
/* Este programa serve de exemplo
Autora: Lucília Ribeiro
Data: agosto/2024 */
real salario;           //salário do funcionário
inteiro: numPecas;     //número de peças
```

### 3.7 EXERCÍCIOS

17) Determine qual é o tipo primitivo de informação presente nas sentenças a seguir:

- a) A placa "Pare" tinha 2 furos.
- b) Josefina subiu 5 degraus para pegar uma maçã boa.
- c) Alberta levou 3,5 horas para chegar ao hospital onde deu à luz uma menina.
- d) Astrogilda pintou em sua camisa: "Preserve o meio ambiente", e ficou devendo 23,00 ao vendedor de tintas.
- e) Felisberto recebeu sua medalha de número 18 por ter alcançado a marca de 57,3 segundos nos 100 metros rasos.

18) Identifique o tipo dos dados: (I)nteiro, (R)eal, (C)aracter e (L)ógico

( ) verdadeiro      ( ) 45,0      ( ) 1234      ( ) 0,0

- |                                 |                                 |                                  |                                    |
|---------------------------------|---------------------------------|----------------------------------|------------------------------------|
| <input type="checkbox"/> "aula" | <input type="checkbox"/> "c*d"  | <input type="checkbox"/> -234    | <input type="checkbox"/> "1 2 3 4" |
| <input type="checkbox"/> -0,342 | <input type="checkbox"/> 35,23  | <input type="checkbox"/> "34"    | <input type="checkbox"/> -18,589   |
| <input type="checkbox"/> " "    | <input type="checkbox"/> -354,0 | <input type="checkbox"/> -15,0   | <input type="checkbox"/> falso     |
| <input type="checkbox"/> 0      | <input type="checkbox"/> 897    | <input type="checkbox"/> "falso" | <input type="checkbox"/> -23       |

19) Assinale os identificadores válidos e, para cada identificador não assinalado, explique a razão de ele ser inválido.

- |         |        |             |            |            |
|---------|--------|-------------|------------|------------|
| a) (X)  | b) U2  | c) AH!      | d) "ALUNO" | e) #55     |
| f) KM/L | g) UYT | h) ASDRUBAL | i) AB*C    | j) O&O     |
| l) P{O} | m) B52 | n) Rua      | o) CEP     | p) dia/mês |

20) Supondo que as variáveis *nota*, *nome*, *matricula*, *sexo* sejam utilizadas para armazenar a nota do aluno, nome do aluno, o número da matrícula e o sexo, declare-as corretamente na linguagem algorítmica, associando o tipo primitivo adequado ao dado que será armazenado.

21) Encontre os erros da seguinte declaração de variáveis:

```
inteiro endereco, numFilhos;
caracter idade, x;
real xpto, c, peso, R$;
logico resposta, c;
```

-X-

## 4

**Expressões e Operadores**

*“A chave usada sempre brilha”*  
(Benjamim Franklin)

**4.1 INTRODUÇÃO**

Assim como na matemática, realizamos diversas operações dependendo do tipo de dado utilizado nos algoritmos.

**4.2 EXPRESSÕES ARITMÉTICAS**

As expressões aritméticas são escritas linearmente usando a notação matemática, tendo como resposta sempre um valor numérico. Seus operandos são constantes ou variáveis do tipo numérico (inteiro ou real) e seus operadores são operadores aritméticos, ou seja, símbolos que representam as operações básicas da matemática.

O conjunto de operações básicas adotadas nos algoritmos são: Adição, Subtração, Multiplicação e Divisão.

Operador	Função	Exemplos
+	Adição	$5 + 1$ , $A + B$
-	Subtração	$5 - 2$ , $X - Z$
*	Multiplicação	$4 * 2$ , $Y * W$
/	Divisão	$27/3$ , $B1/B2$

As operações de radiciação e potenciação serão realizadas através do uso das palavras-chave **sqrt** e **pot**, conforme tabela abaixo.

Operador	Função	Significado	Exemplos
pot(X,Y)	Potenciação	X elevado a Y	pot(2,3) = 8
sqrt(X)	Radiciação	Raiz quadrada de X	sqrt(9) = 3

Algumas outras operações matemáticas não-convencionais, mas úteis na construção de algoritmos são: o resto da divisão e o quociente da divisão inteira, mostrados na tabela a seguir.

Operador	Função	Exemplos
mod	Resto da divisão	$9 \text{ mod } 4$ resulta em 1
div	Quociente da divisão	$9 \text{ div } 4$ resulta em 2

**Prioridades:** Na resolução das expressões aritméticas, as operações guardam uma hierarquia entre si.

Operadores	Prioridade
1ª	parênteses mais internos
2ª	rad, pot
3ª	/, *, div, mod
4ª	+, -

Para operações de mesma prioridade, devemos resolver a expressão na seqüência existente, ou seja, da esquerda para a direita. Para alterar a prioridade da tabela, utilizamos parênteses mais internos.

**Exemplos:**

- a)  $5 + 9 + 7 + 8/4$   
 $5 + 9 + 7 + 2$   
 23
- b)  $1 - 4 * 3/6 - \text{pot}(3,2)$   
 $1 - 4 * 3/6 - 9$   
 $1 - 12/6 - 9$   
 $1 - 2 - 9$   
 - 10
- c)  $\text{pot}(5,2) - 4/2 + \text{rad}(1 + 3 * 5)/2$   
 $\text{pot}(5,2) - 4/2 + \text{rad}(1 + 15)/2$   
 $\text{pot}(5,2) - 4/2 + \text{rad}(16)/2$   
 $25 - 4/2 + 4/2$   
 $25 - 2 + 2$   
 25

**4.3 EXPRESSÕES LÓGICAS**

As expressões lógicas são expressões que trabalham com operadores relacionais ou lógicos, tendo como resposta um valor lógico (verdadeiro ou falso).

Uma relação é uma comparação entre valores do mesmo tipo primitivo. Esses valores podem ser representados por constantes, variáveis ou expressões aritméticas.

Os operadores relacionais utilizados estão listados na tabela a seguir. O resultado obtido de uma relação é SEMPRE um valor lógico.

Operador	Descrição
==	Igual a
!=	Diferente de
<=	Menor ou Igual a
>=	Maior ou Igual a
<	Menor que
>	Maior que

**Exemplos:**

- a)  $A + B == C$   
 (Essa relação retornará verdadeiro se o valor de  $A + B$  for igual ao de  $C$ , e retornará falso caso contrário).
- b)  $2 * 4 == 24 / 3$   
 $8 == 8$   
 V
- c)  $15 \text{ mod } 4 < 19 \text{ mod } 6$   
 $3 < 1$   
 F

Os operadores lógicos utilizados com expressões lógicas bem como as prioridades estão na seguinte tabela:

Operador	Comando	Descrição	Prioridade
E	&&	Conjunção	2ª
OU		Disjunção	3ª
NÃO	!	Negação	1ª

**Tabela-verdade** é o conjunto de todas as possibilidades combinatórias entre os valores de diversas variáveis lógicas e um conjunto de operadores lógicos.

OPERAÇÃO DE NEGAÇÃO	
A	NÃO A
V	F
F	V

OPERAÇÃO DE CONJUNÇÃO		
A	B	A and B (A && B)
V	V	V
V	F	F
F	V	F
F	F	F

OPERAÇÃO DE DISJUNÇÃO		
A	B	A or B (A    B)
V	V	V
V	F	V
F	V	V
F	F	F

**Exemplos:**

- a)  $2 < 5 \ \&\& \ 15 / 3 = 5$   
 $2 < 5 \ \&\& \ 5 = 5$   
 $V \ \&\& \ V$   
 $V$
- b)  $2 < 5 \ || \ 15 / 3 = 5$   
 $2 < 5 \ || \ 5 = 5$   
 $V \ || \ V$   
 $V$
- c)  $F \ || \ 20 \ \text{div} \ (18/3) \ != \ (21/3) \ \text{div} \ 2$   
 $F \ || \ 20 \ \text{div} \ 6 \ != \ 7 \ \text{div} \ 2$   
 $F \ || \ 3 \ != \ 3$   
 $F \ || \ F$   
 $F$
- d)  $\text{NÃO } V \ || \ \text{pot}(3,2)/3 < 15 - 35 \ \text{mod} \ 7$   
 $\text{NÃO } V \ || \ 9/3 < 15 - 0$   
 $\text{NÃO } V \ || \ 3 < 15$   
 $\text{NÃO } V \ || \ V$   
 $F \ || \ V$   
 $V$

## 4.4 EXERCÍCIOS

22) Supondo que A, B e C são variáveis do tipo inteiro, com valores iguais a  $A = 5$ ,  $B = 10$  e  $C = -8$  e uma variável real  $D = 1,5$ . Quais os resultados das expressões aritméticas a seguir?

- a)  $2 * A \ \text{mod} \ 3 - C$   
b)  $\text{sqrt}(-2 * C) \ \text{div} \ 4$   
c)  $((20 \ \text{div} \ 3) \ \text{div} \ 3) + \text{pot}(8,2)/2$   
d)  $(30 \ \text{mod} \ 4 * \text{pot}(3,3)) * -1$   
e)  $\text{pot}(-C,2) + (D*10)/A$   
f)  $\text{sqrt}(\text{pot}(A, B/A)) + C * D$

23) Escreva as expressões algébricas em forma de algoritmo. Não se preocupe nesse momento em atribuir o resultado da expressão a uma variável.

Exemplo: Expressão:  $x + vy$   
 Algoritmo:  $x + v * y$

- a)  $a + bc + d$
- b)  $(a+b) c + d (a-2b)$
- c)  $(x+y) (x-y)$

24) Verifique as expressões abaixo e diga qual o resultado das mesmas (verdadeiro ou falso).

- a)  $3 * 5 \text{ div } 4 \leq \text{pot}(3,2) / 0,5$
- b)  $2 + 8 \text{ mod } 7 \geq 3 * 6 - 15$

25) Determine os resultados obtidos na avaliação das expressões lógicas seguintes.

- a)  $X * X + Y > Z$ . Considere  $X = 1, Y = 2, Z = 5$
- b)  $X * X + Y > Z$ . Considere  $X = 4, Y = 3, Z = 1$
- c)  $(\text{NOME} = \text{"JORGE"}) \text{ E } \text{SIM} \text{ OU } (X < Z * 10)$ .  
 Considere  $X = 2, Z = 9, \text{NOME} = \text{"MARIA"}, \text{SIM} = \text{F}$

26) Determine o resultado das expressões a seguir:

- a)  $\text{NÃO } (5 \neq 10/2 \text{ || } V \ \&\& \ 2 - 5 > 5 - 2 \text{ || } V)$
- b)  $\text{pot}(2,4) \neq 4 + 2 \text{ || } 2 + 3 * 5 / 3 \text{ mod } 5 < 0$

27) Determine o resultado obtidos na avaliação das expressões lógicas apresentadas a seguir, sabendo que  $A = 2, B = 7, C = 3,5$  e que  $L$  é uma variável lógica com o valor  $F$ .

- a)  $B == A * C \ \&\& \ (L \text{ || } V)$
- b)  $B > A \text{ || } B = \text{pot}(A,A)$
- c)  $L \ \&\& \ B \text{ div } A \geq C \text{ || } \text{NÃO } A \leq C$
- d)  $\text{NÃO } L \text{ || } V \ \&\& \ \text{sqrt}(A+B) \geq C$
- e)  $B/A == C \text{ || } B/A \neq C$
- f)  $L \text{ || } \text{pot}(B,A) \leq C * 10 + A * B$

28) Faça a declaração de 3 variáveis do tipo inteiro, 2 variáveis do tipo real, 2 variáveis do tipo caracter e 1 variável do tipo lógico.

-X-

# 5

## Estrutura Sequencial

*“Tente terminar e nunca vacile;  
Nada é mais difícil, mas a pesquisa descobrirá.”*  
(Robert Herrick)

### 5.1 COMANDO DE ATRIBUIÇÃO

Um comando de atribuição é usado para atribuir valores a variáveis. O tipo do dado deve ser compatível com o tipo da variável, ou seja, só se pode atribuir um valor lógico a uma variável capaz de comportá-lo, isto é, a uma variável declarada como sendo do tipo lógico. Na linguagem algorítmica, o comando de atribuição tem o formato apresentado a seguir, onde **identificador** é o nome da variável à qual está sendo atribuído o valor, **=** é o símbolo de atribuição e **expressão** pode ser uma constante, variável, valor ou expressão cujo tipo seja compatível com o da variável à qual o valor está sendo atribuído.

**Formato:** `identificador = expressão;`

**Exemplos:**

```

caracter cor;
real media, n1, n2;
logico teste;
inteiro x, y, z;

cor = "Verde";
n1 = 10;
n2 = 4,5;
media = (n1+n2)/2;
teste = FALSO;
y = 42;
x = y;
z = 0;

```

É importante ressaltar que do lado esquerdo do símbolo de atribuição deve existir apenas um identificador.

### 5.2 COMANDOS DE ENTRADA E SAIDA

Os comandos de entrada e saída têm como finalidade a interrupção do programa a fim de que o usuário possa entrar com dados (entrada), através de unidades de entrada (normalmente o teclado), ou o programa possa fornecer informações a respeito dos dados (saída), através de unidades de saída (normalmente monitor ou impressora).

#### 5.2.1 COMANDO DE ENTRADA

O comando denominado **leia** é o comando de entrada de dados. Seu objetivo é atribuir o dado a ser fornecido à variável identificada. O formato está listado a seguir. **leia** é a palavra-chave do comando de entrada, **lista de identificadores** é o nome das variáveis, separados por vírgula, nas quais serão armazenados os valores provenientes do meio de entrada.

**Formato:** `leia (lista de identificadores);`

**Exemplos:**

```

leia (codigo);
leia (nota1, nota2);
leia (nome);
leia (endereco);

```

### 5.2.2 COMANDO DE SAÍDA

O comando denominado **escreva** é o comando de saída de dados, que tem como objetivo mostrar o conteúdo de uma variável, constante ou expressão. O formato é apresentado a seguir, onde **escreva** é a palavra-chave do comando de saída; **lista de identificadores** e/ou **expressões** são os nomes das variáveis, os nomes das constantes ou as expressões, que se deseja fornecer como resultados.

**Formato:** escreva(lista de identificadores e/ou expressões)

**Exemplos:**

```
escreva(x);
escreva("Bom dia", nome);
escreva("Você pesa ", peso, "quilos");
```

## 5.3 ESTRUTURA SEQUENCIAL

A estrutura sequencial de um algoritmo corresponde ao fato de que o conjunto de ações primitivas será executado em uma sequência linear de cima para baixo e da esquerda para a direita, ou seja, na mesma ordem em que foram escritas.

As ações serão seguidas por ponto-e-vírgula (;). O ponto-e-vírgula objetiva separar uma ação da outra e auxiliar na organização sequencial das ações, visto que, após encontrar um (;) o próximo comando da sequência deve ser executado.

**Exemplo 1:**

```
Algoritmo NomePrograma// identificação do início do algoritmo {
    // declaração de variáveis
    // corpo do algoritmo
    ação 1;
    ação 2;
    ação 3;
    .
    .
    .
    ação n;
} // fim do algoritmo
```

**Exemplo 2:** Escreva um algoritmo que calcule a média aritmética entre quatro notas quaisquer fornecidas por um aluno.

```
Algoritmo Exemplo2 {
    // declaração de variáveis
    real n1, n2, media; //variáveis que armazenarão as notas e a média
    escreva("Digite a Nota1: ");
    leia (n1);
    escreva("Digite a Nota2: ");
    leia (n2);
    // processamento, ou seja, cálculo da média aritmética das notas
    media = (n1 + n2) / 2;
    // saída de dados
    escreva("A média das notas informadas é: ", media);
}
```

## 5.4 TESTE DE MESA

A simulação ou **teste de mesa** tem por objetivo detectar, caso existam, erros de lógica na descrição do algoritmo. Partindo de dados escolhidos (com resposta conhecida) simulamos a execução do algoritmo e comparamos a resposta obtida com a resposta esperada. Se não houver coincidência está detectado um erro, embora a recíproca não seja verdadeira.

A escolha dos dados para os testes é de muita importância na simulação. As amostras de dados devem ser escolhidas de modo que provoquem a execução de todas as instruções presentes no algoritmo, testando todas as possibilidades diferentes de saída.

Na simulação são listadas todas as variáveis utilizadas no algoritmo e registrados todos os valores assumidos pelas variáveis, na ordem em que ocorrem.

É conveniente identificar com um número cada uma das instruções contidas no algoritmo. Dessa forma será possível localizar facilmente a variável que foi definida ou alterada naquele momento.

## 5.5 EXERCÍCIOS

29) Encontre os erros dos comandos de atribuição a seguir:

```
logico a;
real b, c;
inteiro d;
a = b == c;
d = b;
c + 1 = b + c;
c && b = 3,5;
b = pot(6,2)/3 <= sqrt(9) * 4;
```

30) O que será atribuído às variáveis A, X e B?

```
logico a, b;
inteiro: x;

x = 8 + 13 div 5;
b = 5 == 3;
a = b;
```

31) Dadas as declarações abaixo, assinale os comandos de atribuição inválidos:

```
inteiro num;
real soma, x;
caracter símbolo, nome, cor, dia;
logico cod, teste, tudo;

( ) símbolo = 5;           ( ) soma = num + 2 * x;
( ) teste = cor;          ( ) tudo = soma;
( ) cor = "PRETO";       ( ) x = x + 1;
( ) num = "*ABC*";       ( ) dia = "SEGUNDA";
```

32) Explique o que está acontecendo em cada linha do trecho de algoritmo abaixo e qual é o resultado de cada ação executada:

```
Algoritmo Exemplo {
  inteiro x, y;
  real z;
  escreva("Digite um valor: ");
  leia(x);
  escreva(x, "elevado ao cubo = ", pot(x,3));
  escreva("Digite outro valor: ");
  leia(y);
  escreva(x + y);
  z = x / y;
  escreva(z);
  z = z + 1;
  x = (y + x) mod 2;
  escreva(x);
}
```

33) Escreva um algoritmo que receba o valor do salário de um funcionário e o valor do salário mínimo. Calcule e escreva quantos salários mínimos ganha esse funcionário.

34) Faça um algoritmo que receba o nome e o salário de um funcionário, calcule e imprima o nome do funcionário e o valor do imposto de renda a ser pago, sabendo que o imposto equivale a 5% do salário.

35) Faça um algoritmo que receba a idade de uma pessoa em anos, calcule e imprima essa idade em: a) meses, b) dias, c) horas e d) minutos.

36) Escreva um algoritmo que receba dois números inteiros, calcule e escreva:

- a) soma dos dois números
- b) subtração do primeiro pelo segundo
- c) subtração do segundo pelo primeiro
- d) multiplicação dos dois números
- e) divisão do primeiro pelo segundo
- f) divisão do segundo pelo primeiro
- g) o primeiro elevado ao quadrado

37) Sabe-se que o quilowatt de energia custa um quinto do salário mínimo. Escreva um algoritmo que receba o valor do salário mínimo e a quantidade de quilowatts gasta por uma residência. Calcule e imprima:

- a) o valor, em reais, de cada quilowatt;
- b) o valor, em reais, a ser pago por essa residência;
- c) novo valor a ser pago por essa residência, a partir de um desconto de 15%.

38) Faça um algoritmo que receba o ano do nascimento de uma pessoa e o ano atual. Calcule e imprima: a) a idade dessa pessoa; b) essa idade convertida em semanas.

39) Escreva um algoritmo que calcule e imprima a área de um retângulo. O algoritmo deve solicitar ao usuário as informações necessárias para esse cálculo.

40) Faça um algoritmo que calcule e imprima o valor do FGTS a ser descontado de um empregado, sabendo que o desconto do FGTS é de 8% sobre o salário bruto (fornecido pelo usuário). O algoritmo deve, ainda, calcular e imprimir o valor do salário líquido. Nesse exemplo, o valor do salário líquido é representado pelo salário bruto menos o valor do FGTS.

41) Escreva um algoritmo que leia 2 valores A e B e calcule o valor de C, sabendo que  $C = (A + B) * B$ . O algoritmo deve imprimir os valores de A, B e C.

-X-

## 6

**Estruturas Condicionais**

*“Numa sociedade de lobos, é preciso aprender a uivar.”*  
(Mme. Du Barry)

**6.1 ESTRUTURA CONDICIONAL SIMPLES**

Uma estrutura de seleção permite escolher um grupo de ações a ser executado quando determinadas condições são ou não satisfeitas. As condições podem ser representadas por expressões lógicas ou relacionais.

**Formato:**

```
se (condição)
    comando1;      // ação primitiva
```

Onde (condição) é uma expressão que, quando avaliada, vai gerar um resultado lógico (verdadeiro ou falso). Na seleção simples, se (condição) for verdadeira, a ação primitiva (comando1) sob a cláusula então é executada. Caso contrário, nenhum comando é executado.

Quando existirem diversas ações a serem executadas, é necessário usar um bloco, delimitado por chaves { }, conforme mostrado a seguir.

```
se (condição)
{ // início do bloco verdade
    comando1;      // sequência de comandos
    comando2;
    .
    .
    .
    comandoN;
}
```

Se (condição) for verdadeira, os comandos comando1, comando2, ..., comandoN (bloco verdade) são executados. Caso contrário ((condição) é falsa), nenhum comando é executado, encerrando-se a seleção.

**Exemplo 1:** Escreva um algoritmo para calcular a média de quatro notas de um aluno e caso a mesma seja igual ou superior a 7, seu algoritmo deve imprimir uma mensagem sobre a aprovação do aluno.

```
Algoritmo MediaAlunos {
    real n1, n2, n3, n4, media;
    escreva ("Informe as quatro notas do aluno: ");
    leia (n1, n2, n3, n4);
    media = (n1 + n2 + n3 + n4) / 4;
    escreva ("A média do aluno é: ", media);
    se (media >= 7) {
        escreva ("Aluno aprovado");
    }
}
```

**6.2 ESTRUTURA CONDICIONAL COMPOSTA**

Quando houver duas alternativas que dependem de uma mesma condição, uma da condição ser verdadeira e a outra da condição ser falsa, usamos a estrutura de seleção composta.

**Formato:**

```

se (condição)
{
    // início do bloco verdade
    comando1;           // sequência de comandos
    comando2;
    .
    .
    .
    comando10;
} // fim do bloco verdade
senão
{
    // início do bloco falso
    comandoA;           // sequência de comandos
    comandoB;
    .
    .
    .
    comandoZ;
} // fim do bloco falso

```

As regras para utilização dos delimitadores { e } tanto dentro do então, quanto dentro senão, são as mesmas apresentadas para a estrutura de seleção simples.

**Exemplo 2:** Escreva um algoritmo que verifique a igualdade de dois números inteiros fornecidos pelo usuário.

```

Algoritmo NumerosIguais {
    inteiro num1, num2;
    escreva("Informe dois números inteiros: ");
    leia(num1, num2);
    se (num1 == num2)
    {
        escreva("Os números informados são iguais.");
    }
    senão
    {
        escreva("Os números informados são diferentes.");
    }
    escreva("Fim de Programa");
}

```

**Exemplo 3:** Escreva um algoritmo que receba como entrada três números inteiros, realize sua soma e informe se essa soma é maior ou igual a 100. Em caso contrário, informe uma mensagem que a soma é menor do que 100.

```

Algoritmo SomaMaiorCem {
    inteiro num1, num2, num3, soma;
    escreva("Informe três números inteiros: ");
    leia (num1, num2, num3);
    soma = num1 + num2 + num3;
    se (soma >= 100)
    {
        escreva("A soma dos números é maior ou igual a 100");
    }
    senão
    {
        escreva("A soma dos números é menor do que 100");
    }
    escreva("Fim");
}

```

## 6.3 ESTRUTURA CONDICIONAL ENCADEADA

Quando agrupamos várias seleções, estamos formando uma seleção encadeada.

### 6.3.1 SELEÇÃO ENCADEADA HETEROGÊNEA

Quando não conseguimos identificar um padrão lógico de construção em uma estrutura de seleção encadeada, temos uma estrutura de seleção encadeada heterogênea.

**Formato:**

```

se (condição1)
{
    se (condição2)
    {
        comando1;
        comando2;
        comando3;
    }
}
senão
{
    se (condição3)
    {
        comando4;
        comando5;
        comando6;
    }
    senão
    {
        se (condição4)
        {
            se (condição5)
            {
                comando7;
            }
            senão
            {
                comando8;
            }
        }
    }
}

```

Para resumir as variações possíveis da seleção encadeada mostrada anteriormente, temos a seguinte tabela de decisão:

Condição 1	Condição 2	Condição 3	Condição 4	Condição 5	Ação executada
V	V	-	-	-	comando1, comando2, comando3
F	-	V	-	-	comando4, comando5, comando6
F	-	F	V	V	comando7
F	-	F	F	-	comando8

### 6.3.2 SELEÇÃO ENCADEADA HOMOGÊNEA

Quando temos diversas estruturas de seleção encadeadas que seguem um determinado padrão lógico, temos uma estrutura de seleção encadeada homogênea.

Neste exemplo, após cada então existe outro **se** e não existem **senões**. Temos, então, uma estrutura de seleção encadeada homogênea.

**Formato:**

```

se (condição1)
{
    se (condição2)
    {
        se (condição3)
        {
            se (condição4)
            {
                comando1;
            }
        }
    }
}

```

O exemplo anterior é equivalente a:

```

se ((condição1) && (condição2) && (condição3) && (condição4))
{
    comando1;
}

```

**Exemplo 4:** Supondo que uma variável X possa assumir apenas quatro valores, V1, V2, V3 e V4 e que exista um comando diferente a ser executado para cada valor armazenado em X. Nesse exemplo, somente um, e apenas um comando, pode ser executado nos testes, ou seja, se X é igual a V3, ele não é igual a V1, nem a V2 e nem a V4.

```

se (x == v1)
{
    comando1;
}
se (x == v2)
{
    comando2;
}
se (x == v3)
{
    comando3;
}
se (x == v4)
{
    comando4;
}

```

X == V1	X == V2	X == V3	X == V4	Ação
V	F	F	F	comando1
F	V	F	F	comando2
F	F	V	F	comando3
F	F	F	V	comando4

Não temos aqui uma estrutura de seleção encadeada, pois as seleções não estão interligadas. Dessa maneira, todas as condições (X == Vn) estão sendo avaliadas e acontecerão testes desnecessários.

Com o objetivo de melhorar a performance do exemplo apresentado anteriormente, podemos usar um conjunto de seleções encadeadas, como mostrado a seguir.

```

se (x == v1)
{
    comando1;
}
senão
{
    se (x == v2)
    {
        comando2;
    }
    senão
    {
        se (x == v3)
        {
            comando3;
        }
        senão
        {
            se (x == v4)
            {
                comando4;
            }
        }
    }
}

```

X == V1	X == V2	X == V3	X == V4	Ação
V	-	-	-	comando1
F	V	-	-	comando2
F	F	V	-	comando3
F	F	F	V	comando4

Nesse caso, o número médio de testes a serem executados foi reduzido. Se o conteúdo de X for igual a V2, por exemplo, serão executados apenas dois testes ( $X==V1$ ) e ( $X==V2$ ) e um comando (comando2).

Na estrutura anterior, para esse exemplo de V2, são inspecionadas quatro condições, embora um único comando (comando2) seja executado.

## 6.5 EXERCÍCIOS

42) A nota final de um estudante é calculada a partir de três notas atribuídas respectivamente a um trabalho de laboratório, a uma avaliação semestral e a um exame final. A média das três notas mencionadas anteriormente obedece aos pesos a seguir:

Nota	Peso
Trabalho de laboratório	2
Avaliação semestral	3
Exame final	5

Faça um algoritmo que receba as três notas, calcule e mostre a média ponderada e o conceito segundo mostrado abaixo:

Média Ponderada	Conceito
8,0 ●----● 10,0	A
7,0 ●----○ 8,0	B
6,0 ●----○ 7,0	C
5,0 ●----○ 6,0	D

0,0 •---○ 5,0

E

43) Faça um algoritmo que receba dois números e mostre o maior.

44) Escreva um algoritmo que receba como entrada três números inteiros e verifique se a soma dos mesmos é maior ou igual a 100. O algoritmo deve emitir uma mensagem em caso positivo.

45) Escreva um algoritmo que receba o valor de um depósito e o valor da taxa de juros, calcule e mostre o valor do rendimento e o valor total depois do rendimento.

46) Escreva um algoritmo que receba uma medida em pés, faça as conversões a seguir e mostre os resultados.

- a) polegadas
- b) jardas
- c) milhas

Sabe-se que:

- 1 pé = 12 polegadas
- 1 jarda = 3 pés
- 1 milha = 1760 jardas

47) Escreva um algoritmo que receba três números obrigatoriamente em ordem crescente e um quarto número que não siga esta regra. Mostre, em seguida, os quatro números em ordem decrescente.

48) Escreva um algoritmo verifique a validade de uma senha fornecida pelo usuário. A senha válida é um conjunto de caracteres "ASDFG". O algoritmo deve imprimir uma mensagem de permissão ou de negação de acesso.

49) Escreva um algoritmo que receba o valor do salário mínimo, o número de horas trabalhadas, o número de dependentes do funcionário e a quantidade de horas extras trabalhadas. Calcule e mostre o salário do funcionário de acordo com as regras a seguir:

- o valor da hora trabalhada é igual a 1/5 do salário mínimo;
- o salário do mês é igual ao número de horas trabalhadas multiplicado pelo valor da hora trabalhada;
- para cada dependente acrescentar R\$ 32,00;
- para cada hora extra trabalhada, calcular o valor da hora trabalhada acrescida de 50%;
- o salário bruto é igual ao salário do mês mais o valor dos dependentes mais o valor das horas extras;
- calcular o valor do imposto de renda retido na fonte de acordo com a tabela a seguir:

IRRF	Salário bruto
Isento	Inferior a R\$ 200,00
10%	De R\$ 200,00 até R\$ 500,00
20%	Superior a R\$ 500,00

- o salário líquido é igual ao salário bruto menos IRRF;
- a gratificação segue a tabela a seguir:

Salário Líquido	Gratificação
Até R\$ 350,00	R\$ 100,00
Superior a R\$ 350,00	R\$ 50,00

- o salário do funcionário é igual ao salário líquido mais gratificação.

50) Escreva um algoritmo que leia um número e, caso ele seja positivo, imprima seu inverso. Caso contrário, imprima seu valor absoluto.

51) Escreva um algoritmo para resolver equações do 2º grau.  
 $ax^2 + bx + c = 0$ .

A variável  $a$  deve ser diferente de zero.

$\Delta < 0 \rightarrow$  não existe raiz real

$\Delta = 0 \rightarrow$  existe uma raiz real  $x = -b/(2*a)$

$\Delta > 0 \rightarrow$  existem duas raízes reais:  $x_1 = (-b + \sqrt{\Delta})/(2*a)$  e  $x_2 = (-b - \sqrt{\Delta})/(2*a)$

52) Escreva um algoritmo que leia 2 valores  $A$  e  $B$  e calcule o valor de  $C$ , sabendo que  $C = (A + B) * B$ . O algoritmo deve imprimir os valores de  $A$ ,  $B$  e  $C$ .

53) Deduza o que ficará armazenado nas variáveis do trecho de algoritmo a seguir, sabendo que as variáveis armazenam valores do tipo inteiro.

```
se (a > b)
{
    aux = a;
    a = b;
    b = aux;
}
se (a > c)
{
    aux = a;
    a = c;
    c = aux;
}
se (b > c)
{
    aux = b;
    b = c;
    c = aux;
}
```

54) Um supermercado deseja reajustar os preços de seus produtos usando o seguinte critério: o produto poderá ter seu preço aumentado ou diminuído. Para alterar o preço, o produto deve preencher pelo menos um dos requisitos a seguir:

Requisitos		Reajustes	
Venda Média Mensal	Preço Atual	% de acréscimo	% de desconto
< 500	< R\$ 30,00	10	-
>= 500 e < 1200	>= R\$ 30,00 e < R\$ 60,00	15	-
>= 1200	>= R\$ 80,00	-	20

Faça um algoritmo que receba o preço atual e a venda mensal média do produto, calcule e mostre o novo preço.

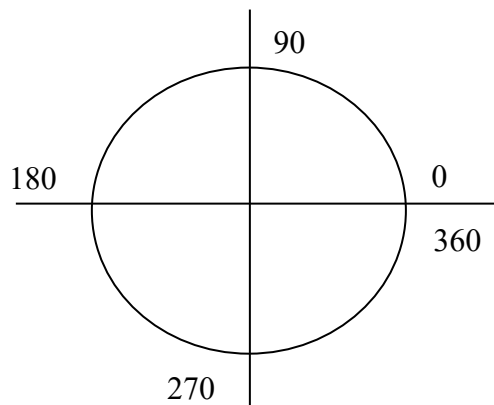
55) Escreva um algoritmo que receba a altura e o peso de uma pessoa. De acordo com a tabela a seguir, verifique e mostre qual a classificação dessa pessoa.

Altura	Peso		
	Até 60	Entre 60 e 90 (inclusive)	Acima de 90
Menores que 1,20	A	D	G
De 1,20 a 1,70	B	E	H
Maiores que 1,70	C	F	I

56) Escreva um algoritmo que receba a altura e o sexo de uma pessoa, calcule e imprima o seu peso ideal, usando as seguintes fórmulas:

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

57) Escreva um algoritmo que receba a medida de um ângulo em graus (número inteiro). Calcule e mostre o quadrante em que se localiza esse ângulo. Considere os quadrantes da trigonometria e para ângulos maiores que  $360^\circ$  ou menores que  $-360^\circ$ , reduzi-los, mostrando também o número de voltas e o sentido da volta (horário ou anti-horário).



58) O IMC – Índice de Massa Corporal é um critério da Organização Mundial de Saúde para dar uma indicação sobre a condição de peso de uma pessoa adulta. A fórmula é  $IMC = \text{peso} / (\text{altura})^2$ . Elabore um algoritmo que leia o peso e a altura de um adulto e mostre sua condição.

IMC em adultos	Condição
abaixo de 18.5	abaixo do peso
entre 18.5 e 25	peso normal
entre 25 e 30	acima do peso
acima de 30	obeso

59) Escreva um algoritmo que receba a idade de um nadador e imprima a sua categoria seguindo as regras:

Categoria	Idade
Infantil A	5 – 7 anos
Infantil B	8 – 10 anos
Juvenil A	11 – 13 anos
Juvenil B	14 – 17 anos
Sênior	maiores de 18 anos

-X-

# 7

## Seleção de Múltipla Escolha

*“Numa sociedade de lobos, é preciso aprender a uivar.”*  
(Mme. Du Barry)

### 7.1 SELEÇÃO DE MÚLTIPLA ESCOLHA

Uma estrutura de seleção de múltipla escolha é usada quando ocorrem situações onde se um comando for executado, os demais não serão, ou seja, quando um conjunto de valores discretos precisa ser testado e ações diferentes são associadas a esses valores. Observe o seguinte exemplo:

```

se (x == v1)
{
    comando1;
}
senão
{
    se (x == v2)
    {
        comando2;
    }
    senão
    {
        se (x == v3)
        {
            comando3;
        }
        senão
        {
            se (x == v4)
            {
                comando4;
            }
        }
    }
}

```

Esse exemplo pode ser escrito usando uma seleção de múltipla escolha. O uso da estrutura escolha pode ser para variáveis dos tipos inteiro, caracter e lógico.

```

escolha x
{
    caso v1: comando1;
    caso v2: comando2;
    caso v3: comando3;
    caso v4: comando4;
}

```

Para se executar um comando que possui mais de um valor em que se verifica sua necessidade, todos esses valores são agrupados em um único caso (os valores são separados por vírgula ou, caso seja um intervalo, separados por ..).

Para se executar um comando que se verifica com todos os outros valores, exceto os discriminados caso a caso, se inclui outra situação: caso contrário.

Se dentro de um caso for preciso colocar mais de um comando, é necessário usar os delimitadores { e }.

**Exemplo:**

```

se (x == v1)
{
    comando1;
}
senão
{
    se (x == v2)
    {
        comando2;
    }
    senao
    {
        se (x == v3)
        {
            comando2;
        }
        senao
        {
            se (x == v4)
            {
                comando3;
            }
            senao
            {
                se (x == v5)
                {
                    comando4;
                }
                senao
                {
                    comando5;
                }
            }
        }
    }
}

```

Reescrevendo o exemplo anterior com a estrutura de múltipla escolha:

```

escolha x
{
    caso v1: comando1;
    caso v2, v3: comando2;
    caso v4: comando3;
    caso v5: comando4;
    caso contrario: comando5;
}

```

**Exemplo 1:** Escreva um algoritmo que diga se um caracter digitado é: uma vogal maiúscula, um número inteiro de 0 a 9 ou um operador aritmético. Se o usuário informar algo diferente do que foi mencionado anteriormente, imprimir uma mensagem avisando-o desse fato.

```

Algoritmo Letras
{
    caracter opcao;
    escreva("Análise de Caracter Digitado");
    escreva("Informe um caracter: ");
    leia(opcao);
    escolha opcao
    {
        caso 'A', 'E', 'I', 'O', 'U': escreva("É uma vogal maiúscula");
    }
}

```

```

    caso '0'..'9': escreva("É um número inteiro de 0 a 9");
    caso '+', '-', '*', '/': escreva("É um operador matemático");
    caso contrario: escreva("Outro caracter");
  }
}

```

**Exemplo 2:** Construa um algoritmo que, tendo como dados de entrada o preço de um produto e seu código de origem, mostre o preço junto de sua procedência. Se o código não for nenhum dos especificados, o produto deve ser encarado como importado. Considere a tabela de códigos apresentada abaixo:

Código de origem	Procedência
1	Sul
2	Norte
3	Leste
4	Oeste
5 ou 6	Nordeste
7, 8 ou 9	Sudeste
10 até 20	Centro-oeste
25 até 30	Nordeste

Algoritmo Produto

```

{
  real preco;
  inteiro codigo;
  escreva("Informe o preço do produto: ");
  leia(preco);
  escreva("Informe o código de origem: ");
  leia(codigo);
  escolha codigo
  {
    caso 1: escreva("Preço: ", preco, " - Sul");
    caso 2: escreva("Preço: ", preco, " - Norte");
    caso 3: escreva("Preço: ", preco, " - Leste");
    caso 4: escreva("Preço: ", preco, " - Oeste");
    caso 5, 6, 25..30: escreva("Preço: ", preco, " - Nordeste");
    caso 7, 8, 9: escreva("Preço: ", preco, " - Sudeste");
    caso 10..20: escreva("Preço: ", preco, " - Centro-oeste");
    caso contrario: escreva("Preço: ", preco, " - Produto Importado");
  }
}

```

## 7.2 EXERCÍCIOS

60) Dado o algoritmo a seguir, responda:

```

Algoritmo Ex60
{
  logico A, B, C;
  se (A)
  {
    comando1;
  }
  senao
  {
    se (B)
    {
      se (C)
      {
        comando2;
      }
    }
  }
}

```

```

        comando3;
        comando4;
    }
}
comando5;
}
comando6;
}

```

- Se A = verdade, B = verdade, C = falsidade, quais comandos serão executados?
- Se A = falsidade, B = verdade, C = falsidade, quais comandos serão executados?
- Se A = falsidade, B = verdade, C = verdade, quais comandos serão executados?
- Valores de A, B, C para que somente os comandos C5 e C6 sejam executados?
- Quais são os valores de A, B, C para que somente o comando C6 seja executado?

61) Escreva um algoritmo que leia o código de um determinado produto e mostre a sua classificação de acordo com a tabela apresentada a seguir:

<b>Código</b>	<b>Classificação</b>
1	Alimento não-perecível
2, 3 ou 4	Alimento perecível
5 ou 6	Vestuário
7	Higiene pessoal
8 até 15	Limpeza e utensílios domésticos
Qualquer outro código	Inválido

-X-

## 8

**Estruturas de Repetição**

*“Quem não compreende um olhar, tampouco compreenderá uma longa explicação.”*  
(Mário Quintana)

**8.1 ESTRUTURAS DE REPETIÇÃO**

As estruturas de repetição são usadas quando queremos executar uma mesma sequência de comandos várias vezes, provocando sempre um retrocesso para o início dessa sequência. Isso ocasiona a repetição de certo trecho do algoritmo um número de vezes (que pode ser indeterminado, porém finito).

Aos trechos do algoritmo que são repetidos damos o nome de laços de repetição, *loops* ou *looping*.

**8.2 ESTRUTURA DE REPETIÇÃO COM VARIÁVEL DE CONTROLE**

A estrutura de repetição com variável de controle, conhecida como **para**, repete a execução de um bloco de comandos um número predeterminado de vezes, pois possui limites fixos.

**Sintaxe:**

```
para (i de vi ate vf passo p)
{
    comando1;
    comando2;
    comando3;
    .
    .
    .
    comandoN;
}
```

Onde: **i** é a variável de controle, **vi** é o valor inicial da variável **i**, **vf** é o valor final da variável **i** (valor até o qual ela vai chegar) e **p** é o valor do incremento dado à variável **i**.

**Exemplo 1:** Escreva um algoritmo que imprima todos os números inteiros de 1 a 50.

```
Algoritmo NumInt
{
    inteiro i;
    escreva("Imprimindo os números inteiros de 1 a 50");
    para (i de 1 ate 50 passo 1)
    {
        escreva(i, " ");
    }
}
```

**Exemplo 2:** Escreva um algoritmo que imprima todos os números inteiros de 1 até 50, em ordem decrescente.

```
Algoritmo NumIntDecrescente
{
    inteiro i;
    escreva("Números inteiros de 1 a 50 em ordem decrescente");
    para (i de 50 ate 1 passo -1)
    {
```

```

        escreva(i, " ");
    }
}

```

### 8.3 ESTRUTURA DE REPETIÇÃO COM TESTE NO INÍCIO

É uma estrutura de controle do fluxo de execução que permite repetir diversas vezes um mesmo trecho do algoritmo, sempre verificando antes de cada execução se é permitido executar o mesmo trecho.

A estrutura de repetição com teste no início denomina-se **enquanto** e permite que um bloco de comandos ou uma ação primitiva seja repetida enquanto uma determinada condição for verdadeira.

Quando o resultado da condição for falso, o comando de repetição é abandonado. Caso o resultado da condição seja falso já da primeira vez, os comandos que se encontram dentro do enquanto não são executados nenhuma vez.

#### Sintaxe:

```

enquanto (condicao)
{
    comando1;
    comando2;
    . . .
    comandoN;
}

```

**Exemplo 3:** Cálculo da média das 4 notas por aluno, considerando 50 alunos. Escrever mensagem de aprovado, caso a média seja maior ou igual a 7 e de reprovado, caso a mesma seja menor do que 7. Cuidado com dados inválidos.

```

Algoritmo Medias
{
    real n1,n2,n3,n4,media;
    inteiro cont; // declaração do contador
    cont = 0;
    enquanto (cont < 50)
    {
        escreva("Informe quatro notas: ");
        leia(n1, n2 ,n3 ,n4);
        enquanto ((n1<0) ou (n1>10) ou (n2<0) ou (n2>10) ou (n3<0) ou
        (n3>10) ou (n4<0) ou (n4>10))
        {
            escreva("Notas inválidas. Informe novamente: ");
            leia(n1,n2,n3,n4);
        }
        media = (n1 + n2 + n3 + n4) / 4;
        escreva ("Média Anual: ", media);
        se (media >= 7,0)
        {
            escreva("Aluno Aprovado");
        }
        senao
        {
            escreva("Aluno Reprovado");
        }
        cont = cont + 1;
    }
}

```

#### 8.4 ESTRUTURA DE REPETIÇÃO COM TESTE NO FINAL

A estrutura de repetição com teste no final (faça ... enquanto) consiste em uma estrutura de controle do fluxo de execução que permite repetir diversas vezes um mesmo trecho do algoritmo. Porém, diferentemente do enquanto, a condição é testada sempre no final.

Assim, o **faça ... enquanto** é usado para realizar a repetição com teste no final, permitindo que um bloco ou ação primitiva seja repetido até que uma determinada condição se torne verdadeira.

Dessa forma, é possível observar que um bloco (ou ação primitiva) é executado pelo menos uma vez, independente da validade da condição. Isto ocorre porque a inspeção da condição é feita após a execução do bloco, o que representa a característica principal dessa estrutura de repetição.

##### Sintaxe:

```
faça
{
    comando1;
    comando2;
    . . .
    comandoN;
} enquanto (condicao);
```

**Exemplo4:** Escreva um algoritmo que receba 2 notas de um aluno e calcule sua média. Para cada nota, se o usuário entrar com um valor maior do que 10 ou menor do que zero, o programa deve sempre pedir para que ele digite novamente a nota correta em um intervalo entre 0 e 10.

```
Algoritmo Notas
{
    real n1, n2, media;
    faça
    {
        escreva("Informe a primeira nota (entre 0 e 10): ");
        leia(n1);
    } enquanto (n1 < 0) ou (n1 > 10);
    faça
    {
        escreva("Informe a segunda nota (entre 0 e 10): ");
        leia(n2);
    } enquanto (n2 < 0) ou (n2 > 10);
    media = (n1 + n2) / 2;
    escreva("A média das duas notas é: ", media);
}
```

#### 8.5 EXERCÍCIOS

62) Uma loja usa os seguintes códigos para as transações de cada dia:

"d" – para compras à vista em dinheiro;

"c" – para compras à vista no cartão.

É dada uma lista de transações contendo o valor de cada compra e o respectivo código da transação. Considere que houve 25 transações no dia.

Faça um algoritmo que calcule e imprima:

- valor total das compras à vista em dinheiro;

- valor total das compras no cartão;

- valor total das compras efetuadas.

63) Escreva um algoritmo que receba a idade de 10 pessoas, calcule e imprima a quantidade de pessoas com idade maior ou igual a 18 anos.

64) Escreva um algoritmo que conte quantos números pares existem de 0 a 20. (Use a estrutura faça ... enquanto).

65) Escreva um algoritmo que calcule a tabuada de multiplicação (de 1 a 10) de um número inteiro positivo qualquer digitado pelo usuário usando faça ... enquanto. Reescreva o mesmo algoritmo usando enquanto. O algoritmo pode ser executado várias vezes até que o usuário deseje sair do mesmo.

**-X-**

# 9

## Vetores e Matrizes

*“Lembrança é quando, mesmo sem autorização, seu pensamento reapresenta um capítulo”*  
(Mário Quintana)

### 9.1 ESTRUTURA DE DADOS

Estruturas de dados são tipos construídos, compostos de tipos primitivos. É um conjunto de elementos.

### 9.2 VARIÁVEIS COMPOSTAS HOMOGÊNEAS

Variável Composta Homogênea é composta de variáveis com o mesmo tipo primitivo, formando, assim, um conjunto homogêneo de dados. Podem ser unidimensionais, também chamadas de Vetores ou bidimensionais, chamadas de matrizes.

### 9.3 VETORES - Variáveis Compostas Unidimensionais

Os vetores são variáveis compostas homogêneas unidimensionais. Eles representam uma seqüência de variáveis, todas do mesmo tipo, com o mesmo identificador (mesmo nome).

Como as variáveis têm o mesmo nome, o que as distingue é um índice que referencia sua localização dentro da estrutura. Para declarar a variável, utilizamos a seguinte sintaxe descrita abaixo, onde **i** é a quantidade de elementos do vetor e **tipo primitivo** é qualquer um dos tipos básicos.

#### Sintaxe:

```
tipo primitivo de dado indentificador[i];
```

A figura a seguir mostra como o vetor `vetorClasse` pode ser representado. Nesse exemplo, a primeira posição do vetor é 1 e a última é 40.

`vetorClasse`

8,2	6,8	9,2	3,1	...	5,2	9,6
1	2	3	4		39	40

#### 9.3.1 MANIPULAÇÃO DE VETORES

O nome do vetor é determinado através do identificador que foi usado na definição de variáveis.

A posição é determinada por meio de uma constante, de uma expressão aritmética ou de uma variável que estiver dentro dos colchetes e também é denominada índice.

`vetorClasse[4]`

1	2	3	4		39	40

Após isolar um único elemento do vetor, podemos manipulá-lo através de qualquer operação de atribuição, entrada ou saída.

**Atribuição:**

```
vetorClasse[4] = 8,6;
```

```
a[3] = 6;
```

**Leitura de dados e atribuição dos mesmos a um vetor:**

```
para (i de 1 até 15 passo 1)
{
  escreva("Digite o numero ", i, ": ");
  leia(a[i]);
}
```

**Escrita (ou impressão) dos elementos de um vetor:**

```
para (i de 1 até 8 passo 1)
{
  escreva("Elemento[", i, "]: ", b[i]);
}
```

**Exemplo 1:** Escreva um algoritmo que calcule a média aritmética geral de uma classe com 10 alunos e imprima a média e a quantidade de notas acima da média calculada.

```
Algoritmo MediaAritmetica
{
  inteiro notaAcima = 0;
  real A, B, C, D, E, F, G, H, I, J, mediaNota;
  faça
  {
    escreva("Informe 10 notas: ");
    leia(A,B,C,D,E,F,G,H,I,J);
  } enquanto ((A<0) ou (A>10) ou (B<0) ou (B>10) ou (C<0) ou (C>10) ou (D<0) ou
(D>10) ou (E<0) ou (E>10) ou (F<0) ou (F>10) ou (G<0) ou (G>10) ou (H<0) ou
(H>10) ou (I<0) ou (I>10) ou (J<0) ou (J>10));
  mediaNota = (A + B + C + D + E + F + G + H + I + J) / 10;
  se (A > mediaNota)
  {
    notaAcima = NotaAcima + 1;
  }
  se (B > mediaNota)
  {
    notaAcima = NotaAcima + 1;
  }
  se (C > mediaNota)
  {
    notaAcima = NotaAcima + 1;
  }
  se (D > mediaNota)
  {
    notaAcima = NotaAcima + 1;
  }
  se (E > mediaNota)
  {
    notaAcima = NotaAcima + 1;
  }
  se (F > mediaNota)
  {
    notaAcima = NotaAcima + 1;
  }
  se (G > mediaNota)
  {
    notaAcima = NotaAcima + 1;
  }
}
```

```

se (H > mediaNota)
{
    notaAcima = NotaAcima + 1;
}
se (I > mediaNota)
{
    notaAcima = NotaAcima + 1;
}
se (J > mediaNota)
{
    notaAcima = NotaAcima + 1;
}
escreva("A média é de: ", mediaNota);
escreva("Existem ", notaAcima, "notas acima da media!");
}

```

O algoritmo desenvolvido no exemplo anterior torna impraticável para uma grande quantidade de notas. Imagine que existissem 100 alunos na sala. Teríamos que declarar 100 variáveis diferentes, uma para cada nota.

Assim, é muito mais coerente usar uma única variável que comporte vários dados, ou seja, um vetor armazenando cada nota em uma posição diferente do mesmo.

Para acessar cada posição do vetor usaremos uma variável como índice.

Assim, reescrevendo o exercício anterior usando um vetor, teremos:

```

Algoritmo OutroExemplo
{
    real nota[10], soma, mediaNota;
    inteiro notaAcima, i;
    notaAcima = 0;
    soma = 0;
    para (i de 1 ate 10 passo 1)
    {
        faça
        {
            escreva("Informe a nota do aluno: ", i, ": ");
            leia(nota[i]);
        } enquanto ((nota[i] < 0) ou (nota[i] > 10));
        soma = soma + nota[i];
    }
    medianota = soma / 10;
    para (i de 1 ate 10 passo 1)
    {
        se (nota[i] > medianota)
        {
            notaAcima = notaAcima + 1;
        }
    }
    escreva("A média é de: ", medianota);
    escreva("Existem ", notaAcima, "notas acima da media!");
}

```

#### 9.4 MATRIZES - Variáveis Compostas Multidimensionais

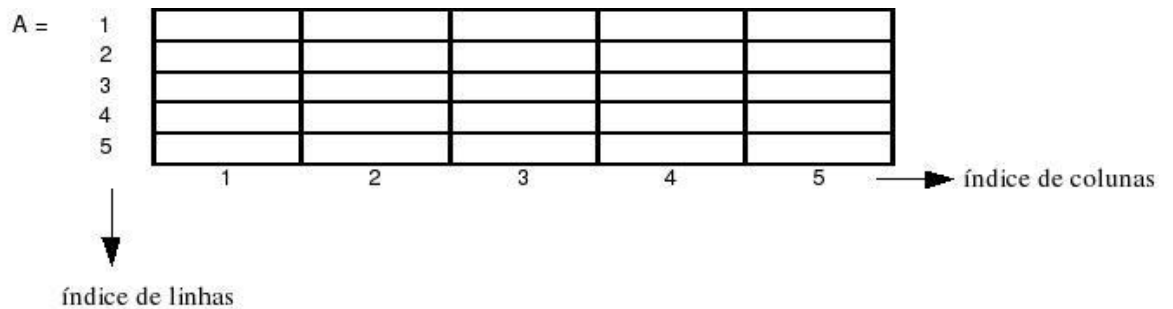
Os vetores têm como característica principal a necessidade de apenas um índice para endereçamento. São estruturas UNIDIMENSIONAIS. Uma estrutura que precisa de mais de um índice é denominada de Estrutura Composta Multidimensional (ou Matriz).

No caso da estrutura ter duas dimensões temos uma Estrutura Composta Bidimensional.

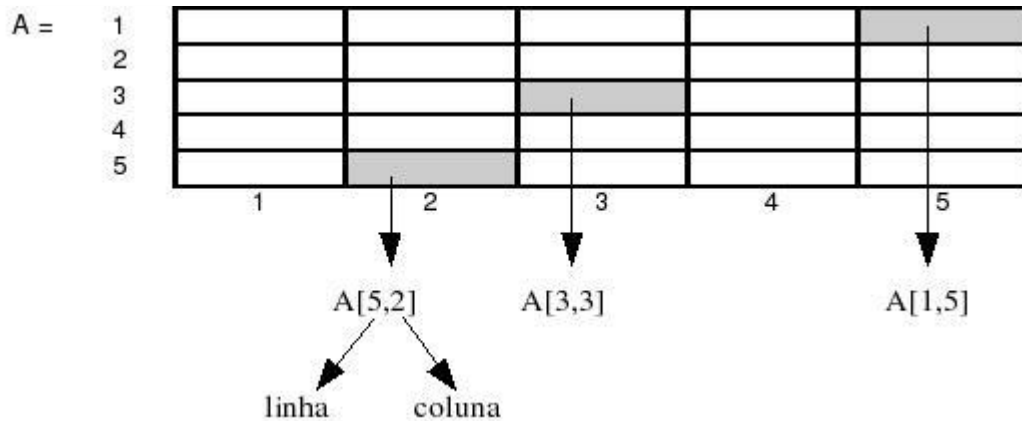
### 9.4.1 DECLARAÇÃO DE MATRIZES

tipo primitivo identificador [linha, coluna];

onde **linha** é a quantidade de linhas que a matriz terá e **coluna** a quantidade de colunas. Veja o exemplo de uma matriz A 5 x 5:



### 9.4.2 MANIPULAÇÃO DE MATRIZES



### 9.4.3 PERCORRENDO UMA MATRIZ BIDIMENSIONAL

**Exemplo 2:** Seja a matriz abaixo. Descobrir qual a linha da matriz A que possui a maior quantidade de "0"s.

A =

1	9	1	0	10	2
2	0	0	15	9	8
3	1	2	3	0	5
4	1	0	0	0	0
5	9	7	10	9	9
6	3	4	2	0	1

1 2 3 4 5

Algoritmo LinhaZerada

```
{
  inteiro mat[6,5], i, j, maiorQtde = 0, numLinha = 0, qtdeLinha;
  para (i de 1 ate 6 passo 1)
  {
    para (j de 1 ate 5 passo 1)
    {
      escreva("Informe o elemento (", i " , ", j, "): ");
      leia(mat[i,j]);
    }
  }
  para (i de 1 ate 6 passo 1)
  {
```

```

        para (j de 1 ate 5 passo 1)
            escreva(mat[i,j], " ");
        }
    }
    para (i de 1 ate 6 passo 1)
    {
        qtdeLinha = 0;
        para (j de 1 ate 5 passo 1)
        {
            se (mat[i,j] == 0)
            {
                qtdeLinha = qtdeLinha + 1;
            }
        }
        se (qtdeLinha > maiorQtde)
        {
            numLinha = i;
            maiorQtde = qtdeLinha;
        }
    }
    se (numLinha != 0)
    {
        escreva("Linha com maior qtde de zeros é: ", numLinha);
        escreva("Essa linha possui ", maiorQtde, "zeros. ");
    }
    senao
    {
        escreva("Nenhuma linha da matriz possui zeros");
    }
    escreva("Fim de Programa");
}

```

**Exemplo 3:** Seja a matriz do exemplo anterior. Descubrir qual a coluna da matriz que possui a maior quantidade de "0"s.

```

Algoritmo ColunaZerada
{
    inteiro mat[6,5], i, j, maiorQtde = 0, numColuna = 0, qtdeCol;
    para (i de 1 ate 6 passo 1)
    {
        para (j de 1 ate 5 passo 1)
        {
            escreva("Informe o elemento (", i " , ", j, "): ");
            leia(mat[i,j]);
        }
    }
    para (i de 1 ate 6 passo 1)
    {
        para (j de 1 ate 5 passo 1)
            escreva(mat[i,j], " ");
    }
    para (j de 1 ate 5 passo 1)
    {
        qtdeCol = 0;
        para (i de 1 ate 6 passo 1)
        {
            se (mat[i,j] == 0)
            {
                qtdeCol = qtdeCol + 1;
            }
        }
        se (qtdeCol > maiorQtde)
        {

```

```

        numCol = j;
        maiorQtde = qtdeCol;
    }
}
se (numCol != 0)
{
    escreva("Coluna com maior qtde de zeros é: ", numCol);
    escreva("Essa coluna possui ", maiorQtde, "zeros. ");
}
senao
{
    escreva("Nenhuma coluna da matriz possui zeros");
}
escreva("Fim de Programa");
}

```

## 9.5 EXERCÍCIOS

66) Escreva um algoritmo que preencha um vetor de 100 elementos inteiros, colocando 0 na posição correspondente a um número par e 1 na posição correspondente a um número ímpar.

67) Escreva um algoritmo que alimente os dados de dois vetores inteiros de 20 posições, efetue as respectivas operações indicadas por um outro vetor de 20 posições de caracteres também fornecido pelo usuário, contendo as quatro operações aritméticas em qualquer combinação e armazenando os resultados em um terceiro vetor.

68) Sendo o vetor V descrito abaixo e as variáveis  $X = 2$  e  $Y = 4$ , escreva o valor correspondente às solicitações

2	6	8	3	10	9	1	21	33	14
1	2	3	4	5	6	7	8	9	10

- a)  $V[X+1]$       b)  $V[X+2]$       c)  $V[X+3]$       d)  $V[X*4]$       e)  $V[X*1]$   
 f)  $V[X*2]$       g)  $V[X*3]$       h)  $V[V[X+Y]]$       i)  $V[X+Y]$       j)  $V[8-V[2]]$   
 l)  $V[V[4]]$       m)  $V[V[V[7]]]$       n)  $V[V[1] * V[4]]$       o)  $V[X+4]$

69) Construa um algoritmo que efetue a leitura, a soma e a impressão do resultado entre duas matrizes quadradas inteiras que comportem 25 elementos.

70) Dada uma matriz B de dimensão  $N \times M$  ( $N \leq 20$  e  $M \leq 20$ ), calcule  $C = K * B$ , sendo K um escalar fornecido pelo usuário. Imprima a matriz original (B) e a matriz resultante da multiplicação de B por K. Seu algoritmo deve pedir ao usuário para informar a dimensão da matriz a ser digitada, considerando as restrições para N e M definidas anteriormente.

71) Escreva um algoritmo que leia uma matriz quadrada A de dimensão  $N \times N$  ( $N \leq 20$ ) de valores inteiros, calcule e imprima a soma dos elementos da diagonal secundária. Coloque os elementos da diagonal secundária em um vetor V. Seu algoritmo deve pedir ao usuário para informar a dimensão da matriz a ser digitada, considerando a restrição para N definida anteriormente.

72) Dada uma matriz A de dimensão  $N \times M$  ( $N \leq 20$  e  $M \leq 20$ ), calcule sua transposta. Imprima a matriz original e a sua transposta. Seu algoritmo deve pedir ao usuário para informar a dimensão da matriz a ser digitada, considerando as restrições para N e M definidas anteriormente.

-X-

# 10

## Registros

*“Lembrança é quando, mesmo sem autorização,  
seu pensamento reapresenta um capítulo”*  
(Mário Quintana)

### 10.1 ESTRUTURA DE DADOS

Conforme já foi visto, as variáveis compostas homogêneas são os vetores e as matrizes, pois são constituídos de elementos do mesmo tipo de dado. Estruturas de dados formadas por elementos de vários tipos de dados, são chamadas de Variáveis Compostas Heterogêneas.

### 10.2 REGISTROS

Registro é um conjunto em que os elementos não são do mesmo tipo, ou seja, é um conjunto heterogêneo de dados.

#### Exemplo 1: Passagem de Ônibus

Número: _____	Fumante: ( ) Sim ( ) Não
De: _____	Para: _____
Data: ___/___/___	Horário: _____ Poltrona: _____
Nome do passageiro: _____	Idade: _____

A passagem de ônibus é formada pelas seguintes informações de tipos diferentes:

- Número da passagem (inteiro)
- Fumante (logico)
- Origem e Destino (caracter)
- Data (caracter)
- Horário (caracter)
- Poltrona (inteiro)
- Nome do passageiro (caracter)
- Idade (inteiro)

Essas informações são chamadas de CAMPOS. Assim, um registro é composto por campos.

#### 10.2.1 DECLARAÇÃO DE UM REGISTRO

Considerando o exemplo mencionado anteriormente:

```
registro: passagem
{
    inteiro nPass, nPol, idade;
    caracter origem, destino, data, horario, nome;
    logico fumante;
}
```

#### 10.2.2. MANIPULAÇÃO DE UM REGISTRO

```
leia (passagem.nPol);
escreva (passagem.fumante);
```

É utilizado o ponto ( . ) para separar o nome do registro do nome do campo.

### 10.2.3. REGISTRO DE CONJUNTOS

O registro mostrado anteriormente possui apenas campos de dados de tipos primitivos (inteiro, caracter, logico). Entretanto, é possível termos campos que são compostos também.

#### Exemplo 2: Estoque

Nome: _____												
Código: _____ Preço: _____												
Baixa:												
<table border="1" style="margin-left: 40px;"> <tr> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">2</td> <td style="text-align: center;">3</td> <td style="text-align: center;">4</td> <td style="text-align: center;">5</td> <td style="text-align: center;">6</td> </tr> </table>							1	2	3	4	5	6
1	2	3	4	5	6							

```
registro produto
{
    inteiro baixa[6], cod;
    caracter nome;
    real preco;
}
```

Modificando o exemplo anterior, fazendo com que o registro de estoque de um produto possa conter as baixas de 4 semanas (usando uma matriz para isso). **Exemplo 3:**

Nome: _____																																			
Código: _____ Preço: _____																																			
Baixa:																																			
<table border="1" style="margin-left: 40px;"> <tr> <td></td> <td style="text-align: center;">1</td> <td style="text-align: center;">2</td> <td style="text-align: center;">3</td> <td style="text-align: center;">4</td> <td style="text-align: center;">5</td> <td style="text-align: center;">6</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> </tr> <tr> <td style="text-align: center;">2</td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> </tr> <tr> <td style="text-align: center;">3</td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> </tr> <tr> <td style="text-align: center;">4</td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> <td style="width: 30px; height: 20px;"></td> </tr> </table>		1	2	3	4	5	6	1							2							3							4						
	1	2	3	4	5	6																													
1																																			
2																																			
3																																			
4																																			

```
registro produto
{
    inteiro baixa[4,6], cod;
    caracter nome;
    real preco;
}
```

### 10.2.4. MANIPULAÇÃO DE REGISTRO DE CONJUNTOS

- Acessa quanto foi vendido no terceiro dia da quarta semana.: `produto.baixa[4,3]`

- Imprime as baixas da segunda semana:

```
para (i de 1 ate 6 passo 1)
{
    escreva (produto.baixa[2,i]);
}
```

- Totaliza, por dia da semana, todas as baixas do mês.

```
para (j de 1 ate 6 passo 1)
{
    total = 0;
    para (i de 1 ate 4 passo 1)
    {
        total = total + produto.baixa[i,j];
    }
}
```

```

    }
    escreva("No dia ", j, " houve ", total, " baixas no mês");
}

```

### 10.2.5. CONJUNTO DE REGISTROS

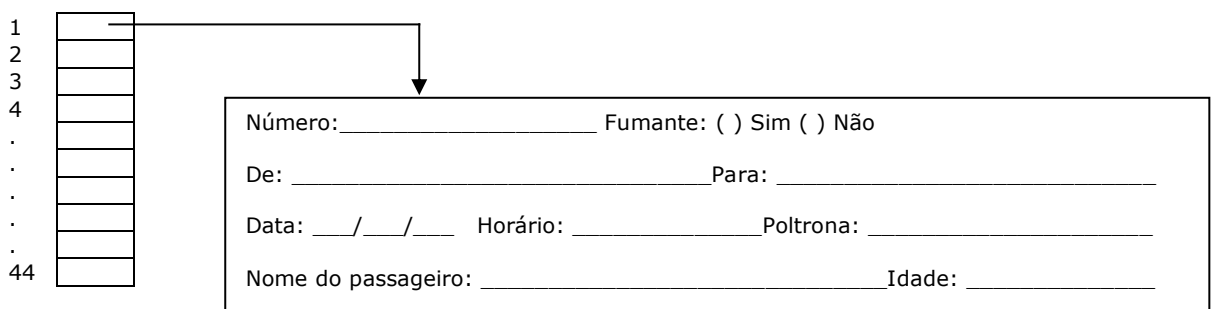
Em vetores e matrizes usamos tipos de dados primitivos sendo elementos dessas estruturas. Entretanto, podemos usar como componentes dessa estrutura não apenas um tipo primitivo, mas também os registros!

**Exemplo 4:** Ônibus de viagem com 44 lugares.

Suponha que quiséssemos manter um registro de informações relativas às passagens de todos os lugares de um ônibus.

Assim, usaríamos um registro referente a cada poltrona (Passagem) e para agrupar todos eles utilizaríamos um conjunto desses registros.

Como o ônibus tem 44 lugares, numerados sequencialmente de 1 a 44, pode-se, para uni-los, criar um vetor no qual cada posição é um elemento do tipo registro (Passagem).



Exemplo:

```

registro: passagem
{
    inteiro nPass, nPol, idade;
    caracter origem, destino, data, horario, nome;
    logico fumante;
}
passagem onibus[44];

```

**Exemplo 5:** Declarar um conjunto de registro que comporte as informações de estoque de 500 produtos. Considere as informações mostradas no registro do Exemplo 3.

```

registro produto
{
    Inteiro baixa[4,6], cod;
    caracter nome;
    real preco;
}
produto estoque[500]

```

### 10.2.6 MANIPULAÇÃO DE CONJUNTO DE REGISTROS

- Acessa a baixa do décimo produto, da terceira semana e do quarto dia da semana:  
estoque[10].baixa[3,4]

- Imprime o total de movimentação do estoque para cada um dos 500 produtos:

```

para (n de 1 até 500 passo 1)
{
    total = 0;
    para (i de 1 até 4 passo 1)
    {
        para (j de 1 até 6 passo 1)

```

```

        {
            total = total + produtos[n].baixa[i,j];
        }
    }
    escreva("Movimentação do produto ", estoque[n].nome,"foi ",total);
}

```

### 10.3 EXERCÍCIOS

73) Faça um algoritmo que realize o cadastro de contas bancárias com as seguintes informações: número da conta, nome do cliente e saldo. O banco permitirá o cadastramento de apenas 15 contas e não pode haver mais de uma conta com o mesmo número. Crie o menu de opções a seguir:

1. Cadastrar contas
2. Visualizar todas as contas de um determinado cliente
3. Excluir a conta com o menor saldo (não existem saldos iguais)
4. Sair do programa

74) Escreva um algoritmo para ler o código, o sexo (M – masculino; F – feminino) e o número de horas/aulas dadas no mês dos professores de uma escola, sabendo que um professor ganha R\$ 24,00 hora/aula e que a escola possui dez professores. Após a leitura, mostre:

- a) Uma listagem contendo o código, o salário bruto, o desconto e o salário líquido de todos os professores;
- b) A média aritmética dos salários brutos dos professores do sexo masculino;
- c) O código da professora que ganha o maior salário bruto.

Os descontos são assim calculados:

<b>Sexo</b>	<b>Até 70 ha ao mês</b>	<b>Mais que 70 ha ao mês</b>
Masculino	10%	8%
Feminino	7%	5%

-X-

# 11

## Modularização

“Algumas questões não resolvidas, atolam-se em detalhes.”  
(Stanislaw Lem)

### 11.1 DECOMPOSIÇÃO

Um problema complexo pode ser simplificado quando dividido em vários subproblemas. Para acompanhar essa abordagem, serão apresentados conceitos e técnicas que permitem a divisão de um algoritmo em módulos.

A decomposição de um problema é fator determinante para a redução da complexidade. Lembremos que Complexidade é sinônimo de Variedade, ou seja, a quantidade de situações diferentes que um problema pode apresentar. Assim, quando decomposmos um problema em subproblemas, estamos invariavelmente dividindo também a complexidade e, por consequência, simplificando a resolução. Outra grande vantagem da decomposição é que permite focalizar a atenção em um problema pequeno de cada vez, o que ao final produzirá uma melhor compreensão do todo.

Passos para orientar o processo de decomposição:

1. Dividir o problema em suas partes principais.
2. Analisar a divisão obtida para garantir coerência.
3. Se alguma parte ainda permanecer complexa, decompô-la também.
4. Analisar o resultado para garantir entendimento e coerência.

Esse processo de decomposição também é conhecido como **Refinamentos Sucessivos**, porque se parte de um problema complexo e abrangente, que é sucessivamente dividido até resultar em problemas mais simples e específicos.

À técnica de Refinamentos Sucessivos também se dá o nome de *Top-Down*, uma vez que se parte de conceitos mais abrangentes (abstratos) até atingir o nível de detalhamento desejado.

Também existe, apesar de menos difundida, uma técnica exatamente inversa, conhecida por *Bottom-Up*. Consiste em partir dos conceitos mais detalhados e ir agrupando-os sucessivamente em níveis mais abrangentes, até atingir o nível de abstração desejado.

O processo de compreensão é frequentemente mais natural quando se usa a técnica *Top-Down*. Por exemplo, é mais fácil compreender um automóvel partindo-se do todo até o último parafuso do que do parafuso até o todo. Certamente existem exceções. Por exemplo, é mais fácil entender operações aritméticas mais abstratas, como potenciação e radiciação, se antes soubermos somar e subtrair.

### 11.2 MÓDULOS

Depois de decompor um problema complexo em subproblemas, podemos construir um subalgoritmo ou módulo para cada subproblema.

Um **módulo** é um grupo de comandos, constituindo um trecho de algoritmo, com uma função bem definida e o mais independente possível em relação ao resto do algoritmo.

Assim sendo, ao se elaborar um algoritmo para calcular o salário líquido de um empregado, tem-se as seguintes etapas:

```
Algoritmo Exemplo
{
  Leia os dados do funcionário;
  Determine o salário;
```

```

    Escreva o salário;
}

```

O comando “Determine o salário” pode ser refinado assim:

```

Ref. Determine o salário
{
    Calcule as vantagens;
    Calcule as deduções;
    SalarioLiquido = vantagens - deduções;
}

```

Na elaboração do refinamento acima, não houve preocupação de como o processo de cálculo das vantagens e das deduções seria efetuado. Essas ações constituem funções bem definidas no algoritmo e que serão executadas por módulos específicos. Nesta fase do projeto do algoritmo, pode-se, então, optar pela elaboração de módulos para o cálculo das vantagens e cálculo das deduções. O mesmo refinamento ficaria então:

```

Ref. Determine o salário
{
    Ative o módulo “Cálculo das vantagens”;
    Ative o módulo “Cálculo das deduções”;
    SalarioLiquido = vantagens - deduções;
}

```

A versão final modularizada do algoritmo ficaria então constituída do algoritmo seguinte, mais os módulos:

```

Algoritmo
{
    Leia os dados do funcionário;
    Ative o módulo “Cálculo das vantagens”;
    Ative o módulo “Cálculo das deduções”;
    SalarioLiquido = vantagens - deducoes;
    Escreva SalarioLiquido;
}

Módulo Cálculo das vantagens
{
    salarioBruto = numHoras * salarioHora;
    salarioFamilia = numFilhos * valorFilho;
    vantagens = salarioBruto + salarioFamilia;
}

Módulo Cálculo das deduções
{
    inss = salarioBruto * (8 / 100);
    irpf = salarioBruto * taxa;
    deducoes = inss + irpf;
}

```

A experiência recomenda que os módulos de um programa tenham tamanho limitado. Módulos muito grandes são difíceis de ser compreendidos e, em geral, são multifuncionais. Um outro aspecto importante é a possibilidade de cada módulo poder definir as próprias estruturas de dados, suficientes e necessárias apenas para atingir o objetivo final do módulo. Todo módulo é constituído por uma sequência de comandos que operam sobre um conjunto de objetos, que podem ser globais ou locais.

**Objetos globais** são entidades que podem ser usadas em módulos internos a outros módulos do algoritmo onde foram declaradas. **Objetos locais** são entidades que só podem ser usadas no módulo do algoritmo onde foram declaradas. Estes objetos não possuem qualquer significado fora deste módulo. São exemplos de objetos globais ou locais: variáveis, arquivos, outros módulos, etc.

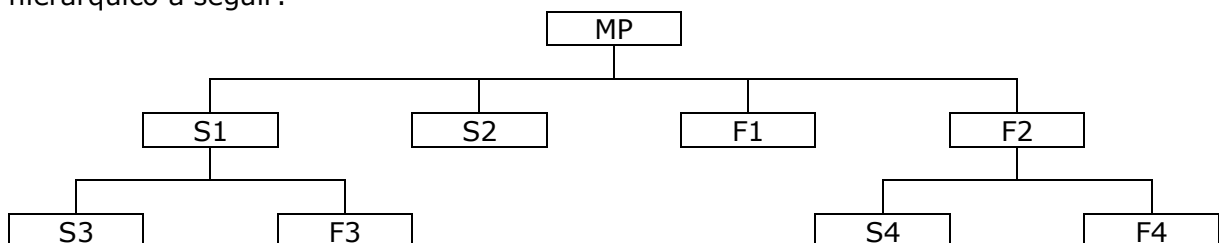
A decisão pela modularização traz benefícios tais como:

- a) a independência do módulo permite uma manutenção mais simples e evita efeitos colaterais em outros pontos do algoritmo;
- b) a elaboração do módulo pode ser feita independentemente e em época diferente do restante do algoritmo;
- c) testes e correções dos módulos podem ser feitos em separado;
- d) um módulo independente pode ser utilizado em outros algoritmos que requeiram o mesmo processamento por ele executado.

Dentre as ferramentas para modularização podemos destacar as **sub-rotinas** e as **funções**. Essas ferramentas são módulos de programação que servem basicamente a três objetivos:

- Evitar que uma certa sequência de comandos necessária em vários locais de um algoritmo tenha que ser escrita repetidamente nestes locais;
- Dividir e estruturar um algoritmo em partes fechadas e logicamente coerentes
- Aumentar a legibilidade de um algoritmo.

Sub-rotinas e funções são módulos hierarquicamente subordinados a um algoritmo, comumente chamado de módulo principal. Da mesma forma, uma sub-rotina ou uma função pode conter outras sub-rotinas e funções aninhadas, como pode ser visto no diagrama hierárquico a seguir:



Essa figura ilustra um algoritmo constituído de um módulo principal MP, as sub-rotinas S1, S2, S3 e S4 e as funções F1, F2, F3 e F4. A notação algorítmica fica da seguinte forma:

```

Algoritmo
{
  Subrotina S1
  {
    Subrotina S3
    {
      comando1;
      comando2;
    }
    Função F3
    {
      comando3;
    }
  }
  Subrotina S2
  {
    comando4;
    comando5;
  }
  Função F1
  {
    comando6;
  }
  Função F2
  {
    Subrotina S4
    {
      comando7;
      comando8;
    }
    Função F4
  }
}

```

```

    {
        comando9;
    }
}

```

Tendo em vista os conceitos de objeto local e global, a função F3, declarada dentro da sub-rotina S1 é conhecida apenas em S1. Entretanto, a sub-rotina S2, declarada dentro do módulo principal, é conhecida em todo o algoritmo e pode ser ativada dentro de S1.

A declaração de uma sub-rotina ou função é constituída de um **cabeçalho** e de um **corpo**. O cabeçalho, que identifica a sub-rotina ou função, contém o seu nome e a lista de parâmetros formais. O corpo contém declarações locais e os comandos da sub-rotina ou função. A ativação é feita através da referência a seu nome e a indicação dos parâmetros atuais.

### 11.3 SUB-ROTINA

Uma sub-rotina é declarada conforme a seguir, onde **sub-rotina** é uma palavra chave, **NOME** é o nome dado à sub-rotina e **lista de parâmetros formais** é a lista de objetos que serão substituídos por outros objetos, fornecidos quando da chamada da sub-rotina.

```

Subrotina NOME (lista-de-parâmetros-formais)
{
    Declarações dos objetos locais à sub-rotina;
    Comandos da sub-rotina;
}

```

A chamada de uma sub-rotina é feita com uma referência a seu nome e a indicação dos parâmetros reais no local do algoritmo onde a sub-rotina deve ser ativada, ou seja, onde a sua execução deve ser iniciada. A forma geral para o comando de ativação de uma sub-rotina é mostrada a seguir, onde **NOME** é o nome dado à sub-rotina, e **lista de parâmetros reais** é a lista de objetos que substituirão os parâmetros formais durante a execução da sub-rotina. Os parâmetros reais devem concordar em número, ordem e tipo com os parâmetros formais.

```

NOME (lista-de-parâmetros-reais);

```

Ao terminar a execução dos comandos de uma sub-rotina, o fluxo de controle retorna ao comando seguinte àquele que provocou a chamada. Desta maneira, a execução de uma sub-rotina se constitui de uma transferência temporária da execução do módulo chamador para a sub-rotina, retornando depois ao algoritmo que a chamou.

**Exemplo 1:** Dados três valores distintos, coloca-os em ordem crescente.

```

Algoritmo OrdemCrescente
{
    inteiro aux, n1, n2, n3;
    leia (n1, n2, n3);
    se ((n1 > n2) ou (n1 > n3))
    {
        se (n2 < n3)
        {
            aux = n1;
            n1 = n2;
            n2 = aux;
        }
        senão
        {
            aux = n1;
            n1 = n3;
            n3 = aux;
        }
    }
}

```

```

se (n2 > n3)
{
    aux = n2;
    n2 = n3;
    n3 = aux;
}
escreva (n1, n2, n3);
}

```

Observe que há repetição de um grupo de comandos que diferem entre si devido às variáveis utilizadas:

```

aux = n1;
n1 = n2;
n2 = aux;

```

```

aux = n1;
n1 = n3;
n3 = aux;

```

```

aux = n2;
n2 = n3;
n3 = aux;

```

Todas estas repetições têm por objetivo a troca de valores de 2 variáveis, podendo ser substituídas por uma única sub-rotina:

```

Algoritmo OrdemCrescente;
{
    Subrotina Troca(inteiro a, inteiro b)
    {
        inteiro aux;
        aux = a;
        a = b;
        b = aux;
    }

    inteiro n1, n2, n3;
    leia (n1, n2, n3);
    se ((n1 > n2) ou (n1 > n3))
    {
        se (n2 < n3)
        {
            troca(n1, n2);
        }
        senão
        {
            troca(n1, n3);
        }
    }
    se (n2 > n3)
    {
        troca(n2, n3);
    }
    escreva (n1, n2, n3);
}

```

Cada vez que a sub-rotina TROCA for ativada, os comandos dentro dela são executados tendo em vista os valores contidos nos parâmetros atuais e, em seguida, a sequência do algoritmo retorna ao comando imediatamente seguinte ao da ativação.

A vinculação entre módulos pode ser feita através da transferência ou passagem de parâmetros, que associam parâmetros reais com os parâmetros formais. Dentre os modos de transferência de parâmetros, pode-se destacar a passagem por valor e por referência.

### 11.3.1 PASSAGEM DE PARÂMETROS POR VALOR

A passagem de parâmetro por valor caracteriza-se pela não alteração do valor do parâmetro real quando o parâmetro formal é manipulado dentro da sub-rotina. Assim sendo, o valor passado pelo parâmetro real é copiado para o parâmetro formal, que no caso assume o papel de variável local da sub-rotina. Desta forma, qualquer modificação que ocorra na variável local da sub-rotina não afetará o valor do parâmetro real correspondente, ou seja, o processamento é executado somente dentro da sub-rotina, ficando o resultado obtido “preso” na sub-rotina. Como exemplo considere o exemplo a seguir:

#### Exemplo 2: Cálculo do fatorial de um número

```

Algoritmo Fatorial
{
  Subrotina Fatorial(inteiro n)
  {
    inteiro i, fat;
    fat = 1;
    para (i de 1 até n passo 1)
    {
      fat = fat * i;
    }
    escreva (fat);
  }

  inteiro limite;
  escreva("Qual fatorial: ");
  leia (limite);
  Fatorial(limite);
}

```

Neste exemplo, é indicado o uso da passagem de parâmetro por valor. No caso, a variável **n** é o parâmetro formal, que receberá o valor fornecido à variável **limite** por meio da sub-rotina FATORIAL. Esse valor estabelece o número de vezes que o looping deve ser executado. Dentro do procedimento é encontrado a variável **fat** que irá realizar um efeito de acumulador, tendo ao final do looping o valor da fatorial do valor informado para o parâmetro **n**. Ao término do looping, a instrução **escreva(fat)** imprime o valor da variável **fat**, que somente é válida dentro da sub-rotina e por esta razão ficará “preso” dentro da mesma. A passagem de parâmetro por valor é utilizada somente para a entrada de um determinado valor.

### 11.3.2 PASSAGEM DE PARÂMETROS POR REFERÊNCIA

Ao utilizar a passagem de parâmetros por referência, eles deverão ser indicados não só no início da sub-rotina, mas também junto do programa principal, uma vez que este tipo de parâmetro devolve para o módulo principal do programa um resultado.

#### Exemplo 3: Cálculo do fatorial de um número

```

Algoritmo Fatorial
{
  Subrotina Fatorial(inteiro n, *fat)
  {
    inteiro i;
    fat = 1;
    para (i de 1 até n passo 1)
    {
      fat = fat * i;
    }
  }

  inteiro limite, retorne;
  escreva("Qual fatorial: ");
}

```

```

    leia (limite);
    Fatorial(limite, retorne);
    escreva (retorne);
}

```

Neste exemplo, é indicado o uso da passagem de parâmetro por referência (variável **fat** por meio do símbolo \*). A variável **n** neste exemplo continua sendo do tipo passagem de parâmetro por valor, pois ela receberá o valor fornecido à variável **limite**, por meio da sub-rotina FATORIAL. Esse valor estabelece o número de vezes que o looping deve ser executado. Dentro do procedimento é encontrada a variável **fat** que é do tipo passagem de parâmetro por referência e possui no final o valor acumulado do cálculo da fatorial. Ao término do looping, o valor da variável **fat** é transferido para fora da rotina, ou seja, é transferido para a variável **retorne** do programa principal. Então, a instrução **escreva**(retorne) imprime o valor recebido de dentro da sub-rotina por meio da variável **fat**. A passagem de parâmetro por referência é utilizada para que se tenha a saída de um determinado valor de dentro de uma sub-rotina.

#### 11.4 FUNÇÃO

As funções, embora bastante semelhantes às sub-rotinas, têm a característica especial de retornar ao algoritmo que as chamou um valor associado ao nome da função. Esta característica permite uma analogia com o conceito de função matemática.

A declaração de uma função é idêntica à de uma sub-rotina, com exceção de que é necessário o seu tipo, ou seja, o tipo do valor que será retornado. Além de numéricas, as funções podem ser lógicas ou literais. No formato abaixo, **função** é uma palavra chave, **tipo** é o tipo do valor que será retornado, **NOME** é o nome dado à função e **lista de parâmetros formais** é a lista dos objetos que serão substituídos por outros objetos, fornecidos quando da chamada da função.

```

Função tipo NOME(lista-de-parâmetros-formais)
{
    Declarações dos objetos locais à função;
    Comandos da função;
}

```

A chamada de uma função é feita com uma referência a seu nome e a indicação dos parâmetros reais em uma expressão. A seguir, apresenta-se a forma geral para a ativação da função, onde **NOME** é o nome dado à função, e **lista de parâmetros reais** é a lista de objetos que substituirão os parâmetros formais durante a execução da função. Os parâmetros reais devem concordar em número, ordem e tipo com os parâmetros formais.

```

NOME(lista-de-parâmetros-reais);

```

O fluxo de controle é desviado para a função, no momento em que ela é ativada. Ao terminar a execução dos comandos da função, o fluxo de controle retorna ao comando seguinte àquele onde ela foi ativada. São válidas aqui as considerações já feitas sobre passagem de parâmetros.

**Exemplo 4:** Escrever um algoritmo que retorne o valor da soma dos números **a** e **b**.

```

Algoritmo CalculaSoma
{
    Função real Soma(real a, real b)
    {
        real total;
        total = a + b;
        retorne total;
    }

    real n1, n2;
    escreva("Informe o primeiro número: ");
    leia (n1);
}

```

```

    escreva("Informe o segundo número: ");
    leia (n2);
    escreva ("A soma dos números é: ", Soma(n1, n2));
}

```

**Exemplo 5:** Escrever um algoritmo que apresente uma mensagem informando se dois números são iguais ou diferentes

```

Algoritmo Comparação
{
    Função logica Compara(inteiro a, inteiro b)
    {
        retorne a == b;
    }

    inteiro n1, n2;
    escreva("Informe o primeiro número: ");
    leia (n1);
    escreva("Informe o segundo número: ");
    leia (n2);
    se (Compara(n1, n2))
    {
        escreva ("Números Iguais");
    }
    senão
    {
        escreva ("Números Diferentes");
    }
}

```

**Exemplo 6:** Criar uma função que efetue o cálculo, segundo o parâmetro de operação fornecido. Assim, essa função deverá receber três parâmetros, sendo os dois números mais o operador para cálculo. Se for "+", soma os números, se for "-", subtrai, se for "\*", multiplica e se o operador for "/", divide os valores.

```

Algoritmo Calculadora
{
    Subrotina Entrada
    {
        escreva("Informe o primeiro número: ");
        leia (a);
        escreva("Informe o segundo número: ");
        leia (b);
    }

    Função real Calculo(real a, real b, caracter op)
    {
        real result;
        escolha op
        {
            caso "+": result = a + b;
            caso "-": result = a - b;
            caso "*": result = a * b;
            caso "/": result = a / b;
        }
        retorne result;
    }

    Subrotina Saida
    {
        escreva("O resultado é: ",r);
    }

    Subrotina Soma

```

```

    {
        escreva("Soma");
        Entrada;
        r = Calculo(a, b, "+");
        Saida;
    }

Subrotina Subtrai
{
    escreva("Subtração");
    Entrada;
    r = Calculo(a, b, "-");
    Saida;
}

Subrotina Multiplica
{
    escreva("Multiplicação");
    Entrada;
    r = Calculo(a, b, "*");
    Saida;
}

Subrotina Divide
{
    escreva("Divisão");
    Entrada;
    r = Calculo(a, b, "/");
    Saida;
}

real r, a, b;
inteiro opcao;
opcao = 0
enquanto (opcao != 5)
{
    escreva("1 - Soma");
    escreva("2 - Subtrai");
    escreva("3 - Multiplica");
    escreva("4 - Divide");
    escreva("5 - Sai do Programa");
    leia(opcao);
    escolha opcao
    {
        caso 1: Soma;
        caso 2: Subtrai;
        caso 3: Multiplica;
        caso 4: Divide;
        caso 5: escreva("Fim de Programa");
        caso contrário: escreva ("Opção Inválida");
    }
}
}

```

## 11.5 EXERCÍCIOS

75) Faça uma função que retorne 1 se o número digitado for positivo ou 0 se o número for negativo.

76) Faça uma função que receba dois números positivos por parâmetro e retorne a soma dos **N** números inteiros existentes entre eles.

77) Faça uma função que receba três números inteiros: **a**, **b** e **c**, onde **a** é maior que 1. A função deve somar todos os inteiros entre **b** e **c** que sejam divisíveis por **a** (inclusive **b** e **c**) e retornar o resultado para a função principal.

78) Faça uma função que transforme e mostre segundos em horas, minutos e segundos. Todas as variáveis devem ser passadas como parâmetro, não havendo variáveis globais.

79) Faça uma função que receba como parâmetro um inteiro no intervalo de 1 a 9 e mostre a seguinte tabela de multiplicação (no exemplo,  $n = 9$ ):

1								
2	4							
3	6	9						
4	8	12	16					
5	10	15	20	25				
6	12	18	24	30	36			
7	14	21	28	35	42	49		
8	16	24	32	40	48	56	64	
9	18	27	36	45	54	63	72	81

80) Faça uma sub-rotina que receba as 3 notas de um aluno como parâmetros e uma letra. Se a letra for **A** o procedimento calcula a média aritmética das notas do aluno, se for **P** o procedimento calcula a média ponderada com pesos 5, 3 e 2. A média calculada deve ser devolvida ao programa principal para, então, ser mostrada.

81) Faça uma sub-rotina que receba, por parâmetro, a hora de início e a hora de término de um jogo, ambas subdivididas em dois valores distintos: horas e minutos. O procedimento deve retornar a duração expressa em minutos, considerando que o tempo máximo de duração de um jogo é de 24 horas e que o jogo pode começar em um dia e terminar no outro.

82) Faça uma sub-rotina que leia cinco valores inteiros e retorne o maior e o menor deles.

83) Faça uma função que receba, por parâmetro, uma matriz  $A[5,5]$  e retorne a soma dos seus elementos.

-X-

# Listas de Exercícios

---

## LISTA 1: LÓGICA

---

**01.** Com as informações a seguir, determine a extensão, a altura, a quantidade de operários e o mês previsto para o término da construção de cinco novos viadutos em uma grande cidade:

- a) O viaduto mais curto não tem uma altura de 160 metros.
- b) A ponte cuja construção está prevista para terminar em janeiro e na qual trabalham 2850 operários não terá uma extensão de 1900 metros, mas terá uma altura de 130 ou 160 metros.
- c) Os 2800 operários não trabalham na obra de um viaduto com 140 metros de altura.
- d) O viaduto que ficará pronto em abril é 300 metros mais extenso do que o mais alto entre os cinco viadutos em construção.
- e) A ponte de 1600 metros, na qual trabalham 2750 operários, não ficará pronta em fevereiro nem em maio.
- f) O viaduto cuja construção está prevista para terminar em março será mais alto do que aquele de 1900 metros de extensão, porém será mais baixo do que aquele que está sendo construído por 2900 trabalhadores.
- g) O viaduto em cuja construção trabalham 2700 operários não será inaugurado em fevereiro e não será o mais longo de todos.
- h) A ponte de 1600 metros não terá uma altura de 160 metros.

**02.** Cinco moradores de condomínios diferentes sofreram durante algum tempo com o barulho de uma obra realizada perto de seus locais de moradia. Descubra a quantidade de prédios de cada condomínio, a distância da obra e a duração do inconveniente barulho:

- a) A obra que tanto incomodou Alberto durou 3 meses a mais ou a menos do que a obra mais distante.
- b) A obra realizada mais rapidamente tinha uma distância 50 metros menor do que aquela realizada no condomínio em que mora Mariana e 100 metros maior do que os barulhentos trabalhos que duraram 4 meses.
- c) Beatriz tinha que aumentar constantemente o volume da televisão em função de uma obra 150 metros mais longe ou mais perto do que aquela realizada no condomínio com 10 prédios.
- d) O barulho ouvido por Lúcia, que não mora em um condomínio com 10 nem com 17 prédios, durou 2 meses a mais do que o barulho que tirou o sossego daquele(a) que reside em um condomínio de 12 edifícios e um mês a menos do que o barulho vindo de uma obra 250 metros distante de uma das residências.
- e) A obra realizada no condomínio com 14 edifícios era mais distante do que a obra suportada por Cláudio e menos distante do que aquela que durou 6 meses.



## LISTA 2: ESTRUTURA SEQUENCIAL

---

- 01.** Faça um algoritmo que escreva seu nome, endereço e seu time favorito.
- 02.** Faça um algoritmo que leia três números, calcule e escreva a multiplicação dos três.
- 03.** Faça um algoritmo que leia a idade do pai e a idade do filho, calcule e escreva a diferença de idade dos dois.
- 04.** Faça um algoritmo que dado um número inteiro de 3 algarismos, inverter a ordem de seus algarismos e mostrar o novo número formado. Considere que os três algarismos sejam diferentes de zero.
- 05.** Faça um algoritmo que leia as medidas de um retângulo (comprimento e largura), calcule e escreva sua área.
- 06.** João faz economias em dólar e deseja saber quanto vale em reais, faça um algoritmo que leia a quantidade de dólares que ele possui e o valor atual do dólar, calcule e escreva o valor convertido.
- 07.** Uma pessoa construindo sua residência resolveu colocar em sua casa uma caixa para servir como reservatório de água. Considerando que a caixa seja retangular, faça um algoritmo que leia as dimensões da caixa (comprimento, altura e largura), calcule e escreva o volume de água que pode ser armazenado.
- 08.** O critério de notas semestrais numa faculdade consiste em dois bimestres, sendo que, cada nota varia de 0 a 10 e tem os respectivos pesos 4 e 6. Elabore um algoritmo que leia as notas bimestrais, calcule e escreva a media semestral.
- 09.** Um canal de notícias internacionais, previa temperatura máxima para Brasília de 85 graus Fahrenheit. Escrever um programa que lhe permita converter esta temperatura (e qualquer outra) para graus Celsius, sabendo que a relação entre elas é  $C=5/9*(F-32)$ .
- 10.** A conta de água de uma residência é o resultado da soma da tarifa de água com a tarifa de esgoto. Faça um algoritmo que leia a tarifa de água, calcule a tarifa de esgoto (80% da tarifa de água) e escreva o valor da conta a ser paga.
- 11.** O valor do ICMS pago na venda de um produto é de 12%. Faça um algoritmo que leia o valor do produto vendido, calcule e escreva o ICMS pago.
- 12.** Maria e José resolveram abrir uma poupança conjunta, os dois têm economias guardadas, faça um algoritmo que leia o valor da economia de cada um, calcule e escreva a porcentagem de participação de cada sócio.
- 13.** Luciana distribui sua renda mensal da seguinte forma: 10% saúde, 25% educação, 30% alimentação, 10% vestuário; 5% lazer, 20% outros. Faça um algoritmo que leia a renda mensal líquida de Luciana, calcule e escreva o valor aplicado em cada item acima citado.
- 14.** Faça um algoritmo que leia quantos minutos por dia uma pessoa pode estudar Algoritmos, calcule e escreva ao final de 90 dias quanto tempo em horas ela estudou.

- 15.** Faça um algoritmo que leia a distância em centímetros entre duas Universidades, calcule e escreva a distância em KM.
- 16.** Faça um algoritmo que leia o salário mensal de um funcionário e o percentual de reajuste a ser aplicado. Calcule e escreva o valor do novo salário.
- 17.** Uma instituição de ensino realizou uma pesquisa sobre os eleitores de um município que participaram da última eleição. Faça um algoritmo que leia o total de votos brancos, nulos e válidos. Calcule e escreva o percentual que cada um representa em relação ao total de eleitores.
- 18.** Efetuar o cálculo da quantidade de litros de combustível gastos em uma viagem, sabendo-se que o carro faz 12 Km com um litro. Deverão ser fornecidos o tempo gasto na viagem e a velocidade média. Utilizar as seguintes fórmulas:  
Distância = Tempo x Velocidade  
Litros Usados = Distância / 12
- O algoritmo deverá apresentar os valores da velocidade média, tempo gasto na viagem, distância percorrida e a quantidade de litros utilizados na viagem.
- 19.** Criar um algoritmo que leia o peso de uma pessoa, só a parte inteira, calcular e imprimir:  
a) o peso da pessoa em gramas  
b) novo peso, em gramas, se a pessoa engordar 12%
- 20.** Criar um algoritmo que leia um número entre 0 e 60 e imprimir o seu sucessor, sabendo que o sucessor de 60 é 0. Não pode ser utilizado nenhum comando de seleção e nem de repetição.
- 21.** Faça um algoritmo que dado um número inteiro que representa um número binário de cinco dígitos, determinar o seu equivalente decimal.



## LISTA 3: ESTRUTURA CONDICIONAL

---

**01.** Escreva um programa que verifique a igualdade de dois números inteiros e informe ao usuário se os números são ou não iguais.

**02.** Escreva um programa que receba como entrada três números inteiros, realize sua soma, verifique se essa soma é maior ou igual a 100 e informe ao usuário o valor da soma e se a soma é maior ou igual a 100 ou se é menor do que 100.

**03.** Implemente um programa que receba como dados de entrada as quatro notas de um aluno, calcule e imprima a média aritmética das notas. O programa deve informar ao usuário as seguintes mensagens: "Aluno Aprovado" para média superior ou igual a 7,0 ou "Aluno Reprovado" para média inferior a 7,0.

**04.** Uma empresa resolve dar um aumento de 30% aos funcionários que recebem um salário inferior a R\$ 500,00. Implemente um programa que receba como dado de entrada o salário de um funcionário e imprima o valor do salário reajustado, caso o funcionário tenha direito ao aumento. Se o funcionário não tiver direito ao aumento, informe isso através de uma mensagem.

**05.** Implemente um programa que verifique a validade de uma senha fornecida pelo usuário. A senha é 1234567. O programa deve imprimir uma mensagem de permissão ou de negação de acesso.

**06.** Implemente um programa que receba dois números e imprima o menor deles.

**07.** Implemente um programa que receba a altura e o sexo de uma pessoa, calcule e imprima o seu peso ideal, usando as seguintes fórmulas:

- homens:  $(72.7 * altura) - 58$
- mulheres:  $(62.1 * altura) - 44.7$

**08.** Em uma determinada disciplina, a nota final do estudante é calculada a partir de 3 notas atribuídas a um trabalho, a um teste em laboratório e a uma prova escrita em sala de aula, respectivamente. As notas variam de 0 a 10 e a nota final é a média ponderada das 3 notas anteriormente citadas. A seguir são apresentados os pesos das notas:

- Trabalho: peso 2
- Teste em laboratório: peso 2
- Prova escrita em sala de aula: peso 6

Sabendo que foram dadas 64 aulas, implemente um programa que receba as 3 notas do estudante e o número de faltas, calcule e imprima a média final e uma mensagem de acordo com a seguinte regra:

- Se o número de faltas for superior a 25% das aulas dadas, REPROVADO POR FALTA
- Se o número de faltas for inferior ou igual a 25% das aulas dadas e a média final for igual ou superior a 7,0: ALUNO APROVADO
- Se o número de faltas for inferior ou igual a 25% das aulas dadas e a média final for inferior a 7,0 mas superior ou igual a 5,0: PROVA FINAL
- Se a média final for inferior a 5,0: ALUNO REPROVADO

**09.** Implemente um programa que receba a idade de uma pessoa e imprima mensagem de acordo com os critérios:

- Menor de 16 anos: MENOR
- Entre 16 e 18 anos: EMANCIPADO
- Maior de 18 anos: MAIOR

- 10.** Implemente um programa que receba como entrada o salário de um funcionário da empresa e imprima o salário reajustado de acordo com as seguintes regras:
- salários até R\$ 300,00, reajuste de 50%;
  - salários maiores que R\$ 300,00, reajuste de 30%.

**11.** Fazer um programa que leia três valores inteiros, determine e imprima o menor deles.

**12.** Dados três valores, X, Y, Z, verificar se eles podem ser os comprimentos dos lados de um triângulo e, se forem, verificar se é um triângulo equilátero, isósceles ou escaleno. Se eles não formarem um triângulo, escrever uma mensagem.

Propriedade: o comprimento de cada lado de um triângulo é menor que a soma dos comprimentos dos outros dois lados

Equilátero: três lados iguais

Isósceles: dois lados iguais

Escaleno: três lados diferentes

**13.** Dados três valores inteiros distintos, coloca-los em ordem crescente.

**14.** Desenvolva um programa que determine se um cliente de um banco excedeu o limite de crédito de sua conta. Os seguintes dados de cada cliente são solicitados:

- Número da conta
- Limite do Crédito
- Saldo no início do mês
- Total de depósitos
- Total de retiradas

O programa deve receber esses dados, calcular o novo saldo (= saldo inicial + total de depósitos – total de retiradas) e determinar se o novo saldo (caso negativo), supera o limite de crédito do cliente. Se superar, imprimir o número da conta do cliente, e uma mensagem indicando: CRÉDITO EXCEDIDO EM tantos reais.

**15.** Uma grande companhia química paga seus vendedores por comissão. Os vendedores recebem R\$ 200,00 por semana mais 9 por cento de suas vendas brutas naquela semana. Por exemplo, um vendedor que vender o equivalente a R\$ 500,00 em produtos em uma semana recebe R\$ 200,00 mais 9 por cento de R\$ 500,00, ou um total de R\$ 745,00. Se por acaso, as vendas ultrapassarem R\$ 1000,00, o vendedor recebe um prêmio de R\$ 800,00. Desenvolva um programa que receba as vendas brutas de um vendedor na última semana, calcule seu salário e o exiba.

**16.** Desenvolva um programa que determine o pagamento bruto de um empregado. A companhia paga o valor de uma "hora normal" pelas primeiras 40 horas trabalhadas e paga o valor de uma "hora extra" (uma vez e meia a hora normal) para cada hora trabalhada depois de completadas as primeiras 40 horas. Solicite o número de horas trabalhadas, o valor da hora normal do trabalhador e exiba o pagamento bruto do funcionário.

**17.** Um palíndromo é um número que é lido da mesma forma tanto da direita para a esquerda como da esquerda para a direita. Por exemplo, cada um dos inteiros seguintes, de cinco dígitos, é palíndromo: 12321, 55555, 45554 e 11611. Escreva um programa que leia um número de cinco dígitos e determine se ele é palíndromo ou não. (Sugestão: use os operadores divisão e resto para separar o número em seus algarismos isolados)

**18.** Uma empresa deseja transmitir dados através do telefone, mas existe a preocupação de que seus telefones possam estar grampeados. Todos os seus dados são transmitidos como inteiros de quatro dígitos. A empresa pediu a você que escrevesse um programa

para criptografar os dados de forma que eles possam ser transmitidos com mais segurança. Seu programa deve ler um inteiro de quatro dígitos e criptografá-lo da seguinte maneira: substitua cada dígito pelo resultado da expressão  $((\text{soma daquele dígito com } 7) \% 10)$  (ou seja, o resto da divisão por 10 do número obtido pela soma daquele dígito com 7). Depois, troque o primeiro dígito pelo terceiro e troque o segundo dígito pelo quarto. A seguir, imprima o inteiro criptografado.

**19.** Faça um programa que peça um número ao usuário e imprima se o número é par ou ímpar.

**20.** A prefeitura de Aparecida de Goiânia abriu uma linha de crédito para os funcionários estatutários. O valor máximo da prestação não poderá ultrapassar 30% do salário bruto. Fazer um algoritmo que permita entrar com o salário bruto e o valor da prestação e informar se o empréstimo pode ou não ser concedido.



## LISTA 4: SELEÇÃO DE MÚLTIPLA ESCOLHA

---

**01.** O prefeito do Rio de Janeiro contratou uma firma especializada para manter os níveis de poluição considerados ideais para um país do 1º mundo. As indústrias, maiores responsáveis pela poluição, foram classificadas em três grupos. Sabendo-se que a escala utilizada varia de 0,05 e que o índice de poluição aceitável é até 0,25, fazer um algoritmo que possa imprimir intimações de acordo com o índice e a tabela a seguir:

índice	indústrias que receberão intimação
0,3	1º grupo
0,4	1º e 2º grupo
0,5	1º, 2º e 3º grupos

**02.** Faça um algoritmo que leia um número que represente um determinado mês do ano. Após a leitura escreva por extenso qual o mês lido. Caso o número digitado não esteja na faixa de 1..12 escreva uma mensagem informando o usuário do erro de digitação.

**03.** Faça um algoritmo que leia um número qualquer. Caso o número seja par menor que 10, escreva "Número par menor que Dez"; caso o número digitado seja ímpar menor que 10, escreva "Número ímpar menor que Dez"; caso contrário escreva "Número fora do Intervalo".

**04.** Uma empresa irá dar aumento de salário aos seus funcionários de acordo com a categoria de cada empregado. O aumento seguirá as seguintes regras:

- Funcionários das categorias A, C, F e H receberão 10% de aumento.
- Funcionários das categorias B, D, E, I, J e T receberão 15% de aumento.
- Funcionários das categorias K e R receberão 25% de aumento.
- Funcionários das categorias L, M, N, O, P, Q, R e S receberão 35% de aumento.
- Funcionários das categorias U, V, X, Y, W e Z receberão 50% de aumento.

Faça um algoritmo que leia o nome, a categoria e o salário do funcionário e mostre o salário reajustado.

**05.** Elabore um algoritmo que calcule o que deve ser pago por um produto, considerando o preço normal de etiqueta e a escolha da condição de pagamento. Use os códigos da tabela a seguir para ler qual a condição de pagamento escolhida e efetuar o cálculo adequado.

---

Código	Condição de Pagamento
1	À vista em dinheiro ou cheque, recebe 10% de desconto
2	À vista no cartão de crédito, recebe 5% de desconto
3	Em duas vezes, preço normal de etiqueta sem juros
4	Em três vezes, preço normal de etiqueta mais juros de 10%



## LISTA 5: REVISÃO

---

**01.** O número 3025 possui a seguinte característica mostrada abaixo. Fazer um programa que leia um número inteiro de quatro dígitos e diga se tal número possui ou não tal característica

$$\begin{cases} 30 + 25 = 55 \\ 55^2 = 3025 \end{cases}$$

**02.** Numa certa loja de eletrodomésticos, o funcionário encarregado da seção de televisores recebe, mensalmente, um salário fixo mais comissão. Essa comissão é calculada em relação ao tipo e ao número de televisores vendidos por mês, obedecendo à tabela abaixo:

TIPO	Nº DE TELEVISORES VENDIDOS	COMISSÕES
a cores	maior ou igual a 10	R\$ 50,00 por televisor vendido
	menor do que 10	R\$ 5,00 por televisor vendido
preto e branco	maior ou igual a 20	R\$ 20,00 por televisor vendido
	menor do que 20	R\$ 2,00 por televisor vendido

Sabe-se, ainda, que ele tem um desconto de 8% sobre seu salário fixo para o INSS. Se o seu salário total (fixo + comissões - INSS) for maior ou igual a R\$ 500,00 ele ainda terá um desconto de 15%, sobre esse salário total, relativo ao imposto de renda retido na fonte. Faça um programa que leia o valor do salário fixo, o número de televisores a cores e o número de televisores preto e branco vendidos; calcule e escreva o seu salário líquido.

**03.** Numa fábrica trabalham homens e mulheres divididos em três classes:

- A - os que fazem até 30 peças por mês;
- B - os que fazem de 31 a 35 peças por mês;
- C - os que fazem mais de 35 peças por mês.

A classe A recebe salário-mínimo. A classe B recebe salário-mínimo e mais 3% do salário-mínimo por peça, acima das 30 iniciais. A classe C recebe salário-mínimo e mais 5% do salário-mínimo por peça acima das 30 iniciais.

Fazer um programa que leia o número de peças fabricadas no mês, e calcule e escreva o salário do operário. Considere o salário mínimo como uma constante no valor de R\$ 240,00.

**04.** Uma grande companhia química paga seus vendedores por comissão. Os vendedores recebem R\$ 200,00 por semana mais 9 por cento de suas vendas brutas naquela semana. Por exemplo, um vendedor que vender o equivalente a R\$ 500,00 em produtos em uma semana recebe R\$ 200,00 mais 9 por cento de R\$ 500,00, ou um total de R\$ 245,00. Se por acaso, as vendas ultrapassarem R\$ 1.000,00, o vendedor recebe um prêmio de R\$ 800,00. Desenvolva um programa em Pascal que receba as vendas brutas de um vendedor na última semana, calcule seu salário e o exiba.

**05.** Uma empresa deseja transmitir dados através do telefone, mas existe a preocupação de que seus telefones possam estar grampeados. Todos os seus dados são transmitidos como inteiros de quatro dígitos. A empresa pediu a você que escrevesse um programa para criptografar os dados de forma que eles possam ser transmitidos com mais segurança. Seu programa deve ler um inteiro de quatro dígitos e criptografá-lo da seguinte maneira: substitua cada dígito pelo resultado da expressão ((soma daquele dígito com 9) % 5) (ou seja, o resto da divisão por 5 do número obtido pela soma daquele dígito com 9). Depois, troque o primeiro dígito pelo quarto e troque o segundo dígito pelo terceiro. A seguir, imprima o inteiro criptografado.

**06.** Um endocrinologista deseja controlar a saúde de seus pacientes e, para isso, se utiliza do Índice de Massa Corporal (IMC). Sabendo-se que o IMC é calculado através da fórmula **IMC= peso / altura <sup>2</sup>**, criar um algoritmo que apresente o nome do paciente e sua faixa de risco, baseando-se na seguinte tabela:

IMC	FAIXA DE RISCO
abaixo de 20	abaixo do peso
a partir de 20 até 25	normal
acima de 25 até 30	excesso de peso
acima de 30 até 35	obesidade
acima de 35	obesidade mórbida

**07.** Desenvolva um programa que determine o pagamento bruto de um empregado. A companhia paga o valor de uma "hora normal" pelas primeiras 40 horas trabalhadas e paga o valor de uma "hora extra" (uma vez e meia a hora normal) para cada hora trabalhada depois de completadas as primeiras 40 horas. Solicite o número de horas trabalhadas, o valor da hora normal do trabalhador e exiba o pagamento bruto do funcionário.

**08.** Uma locadora de filmes tem a seguinte regra para aluguel de fitas:

- Segunda, Terça e Quinta (2, 3 e 5), um desconto de 40% em cima do preço normal. E Quarta, Sexta, Sábado e Domingo (4, 6, 7 e 1), preço normal.
- Aluguel de Fitas Comuns: preço normal e Aluguel de Lançamentos: 15% em cima do preço normal

Desenvolver um programa para ler o preço normal da fita, o dia da semana (1 a 7) e se ele é lançamento (1) ou não (2). Calcule e imprima o preço final a ser pago pelo cliente.

**09.** Em uma danceteria o preço da entrada sofre variações. Segunda, Terça e Quinta (2, 3 e 5), ela oferece um desconto de 25% do preço normal de entrada. Nos dias de músicas ao vivo, o preço da entrada ainda é acrescido em 15% em relação ao preço normal da entrada. Fazer um programa que leia o preço normal da entrada, o dia da semana (1 a 7) e se é dia de música ao vivo (1) ou não (2). Calcular e imprimir o preço final que deverá ser pago pela entrada.

**10.** Desenvolver um programa que determine imposto de renda cobrado de um funcionário pelo governo. Seu programa deverá ler o salário anual do funcionário e o imposto já pago. O imposto é calculado conforme a tabela abaixo:

SALÁRIO ANUAL	IMPOSTO BRUTO
mais de 12 salários mínimos	20% do salário anual
de 5 a 12 salários mínimos	8% do salário anual
menos de 5 salários mínimos	Isento

O programa calculará e imprimirá o imposto a ser pago ou devolvido, que é a diferença entre o imposto já pago e o imposto bruto. Se a diferença for negativa sair a mensagem de "imposto a restituir", caso contrário "imposto a receber". Após a mensagem, imprima o valor do imposto a pagar/restituir. Considere o salário mínimo como uma constante no seu programa igual a 240.00.

Exemplo: Salário Anual = 12000,00. Imposto Pago = 700,00  
 Bem, o salário anual é maior que 12 salários-mínimos (12\*240), portanto utilizarei a taxa de 20% do salário anual = 2400,00 (imposto bruto). Considerando que paguei 700,00, o imposto líquido será 2400 – 700 = 1700 a ser pago.

**11.** Um certo aço é classificado de acordo com o resultado dos três testes abaixo, que devem determinar se o mesmo satisfaz as seguintes especificações:

- Conteúdo de Carbono abaixo de 7
- Dureza Rockwell maior do que 50
- Resistência à tração maior do que 80.000 psi

Ao aço é atribuído o grau 10 se passar por todos os testes; 9 se passar somente nos testes 1 e 2; 8 se passar no teste 1; 7 se não passar em nenhum dos 3 testes.

Desenvolver um programa que leia o conteúdo do carbono (CC), a dureza Rockwell (DR) e a resistência à tração (RT) e forneça a classificação do aço.

**12.** Dado um número de 3 algarismos construir outro número de 4 algarismos de acordo com a seguinte regra:

- os três primeiros algarismos, contados da esquerda para a direita são iguais aos do número dado;
- o quarto algarismo é um dígito de controle calculado da seguinte forma: primeiro algarismo + segundo algarismo \* 3 + terceiro algarismo \* 5; o dígito de controle é igual ao resto da divisão dessa soma por 7.

Exemplo: Número 479 (três algarismos)

Novo Número: 974 (três algarismos de trás para frente) e o quarto será:  
 $(4+(7*3)+(9*5))\%7 = 70\%7 = 0$ . O novo número será 9740



## LISTA 6: ESTRUTURAS DE REPETIÇÃO

---

- 01.** Escreva um programa que apresente a série de Fibonacci até o décimo quinto termo. A série de Fibonacci é formada pela sequência: 1, 1, 2, 3, 5, 8, 13, 21, 34, ... etc. Esta série se caracteriza pela soma de um termo atual como seu anterior subsequente, para que seja formado o próximo valor da sequência.
- 02.** Elaborar um programa que calcule potências. O usuário deve digitar a base e o expoente, e o programa deve apresentar o resultado. Esse programa deverá ser executado até que o usuário digite 0 (zero) para a base.
- 03.** Elaborar um programa que apresente os resultados da soma e da média aritmética dos valores pares situados na faixa numérica de 50 a 70.
- 04.** Elaborar um programa que possibilite calcular a área total de uma residência (sala, cozinha, banheiro, quartos, área de serviço, quintal, garagem, etc.) O programa deve solicitar a entrada do nome, a largura e o comprimento de um determinado cômodo. Em seguida, deve apresentar a área do cômodo lido e também uma mensagem solicitando do usuário a confirmação de continuar calculando novos cômodos. Caso o usuário responda N (não), o programa deve apresentar o valor total acumulado da área residencial.
- 05.** Fazer um programa que leia um número indeterminado de dados contendo a idade de um indivíduo. O último dado, que não entrará nos cálculos, contém o valor da idade igual a zero. Calcule e escreva a idade média deste grupo de indivíduos.
- 06.** Tem-se um conjunto de dados contendo a altura e o sexo (masculino, feminino) de 10 pessoas. Fazer um programa que calcule e escreva:
- A maior e a menor altura do grupo
  - A média de altura das mulheres
  - O número de homens
- 07.** Uma certa firma fez uma pesquisa de mercado para saber se as pessoas gostaram ou não de um novo produto lançado no mercado. Para isso, forneceu o sexo do entrevistado e sua resposta (sim ou não). Sabendo-se que foram entrevistadas 10 pessoas, fazer um programa que calcule e escreva:
- O número de pessoas que responderam sim
  - O número de pessoas que responderam não
  - A porcentagem de pessoas do sexo feminino que responderam sim
  - A porcentagem de pessoas do sexo masculino que responderam não
- 08.** A conversão de graus Fahrenheit para centígrados é obtida pela seguinte fórmula:  $C = 5/9 (F - 32)$ . Fazer um programa que calcule e escreva uma tabela de centígrados em função de graus Fahrenheit, que variam de 50 a 150 de 1 em 1.
- 09.** Foi feita uma pesquisa de audiência de canal de TV em várias casas de uma certa cidade, num determinado dia. Para cada casa visitada, é fornecido o número do canal (4, 5, 7, 12) e o número de pessoas que o estavam assistindo naquela casa. Se a televisão estivesse desligada, nada era anotado, ou seja, esta casa não entrava na pesquisa. Fazer um programa que:
- Leia um número indeterminado de dados, sendo que o "FLAG" corresponde ao número do canal igual a zero
  - Calcule a porcentagem de audiência para cada emissora
  - Escreva o número do canal e a sua respectiva porcentagem

**10.** Fazer um programa que calcule e escreva o valor de S:  
 $S = 1/1 + 3/2 + 5/3 + 7/4 + \dots + 99/50$

**11.** Fazer um programa que calcule e escreva o valor de S:  
 $S = 37 \times 38/1 + 36 \times 37/2 + 35 \times 36/3 + \dots + 1 \times 2/37$

**12.** Fazer um programa que calcule e escreva o valor de S:  
 $S = 1/1 + 2/4 + 3/9 + 4/16 + 5/25 + 6/36 + \dots + 10/100$

**13.** Fazer um programa que calcule e escreva o valor de S:  
 $S = 1000/1 + 997/2 + 994/3 + 991/4 + \dots$

**14.** Fazer um programa que calcule e escreva o valor de S:  
 $S = 480/10 + 475/11 + 470/12 + 465/13 + \dots$

**15.** Escreva um programa que some uma seqüência de inteiros. Admita que o primeiro inteiro lido especifica o número de valores que ainda devem ser fornecidos. Seu programa deve ler apenas um valor cada vez que **leia** for executado. Uma seqüência típica de entrada poderia ser: 5 100 200 300 400 500, onde o **5** indica que os **5** valores subseqüentes devem ser somados.

**16.** Uma loja de venda de produtos por reembolso postal vende cinco produtos diferentes cujos preços de varejo são mostrados na tabela a seguir:

Número do Produto	Preço de Varejo
1	2,98
2	4,50
3	9,98
4	4,49
5	6,87

Escreva um programa que leia uma série de pares de números como se segue: (número do produto, quantidade vendida em um dia). Seu programa deve usar uma instrução **escolha** para ajudar a determinar o preço de varejo de cada produto. Seu programa deve calcular e mostrar o valor total de todos os produtos vendidos na semana passada.

**17.** Fazer um programa para ler a base e a altura de 10 triângulos e imprimir suas áreas

**18.** Fazer um programa para ler um valor x qualquer e calcular  
 $Y = (x+1)+(x+2)+(x+3)+(x+4)+\dots+(x+100)$

**19.** Fazer um programa para gerar os termos da seguinte Progressão Geométrica: 3, 9, 27, 81,.... Calcule e mostre o 20º termo e a soma dos termos da PG.

**20.** Dado um número indeterminado de funcionários, onde são fornecidos o Nome, Número de Dependentes e o Número de Horas Trabalhadas. Pede-se um programa que imprima, para cada funcionário, o seu Nome, Salário Bruto, Salário Líquido e o Valor Descontado. A empresa paga R\$ 12,50 por hora de trabalho e R\$ 125,55 por dependente, e ainda faz um desconto de 12% sobre o salário bruto. Pede-se ainda que seja impresso o total de funcionários da empresa e o total gasto com salários. Para encerrar a entrada de dados, considere o nome igual a "FIM"



## LISTA 7: VETORES E MATRIZES

---

### VETORES

**01.** Elabore um Programa que leia uma sequência de números, e os mostre por ordem inversa.

**02.** Elabore um Programa que dada uma sequência de números, indique qual a porcentagem que cada um representa em relação ao total.

**03.** Elabore um programa que:

- Leia 100 valores numéricos e os armazene num vetor A
- Calcule e escreva

Onde  $a_i$  é o  $i$ -ésimo valor armazenado na variável A

$$S = \sum_{i=1}^{100} \frac{i}{a_i}$$

- Calcule e escreva quantos termos da série têm o numerador inferior ao denominador

Exemplo:

A				
-48	37	99,2	...	16,3
1	2	3		100

$$\text{Somatório} = \frac{1}{-48} + \frac{2}{37} + \frac{3}{99,2} + \dots + \frac{100}{16,3}$$

**04.** Fazer um algoritmo que: a) leia uma frase de 80 caracteres, incluindo brancos; b) conte quantos brancos existem na frase; c) conte quantas vezes a letra A aparece; d) conte quantas vezes ocorre um mesmo par de letras na frase e quais são elas; e) imprima o que foi calculado nos itens b, c e d.

**05.** Dado um conjunto de 100 valores numéricos disponíveis num meio de entrada qualquer, fazer um algoritmo para armazená-los numa variável composta B, e calcular e imprimir o valor do somatório dado a seguir:

$$S = (b_1 - b_{100})^3 + (b_2 - b_{99})^3 + (b_3 - b_{98})^3 + \dots + (b_{50} - b_{51})^3$$

Exemplo:

B				
210	160	...	33	97
1	2		99	100

$$S = (210 - 97)^3 + (160 - 33)^3 + \dots$$

**06.** Fazer um algoritmo que: a) leia um conjunto de valores inteiro correspondentes a 80 notas dos alunos de uma turma, notas estas que variam de 0 a 10; b) calcule a frequência absoluta (número de vezes em que aquela nota aparece no conjunto de dados); c) imprima uma tabela contando os valores das notas (de 0 a 10) e suas respectivas frequências absoluta.

**07.** Fazer um programa que leia diversos pares de conjuntos numéricos e que imprima a identificação dos pares de conjuntos disjuntos (aqueles que não possuem elementos comuns a ambos). Os elementos de cada par de conjuntos são precedidos pelo nome que identifica o par e pelo número de elementos de cada conjuntos. Após o último par de conjuntos vem como identificação do par o literal VAZIO. O número máximo de elementos de cada conjunto é 250.

**08.** Uma grande firma deseja saber quais os três empregados mais recentes. Fazer um programa para ler um número indeterminado de informações (máximo de 300) contendo o número do empregado e o número de meses de trabalho deste empregado e imprimir os três mais recentes. Obs.: A última informação contém os dois números iguais a zero. Não existem dois empregados admitidos no mesmo mês.

**09.** Fazer um algoritmo para corrigir provas de múltipla escolha. Cada prova tem 10 questões, cada questão valendo um ponto. O primeiro conjunto de dados a ser lido será o gabarito para a correção da prova. Os outros dados serão os números dos alunos e suas respectivas respostas, e o último número, do aluno fictício, será 9999. o algoritmo deverá calcular e imprimir:

- a) para cada aluno, o seu número e sua nota;
- b) a porcentagem de aprovação, sabendo-se que a nota mínima é 6;

GABARITO	Nº	NOTA										
<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> </tr> </table>												
RESPOSTAS												
<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> <td style="width: 10%; border: 1px solid black;"> </td> </tr> </table>												

**10.** Fazer um algoritmo que: a) leia o valor de M ( $M \leq 30$ ) e os M valores de uma variável composta A; b) leia o valor de N ( $N \leq 20$ ) e os N valores de uma variável composta B; c) determine o conjunto  $C = A \cup B$  (A união com B), onde C não deverá conter elementos repetidos (A e B não contêm elementos repetidos); d) imprima os elementos contidos em A, B e C.

Fazer um algoritmo que: a) lei o número de elementos do conjunto A ( $NA \leq 100$ ) e seus respectivos elementos em ordem crescente; b) lei o número de elementos do conjunto B ( $NB \leq 100$ ) e seus respectivos elementos em ordem crescente; c) crie e imprima um conjunto C, ordenado, de tamanho  $NA + NB$ , a partir dos conjuntos originais de A e B.

**11.** Escreva um algoritmo que procure por um determinado elemento em um vetor e imprima quantas vezes ele aparece no vetor (caso ele esteja no vetor). Seu algoritmo deve, inicialmente, ler os valores e armazená-los em um vetor de 8 posições.

**12.** Escreva um algoritmo que descubra qual é o maior elemento de um vetor e o coloque na última posição do mesmo, comparando pares de elementos e permutando-os quando estiverem fora de ordem.

Exemplo com um vetor de 5 elementos:  $A = [8 \ 7 \ 3 \ 1 \ 2]$

```

→      7 8
        3 8
          1 8
            2 8
  
```

**13..** Escreva um algoritmo que faça o mesmo procedimento do algoritmo do exercício anterior para todos os elementos do vetor.

**14.** Escreva um algoritmo que inverta a ordem de um vetor A de N posições (com  $N \leq 100$ ) sem usar um vetor auxiliar. Leia os elementos a serem colocados no vetor, bem como a dimensão do mesmo.

**15.** Dados dois vetores A e B em ordem crescente (assuma que A e B serão digitados em ordem crescente), construa um vetor C resultante da intercalação destes 2 vetores de modo que C já seja gerado em ordem crescente. Considere que a dimensão máxima de C é 100. Entretanto, os vetores podem ter dimensões distintas (ex.: A ter 6 elementos e B ter 3). Peça ao usuário para informar as dimensões e os elementos dos vetores. Não é permitido colocar um vetor após o outro em C e fazer a ordenação em seguida. Observação: O primeiro vetor tem dimensão necessariamente  $\geq 1$ . O usuário pode escolher a

dimensão do primeiro vetor = 100. Neste caso, não há intercalação de vetores e o vetor resultante é o primeiro vetor. Caso contrário, há intercalação de vetores e, neste caso, o segundo vetor tem dimensão necessariamente  $\geq 1$ .

**16.** Encontre o maior elemento de um vetor A e coloque-o na 1ª posição do mesmo (Efetue uma troca com o maior elemento cuja posição deverá estar armazenada na variável POS, se necessário, isto é, se o primeiro elemento já não for o maior de todos). Dimensão máxima do vetor = 100.

**17.** Ordene o vetor A em ordem decrescente, adotando o seguinte procedimento:

- coloque na 1ª posição do vetor o maior elemento do mesmo
- coloque na 2ª posição do vetor o segundo maior elemento do mesmo;
- coloque na 3ª posição do vetor o terceiro maior elemento do mesmo etc....

**18.** Dado um vetor ordenado em ordem crescente de valores do tipo caracter (assuma que os elementos do vetor serão digitados em ordem crescente), verifique se uma CHAVE (dada) pertence ao vetor (CHAVE → elemento a ser pesquisado). Compare a CHAVE com o elemento do meio do vetor. Se a CHAVE for maior que tal elemento, restrinja a busca à segunda metade do vetor. Em caso contrário, pesquise na primeira metade. Repita o processo até que a CHAVE seja encontrada. Imprima a posição do vetor onde a CHAVE ocorre ou uma mensagem, caso você conclua que a pesquisa não foi bem sucedida.  
Exemplo: Seja VET o vetor abaixo e CHAVE = G.

VET

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

CHAVE > VET[MEIO] → procurar na segunda metade

E	F	G	H
---	---	---	---

CHAVE > VET[MEIO] → procurar na segunda metade

G	H
---	---

Tal método chama-se Pesquisa Binária.

Neste caso, a pesquisa foi concluída com sucesso. O elemento procurado se encontra na posição POS = 7.

## MATRIZES

**19.** Elabore um Programa que:

- Leia uma matriz inteira A de M x N, onde os elementos de cada linha e os valores de M e N são fornecidos (  $M \leq 20$ ,  $N \leq 10$ );
- Imprima a matriz lida;
- Calcule e imprima uma matriz modificada B (M x N+1), sendo que os elementos da (N+1)-ésima coluna são formados com os produtos dos elementos da mesma linha.

EXEMPLO:

A

2	3
4	5

B

2	3	6
4	5	20

**20.** Elabore um Programa que leia e imprima uma matriz cujo conteúdo é a população dos 10 municípios mais populosos de cada um dos 26 estados brasileiros. Determinar e imprimir o número do município mais populoso e o número do estado a que pertence. Considerando que a primeira coluna contém sempre a população da capital do estado, calcular a média da população das capitais dos 26 estados.

	1	2	...	10
1				
2				
.				
.				
.				
26				

POPULAÇÃO[i,j]

População do j-ésimo município do i-ésimo estado.

**21.** A composição dos custos das diversas atividades de construção de um prédio é feita a partir da elaboração de um quadro de quantitativos dos diversos recursos envolvidos em cada atividade. Estes recursos são de vários tipos e envolvem principalmente os custos mais diretos, como, por exemplo, matérias-primas, mão-de-obra, hora de equipamento etc.

Sendo conhecidos os custos unitários para cada recurso envolvido, chega-se facilmente ao custo final unitário de cada atividade. A este custo são acrescentados os percentuais de custos indiretos (administrativos), impostos, depreciação de equipamentos, leis sociais etc., totalizando o preço final para a execução de cada fase.

Este procedimento básico é adotado em várias empreiteiras de obras e o objetivo deste trabalho é fazer um algoritmo que execute estes cálculos para auxiliar o analista de custos de uma empreiteira.

Supondo-se que na execução do prédio são realizados quatro tipos de atividades e que cada uma consome os recursos especificados na tabela dada a seguir e que as despesas indiretas (administração) são levantadas a cada mês, fazer um programa que:

- Leia o percentual de administração do mês;
- Leia os custos unitários dos sete recursos envolvidos;
- Leia um conjunto de 4 atividades contendo os quantitativos de recursos envolvidos em cada atividade;
- Calcule e imprima:
  - i. O preço unitário de custo (direto + administração) de cada atividade;
  - ii. O preço unitário que a empreiteira deve cobrar em cada atividade, para que tenha 36% de lucro;
  - iii. Considerando o percentual de 16% para leis sociais, incidentes sobre a mão-de-obra, quanto deve ser recolhido para cada unidade de atividade;
  - iv. Considerando o percentual de administração fornecido + 36% de lucro + 16% de leis sociais, qual será o preço a ser cobrado pela empreiteira para a construção de uma obra que envolva as seguintes atividades: 50 m<sup>3</sup> de fundação; 132 m<sup>2</sup> de alvenaria, 200 m<sup>3</sup> de estrutura e 339 m<sup>2</sup> de acabamento;
  - v. Para a mesma obra acima, qual será a quantidade total de cada recurso envolvido?

Recurso Atividade	Cimento (Kg)	Areia (m <sup>3</sup> )	Brita (m <sup>3</sup> )	Pe- dreiro (h)	Ser- vente (h)	Tijolo (un)	Beto- neira (h)
Fundação m <sup>3</sup>	50	0,4	0,6	5	3	0	3
Alvenaria m <sup>2</sup>	20	0,3	0	2	1	100	1
Estrutura m <sup>3</sup>	70	0,3	0,7	6	3	0	35
Acabamento m <sup>2</sup>	40	0,2	0	9	5	0	1

**22.** Escreva um algoritmo que leia uma matriz quadrada A de dimensão N x N (N ≤ 20) de valores inteiros, calcule e imprima a soma dos elementos da diagonal secundária. Coloque os elementos da diagonal secundária em um vetor V. Seu algoritmo deve pedir ao usuário para informar a dimensão da matriz a ser digitada, considerando a restrição para N definida anteriormente.

**23.** Dada uma matriz A de dimensão N x M (N ≤ 20 e M ≤ 20), calcule sua transposta. Imprima a matriz original e a sua transposta. Seu algoritmo deve pedir ao usuário para

informar a dimensão da matriz a ser digitada, considerando as restrições para N e M definidas anteriormente.

**24.** Fazer um programa que carregue o relacionado abaixo. O programa deve mostrar para cada produto, a loja que tem o menor preço

- Um vetor com oito posições com os códigos das lojas;
- Um outro vetor com quatro posições com os códigos dos produtos;
- Uma matriz com os preços de todos os produtos em cada loja;

**25.** Faça um programa que carregue uma matriz 3 x 4, calcule e mostre:

- A quantidade de elementos pares;
- A soma dos elementos ímpares;
- A média de todos elementos.

**26.** Faça um programa que:

- Receba o preço de 10 produtos e armazene-os em um vetor;
- Receba a quantidade estocada de cada um desses produtos em cinco armazéns diferentes, utilizando uma matriz 5 x 10.

Calcule e mostre:

- a) A quantidade de produtos estocados em cada um dos armazéns;
- b) A quantidade de cada um dos produtos estocados em todos os armazéns juntos;
- c) O preço do produto que possui maior estoque em um único armazém;
- d) O menor estoque armazenado;
- e) O custo de cada armazém.

**27.** Faça um programa que carregue uma matriz 4 x 5, calcule e mostre um vetor com cinco posições, onde cada posição contém a soma dos elementos de cada coluna da matriz. Mostre apenas os elementos do vetor maiores que 10. se não existir nenhum elemento maior que 10 mostre uma mensagem.

**28.** Faça um programa que receba a idade de 8 alunos e armazene-as em um vetor, em um outro vetor armazene o código de 5 disciplinas e em uma matriz armazene a quantidade de provas que cada aluno fez em cada disciplina. Calcule e mostre:

- a) A quantidade de alunos com idade entre 18 e 25 anos e que fizeram mais de duas provas em uma disciplina com código digitado pelo usuário. O usuário pode digitar um código que não está cadastrado; nesse caso, mostrar mensagem.
- b) Uma listagem com o número do aluno e o código da disciplina dos alunos que fizeram menos de três provas. Analisar cada disciplina.
- c) A média de idade dos alunos que não fizeram nenhuma prova em alguma disciplina. Cuidado para não contar duas vezes o mesmo aluno.

**29.** Escreva um algoritmo que receba uma matriz de números inteiros NxN, com N no máximo 100, e mostre uma mensagem dizendo se a matriz digitada é *simétrica*. Uma matriz simétrica possui  $A[i,j] = A[j,i]$ .

**30.** Escreva um algoritmo que receba uma matriz de números inteiros NxN, com N no máximo 100, e verifique se essa matriz forma o chamado *quadrado mágico*. Um quadrado mágico é formado quando a soma dos elementos de cada linha é igual à soma dos elementos de cada coluna e igual à soma dos elementos da diagonal principal e igual, também, à soma dos elementos da diagonal secundária.



## LISTA 8: REGISTROS

---

**01.** Considere as seguintes definições em seu algoritmo. Usando tais definições, escreva um algoritmo que imprima o produto de maior saída.

```
Registro regProduto
{
    inteiro baixa[4,6], cod;
    caracter nome;
    real preco;
}
regProduto produto[500];
```

**02.** Considerando as definições apresentadas para o exercício anterior, elabore um algoritmo que imprima um relatório com a semana de maior saída de cada produto.

**03.** Uma determinada biblioteca possui obras de ciências exatas, humanas e biológicas, totalizando 300 volumes, 100 de cada área. O proprietário resolveu informatizá-la e, para tal, agrupou as informações sobre cada livro do seguinte modo:

Código de Catalogação: \_\_\_\_\_ Doado: \_\_\_\_\_  
Nome da Obra: \_\_\_\_\_  
Nome do Autor: \_\_\_\_\_  
Editora: \_\_\_\_\_ Número de páginas: \_\_\_\_\_

a) Construa um algoritmo que declare tal estrutura e reúna todas as informações de todas as obras em 3 vetores distintos para cada área. Faça com que o código de catalogação seja gerado automaticamente (iniciando pelo código = 1). Cuidado com dados inválidos para doação e para número de páginas.

b) Elabore um trecho de algoritmo que, usando como premissa o que foi feito no item a), realize uma consulta às informações. O usuário fornecerá o código da obra e sua área. Existindo tal livro, o algoritmo informa os dados da obra. Caso contrário, ele imprime uma mensagem informando a não-existência ou o não-cadastramento do livro naquela área. A consulta repete-se até que o usuário forneça o código finalizador com valor -1.

**04.** Utilizando o exercício anterior sobre biblioteca, escreva um programa que liste todas as obras de cada área que representem livros doados.

**05.** Faça um programa que realize o cadastro de contas bancárias com as seguintes informações: número da conta, nome do cliente e saldo. O banco permitirá o cadastramento de apenas 15 contas e não pode haver mais de uma conta com o mesmo número. Crie o menu de opções a seguir:

Menu de opções:

1. Cadastrar contas
2. Visualizar todas as contas de um determinado cliente
3. Excluir a conta com menor saldo (supondo a não existência de saldos iguais)
4. Sair



## LISTA 9: MODULARIZAÇÃO

---

**01.** A avaliação de aproveitamento de uma certa disciplina é feita através de 4 provas mensais no valor de 20 pontos e uma prova final no valor de 40 pontos. A nota final é obtida somando-se as 3 melhores notas, dentre as provas mensais, com a nota da prova final. O conceito final é dado atendendo-se ao seguinte critério:

Faixa	Conceito
90 a 100	A
80 a 89	B
70 a 79	C
60 a 69	D
40 a 59	E
0 a 39	F

Fazer uma sub-rotina que, recebendo como parâmetro 4 (quatro) números inteiros, devolva ao módulo que a chamou a soma dos 3 maiores números dentre os 4 números recebidos.

Fazer um algoritmo que:

- Leia um conjunto de 80 registros contendo, cada um, o número do aluno, as 4 notas mensais e a nota da prova final
- Calcule, para cada aluno, sua nota final, utilizando a sub-rotina anterior
- Verifique o conceito obtido por cada aluno
- Escreva, para cada aluno, o seu número, sua nota final e seu conceito.

**02.** Construir uma função que receba como parâmetro de entrada um número inteiro positivo e devolva um dígito verificador conforme o processo de cálculo descrito a seguir. Utilize como exemplo o número 811057, que é usado como 8110573:

- Cada algarismo do número é multiplicado por um peso começando de 2 e crescendo de 1, da direita para a esquerda:  $8 \times 7$ ,  $1 \times 6$ ,  $1 \times 5$ ,  $0 \times 4$ ,  $5 \times 3$ ,  $7 \times 2$ ;
- Somam-se as parcelas obtidas:  $56 + 6 + 5 + 0 + 15 + 14 = 96$ ;
- Calcula-se o resto da divisão desta soma por 11:  $96 / 11 = 8$ , e sobra resto 8
- Subtrai-se de 11 o resto obtido:  $11 - 8 = 3$
- Se o valor encontrado for 10 ou 11, o dígito verificador será 0; nos outros casos, o dígito verificador é o próprio valor encontrado.

Escrever um algoritmo capaz de:

- a) ler um conjunto de 100 registros contendo, cada um, o nome de uma pessoa e seu número de CPF
- b) imprimir, para cada pessoa, os seus dados de entrada mais a mensagem "válido" ou "inválido", conforme a situação do número de CPF.
- c) utilize a função acima para calcular os dígitos verificadores.

OBS.: um nº de CPF é validado através de seus dois últimos dígitos (dígitos verificadores, denominados *controle*). Por exemplo, o CPF 230.860.256-20 é validado pelos dígitos verificadores 20. o esquema de verificação é o seguinte:

230860256 (função) dígito verificador igual a 2

2308602562 (função) dígito verificador igual a 0

**03.** Fazer uma sub-rotina, cujo cabeçalho é dado por: QUANTOSDIAS(dia,mês,ano,n), onde **dia**, **mês** e **ano** são parâmetros de entrada; **n** é um parâmetro de saída que conterá o número de dias do ano até a data fornecida.

Para calcular quantos dias tem o ano até uma data fornecida, é preciso verificar o número de dias de cada mês. O mês de fevereiro pode ter 28 ou 29 dias, dependendo de o ano ser

bissexto ou não. Um ano é bissexto se for divisível por 400, ou se for divisível por 4 e não o for por 100. Fazer um algoritmo que:

a) leia um conjunto de registro contendo, cada um, duas datas. O último registro, que será utilizado como *flag*, conterà os valores 0,0,0,0,0,0;

b) verifique se as datas estão corretas (mês entre 1 e 12, dia de acordo com o mês e se ambas estão dentro do mesmo ano). Se alguma das datas não estiver correta, escreva "data inválida";

c) verifique, se as datas estiverem corretas, qual a diferença, em dias, entre essas duas datas

**04.** Fazer uma função que converta graus para radianos.

**05.** Criar uma função que possa calcular a raiz de um número em qualquer índice. Esta é uma função importante, pois a maioria das linguagens só oferece a função raiz quadrada. Lembrando que  $A^n = \exp(n * \log(A))$

**06.** Criar uma função que verifique se um número é primo

**07.** Foi realizada uma pesquisa de algumas características físicas de cinco habitantes de uma certa região. De cada habitante foram coletados os seguintes dados: sexo, cor dos olhos (A – azuis ou C – castanhos), cor dos cabelos (L – louros, P – pretos ou C – castanhos) e idade.

a) Faça uma função que leia esses dados em um vetor. Determine, por meio de outra função, a média de idade das pessoas com olhos castanhos e cabelos pretos. Mostre esse resultado no programa principal.

b) Faça uma função que determine e devolva ao programa principal a maior idade entre os habitantes.

c) Faça uma função que determine e devolva ao programa principal a quantidade de indivíduos do sexo feminino cuja idade está entre 18 e 35 (inclusive) e que tenham olhos azuis e cabelos louros

**08.** Criar um algoritmo que funcione de acordo com o menu a seguir, sabendo-se que os vetores têm dimensão 5. Os itens 1 a 5 são funções. Lembre-se de exibir uma mensagem se a opção 3, 4 ou 5 for digitada antes da 1 e 2, pois são dependentes:

- 1 – Dados do vetor A
- 2 – Dados do vetor B
- 3 – Imprime vetores
- 4 – Soma vetores
- 5 – Subtrai vetores
- 6 – Sai do programa



## Bibliografia

*"Só sei que nada sei"* (Sócrates)

Esta apostila foi totalmente baseada nos livros:

FORBELLONE, A. L. V. e EBERSPACHER, H. F. *Lógica de Programação – A Construção de Algoritmos e Estrutura de Dados*. 2ª Edição. Makron Books. 2000

FARRER, Harry e outros. *Algoritmos Estruturados*. 3ª Edição. LTC. 1999.

