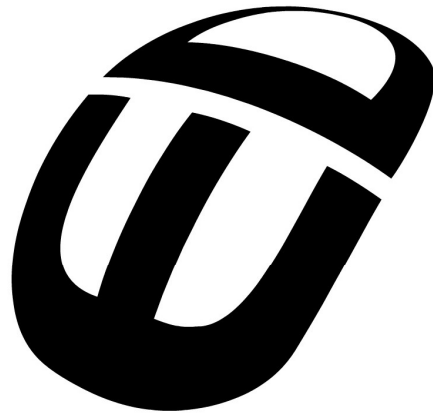


PROJETO E-JOVEM



EDUCANDUS

APOSTILA ALGORITMOS

ÍNDICE

1.	ALGORITMO NO COTIDIANO:	4
2.	MELHORIAS NO ALGORITMO:	5
3.	ALGORITMO ESTRUTURADO:	6
4.	LINGUAGEM DE PROGRAMAÇÃO:.....	8
5.	ITENS FUNDAMENTAIS:.....	10
6.	EXPRESSÕES ARITMÉTICAS	14
7.	EXPRESSÕES LITERAIS.....	17
8.	ESTRUTURA SEQÜENCIAL E CONDICIONAL SIMPLES.....	20
9.	ANINHAMENTO DE SE – ESTRUTURA CONDICIONAL SIMPLES II.....	21
10.	ESTRUTURA CONDICIONAL COMPOSTA I	22
11.	ESTRUTURA CONDICIONAL COMPOSTA II.....	22
12.	ESTRUTURA DE REPETIÇÃO DETERMINADA.....	23
13.	ESTRUTURA DE REPETIÇÃO INDETERMINADA	25
14.	COMBINANDO ESTRUTURA DE REPETIÇÃO E ESTRUTURA CONDICIONAL	28
15.	VARIÁVEIS COMPOSTAS HOMOGÊNEAS.....	28
16.	VARIÁVEIS COMPOSTA HETEROGÊNEAS - REGISTROS.....	30
17.	ARQUIVOS	31
18.	SUBROTINA	34
19.	FUNÇÃO:	35
20.	EXERCÍCIOS RESOLVIDOS	37
21.	FIM ALGORITMO	47
22.	RESPOSTA.....	59
23.	EXERCÍCIOS PROPOSTOS	73

1. Algoritmo no Cotidiano:

No nosso dia a dia é comum termos que realizar ações para alcançarmos determinados resultados, às vezes realizamos essas ações de forma coordenada ou de forma não ordenada, com isso surge a questão se sabermos diferenciar um fato imprevisível de uma ação.

Definição:

- **Ação:** é um evento que tem um estado inicial, um período de tempo finito e que produz um resultado esperado.
- **Fato imprevisível:** é uma situação que acontece sem que haja um tempo e também um resultado definido.

Para que seja possível a execução de uma ação de forma eficiente, os passos devem ser descritos de forma clara. Um exemplo para um melhor entendimento seria a contagem de de letras de uma palavra qualquer, vamos aos passos:

- Escolher a palavra;
- Contar quantas letras ela possui
- Escrever o resultado.

Sendo assim por definição temos que o conjunto desses passos necessários para se realizar uma ação é denominado **Algoritmo**.

Algoritmo: é uma seqüência de instruções ordenadas e que, se corretamente seguida, produz um resultado previsível.

Algumas ações do nosso dia a dia como: fazer um bolo, tomar banho, trocar uma lâmpada seguem procedimentos padrões para chegar a um objetivo. Podemos assim considerá-los como algoritmos do nosso cotidiano.

2. Melhorias no Algoritmo:

Quando somos encarregados de realizar uma determinada ação sempre tentamos melhorar a forma de execução dessa ação, buscando talvez a realização em um tempo menor ou melhorar a qualidade naquilo que se busca. O mesmo se aplica aos algoritmos, nunca devemos interpretá-los como definitivos, sempre poderão ser melhorados.

Você deve estar se perguntando por que melhorar um algoritmo? Nem sempre os algoritmos estão descritos de forma que possam ser compreendidos ou estão gerando resultados não esperados, quando isso acontecer eles devem ser refeitos ou detalhados em mais instruções.

Analise o seguinte algoritmo de receita de bolo:

- **Adicione os seguintes ingredientes:**
 - Ovos;
 - Farinha de Trigo;
 - Leite;
 - Açúcar;
 - Fermento em pó;
- **Misture**
- **Leve ao Forno**

Você acha que com essas informações teríamos um bolo ao final? A resposta é não, pois não vou poder fazer um bolo apenas colocando os ingredientes, é preciso definir as medidas dos ingredientes para chegarmos corretamente ao bolo, o nosso algoritmo ficaria melhor da seguinte forma:

- **Adicione os seguintes ingredientes:**
 - 4- Ovos;
 - 2 copos e meio de farinha de trigo;
 - 1 copo de leite;
 - 2 copos e meio de açúcar;
 - 1 colher de fermento em pó;
- **Misture**
- **Leve ao Forno por 25 minutos.**

Mas ele está perfeito? Provavelmente um cozinheiro profissional poderia melhorá-lo.

2.1 Características de um algoritmo

- **Definição Exata:** Para que um algoritmo seja considerado exato deve descrever todas as instruções de forma clara e também nunca deixar dúvida no que é para ser feito.
- **Eficiência:** Buscar sempre o menor tempo possível para a execução das atividades e também utilizar de forma inteligente os recursos disponíveis.

Um algoritmo de qualidade deve reunir essas duas características. Uma boa estratégia para desenvolver um algoritmo é:

1. Entender o problema completamente;
2. Descrever todos os mínimos detalhes;
3. Detalhar o problema de forma seqüencial.

3. Algoritmo Estruturado:

Todo algoritmo deve ser escrito de forma clara e precisa e com isso é muito importante que ele seja escrito seguindo um padrão de forma que possa ser interpretado por todos.

Abaixo mostramos informações básicas de um algoritmo:

- Nome:** Identificador do programa
- Variáveis:** Variáveis que são utilizadas no programa
- Procedimentos:** procedimentos que podem ser utilizados no programa
- Funções:** Funções que podem ser utilizados no programa
- Bloco de Ações:** As ações que o programa vai executar.

Mostramos abaixo um exemplo de um algoritmo padrão estruturado, a linguagem que vamos utilizar a partir daqui é o PORTUGOL (Pseudo_liguagem criada para tornar o ensino da lógica de programação o mais simples possível), segue:

Receita do Bolo:**Algoritmo** Receita_Bolo**Variáveis**

panela,ovos,copo_farinha,copo_acucar,copo_leite,colher_fermento

Procedimentos

misture, leve_ao_forno

Funções

espere

Início

ovos:= 4;

copo_farinha:=2;

copo_acucar:=1;

copo_leite:=1;

panela:= ovos+copo_farinha+copo_acucar+copo_leite;

misture

leve_ao_forno

espere 25

fim**3.1. Etapas para a construção de algoritmo**

- **Problema:** Identificar o problema é o primeiro passo no processo de construção de algoritmo;
- **Análise:** Entender o problema é primordial para a resolução do mesmo.
- **Desenvolvimento da solução:** Desenvolvimento do algoritmo;
- **Testes:** Executar o algoritmo com dados conhecidos para obter um resultado esperado;
- **Alterações:** Realizar alterações buscando sempre a velocidade e qualidade;
- **Algoritmo Final;**

Você deve estar se perguntando como vou poder testar se meu algoritmo está correto ou não? Existem alguma técnicas para realizar testes em algoritmo. Abordaremos a técnica do Chinesinho

Chinesinho: Consiste em executar fielmente o que está escrito no algoritmo, utilizando papel e caneta onde anotamos os valores inseridos e modificados pelo programa e ao final da execução saberemos se o resultado é realmente o esperado.

Exemplo:

Algoritmo Soma

Variáveis

a: inteiro

b: inteiro

Início

```
Escreva "Digite o valor de a: "           // nesse momento a = 0
Receba a;                                 // valor digitado pelo usuário a = 3
Escreva "Digite o valor de b: "          // nesse momento b = 0
Receba b;                                 // valor digitado pelo usuário b = 2

Fim imprima a+b                          // Resultado do algoritmo = 5
```

Como sabemos a soma de $3+2 = 5$, então o concluímos que o algoritmo está correto.

4. Linguagem de Programação:

Uma **linguagem de programação** é um método padronizado para expressar instruções para um computador, através da linguagem o programador pode definir precisamente o que o computador irá executar num determinado momento.

4.1. Tipos de Linguagem

Abaixo listamos os tipos de linguagens separados por seu tipo de paradigma de programação.

- **Linguagem Natural;**
- **Linguagem de programação estruturada;**
- **Linguagem Funcional;**
- **Linguagem de programação lógica;**
- **Linguagem de Programação Orientada a Objetos (o.o);**

Vamos explicar apenas os tipos de linguagens mais importantes e mais utilizados na prática.

Linguagem de programação estruturada: é a forma de programar que defende que todos os programas podem ser escritos reduzidos a três estruturas.

4.2. Estruturas:

- Seqüência
- Decisão
- Iteração

Abaixo temos uma lista das linguagens de programação estruturada mais conhecidas no mercado:

- PASCAL
- CLIPPER
- C
- VISUAL BASIC
- DELPHI

Podemos afirmar que a programação estruturada ainda é marcadamente influente, uma vez que grande parte das pessoas ainda aprendem programação através dela.

4.3. Linguagem de Programação Orientada a Objetos (P.o.o)

Para muitos uma linguagem complicada de se entender, para outros a melhor forma de expressar a vida real em um programa de computador.

Definição: É um paradigma de análise, projeto e programação de sistemas de *software* baseado na composição e interação entre diversas unidades de software chamadas de objetos.

Não vamos nos prolongar no estudo da Programação Orientada a Objetos nesta apostila, pois esse assunto será abordado em outro momento.

Abaixo temos uma lista das linguagens de programação Orientada a Objetos mais conhecidas no mercado:

- RUBY
- PYTHON
- SMALTALK
- C++
- C#
- JAVA

Por exigir formas de pensar relativamente complexas, a programação orientada a objetos até hoje ainda não é bem compreendida ou usada pela maioria.

5. Itens Fundamentais:

Para podermos prosseguir com nossos estudos adotaremos algumas convenções e conjuntos de regras. É importante saber que essas regras não são universais, mas servirão para facilitar nossos estudos.

5.1. **Constantes:** Entende-se por uma informação constante, aquela que não sofre nenhuma alteração no decorrer do tempo e em um algoritmo não é diferente uma constante nada mais é que um valor utilizado pelo algoritmo e que não sofre alterações no decorrer da execução do programa.

As constantes podem ser:

- **Numérica:**

1. O uso de uma constante numérica deve seguir o padrão decimal, podendo não existir a parte fracionária.

Exemplos:

- a) 45
- b) 13,9
- c) 1001,5

2. Em caso de constantes com exponencial, usaremos o símbolo $^$ para indicar o valor exponencial.

Exemplos:

- a) 5^3
- b) 2^2 .

3. Podem ser negativas e positivas de acordo com o sinal informado, quando não houver sinal será positiva.

Exemplos:

- a) 20
- b) -20
- c) -32,5

- **Lógica:** Só poderão ter dois valores **Verdadeiro** ou **Falso**, veremos a melhor utilização em estrutura condicionais mais adiante.

- **Literal:** Poderá conter qualquer conjunto de caracteres, letras, números ou símbolos. Como regra usaremos o valor constante entre aspas duplas "".

Exemplos:

- a) "20"
- b) "educandus"
- c) "e-jovem"
- d) "falso" (Apesar de **Falso** ser um valor para constante lógica, é um literal por estar entre aspas "").

5.2. **Variáveis:** São espaços de memória que são alocados para armazenar informações, um valor ou expressão.

5.3. Por que se precisa de variáveis?

R: Para armazenar valores que serão utilizados posteriormente. Ex.: em um cálculo complexo, resultados intermediários podem ser armazenados e posteriormente processados para se obter o resultado final.

Uma variável só pode existir quando lhe é associada uma “nome”, chamado de Identificador. Toda variável precisa ter um identificador.

Os identificadores seguem regras para sua construção, são elas:

- ✓ Pode ser formado por qualquer caracter;
- ✓ O primeiro caracter deverá ser sempre uma letra.
- ✓ Não é permitido o uso de caracteres especiais;

Uma boa dica para você é sempre identificar a sua variável de forma que represente aquilo que vai ser armazenado: ex: CODIGO, para armazenar código do cliente.

Identificadores válidos:

- a) AB
- b) A3b
- c) Nome

Identificadores inválidos:

- a) 1AB
- b) A?
- c) X+5

As variáveis assim como as constantes podem ser : numéricas, literais e lógicas.

Declaração de Variáveis: para que os programas manipulem valores, estes devem ser armazenados em variáveis e para isso devemos declará-las seguindo um padrão:

No instante de declaração devemos identificar o tipo de dado que irá ser armazenado na variável.

Ex: Declare valor_matricula NUMERICO;

Neste exemplo foi informado que a variável valor_matricula só poderá receber dados numéricos.

5.4. Comentários: Quando escrevemos algoritmos longos e complexos é natural que algumas pessoas e até mesmo você que desenvolveu o código sinta dificuldade em entendê-lo.

Para evitar isso, devemos sempre comentar nossos códigos de forma simples e mais clara possível. Explicando funcionalidades do código.

Para fazer uso dessa técnica basta colocar // e em seguida a frase desejada.

Para fazer comentários em varias linhas use /* conteúdo */.

Exemplos:

- Declare codigo NUMERICO //codigo, armazenar um código do tipo numérico.

- /* as variáveis codigo e valor_mensalidade
 - armazenarão tipos numérico */
 - Declare codigo, valor_mensalidade NUMERICO;

6. Expressões Aritméticas

A tradução das expressões aritméticas para linguagem de computador, geralmente acontece da mesma forma que a usada na matemática salvo algumas exceções, vamos aos exemplos:

- $A + B$ // Adição
- $\text{Total} - \text{desconto}$ //Subtração
- $\text{Mensalidade} * 5$ //Multiplicação
- $X / 3$ //Divisão
- $A * B + C$ // 1 Multiplicação e 1 adição

Observação: Na operação de multiplicação não se pode omitir o operador “ * ” e nem mesmo fazer uso do “ . ”

- $XY + B$ //Errado
- $X.Y + B$ //Errado
- $X*Y + B$ // Correto

6.1. Prioridade a ser obedecida entre os operadores:

1. Operando de multiplicação ou divisão na ordem em que aparecerem.
2. Operando de Adição ou Subtração na ordem em que aparecerem.
3. Em expressões que contêm parênteses, a prioridade é resolver as expressões contidas neles.

6.2. Funções:

Funções descrevem relações matemáticas especiais entre os objetos. Às vezes nos deparamos com problemas maiores que precisa ser decomposto em partes menores. Essas partes menores podem ser chamados repetidamente ao longo da execução do programa e para evitar a repetição e organizarmos melhor nosso código criamos trechos de códigos que podemos chamar de Função, Subrotina ou Procedimento.

Algumas funções:

- ARREDONDA(arg)
- TRUNCA(arg)
- RESTO(arg1,arg2)
- EXP(arg1,arg2)

Obs: Lembre-se que você poderá criar sua própria função ou subrotina.

Exemplos do uso de funções no código:

- $A + \text{RESTO}(7,2)$
- $5 + \text{EXP}(2,8)$
- $A + B * \text{ARREDONDA}(Y)$

6.3. Operadores

Definição: são elementos fundamentais que atuam sobre operandos e produzem determinado resultado.

Na expressão $3 + 2$, o 3 e o 2 são operandos e o sinal de '+' é o operador.

Operadores Unários: operadores que atuam sobre um único operando.

Ex: -3
-1

Operadores Binários: operadores que atuam sobre 2 operandos.

Ex: $3 + 2$
 $5 * 3$

Outra classificação para os operadores é quanto ao tipo de dados que manipulam e o valor resultando de sua avaliação.

Operadores Lógicos (booleanos): são operadores para avaliar expressões lógicas e que devolvem como resultado valores lógicos.

- | | | |
|-------|--------------------------|----------------|
| ➤ NÃO | //negação | //prioridade 1 |
| ➤ E | // e lógico ou conjunção | //prioridade 2 |
| ➤ OU | //ou lógico ou disjunção | //prioridade 3 |

Ex:

- NÃO expressao1
- Expressao1 E Expressao2
- Expressao1 OU Expressao2

Operadores Relacionais:

- MAIOR QUE
- MENOR QUE
- MAIOR OU IGUAL
- MENOR OU IGUAL
- IGUAL
- DIFERENTE

- **Tabela verdade:** A Tabela verdade é uma tabela matemática usada em lógica para determinar se uma expressão é verdadeira e válida.

A	B	não A	não B	A ou B	A e B
F	F	V	V	F	F
F	V	V	F	V	F
V	F	F	V	V	F
V	V	F	F	V	V

Pode
mos
chegar
as
seguintes
conclusões
ao

analisar a tabela verdade acima:

1. O operador lógico NÃO sempre inverte o valor do seu operando
2. para que uma operação lógica OU tenha resultado verdadeiro, basta que um dos seus operandos seja verdadeiro.
3. para que a operação lógica E tenha resultado verdadeiro é necessário que seus dois operandos tenham valor lógico “V”.

Formas de representação:

NÃO	NOT (equivalente em inglês), \neg , \sim , !
E	AND (equivalente em inglês), \wedge , &&
OU	OR (equivalente em inglês), \vee ,
IGUAL	=, ==
DIFERENTE	<>, #, !=, ≠
MAIOR QUE	>
MAIOR OU IGUAL	>=, ≥
MENOR QUE	<
MENOR OU IGUAL	<=, ≤

7. Expressões literais

Constantes Literais: As constante literais em computação geralmente são representadas entre aspas duplas ou simples

Ex: 'Educandus'

Variáveis Literais: são variáveis que assumem valores literais, ou seja, qualquer conjunto de caracteres alfanuméricos.

EX: A= "EDUCANDUS"

7.1. Concatenação

A concatenação consiste em unir dois valores, anexando sempre o segundo valor ao final do primeiro

Consideração:

- Caso um dos valores não seja literal ele será convertido e o operador para realizar essa operação é o "+".

Ex: "SOL" + "DADO" = "SOLDADO"

7.2. Operadores relacionais: são operadores binários que devolvem valores lógicos, verdadeiro ou falso. Estes operadores são somente usados quando se deseja realizar comparações.

Confira abaixo os operadores relacionais.

Operador	Comparação
>	maior que
<	menor que
>=	maior igual
<=	menor igual
=	igual
<>	diferente

Exemplos: suponha que X é igual a 8:

```
X <= 3          //falso
X < 3          //verdadeiro
```

Comando de Atribuição: Este comando nos permite fornecer um valor a uma certa variável, onde o tipo de informação deve ser compatível com o tipo de variável utilizada, ou seja, somente poderemos atribuir “Pedro” a uma variável do tipo caractere.

Este comando é representado pelos sinais (=, := ou □);

Veja abaixo o seguintes exemplos:

Comandos Válidos

```
DECLARE a,b NUMERICO
DECLARE x,y LITERAL
```

```
A=6+2;          X="Educandus";
B=53;           Y = "sol" + "dado";
```

Comandos Inválidos

```
DECLARE a,b NUMERICO
DECLARE x,y LITERAL
```

```
A=6+"2";       X=30;
B="53";        Y = 5 + 4;
```

Comandos de Entrada e Saída: Todo programa executado em um computador (salvo algumas exceções) consiste de três etapas ou pontos de trabalho: a entrada de dados, o seu processamento e a saída dos mesmos.

Para realizarmos essas ações devemos usar os comandos de entrada LEIA() e comandos de saída ESCREVA(). Esses comandos independem do dispositivo utilizado (teclado, discos, impressora, monitor, etc.);

Para cada tipo de dispositivo existe um comando específico:

Dispositivo de Entrada : LEIA()

Dispositivo de Saída: ESCREVA()

Comandos de Entrada: A entrada de dados inseridos pelos usuários será feita através da instrução LER(), que transfere para a memória (variáveis) os dados que os programas irão utilizar.

Sintaxe:

- LER //comando de leitura.
- Parênteses () //de uso obrigatório
- Identificador: //nome da variável a ser lida
- Ponto e vírgula; //determinação do fim do comando.

Exemplo:

```
ALGORITMO entrada_de_dados
DECLARE preço_unit, quant, preco_total NUMERICO
INICIO

    LER (preço_unit, quant);
    preço_total = preço_unit * quant;
    ESCREVER (preço_total);
FIM ALGORITMO entrada_de_dados;
```

Comandos de Saída: consiste no comando pelo qual a informação contida na memória seja colocada nos dispositivos de saída.

Sintaxe:

- ESCREVER OU IMPRIMEIR //comandos de saída.
- Parênteses () //de uso obrigatório
- Identificador e/ou expressão: //dados a serem exibidos
- Ponto e vírgula; //determinação do fim do comando.

Observação:

O comando ESCREVER indica saída no monitor;
O comando IMPRIMIR indica saída na impressora;

Exemplo:

```
ALGORITMO saida_de_dados
DECLARE preço_unit, quant, preco_total NUMERICO
INICIO
    preço_unit = 5.0;
    quant = 10;
```

```
preco_total = preço_unit * quant;  
    ESCREVER (preco_total);  
FIM ALGORITMO saida_de_dados;
```

8. Estrutura seqüencial e condicional simples

Os algoritmos assim como os comandos seguem regras para serem terem sua estrutura escrita.

Veja a ordem

1. Declaração de variáveis e em seguida os comandos;
2. Deverão ser executados de cima para baixo;
3. Sempre que formos iniciar nossos algoritmos, usaremos a palavra ALGORITMO e ao final utilizaremos a palavra FIM ALGORITMO. O nome do algoritmo pode ser descrito ou não.
 - a. Ex: ALGORITMO teste

```
FIM ALGORITMO teste;
```

8.1. Estrutura condicional

Caso lhe fosse pedido para escrever um algoritmo que receba duas notas e no final escreva as notas que forem maiores que 6, certamente você encontraria dificuldade para a criação desse algoritmo. Isto porque encontramos um problema condicional e para resolver esse problema você deve fazer uso de uma estrutura condicional.

A estrutura condicional permite que um bloco de comandos seja executado se uma condição lógica for satisfeita, ou seja, a estrutura condicional só irá executar o bloco interno se a condição tiver o valor verdadeiro.

Notação estrutura condicional:

SE (<condição lógica>) ENTAO

<comando1>

<comando2>

<comando3>

fimse

Exemplo:

```
ALGORITMO
DECLARE nota1, nota2 NUMERICO
INICIO
    LER(nota1, nota2);
    SE (nota1>6) ENTAO

        ESCREVER ("A nota 1 é maior que 6 e o seu valor é: ", nota1);
        fimse
    SE (nota2>6) ENTAO

        ESCREVER ("A nota 2 é maior que 6 e o seu valor é: ", nota2);
        fimse

FIM ALGORITMO
```

9. Aninhamento de SE – Estrutura Condicional Simples II

Notação estrutura condicional aninhada:

```
SE (<condição lógica>) ENTAO

<comando1>
<comando2>
    SE (<condição lógica 2>) ENTAO

        <comando3>
        fimse
    fimse
```

Exemplo: Neste algoritmo vamos receber três valores **a,b,c** e escrever o valor de **a** se for maior que **b**, e em seguida escreve o valor de **b** se for menor **a** e maior que **c**.

```
ALGORITMO
DECLARE a,b,c NUMERICO
INICIO
    LER(a,b,c);
    SE (a>b) ENTAO

        ESCREVER ("O valor de A:", a);

    SE (b>c) ENTAO
```

ESCREVER (“O valor de B:”, b);

fimse

fimse

FIM ALGORITMO

Observação: o segundo SE só testa se **b** é maior que **c**, pois quando é feita a primeira condição (**a>b**), exclui a necessidade se **b** é menor que **a**.

10. Estrutura condicional composta I

Notação estrutura condicional:

SE (<condição lógica>) **ENTAO**

<comando1>

<comando2>

SENAO

<comando3>

<comando4>

fimse

Exemplo:

SE ($x==y$) **ENTAO**

ESCREVER (“x é igual a Y”);

SENAO

ESCREVER (“x é diferente de y”);

fimse

O algoritmo só irá escrever (“x é diferente de y”), se a expressão lógica for falsa.

11. Estrutura condicional composta II: Existem casos em que é necessário se estabelecerem verificações de condições sucessivas. Quando uma ação é executada, ela poderá ainda estabelecer novas condições, isso significa condições dentro de condições. Esse tipo de estrutura poderá ter diversos níveis de condição, porém esse tipo de estrutura torna difícil o entendimento do algoritmo.

Notação estrutura condicional:

```
se (<condição>) entao
    <comandos 1>
senao se (<condição2>) entao
    <comandos 2>
senao
    <comandos 3>
fimse
fimse
```

12. **Estrutura de repetição determinada:** Em vários momentos, na programação, se torna necessário repetir um trecho de um programa um determinado número de vezes. Nesse caso, pode ser criado um laço de repetição que efetue o processamento de um determinado trecho, tantas vezes quantas forem necessárias. Os laços de repetição também são conhecidos por loopings.

12.1. **Classificação:** Os laços são divididos em laços CONTADOS e laços CONDICIONAIS.

○ **Variáveis de Laço:** com a utilização de estruturas de repetição para a elaboração de algoritmos, torna-se necessário o uso de dois tipos de variáveis para a resolução de diversos tipos de problemas: **variáveis contadoras e variáveis acumuladoras.**

▪ **Variáveis contadoras:** É uma variável de controle, “NUMERICA”, que serve para controlar quantas vezes um determinado trecho de programa foi executado.

Inicialmente recebe um valor, geralmente 0 (ZERO), antes do início de uma estrutura de repetição e é incrementada no interior da estrutura de uma valor constante, geralmente 1

Exemplo:

```
...
cont = 0;
<estrutura de repeticao>
...
cont = cont +1;
...
<fim da estrutura de repetição>
...
```

- **Variáveis contadoras:** É uma variável de controle, inteira, que serve para acumular valores.

Inicialmente recebe um valor, geralmente 0 (ZERO), antes do início de uma estrutura de repetição e é incrementada no interior da estrutura de uma valor variável, geralmente a variável utilizada na estrutura de controle:

Exemplo:

```
...
cont = 0;
<estrutura_de_repeticao_x>
...
cont = cont +x;
...
<fim_da_estrutura_de_repeticao_x>
```

- **Laços Contados ou Estrutura de Repetição Determinada:** Damos esse nome quando conhecemos a quantidade de vezes que o comando composto no interior do algoritmo será executado.

- **Sintaxe - Forma Geral 1 :**

```
para <variavel> = <valor_inicial> ate <valor_final>

faca <comando_unico>

fim_para
```

Exemplo: algoritmo para escrever 10 vezes a frase “País: Brasil”

```
ALGORITMO repetição
DECLARE i NUMERICO
  PARA i = 1 ate 10
    FACA
      ESCREVER (“País: Brasil”);
    FIM_PARA
  FIM_ALGORITMO repetição
```

- **Sintaxe - Forma Geral 2 :**

```
para <variavel> = <valor_inicial> ate <valor_final>

faca <comando_composto>

fim_para
```

Exemplo: algoritmo para escrever 10 vezes a frase “País: Brasil”

```
ALGORITMO repetição_2
DECLARE i,par NUMERICO
par = 0
  PARA i = 1 ate 100
    FACA
      ESCREVER (“par”);
      Par = par + 2;
    FIM_PARA
FIM_ALGORITMO repetição_2
```

Exercícios resolvidos e propostos;

13. **Estrutura de repetição indeterminada:** Ao contrário das estruturada de repetição determinada os lanços condicionais não sabemos antecipadamente quantas vezes o corpo do laço será executado.

Tipos de Construção: as construções que implementam estruturas de repetição indeterminadas são:

- **Enquanto** – Laço condicional com teste no início;

Caracterizado por uma estrutura que logo no início do laço efetua um teste lógico, para liberar a execução ou não dos comandos que se encontram no interior do laço.

- **Sintaxe - Forma Geral 1 :**

```
ENQUANTO <condição>
```

```
faca <comando_unico>
```

```
fim_enquanto
```

- **Sintaxe - Forma Geral 2 :**

```
ENQUANTO <condição>
```

```
faca <comando_composto>
```

```
fim_enquanto
```

Nesse tipo de laço a variável a ser testada deve possuir um valor associado antes da construção do laço.

Semântica: No início da construção é realizado um teste lógico se o resultado for falso os comandos internos do laço condicional não serão executados e o algoritmo ira executar a próxima instrução ao final do laço.

Caso o teste apresente resultado verdadeiro o comando interno é executado e ao seu término retorna-se ao teste da condição até que a mesma apresente resultado falso os comandos serão executados.

Exemplo - Forma Geral 1:

```
ALGORITMO condicional
DECLARE soma, num NUMERICO
    num = 1;
    soma = num;
    ENQUANTO (soma < 1000 )
        faça ESCREVE (num)
        num = num + 1;
        soma = soma + num;
    fim_enquanto
FIM_ALGORITMO condicional
```

Exemplo - Forma Geral 2:

```
ALGORITMO condicional_2
DECLARE media, soma_notas, cont_alunos NUMERICO
    media = 0;
    soma_notas = 0;
    cont_alunos = 0;
    ENQUANTO (cont_alunos <=3 )
        faça LER (7)
            soma_notas = soma_notas + notas (14+7);
            cont_alunos = cont_alunos + 1(2+1);
        fim_enquanto
    media = soma_notas/cont_alunos (21/3);
    ESCREVER (“A média da turma é: ”, media);
FIM_ALGORITMO condicional_2
```

- **Repita** - Laço condicional com teste no final;

Caracterizado por executar ao menos uma vez os comandos internos ao laço e no final é realizado um teste lógico verificando se é permitido ou não executar os comandos internos.

- **Sintaxe:**

REPITA

<comando_composto>

ATE QUE <condicao>

Nesse tipo de laço a variável a ser testada pode ser inicializada ou lida dentro do laço.

Observação: Nos laços condicionais a variável que é testada tanto no início quanto no final do laço, deve sempre estar associada a um comando que a utilize no interior do laço. Caso isso não ocorra o programa ficará em loop infinito;

Semântica: Como já havia falado anteriormente o comando é executado uma vez e em seguida é testado se pode ou não continuar a ser executado..

Se a condição for falsa o comando é executado e o processo é repetido até que a condição se torne falsa, quando so assim a execução segue pelo comando imediato ao fim do laço.

Exemplo - 1:

```
ALGORITMO soma_salario
DECLARE soma, salario NUMERICO
    soma = 0;
    REPITA

        LER (salario);

        Soma = soma + salario;

        ATE QUE (salario < 0 )

    ESCREVER (soma);
FIM_ALGORITMO soma_salario
```

14. Combinando Estrutura de Repetição e Estrutura condicional

15. Variáveis compostas homogêneas:

Os tipos homogêneos são conjuntos do mesmo tipo básico. A utilização desse tipo de estrutura de dados recebe diversos nomes, tais como: variáveis indexadas, compostas, arranjos, tabelas em memória, arrays (do inglês) vetores e matrizes.

15.1. Variáveis compostas homogêneas - Unidimensionais: *Vetores*

Os **vetores** são uma lista de elementos do mesmo tipo. Todas as variáveis e constantes ocupam um espaço em memória. Este espaço ocupado é um espaço linear. Quando possuímos uma ordem e um índice de acesso aos elementos de um conjunto então temos caracterizado um vetor.

Exemplo:

NOTA									
6,0	7,0	9,0	6,0	5,5	9,1	10,0	4,7	7,4	8,6
1	2	3	4	5	6	7	8	9	10

Declaração:

```
DECLARE lista de identificadores [li:ls]t
```

Onde:

Lista de identificadores são: // nome da variável;

Li : // é o limite inferior do intervalo de variação dos índices;

Ls: // é o limite superior do intervalo de variação dos índices;

T: //é o tipo dos componentes da variável(numérico, literal, lógico)

Obs: O limite superior não deve ser menor que o limite inferior;

Exemplo 1:

```
DECLARE NOTA [1:10] NUMERICO;
```

Exemplo 2:

```
ALGORITMO vetor
  DECLARE moda[] NUMERICO;
  DECLARO i NUMERICO;
INICIO
  i = 0;

  REPITA
    SE(i>7)ENTAO
      INTERROMPA;
    FIM SE
    Moda[i] = 2;
    i = i + 1;
  FIM REPITA
  ESCREVER (moda[1], moda[2], moda[3]);
FIM ALGORITMO vetor
```

15.2. Variáveis compostas homogêneas - Multidimensionais:

Nada mais são do que um conjunto de dados referenciado por um mesmo nome e que necessita de mais de um índice para ter seus elementos individualizados (ou referenciados).

As variáveis compostas multidimensionais mais comuns são as Matrizes e tabelas, onde o primeiro índice se refere ao número linha e o segundo ao número da coluna.

Matrizes:

A matriz mais comum é a de duas dimensões (linha e coluna), por se relacionar diretamente com a utilização de tabelas. Trabalharemos somente com matrizes de 2 dimensões, por serem mais comuns, mas podem ser necessárias, em algum momento, matrizes de 3 ou mais dimensões.

Declaração:

```
DECLARE matriz [1:4,1:3] NUMERICO
```

Onde:

Com essa declaração passa a existir uma matriz $4 \times 3 = 12$ elementos endereçáveis por um par de índices, sendo o primeiro linha e o segundo coluna.

16. Variáveis composta Heterogêneas - Registros

O conceito de registro visa facilitar o agrupamento de variáveis que não são do mesmo tipo, mas que guardam estreita relação lógica. Eles correspondem a conjuntos de posições de memória conhecidos por um mesmo nome e individualizados por identificadores associados a cada conjunto de posições.

Cada componente é individualizado pela explicitação de seu identificador.

Declaração:

```
DECLARE cadastro REGISTRO (nome, rua Literal
                           numero,cep,cpf, NUMERICO
                           Ht[1:3] NUMERICO
                           nascimento NUMERICO
                           tem_dependente LOGICO)
```

A referência ao conteúdo do registro deve ser feita da seguinte maneira:

<Identificador_do_registro>.<identificador_do_componente>

Exemplo: cadastro.nome;

16.1. Registro de conjuntos

Você deve estar se perguntando senão seria possível utilizar vetores e matrizes dentro de um registro? A resposta é sim, isso é possível. Veja como fazer abaixo:

Exemplo:

```
DECLARE cadastro REGISTRO (nome, Literal
                           Cpf NUMERICO
                           Ht[1:3] NUMERICO
                           nascimento NUMERICO
                           tem_dependente LOGICO)
```

```
DECLARE endereco REGISTRO (rua Literal
                           numero,cep NUMERICO)
```

O acesso aos dados se da mesma forma como já foi explicado acima do registro mais externo, depois o mais interno e finalmente o identificador do componente;

Analise o exemplo abaixo:

ALGORITMO exemplo_reg

```
    DECLARE tabela[1:100] REGISTRO (nome LITERAL, codigo NUMERO)
    DECLARE codigoDesejado, i, k NUMERICO
```

```
LER (tabela[1]...tabela[100]);
```

```
INICIO
```

```
    k=1;
```

```
Enquanto (k<=500) faça
```

```
    LER (codigoDesejado);
```

```
    i=1;
```

```
    Enquanto ((tabela[i].codigo != codigoDesejado ) ou (i!=100)) faça
```

```
        i=i+1;
```

```
    Fim enquanto
```

```
    Se (tabela[i].codigo == codigoDesejado) entao
```

```
        ESCREVER (codigoDesejado, tabela[i].nome);
```

```
        Senao
```

```
            Escrever (“Código Inválido”);
```

```
    Fim se
```

```
    k=k+1;
```

```
Fim enquanto
```

```
Fim algoritmo exemplo_reg
```

17. Arquivos

Tendo em vista a necessidade de armazenar pequenas informações podemos fazer uso do trabalho com arquivos, nós podemos criar, acessar e deletá-los.

Para armazenar alguma informação no arquivo é necessário utilizar a função de escrita no arquivo.

Obs: essas operações geralmente são muito demoradas e devemos ter muito cuidado ao utilizar, pois podem deixar nosso algoritmo lento e ineficiente.

O acesso ao arquivo no Algoritmo deve ser feito através de leitura e escrita de registro. Para manusear o arquivo faz-se necessário que ele seja declarado e aberto e ao final do código ou quando houver necessidade o arquivo deve ser fechado(isso é muito importante)

Os arquivos estão organizados basicamente de duas formas:

- **Seqüencial:** os registros são obtidos ou inseridos no arquivo em ordem seqüencial.
- **Direta:** o acesso do registro é feito em ordem aleatória o que o proporciona uma melhor facilidade no acesso aos dados, não sendo necessário pesquisar-se pelo registro, pois o mesmo pode ser obtido diretamente.

Declaração:

Acompanhe a notação abaixo para entender melhor como funciona o processo de declaração de um arquivo em algoritmos:

DECLARE <lista de identificador> ARQUIVO <organização> DE <nome do registro>

Onde:

<Lista de indentificadores > //são os nomes usados para referenciar os arquivos;
ARQUIVO // é uma palavra chave;
< Organização > // tipo de organização do arquivo que pode ser direta ou seqüencial.
DE // é uma palavra chave;
<nome do registro > // nome do registro que será usado para ter acesso aos arquivos.

17.1. Abertura de arquivos

A seguinte notação deve ser utilizada para a abertura dos arquivos:

ABRIR <lista_de_identificadores_de_arquivo><tipo_de_utilizacao>

Onde,

ABRIR: //É uma palavra chave;
<lista_de_identificadores_de_arquivo> //são os nomes dos arquivos.
<tipo_de_utilizacao> //especifica se o arquivo deve ser aberto para leitura, escrita ou ambos, simultaneamente

Exemplos:

ABRIR agenda LEITURA
ABRIR agenda ESCRITA
ABRIR agenda

17.2. Fechamento do arquivo

A seguinte notação deve ser utilizada para desfazer a associação entre o modelo e o arquivo físico:

```
FECHAR<lista_de_identificadores_de_arquivo>
```

Onde,

```
FECHAR //é uma palavra chave;
```

```
<lista_de_identificadores_de_arquivo> //são os nomes dos arquivos.
```

Observação: Sempre que precisar alterar o tipo de acesso ao arquivo, antes você deve fechar o arquivo.

Exemplo:

```
FECHAR agenda;
```

Analise o exemplo abaixo:

```
ALGORITMO exemplo_arq
```

```
DECLARE a,b ARQUIVO SEQUENCIAL DE t
```

```
DECLARE t REGISTRO (nome LITERAL, salario NUMERICO)
```

```
ABRIR a LEITURA
```

```
ABRIR b ESCRITA
```

```
INICIO
```

```
//Leitura do arquivo
```

```
Repita
```

```
LER (a.t);
```

```
//Teste para detectar fim do algoritmo a
```

```
Se (a.FDA) então
```

```
IMTERROMPA;
```

```
Fim se
```

```
ESCREVER (b.t);
```

```
Fim repita
```

```
Fechar a;
```

```
Fechar b;
```

```
Fim algoritmo exemplo_arq
```

Observação sobre o exemplo:

- Não há necessidade de se declarar a variável FDA, já que ela é um atributo que está associado ao arquivo. O sistema operacional é que fica encarregado de atualizá-lo a cada acesso do arquivo.

18. SubRotina

Definição: Consiste numa porção de código que resolve um problema muito específico, parte de um problema maior (a aplicação final).

A utilização das subrotinas nos algoritmos se deu pelas seguintes necessidades:

- Redução de código duplicado num programa;
- Possibilidade de reutilizar o mesmo código sem grandes alterações em outros programas;
- A decomposição de problemas grandes em pequenas partes;
- Melhorar a interpretação visual de um programa;
- Esconder ou regular uma parte de um programa, mantendo o restante código alheio às questões internas resolvidas dentro dessa função;

Dicas:

- As subrotinas (módulos) devem ter um tamanho limitado, pois podem se tornar difíceis de serem compreendidas.
- Todo módulo é constituído por uma seqüência de comandos que operam sobre um conjunto de objetos, que podem ser globais ou locais.

Declaração:

SUBROTINA nome (lista_de_parametros_formal)

Declaração de objetos locais a subrotina

Comandos da Sub-rotina

Fim subrotina

Onde,

lista_de_parametros_formal : é a lista de objetos que serão substituídos por outros objetos. Os parâmetros atuais devem concordar com os parâmetros formais.

Parâmetros de subrotina

Classificação dos parâmetros de uma subrotina:

- **De entrada:** São aqueles que têm seus valores estabelecidos fora da subrotina e não podem ser alterado dentro da subrotina;
- **De saída:** são aqueles que têm seus valores alterados dentro da subrotina;
- **De entrada e Saída:** são aqueles que têm seus valores estabelecidos fora da subrotina e podem ser alterados dentro da subrotina.

Passagem de parâmetros

- **Por valor:** É uma maneira típica de passar parâmetros em funções, quer dizer que a modificação de um parâmetro não actualiza o dado da variável como parâmetro, apesar de mudarmos o valor do parâmetro dentro da função, a variável original não se vê afetada pela mudança.
- **Por resultado:** as alterações feitas nos parâmetros formais dentro de uma subrotina, refletem-se nos parâmetro atuais. Quando a passagem é por resultado significa que o parâmetro é de saída.
- **Por referência:** Neste caso, a mudança do valor de um parâmetro dentro de uma função afeta o valor da variável original.

19. Função:

As funções embora muito semelhante com as subrotinas apresentam uma característica especial de retornar ao algoritmo que as chamou um valor associado ao nome da função.

Declaração: A declaração de uma função é idêntica a de uma subrotina, com exceção de que é necessário o seu tipo, ou seja, o tipo do valor retornado. Além de numéricas as funções podem ser lógicas e literais.

Exemplo:

Funcao TIPO nome (lista_de_parâmetros_formais)

Declaração dos objetos locais à função

Comandos da função

Fim funcao

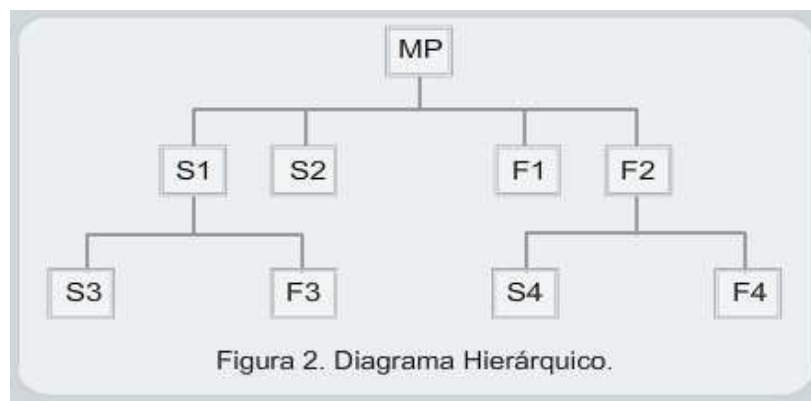
Chamadas de função:

Exemplo:

nome(lista_de_parametros_formais).

Ao terminar a execução dos comandos da função, o fluxo de controle retorna ao comando seguinte àquele que provocou a chamada.

Importante: Subrotinas e Funções são módulos hierarquicamente subordinados a um algoritmo, comumente chamado de módulo principal, da mesma forma subrotinas e funções pode conter outras subrotinas e funções aninhadas veja imagem a seguir:



Exemplo – exercício resolvido - exercício proposto

EXERCÍCIOS RESOLVIDOS

Antes de tudo queremos deixar bem claro para você aluno que as respostas aqui colocadas não são as definitivas, lembre-se na eterna busca pela melhoria do código. O que estamos colocando aqui são apenas formas de se resolver o problema em questão.

1. Escrever um Algoritmo que receba as 4 primeiras notas e ao final escrever se o aluno passou ou foi reprovado. Para o aluno passar será necessário ele atingir uma nota igual ou maior que 6.

Resposta:

ALGORITMO

(Declaração das variáveis que armazenaram os valores das notas)

DECLARE NOTA1, NOTA2, NOTA3, NOTA4, MEDIA NUMERICO;

LEIA NOTA1, NOTA2, NOTA3, NOTA4; (ler as 4 notas)

MEDIA = (NOTA1 +, NOTA2, NOTA3, NOTA4) / 4 ; (Soma as notas e divide por 4 para obter a media)

SE (MEDIA >= 6) (se a Media for igual ou maior que 6 excuta o comando dentro da estrutura)

ENTAO ESCREVA "O ALUNO PASSOU!";

FIM SE;

SE (MEDIA < 6) (se a Media for menor que 6 excuta o comando dentro da estrutura)

ENTAO ESCREVA " ALUNO REPROVADO";

FIM SE;

FIM ALGORITMO

2. Escrever um Algoritmo que receba duas variáveis A e B, e em seguida escreva a maior.

Resposta:

ALGORITMO

DECLARE A, B NUMERICO;

LEIA A, B;

SE (A > B)

ENTAO ESCREVA " O Valor de A é maior :", A;

FIM SE;

SE (B > A)

ENTAO ESCREVA "O Valor de B é maior ", B;

FIM SE;

FIM ALGORITMO

3. Escreva um Algoritmo que receba os valores de A e B, se a soma de A+B for maior que 20 então escreva o total de A+B. Porém, se a soma for menor que 40, leia o valor de C e em seguida escreva a soma de A + B + C.

Resposta:

ALGORITMO

DECLARE A, B, C, SOMA NUMERICO;

LEIA A, B;
SOMA = A + B;

SE (SOMA >20) Se a soma de A+B for maior que 20 então executa os comandos dentro do SE
ENTAO ESCREVA " O total de A + B é :", SOMA; escreve a soma de A + B

SE (SOMA < 40) Se a soma A + B for menor que 40 executa os comandos dentro do SE
ENTAO LEIA C;
SOMA = A +B +C;
ESCREVA "O total de A+ B +C é: ", SOMA;
FIM SE

FIM SE

FIM ALGORITMO

4. Escreva um Algoritmo para o problema abaixo: Em uma escola um aluno precisa obter a média 6 nas 2 primeiras notas para passar de ano. Caso a média das notas seja inferior a 6 ele terá que fazer a recuperação e obter os pontos restantes para passar.

A Regra é a seguinte:

MEDIA = (NOTA1 + NOTA2) / 2;
MEDIA >= 6 PASSOU
MEDIA < 6 = FAZ RECUPERACAO
MEDIA = NOTA1 + NOTA2 + RECUPERACAO
MEDIA >= 6 PASSOU
MEDIA < 6 REPROVADO

Resposta:

ALGORITMO

DECLARE NOTA1, NOTA2, MEDIA, RECUPERACAO NUMERICO;

LEIA NOTA1, NOTA2;

MEDIA = (NOTA1 + NOTA2) / 2;

SE (MEDIA < 6)
ENTAO LEIA RECUPERACAO;
MEDIA = (NOTA1 + NOTA2 + RECUPERACAO) / 3;
FIM SE

SE (MEDIA >= 6)
ENTAO ESCREVA "APROVADO";

FIM SE
SE (MEDIA < 6)
ENTAO ESCREVA "REPROVADO";
FIM SE
FIM ALGORITMO

5. Qual destes não é um exemplo de concatenação válido?

- A – 1 + “CLIENTE” = “UM_CLIENTE”
- b – “Educandus” + WEB = “EducandusWEB”
- c – “Rua ” + 14 = “Rua 14”
- d – 30 + ”Km” = “30Km”
- e - “DES”+”CONTO” = “DESCONTO”

Resposta: A

6. A tabela dada a seguir contém vários itens que estão estocados em vários armazéns de um supermercado. É fornecido, também, o custo de cada um dos produtos armazenados.

ESTOQUE

	PRODUTO1	PRODUTO2	PRODUTO3
ARMAZÉM1	1200	3700	3737
ARMAZÉM2	1400	4210	4224
ARMAZÉM3	2000	2240	2444

CUSTO

CUSTO(R\$)	260.00	420.00	330.00
------------	--------	--------	--------

Fazer um algoritmo que:

- a) Leia o estoque inicial;
- b) Determine e imprima quantos itens estão armazenados em cada armazém;
- c) Qual o armazém que possui a maior quantidade de produto 2 armazenado
- d) O custo total de:
 - 1. Cada produto em cada armazém;
 - 2. Estoque em cada armazém;
 - 3. Cada produto em todos os armazéns;

Resposta:

(Algoritmo de Resolução questões “a” e “b”)

```

ALGORITMO EXEMPLO1
DECLARE a[1:3], p[1:3], i, j, maior, anterior, custo[1:3], estoque[1:3,1:3],
custoArmazemProduto[1:3,1:3]: NUMERICO
LEIA a[1],..., p[1],..., i, j, maior, anterior, custo[1],..., estoque[1,1],..., custoArmazemProduto[1,1],...
INICIO
maior, anterior = 0
i, j = 1 // variáveis contadoras
custo[1] = 260.00 // variável unidimensional custo do Produto 1
custo[2] = 420.00 // variável unidimensional custo do Produto 2
custo[3] = 330.00 // variável unidimensional custo do Produto 3
estoque[1,1] = 1200 // variável multidimensional estoque do Produto 1 no Armazém 1
estoque[1,2] = 3700 // variável multidimensional estoque do Produto 2 no Armazém 1
estoque[1,3] = 3737 // variável multidimensional estoque do Produto 3 no Armazém 1
estoque[2,1] = 1400 // variável multidimensional estoque do Produto 1 no Armazém 2
estoque[2,2] = 4210 // variável multidimensional estoque do Produto 2 no Armazém 2
estoque[2,3] = 4224 // variável multidimensional estoque do Produto 3 no Armazém 2
estoque[3,1] = 2000 // variável multidimensional estoque do Produto 1 no Armazém 3
estoque[3,2] = 2240 // variável multidimensional estoque do Produto 2 no Armazém 3
estoque[3,3] = 2444 // variável multidimensional estoque do Produto 3 no Armazém 3

ENQUANTO i <= 3
FACA
    ENQUANTO j <= 3
    FACA
        LER (estoque[i,j]); // resolve questão “a” – Ler o estoque inicial.
        SE( RESTO(i,3) == 1 ) ENTAO
            a[i] = a[i] + estoque[i,j]; // Itens no armazém 1
        FIM-SE
        SE( RESTO(i,3) == 2 ) ENTAO
            a[i] = a[i] + estoque[i,j]; // Itens no armazém 2
        FIM-SE
        SE( RESTO(i,3) == 0 ) ENTAO
            a[i] = a[i] + estoque[i,j]; // Itens no armazém 3
        FIM-SE
        j = j + 1;
    FIM-ENQUANTO
    i = i + 1;
FIM-ENQUANTO // o comando escreva abaixo resolve a questão “b” – Itens armazenados em cada
armazém
ESCREVER(“Estoque no Armazem2 ”+a[2]); ESCREVER(“Estoque no Armazem3 ”+a[3]);
    
```

(Resolução questões “c” e “d”)

...//continuação exemplo 1

```

i = 1 // reinicializando variável contadora
ENQUANTO i <= 3
    ESCREVER(“Estoque no Armazém ”+ i +” = ”+a[i]);
FACA
FIM-ENQUANTO

i = 1 // reinicializando variável contadora
ENQUANTO i <= 3
FACA
    
```

```

        SE( estoque[i,2] >= anterior ) ENTAO
            maior = estoque[i,2]; // Resolve questão “c” – Armazém que possui maior quantidade do
Produto 2.
        FIM-SE
        anterior = estoque[i,2];
    FIM-ENQUANTO
    ESCRIVER (“Armazém que possui maior quantidade do Produto 2 ” + maior);

    i = 1 // reinicializando variável contadora
    ENQUANTO i <= 3
    FACA
        ENQUANTO j <= 3
        FACA
            SE( RESTO(i,3) == 1 ) ENTAO
                SE( RESTO(j,3) == 1 ) ENTAO
                    custoArmazemProduto [i,j] = custo[i] * estoque[i,j]; // Produto1 -
armazém1
                FIM-SE
                SE( RESTO(j,3) == 2 ) ENTAO
                    custoArmazemProduto [i,j] = custo[i] * estoque[i,j]; // Produto2 -
armazém1
                FIM-SE
                SE( RESTO(j,3) == 0 ) ENTAO
                    custoArmazemProduto [i,j] = custo[i] * estoque[i,j]; // Produto3 -
armazém1
            FIM-SE
        FIM-SE

...//continuação exemplo 1 questão “d.1”

        SE( RESTO(i,3) == 2 ) ENTAO
            SE( RESTO(j,3) == 1 ) ENTAO
                custoArmazemProduto [i,j] = custo[i]* estoque[i,j]; // Produto1-
armazém2
            FIM-SE
            SE( RESTO(j,3) == 2 ) ENTAO
                custoArmazemProduto [i,j] = custo[i]*estoque[i,j]; // Produto2 -
armazém2
            FIM-SE
            SE( RESTO(j,3) == 0 ) ENTAO
                custoArmazemProduto [i,j] = custo[i]*estoque[i,j]; // Produto3 -
armazém2
        FIM-SE
    FIM-SE
    SE( RESTO(i,3) == 0 ) ENTAO
        SE( RESTO(j,3) == 1 ) ENTAO
            custoArmazemProduto [i,j] = custo[i]*estoque[i,j]; // Produto1 -
armazém3
        FIM-SE
        SE( RESTO(j,3) == 2 ) ENTAO
            custoArmazemProduto [i,j] = custo[i]*estoque[i,j]; // Produto2 -
armazém3
        FIM-SE
        SE( RESTO(j,3) == 0 ) ENTAO
            custoArmazemProduto [i,j] = custo[i]*estoque[i,j]; // Produto3 -

```

armazém3

```

                                FIM-SE
                                FIM-SE
                                j = j + 1;
                                FIM-ENQUANTO
                                i = i + 1;
                                FIM-ENQUANTO

                                i, j = 1 //reinicializando variáveis contadoras
                                ENQUANTO i <= 3
                                FACA
                                    ENQUANTO j <= 3 // o algoritmo abaixo resolve a questão d.1
                                    FACA
                                        ESCREVER("Custo do produto " + j + " no armazém " + i + " = " +
                                custoArmazemProduto[i,j]);
                                        j = j + 1;
                                        FIM-ENQUANTO
                                        i = i + 1;
                                FIM-ENQUANTO

                                ...//continuação exemplo 1 questão "d.2"
                                i = 1 //reinicializando variáveis contadoras
                                ENQUANTO i <= 3 // o algoritmo abaixo resolve a questão d.2
                                FACA
                                    ESCREVER("Custo do estoque no armazém " + i + " = " + custo[i]*a[i]);
                                    i = i + 1;
                                FIM-ENQUANTO

                                i = 1 //reinicializando variáveis contadoras
                                ENQUANTO i <= 3 // o algoritmo abaixo resolve a questão d.3
                                FACA
                                    ESCREVER("Custo do estoque no armazém " + i + " = " + custo[i]*p[i]);
                                    i = i + 1;
                                FIM-ENQUANTO
    
```

FIM ALGORITMO EXEMPLO1

7. Declare um arquivo com organização seqüencial para registros com o seguinte formato:

numero_da_conta	nome_cliente	saldo	data_ultima_operacao
-----------------	--------------	-------	----------------------

Resposta:

```

DECLARE agencia ARQUIVO sequencial DE conta
DECLARE conta REGISTRO (numero_da_conta NUMERICO, nome_cliente LITERAL, saldo
NUMERICO, data_ultima_operacao NUMERICO)
    
```

8. Declare um arquivo com organização seqüencial usando o registro dado abaixo:

telefone	cidade	custo_da_ligacao
----------	--------	------------------

Resposta:

DECLARE conta_telefonica ARQUIVO sequencial DE ligacao
 DECLARE ligacao REGISTRO (telefone NUMERICO, cidade LITERAL, custo_da_ligacao NUMERICO)

9. Em certo município, vários proprietários de imóveis estão em atraso com o pagamento do imposto predial. Vamos desenvolver um algoritmo que calcule e escreva o valor da multa a ser paga por esses proprietários considerando que:
 Os dados de cada imóvel: IDENTIFICACAO (de tipo LITERAL), VALOR DO IMPOSTO e NUMERO DE MESES EM ATRASO estão à disposição numa unidade de entrada;

As multas devem ser calculadas a partir do VALOR DO IMPOSTO e de acordo com a seguinte tabela (também à disposição numa unidade de entrada);

VALOR DO IMPOSTO	% POR MÊ EM ATRASO
Até R\$ 5.000,00	1
De R\$ 5.001,00 a R\$ 18.000,00	2
De R\$ 18.001,00 a R\$ 50.000,00	4
De R\$ 50.001,00 a R\$ 120.000,00	7
Acima de R\$ 120.000,00	10

O último registro lido, que não deve ser considerado, contém a identificação do imóvel igual a vazio;

Na saída deverão ser impressos: a identificação do imóvel, valor do imposto, meses em atraso e a multa a ser paga.

As estruturas de dados a serem adotadas para a solução do problema são:

TABELA

LIMITES		PERCENTUAL
DE	ATÉ	
0	5.000	1
5.001	18.000	2
18.001	50.000	4
50.0001	120.000	7

120.001		10
---------	--	----

Essa tabela pode ser representada por uma Variável Composta Homogênea de 5 linhas por 3 colunas. (3 colunas para DE, ATÉ e PERCENTUAL).

Para os dados de cada imóvel, será adotado o seguinte registro (variável composta heterogênea)

IMOVEL

IDENTIFICACAO	IMPOSTO	MESES_ATRASO
(LITERAL)	(NUMERICO)	(NUMERICO)

Resposta:

```

ALGORITMO EXEMPLO1
DECLARE tabela[1:5,1:3] NUMERICO
DECLARE imóvel REGISTRO (identificacao LITERAL, imposto NUMERICO, meses_atraso
NUMERICO)
DECLARE i, multa NUMERICO
LEIA tabela
INICIO
ENQUANTO (imovel.identificacao != "VAZIO")
FACA
    LEIA imovel
    SE (imovel.identificacao != "VAZIO") ENTAO
        i = 6;
        ENQUANTO ((imovel.imposto <= tabela[i,1]) OU (i = 1))
        FACA
            i = i - 1;
        FIM-ENQUANTO

        SE imóvel.imposto >= tabela[i,1] ENTAO
            multa = tabela[i,3] * (imóvel.imposto/100) * imóvel.meses_atraso;
        FIM-SE

    FIM-SE
    ESCREVER imovel.multa;
FIM-ENQUANTO
FIM ALGORITMO EXEMPLO1
    
```

10. Escreva o comando de abertura do arquivo declarado para o registro no seguinte formato abaixo, visto na aula anterior "Arquivo I", de modo a permitir a atualização do registro, isto é, a leitura, modificação dos campos e escrita do registro no mesmo arquivo:

numero_da_conta	nome_cliente	saldo	data_ultima_operacao

Resposta:

ABRIR agencia

11. Suponha-se um arquivo de registros de alunos de um colégio. Cada registro possui um campo para o número de matrícula e uma seqüência de outros campos com outras informações. Observe a figura abaixo que representa a organização deste arquivo seqüencialmente. Um algoritmo para a obtenção de um registro neste arquivo, por exemplo, procurando um registro com um determinado número de matrícula, deverá preocupar-se em pesquisar cada registro até encontrar, ou não, o registro desejado.

MATRÍCULA	OUTRAS INFORMAÇÕES	
820001	ZZZABDEF	00000
820007	KKKZDXZ	140300
820003	ACKYDYA	11111
820005	ZKD AAAA	99990
830008	ZZZ DEEX	1111
830001	KYZ ZZZZ	0010

Resposta:

```

ALGORITMO /* Pesquisa pelo registro cuja matrícula seja 830008, num arquivo seqüencial*/
/* Definição do arquivo de alunos */
DECLARE alunos ARQUIVO sequencial DE dados
DECLARE dados REGISTRO (matricula NUMERICO, outrosdados LITERAL)
/* Definição do arquivo para impressão */
DECLARE listagem ARQUIVO sequencial DE linha
DECLARE linha REGISTRO (componente LITERAL)
/* Abertura dos Arquivos */
ABRIR alunos LEITURA
ABRIR listagem ESCRITA
/* Pesquisa do Registro */
REPITA
    LER(alunos.dados);
    SE(alunos.FDA ou dados.matricula == 830008) ENTAO
        INTERROMPA;
    FIM SE
FIM REPITA
SE(alunos.FDA) ENTAO
    linha.componente = "REGISTRO NÃO ENCONTRADO";
    ESCREVER(listagem.linha);
SENÃO
    ESCREVER(listagem.dados);
FIM SE
/* Fechamento dos arquivos */
FECHAR alunos, listagem
FIM ALGORITMO
    
```

12. Crie a partir de um arquivo seqüencial, um arquivo de acesso direto, com registros contendo a placa, tipo e nome do proprietário de um veículo. Use como chave a placa do veículo.

Resposta:

```

ALGORITMO /* Criação de arquivo de acesso direto a partir de arquivo seqüencial*/
/* Definição das estruturas de dados */
    
```

```
DECLARE fonte ARQUIVO sequencial DE reg /* Arquivo seqüencial contendo os registros dos
veículos */
DECLARE veiculos ARQUIVO direto DE reg /* Arquivo de acesso direto com os registros dos
veículos */
DECLARE reg REGISTRO (placa, tipo, proprietario LITERAL)
/* Abertura dos Arquivos */
ABRIR fonte LEITURA
ABRIR veiculos ESCRITA
/* Criação do arquivo direto */
REPITA
    LER(fonte.reg);
    SE(fonte.FDA) ENTAO
        INTERROMPA;
    FIM SE
    ESCREVER ITEM [placa] veiculos.reg
FIM REPITA
/* Fechamento dos arquivos */
FECHAR fonte
FECHAR veiculos
FIM ALGORITMO
```

13. Fazer uma função que dê o resultado das 4 operações matemáticas básicas: soma, subtração, multiplicação e divisão. Fazer também um sub-rotina para comparar qual dos dois parâmetros é maior, ou se são iguais, e escreva a informação correta. Obs.: Cada operação só realiza o cálculo de dois valores por vez.

Resposta:

```
ALGORITMO
FUNCAO NUMERICO resultado (a, b, operacao)
    DECLARE a, b, operacao
    FACA CASO
        CASO (operacao == 1)
            resultado = a + b;
        CASO (operacao == 2)
            resultado = a - b;
        CASO (operacao == 3)
            resultado = a * b;
        CASO (operacao == 4)
            resultado = a / b;
    FIM-CASO
FIM-FUNCAO
SUBROTINA comparar (a, b)
    DECLARE a, b
    SE (a < b) ENTAO
        ESCREVER (“O primeiro valor é maior que o segundo!”)
    FIM-SE
    SE (a > b) ENTAO
        ESCREVER (“O primeiro valor é menor que o segundo!”)
    FIM-SE
    SE (a == b) ENTAO
```

ESCREVER (“O primeiro valor é igual que o segundo”)

FIM-SE

FIM-SUBROTINA

DECLARE a, b, operacao NUMERICO

ESCREVER (“Digite o primeiro valor: ”);

ESCREVER (“Digite o segundo valor: ”);

ESCREVER (“Digite a operação: 1 para soma ”);

ESCREVER (“Digite a operação: 2 para subtração ”);

ESCREVER (“Digite a operação: 3 para multiplicação ”);

ESCREVER (“Digite a operação: 4 para divisão ”);

LER (valor1, valor2, operacao)

ESCREVER resultado(valor1, valor2, operacao)

comparar(valor1, valor2)

FIM-ALGORITMO

14. Descreva o que o que o seguinte algoritmo faz:

ALGORITMO

DECLARE valor1, valor2, peso1, peso2, media_ponderada NUMERICO;

LEIA valor1, valor2, peso1, peso2;

$media_ponderada = ((valor1 * peso1) + (valor2 * peso2)) / (peso1 + peso2);$

ESCREVA media_ponderada;

FIM ALGORITMO

Resposta:

Primeiro declaramos as variáveis que iram armazenar as notas, os pesos, e também a média ponderada. Em seguida, lemos os valores das notas, e dos pesos, depois multiplicamos cada nota pelo peso correspondente, somamos esses valores multiplicados e dividimos pela soma dos pesos para obtermos o média ponderada de dois valores. Por fim, escrevemos a média ponderada como resultado do algoritmo.

15. Explique o que o código abaixo faz:

```
ALGORITIMO
DECLARE A, B, C NUMERICO;

LEIA A, B, C;

    SE ( A > B )ENTAO
        SE ( A > C ) ENTAO
            ESCREVA “O maior valor é: ”, A;
        SENAO
            ESCREVA “O maior valor é: ”, C;
        FIM SE
    SENAO
        SE ( B > C ) ENTAO
            ESCREVA “O maior valor é: ”, B;
        SENAO
            ESCREVA “O maior valor é: ”, C;
        FIM SE
    FIM SE
FIM ALGORITIMO
```

Primeiro declaramos as variáveis que iram armazenar os valores de A, B e C. Em seguida, lemos os valores e os comparamos para sabermos qual dele é o maior valor. Por fim, escrevemos qual dos três numéricos declarados tem maior valor.

16. Qual(is) dessa(s) estrutura(s) são consideradas Estruturas de Repetição Determinadas?

I –
PARA i = 0 **ATE** 10
FACA i = i + 1;

II –
x = 0;
PARA i = 1 **ATE** 100
FACA x = x + i;

III –
PARA i = 0
FACA i = i * 2;

IV –
PARA i == 0 **ATE** i == 10
FACA i = i + 1;

V –
x = 10;
PARA i = 10 **ATE** 1000

FACA $x = x + i$;

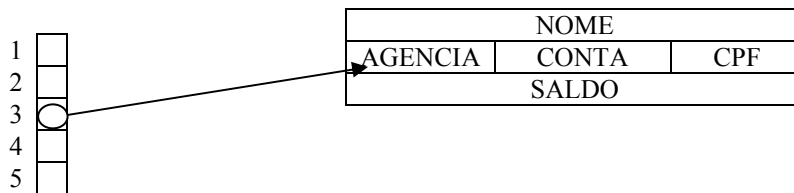
- a) I, II e III.
- b) II, III e V.
- c) I, II e V.
- d) I e II.
- e) I, II, IV e V.

Resposta: C

17. Qual das seguintes assertivas com trechos de código está correta a respeito da estrutura de Conjunto de Registros abaixo?

LISTA

CLIENTE



- I – DECLARE lista[1:5] cliente
 DECLARE cliente REGISTRO (nome LITERAL,
 agencia, conta, cpf, saldo NUMERICO)
- II – DECLARE lista REGISTRO (nome, rua LITERAL,
 numero, cep, saldo NUMERICO)
- III – DECLARE lista[1:5] cliente
 DECLARE cliente REGISTRO (nome, rua LITERAL,
 numero, cep, saldo NUMERICO)
- IV – DECLARE lista REGISTRO (nome LITERAL,
 agencia, conta, cpf, saldo NUMERICO)
- V – DECLARE contas[1:5] cliente
 DECLARE cliente REGISTRO (nome, rua LITERAL,
 numero, cep, saldo NUMERICO)

- a) II, III e V.
- b) II e III.
- c) I, III e V.
- d) I e IV.
- e) I, II e IV.

Resposta: D

18. Fazer um algoritmo que efetue um produto matricial. Seja A ($m \times n$) e B ($n \times m$) as matrizes-fatores, sendo $m \leq 40$ e $n \leq 70$. Deverão ser impressas as matrizes A, B e a matriz-produto obtida.

Resposta:

```

ALGORITMO
DECLARE a[40:70], b[70:40], produto[40:40], m, n NUMERICO
LER a[1,1],..., a[40,70],..., b[1,1], b[70,40]
INICIO
m = 1;
n = 1;
ENQUANTO (m <= 40)
FACA
    ENQUANTO (n <= 70)
    FACA
        produto[m,m] = produto[m,m] + (a[m,n] * b[n,m]);
        n = n + 1;
    FIM-ENQUANTO
    m = m + 1;
    n = 1;
FIM-ENQUANTO

m = 1;
n = 1;
ENQUANTO (m <= 40)
FACA
    ENQUANTO (n <= 70)
    FACA
        ESCREVER (a[m,n]);
        n = n + 1;
    FIM-ENQUANTO
    m = m + 1;
    n = 1;
FIM-ENQUANTO

m = 1;
n = 1;
ENQUANTO (m <= 40)
FACA
    ENQUANTO (n <= 70)
    FACA
        ESCREVER (b[n,m]);
        n = n + 1;
    FIM-ENQUANTO
    m = m + 1;
    n = 1;
FIM-ENQUANTO

m = 1;
n = 1;
ENQUANTO (m <= 40)
FACA
    ENQUANTO (n <= 70)
    FACA
        ESCREVER (produto[m,m]);
        n = n + 1;
    FIM-ENQUANTO
    m = m + 1;
    n = 1;
FIM-ENQUANTO
    
```

```

FACA
    ESCREVER (produto[m,m]);
    n = n + 1;
FIM-ENQUANTO
m = m + 1;
n = 1;
FIM-ENQUANTO
FIM-ALGORITMO
    
```

19. Escreva um algoritmo que represente o formulário digital de cadastramento de clientes de um sistema web real. Além disso, imprima as informações do cliente incluindo os rótulos identificadores do dado em questão.



Resposta:

```

ALGORITMO
DECLARE clientes REGISTRO (cod_cliente NUMERICO,
                           nome_empresa LITERAL,
                           nome_contato LITERAL,
                           cargo LITERAL,
                           endereco LITERAL,
                           cidade LITERAL,
                           regioao LITERAL,
                           cep LITERAL,
                           pais LITERAL,
                           telefone LITERAL,
                           fax LITERAL)

LER (clientes);
INICIO

ESCREVER ("Código do Cliente: ",clientes.cod_cliente);
ESCREVER ("Nome da Empresa: ",clientes.nome_empresa);
ESCREVER ("Nome do Contato: ",clientes.nome_contato);
ESCREVER ("Cargo: ",clientes.cargo);
ESCREVER ("Endereço: ",clientes.endereco);
ESCREVER ("Cidade: ",clientes.cidade);
ESCREVER ("Região: ",clientes.regiao);
ESCREVER ("CEP: ",clientes.cep);
ESCREVER ("País: ",clientes.pais);
    
```

ESCREVER (“Telefone: ”,clientes.telefone);

ESCREVER (“Fax: ”,clientes.fax);

FIM-ALGORITMO

20. Dada a tabela a seguir: (Itens de compras)

	ID	PRODUTO	PREÇO (R\$)
1	001000	Lâmpada	12
2	001050	Queijo	13
3	002000	Espuma de barbear	10
4	002050	Uisque	35
5	003000	Vinho	30
6	003050	Feijão	4
...
99	198050	Arroz	2
100	199000	Barbeador	5

Escrever um algoritmo que, dados 100 CODIGOS DE PRODUTOS (fornecidos em 100 linhas), emite o ID, o PRODUTO e o PREÇO individual dos produtos. A tabela acima também deve ser lida.

Resposta:

```

ALGORITMO
DECLARE tabela[1:100] REGISTRO (nome LITERAL,
                               id, preco NUMERICO)
DECLARE idDesejado, i, k NUMERICO
LER tabela[1]...tabela[100]
INICIO
k = 1
ENQUANTO k <= 100 // 100 id's de produtos fornecidos em linha.
FACA
    LER idDesejado
    i = 1;
    ENQUANTO ((tabela[i].id != idDesejado) OU (i != 101))
    FACA
        i = i + 1;
    FIM-ENQUANTO
    SE (tabela[k].id == idDesejado) ENTAO
        ESCREVER (idDesjado, tabela[k].produto, tabela[k].preco);
    SENA0
        ESCREVER (“ID INVÁLIDO!”);
    FIM-SE
    k = k + 1;
FIM-ENQUANTO
FIM ALGORITMO
    
```

21. Uma pessoa aplicou seu capital a juros e deseja saber, trimestralmente, a posição de seu investimento “c” inicial. Chamando de “i” a taxa de juros do trimestre, escrever uma tabela que dê para cada trimestre o rendimento auferido e o saldo acumulado durante um período de “x” anos, supondo-se que nenhuma retirada tenha sido feita.

- Observações:
- Os valores de c, i e x estão disponíveis numa unidade de entrada.
- O cálculo deve ser feito por uma função.
- A fórmula para capitalização composta é dada por:

$$M_n = c(1 + i)^n$$

Onde:

M_n montante após terem decorridos n trimestres;
 c capital inicial investido;
 i taxa de juros;
 n número de períodos decorridos (trimestres)

Resposta:

```

ALGORITMO MONTANTE
FUNCAO NUMERICO montante(c, i, n)
    DECLARE c, i, n NUMERICO;
    SE (n > 0) ENTAO
        montante = c * (EXP( 1 + i ), n);
    SENAO
        montante = c;
    FIM-SE

FIM-FUNCAO
DECLARE c, i, n NUMERICO
LER (c, i, n); // lida da unidade de entrada.
FAÇA
    ESCREVER (montante (c, i, n));
    
```

FIM-ALGORITMO MONTANTE

Perceba que usamos uma função pré-definida chamada EXP para representar a operação exponencial necessária.

22. Dada a ficha de avaliação abaixo, faça um algoritmo que verifique se algum item obrigatório não foi preenchido, caso algum campo obrigatório da ficha não tenha sido preenchido, deve-se indicar através de uma mensagem que aquele campo era obrigatório e não foi preenchido. Ao final deve-se imprimir todos os campos da ficha com a respectiva indicação do que o campo representa (Ex. Nome: Alexandre Monteiro).

Obs.: Os campos obrigatórios possuem um asterisco (*) ao lado.

AVALIACAO

*ALUNO
ENDERECO
CONTATO
MEDIDAS

ENDERECO

*RUA	*NUMERO	COMPLEMENTO	*CEP	*BAIRRO	*UF	CONTATO
------	---------	-------------	------	---------	-----	---------

CONTATO

TELEFONE	*CELULAR	EMAIL
----------	----------	-------

MEDIDAS

*ALTURA	*PESO	BRACO	PERNA
---------	-------	-------	-------

TELEFONE

*PESSOAL	COMERCIAL
----------	-----------

EMAIL

*PESSOAL	COMERCIAL
----------	-----------

BRACO

ESQUERDO	DIREITO
----------	---------

PERNA

ESQUERDA	DIREITA
----------	---------

Resposta:

ALGORITMO

DECLARE Avaliacao REGISTRO (aluno LITERAL,

endereço Endereço,
medidas Medidas)

DECLARE Endereço REGISTRO (rua LITERAL,

numero NUMERICO,
complemento LITERAL,

cep NUMERICO,

bairro LITERAL,

uf LITERAL,

contato Contato)

DECLARE Contato REGISTRO (telefone Telefone,

celular LITERAL,

email Email)

DECLARE Telefone REGISTRO (pessoal LITERAL,

comercial LITERAL)

DECLARE Email REGISTRO (pessoal LITERAL,

comercial LITERAL)

DECLARE Medidas REGISTRO (altura NUMERICO,

peso NUMERICO,

braco Braco,

perna Perna)

DECLARE Braco REGISTRO (esquerda NUMERICO,

direita NUMERICO)

DECLARE Perna REGISTRO (esquerda NUMERICO,

direita NUMERICO)

LER Avaliação, Endereço, Contato, Telefone, Email, Medidas, Braco, Perna;

INICIO

SE (Avaliacao.aluno != ``) ENTAO

ESCREVER ("Aluno: ", Avaliacao.aluno);

SENAO

ESCREVER ("O preenchimento do campo Aluno é obrigatório.");

FIM-SE

SE (Avaliacao.endereco.rua != ``) ENTAO

ESCREVER ("Rua: ", Avaliacao.endereco.rua);

SENAO

ESCREVER ("O preenchimento do campo Rua é obrigatório.");

FIM-SE

```
SE (Avaliacao.endereco.numero != " ") ENTAO
    ESCREVER ("Endereço: ", Avaliacao.endereco.numero);
SENAO
    ESCREVER ("O preenchimento do campo Endereço é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.cep != " ") ENTAO
    ESCREVER ("CEP: ", Avaliacao.endereco.cep);
SENAO
    ESCREVER ("O preenchimento do campo CEP é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.bairro != " ") ENTAO
    ESCREVER ("Bairro: ", Avaliacao.endereco.bairro);
SENAO
    ESCREVER ("O preenchimento do campo Bairro é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.uf != " ") ENTAO
    ESCREVER ("UF: ", Avaliacao.endereco.uf);
SENAO
    ESCREVER ("O preenchimento do campo UF é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.contato.celular != " ") ENTAO
    ESCREVER ("Celular: ", Avaliacao.endereco.contato.celular);
SENAO
    ESCREVER ("O preenchimento do campo Celular é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.contato.telefone.pessoal != " ") ENTAO
    ESCREVER ("Telefone Pessoal: ", Avaliacao.endereco.contato.telefone.pessoal);
SENAO
    ESCREVER ("O preenchimento do campo Telefone Pessoal é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.contato.email.pessoal != " ") ENTAO
    ESCREVER ("E-mail Pessoal: ", Avaliacao.endereco.contato.email.pessoal);
SENAO
    ESCREVER ("O preenchimento do campo E-mail Pessoal é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.medidas.altura != " ") ENTAO
    ESCREVER ("Altura: ", Avaliacao.endereco.medidas.altura);
SENAO
    ESCREVER ("O preenchimento do campo Altura é obrigatório.");
FIM-SE

SE (Avaliacao.endereco.medidas.peso != " ") ENTAO
    ESCREVER ("Peso: ", Avaliacao.endereco.medidas.peso);
SENAO
    ESCREVER ("O preenchimento do campo Peso é obrigatório.");
FIM-SE
FIM-ALGORITMO
```

23. Vamos criar um outro algoritmo para uma agenda virtual, que durante sua execução, armazenará: Nome, Telefone, endereço e e-mail também de 100 pessoas. Desta vez você deve fazer uso de Variáveis Compostas Multidimensionais, as famosas matrizes.

Resposta:

ALGORITMO

```

    DECLARE agenda [1:100; 1:4] LITERAL
    DECLARE indice1, indice2 NUMERICO
INICIO
    //Início do algoritmo

    //Inicializando indice

    indice1 = 1;

    //Inicializando vetores
    PARA i = 1 ATE 100 FACA
        PARA j = 1 ATE 4 FACA
            agenda [i,j] = "";
        FIM-PARA//Obtendo dados do usuário
    FIM-PARA//Obtendo dados do usuário

    j = 1;

    PARA i = 1 ATE 100 FACA
        ESCREVER('Digite o nome:')
        LER(agenda [i,j])
        ESCREVER ('Digite o Endereco')
        LER(agenda [i,j+1])
        ESCREVER ('Digite o Telefone')
        LER(agenda [i,j+2])
        ESCREVER ('Digite o e-mail')
        LER(agenda [i,j+3])

    FIM-PARA

    //Listando todos os registros em sequência

    PARA i = 1 ATE 100 FACA
        PARA j = 1 ATE 4 FACA
            ESCREVER (agenda [i,j])
            ESCREVER (agenda [i,j])
            ESCREVER (agenda [i,j])
            ESCREVER (agenda [i,j])
        FIM-PARA
    FIM-PARA
FIM-ALGORITMO
```

24. Uma empresa decide dar um aumento de 30% aos funcionários com salários inferiores a R\$ 500,00. Faça um algoritmo que receba o salário do funcionário e mostre o valor do salário reajustado ou uma mensagem caso o funcionário não tenha direito ao aumento.

Resposta:

ALGORITMO 2

DECLARE salario, salario2 NUMERICO

ESCREVA "Insira o valor do salário"

LEIA salario

SE salario < 500 ENTAO

salario2<- salario+(salario*30/100)

ESCREVA "Você vai receber ", salario2." reais."

SENAO

ESCREVA "Você não tem direito a reajuste de salário vai receber ", salario, ." reais."

FIM_SENAO

FIM_ALGORITMO

25. Faça um algoritmo que receba a altura e o sexo de uma pessoa, calcule e mostre o seu peso ideal, utilizando as seguintes fórmulas:

Resposta

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

ALGORITMO 6

DECLARE peso, calculo NUMERICO

sexo LITERAL

ESCREVA "Insira seu peso"

LEIA peso

Escreva "Você é homem ou mulher?"

LEIA sexo

SE sexo="mulher" ENTAO

calculo<- $(72.7*peso)-58$

SENAO

calculo<- $(62.1*peso)-44.7$

FIM_SENAO

ESCREVA "Seu peso ideal é de ",calculo

FIM_ALGORITMO

26. Fazer um algoritmo que dadas as dimensões de um retângulo, calcule a sua área e escreva na tela.

Resposta:

```
Algoritmo
Declare
//Inicio do algoritmo

//Obtendo os dados do usuário
Escreva('Digite o valor da base do retângulo: ')
Leia(Base)
Escreva('Digite o valor da altura do retângulo: ')
Leia(Altura)

//Calculando a área
Area <- Base * altura

Escreva('A área é: ',Area)

FimAlgoritmo
```

27. Fazer um algoritmo para calcular a potência N de um numero X. Ambos serão digitados pelo usuário. E escreva na tela.

Resposta:

```
Algoritmo
Declare
//Inicio do algoritmo
//Obtendo os dados do usuário
Escreva('Digite a base:')
Leia(Base)
Escreva('Digite o expoente:')
Leia(Expoente)

//Inicializando variável resultado
Resultado<- Base
//Calculando
Se Expoente = 0
então
Resultado<-1
senão
Se base = 0
então Resultado <- 0
senão PARA i<- 1 até (Expoente -1)
```

```
FAÇA
Resultado<- Resultado * Expoente
FimPARA
FimSe
FimSe
Escreva('O resultado é: ',Resultado)
FimAlgoritmo
```

28. Dada a expressão:

$$\frac{2 \cdot 10 \cdot 2 \cdot 20 \cdot 2 \cdot 30 \cdot 2 \cdot 40}{1+2+3+4+5+6+7+8}$$

Fazer um algoritmo para mostrar seu resultado.

Resposta

```
Algoritmo
Declare
//Inicio do algoritmo

j<-10
Somar

<- 0
  PARA i <- 1 até 8 Faça
  Calcular<- Calcular + (2*j)
  Somar<- Somar + i
  j<- j + 10
  FimPARA
Resultado<- Calcular / Somar

  Escreva('O resultado é: ', Resultado)

FimAlgoritmo
```

29. Fazer um algoritmo que dados dois números, calcule a razão existente entre eles e monte uma PA de 10 termos e escreva na tela.

Resposta

O calculo da razão é o segundo termo menos o primeiro.

```
Algoritmo
Declare
//Inicio do algoritmo

//Obtendo os dados do usuário
Escreva('Digite o primeiro número')
Leia(PTermo)
```

```
Escreva('Digite o segundo número')
Leia(STermo)

//Calculando a razão
Razao<- STermo - PTermo

Escreva('Os dez primeiros termos desta PA são:')
Escreva(PTermo)
Escreva(STermo)
TAux<-STermo
Para i<-3 até 8 Faça
ProxTermo<- TAux + Razao
    Escreva(ProxTermo)
    TAux<- ProxTermo
FimPara
FimAlgoritmo
```

30. Tendo-se a seguinte expressão matemática:

Soma = X+Y
Mult= X*Y
SubTotal = Mult / Soma
Expressao = [(X*15)+(Y*25)+(X*35)+(Y*45)+(X*55)] / 15
Total= SubTotal + Expressao

Fazer um algoritmo que:

1. Leia os valores X e Y
2. Calcule a soma e a multiplicacao destes valores e armazene em SubTotal
3. Calcule expressao e armazene a última operação de soma na variável Total

Ao final do programa, listar na tela os valores, da seguinte maneira:

Resposta:

```
Valores digitados pelo usuário "VALOR DE X"; "VALOR DE Y"
Subtotal : "VALOR DO SUBTOTAL";
Expressão: "VALOR DA EXPRESSÃO";
Total:"Valor total";
```

```
Algoritmo
Declare
//Inicio do algoritmo
//Obtendo os números do usuário
Escreva('Digite o primeiro número')
leia(X)
Escreva('Digite o Segundo número')
Leia(Y)

Soma <- X + Y
```

```

Mult <- X * Y
SubTotal <- Mult / Soma

Expressao <- 0
i<-15
j<- 1

Enquanto i <= 55 faça
Par<- J mod 2
  Se (par = 0)
    Então Expressao <- Expressao + (X*i)
    Senão Expressao <- Expressao + (Y*i)
  FimSe
i<- i + 10
  Fim Enquanto
Expressao<-Expressao/15

Total <- SubTotal + Expressao
Escreva('Exercicio1')
Escreva("") //isto escreve uma linha vazia
Escreva("") //isto escreve uma linha vazia
Escreva('Os valores digitados pelo usuário
são:X=',X,' Y= ',Y)
Escreva('SubTotal= ',SubTotal)
Escreva('Expressão= ',Expressao)
Escreva('O valor total dos calculos é:', Total)

FimAlgoritmo
    
```

31. Um trabalhador recebeu seu salário e o depositou em sua conta corrente bancária. Esse trabalhador emitiu dois cheques e agora deseja saber seu saldo atual. Sabe-se que cada operação bancária de retirada paga CPMF de 0,38% e o saldo inicial da conta está zerado.

Resposta:

Algoritmo Saldo

Declare

real sal, chq1, chq2, cpmf1, cpmf2, saldo

Início

Ler(sal);

Ler(chq1);

Ler(chq2);

cpmf1←chq1 x 0.38 / 100;

cpmf2←chq2 x 0.38 / 100;

```
saldo ← sal – (chq1 + chq2 + cpmf1 + cpmf2);
```

```
Escrever("O saldo total:" + saldo);
```

Fim Algoritmo Saldo

32. Dados dois números positivos m e n encontre seu maior divisor comum, isto é o maior inteiro positivo que divide tanto m como n . Assuma que m é sempre maior que n , e n diferente de zero.

Resposta:

Algoritmo

início

ler m, n ;

$r = m \% n$; // resto da divisão de m por n

enquanto $r \neq 0$ faça

$m = n$;

$n = r$;

$r = m \% n$;

fim enquanto

imprimir n ;

fim Algoritmo

33. Fazer um Algoritmo que efetue o Produto de $V1$ com $V2$, e que imprima logo depois este resultado.

Resposta:

Algoritmo (Produto)

Declare $V1, V2, Produto$: inteiro;

Início

Ler $V1, V2$

$Produto \leftarrow V1 * V2$;

escreva $Produto$

Fim Algoritmo

34. Fazer um Algoritmo que utiliza a estrutura Repita, que faça uma soma acumulada de 5 números, e logo depois imprima esta Soma.

Resposta:

Algoritmo "Soma Acumulada"

declare Soma, V, K : inteiro

Início

S <- 0

 K <- 0

 Repita

 Ler V

 Soma <- Soma+V

 K <- K+1

 até que (K=5)

 escreva Soma

Fim Algoritmo

35. Fazer um Algoritmo, que leia uma matriz quadrada de dimensão 2, e que depois imprima os valores.

Resposta:

Algoritmo "Leitura_de_Matriz"

Declare

 M : conjunto [1..2, 1..2] de real

 I, J : inteiro

Início

{ Leitura da Matriz }

Para I de 1 até 2 faça

 Para J de 1 até 2 faça

 Ler M [I,J]

Fim Para

Fim Para

{ Imprimi a Matriz lida}

Para I de 1 até 2 faça

Para J de 1 até 2 faça

escreva M [I,J]

Fim Para

Fim Para

Fim Algoritmo

36. Faça um programa que a partir da leitura das medidas dos lados de um retângulo (comprimento (C) e largura (L), lidos do teclado, calcule e imprima a área e o perímetro do retângulo):

Resposta:

Formula: $\text{área} = C \cdot L$ /// $\text{Perímetro} = 2 \cdot C + 2 \cdot L$

Resposta:

Algoritmo

Inicio

Ler(Comprimento)

Ler(Largura)

Área=comprimento+largura

Escrever(área)

Perímetro=2 . comprimento + 2 . largura

Escrever(perímetro)

Fim

Fim Algoritmo

37. Dados dois números inteiros x e y cada um contendo 4 dígitos x ($x_1x_2x_3x_4$) e y ($y_1y_2y_3y_4$). Faça um programa para gerar um terceiro número de 4 dígitos baseado na multiplicação de cada dígito entre x e y , guardando o último dígito da multiplicação como valor para o terceiro número.

Resposta:

Algoritmo

início

```
ler(numero1)
ler(numero2)
```

```
x1=(numero1/1000)
x2=(numero1/100)!10
x3=(numero1/10)!10
x4=(numero1/1)!10
```

```
y1=(numero2/1000)
y2=(numero2/100)!10
y3=(numero2/10)!10
y4=(numero2/1)!10
```

```
n1=(x1*y1)!10
n2=(x2*y2)!10
n3=(x3*y3)!10
n4=(x4*y4)!10
```

```
se(n1<9)
  início
    n1=n1!10
  fim
```

```
se(n2<9)
  início
    n2=n2!10
  fim
```

```
se(n3<9)
  início
    n3=n3!10
  fim
```

```
se(n4<9)
  início
    n4=n4!10
  fim
```

```
n=(n1*1000)+(n2*100)+(n3*10)+(n4*1)
```

```
escrever(n)
```

fim

Fim Algoritmo

38. Faça um programa que leia um número e informe se ele é negativo, positivo ou nulo.

Resposta:

```
Algoritmo
início
ler(numero)
se(numero>0)
início
escrever(Esse número é positivo)
fim
se(numero==0)
início
escrever(Esse número é nulo)
fim
se(numero<0)
início
escrever(Esse número é negativo)
fim se
fim

Fim Algoritmo
```

39. Escreva um algoritmo para calcular o consumo médio de um automóvel (medido em Km/l), dado que são conhecidos a distância total percorrida e o volume de combustível consumido para percorrê-la (medido em litros).

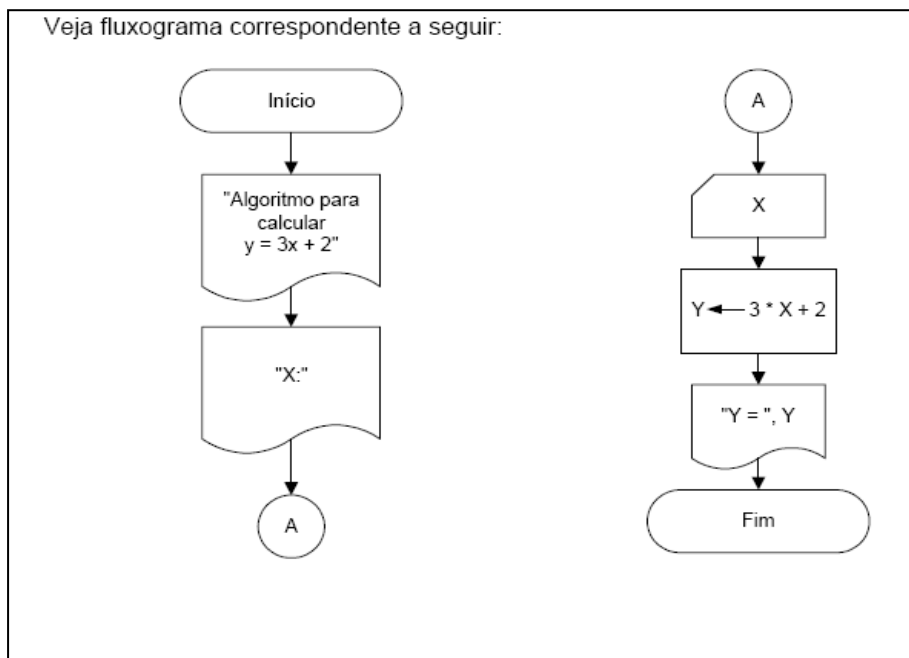
Resposta:

```
Algoritmo CONSUMO_MEDIO
Var CM, DIST, VOL : real

Início
Escreva “Algoritmo para calcular o consumo”
Escreva “Distância total percorrida (Km): ”
Leia DIST
Escreva “Volume de combustível gasto (L): ”
Leia VOL
CM ← DIST / VOL
Escreva “Consumo médio =”, CM, “Km/l”
```

Fim algoritmo CONSUMO_MEDIO

40. Transcreva o fluxograma abaixo para código de algoritmo:



Resposta:

Algoritmo FUNCAO_DE_X

Var X, Y : real

Início

Escreva “Algoritmo para calcular

y = 3x +2”

Escreva “X: ”

Leia X

$Y \leftarrow 3 * X + 2$

Escreva “Y =”, Y

Fim.

41. Dado o seguinte algoritmo

ALGORITMO

```
DECLARE z NUMERICO;
FUNCAO NUMERICO op (x, y)

    DECLARE x, y NUMERICO;
    SE (z == 0) ENTAO
        op = x + y;
    SENAO
        op = x - y;
    FIM-SE

FIM-FUNCAO

DECLARE x, y NUMERICO;
DECLARE res NUMERICO;
LER (z, y, x);
res = op(y, x);
ESCREVER ("O resultado de op sobre x e y é ", res);
```

FIM-ALGORITMO

Responda:

- a) O que será mostrado para o usuário ao final do algoritmo se ele entrar com os seguintes valores em resposta ao comando de entrada:

$$z = 1 / x = 12 / y = 3;$$

Resposta: [O resultado de op sobre x e y é 9].

- b) Por que o valor de Z é conhecido (e pode ser testado) dentro da função OP?

Resposta: Porque "z" é uma variável global (objeto global) o valor de "z" pode ser utilizado em qualquer módulo do programa.

42. Qual desses exemplos citados abaixo pode ser considerado (transformado) no algoritmo em uma variável composta homogênea multidimensional.

- a – Tabelas.
- b – Cubos.
- c – Registros.
- d – Vetores.
- e – Arquivos.

Resposta: D

43. Elabore um algoritmo que leia uma seqüência de n números reais ($n \leq 100$) e imprima-os em ordem inversa.

Resposta

Algoritmo Reverso

DECLARE V : conjunto[100] de real

CONT, N : inteiro

Início

Escreva “Algoritmo Reverso”

Escreva “Nº de Elementos = “

Leia N

Para CONT de 1 até N faça

Escreva “Elemento[“, CONT, “] =”

Leia V[CONT]

Fim_para

Escreva “Números na Ordem Inversa”

Para CONT de 1 até N faça

Escreva V[N – CONT +1]

Fim_para

Fim algoritmo

44. Elabore um algoritmo para realizar a soma de duas matrizes A e B de números reais e de tamanho $m \times n$ ($m \leq 100$ e $n \leq 100$). É necessário ler os números e armazená-los nas matrizes.

Resposta

Algoritmo PESQ_SEQ

DECLARE VALOR : literal[30]

NOMES : conjunto[100] de literal[30]

I : inteiro

ACHOU : lógico

Início

Para I de 1 até 100 faça

 Leia NOMES[I]

 Fim_para

 Leia VALOR

 I \leftarrow 1

 ACHOU \leftarrow .F.

 Enquanto I \leq 100 .E. .NÃO. ACHOU faça

 Se NOMES[I] = VALOR

 então

 ACHOU \leftarrow .V.

 senão

 I \leftarrow I + 1

 Fim_se

 Fim_enquanto

 Se ACHOU

 então

 Escreva VALOR, “ foi encontrado”

 senão

 Escreva VALOR, “ não foi encontrado”

 Fim_se

Fim algoritmo

45. Posso Entrar na Boate? Enunciado: Faça um programa que peça o ano de nascimento de uma pessoa, e diga, se for maior de idade, que pode entrar na boate, e se não for, não pode.

Algoritmo

Declare AnoAtual, AnoNascimento: Numerico

Escreva "Checagem de Idade da Boate Noites Cariocas"

Escreva "Qual é o ano atual?"

Leia AnoAtual

Escreva "Em que ano você nasceu?"

Leia AnoNascimento

Se (AnoAtual - AnoNascimento) \geq 18 entao

 Escreva "Você pode entrar na boate."

senao

 Escreva "Infelizmente, você não pode entrar."

fim se

Fim algoritmo

46. Dois carros percorreram diferentes distâncias em diferentes tempos. Sabendo que a velocidade média é a razão entre a distância percorrida e o tempo levado para percorrê-la, faça um programa que leias as distâncias que cada carro percorreu e o tempo que cada um levou, e indique o carro que teve maior velocidade média.

// Distância percorrida, tempo gasto e velocidade media de cada carro

Declare Distancia1, Distancia2, Tempo1, Tempo2, VelMed1, VelMed2 :

Numerico

// Leitura dos dados

Escreva "Distância percorrida pelo Carro 1:"

Leia Distancia1

Escreva "Tempo gasto pelo Carro 1:"

Leia Tempo1

Escreva "Distância percorrida pelo Carro 2:"

Leia Distancia2

Escreva "Tempo gasto pelo Carro 2:"

Leia Tempo2

// Cálculo e exibição das velocidades médias

VelMed1 \leftarrow Distancia1 / Tempo1

VelMed2 \leftarrow Distancia2 / Tempo2

Escreva "Velocidade média do Carro 1:", VelMed1

Escreva "Velocidade média do Carro 2:", VelMed2

// Resultado

Se VelMed1 > VelMed2 entao

 Escreva "O Carro 1 teve maior velocidade média."

Senao

 Escreva "O Carro 2 teve maior velocidade média."

Fimse

Você deve estar se perguntando: "E se os dois carros tiverem a mesma velocidade média?" Neste caso, nosso programa vai dar uma informação errada, dizendo que o Carro 2 foi mais rápido! Você consegue ver por quê?

Exercícios Propostos

1. Dado um número inteiro de três algarismos, escrever um algoritmo que receba o número e inverta a ordem de seus algarismos. O novo número deve ser exibido ao final.
2. Uma empresa decidiu dar uma gratificação de Natal a seus funcionários, baseada no número de horas extras e no número de horas que o empregado faltou ao trabalho. O valor do prêmio é obtido pela consulta a tabela abaixo, em que H é o número de horas-extras subtraído de dois terços do número de horas-faltas. Escreva um algoritmo que calcule este prêmio para um funcionário.

H	Prêmio
(40, 100]	R\$ 100,00
(30, 40]	R\$ 80,00
(20, 30]	R\$ 60,00
(0, 20]	R\$ 40,00

3. Três números formam um triângulo se o maior dentre eles for menor que a soma dos outros dois. Escreva um algoritmo que receba 3 números inteiros e diga se eles formam um triângulo. Assuma que o usuário digitará os números em ordem decrescente. A seguir, escreva este mesmo algoritmo assumindo que os números podem ser digitados em qualquer ordem.
4. Num frigorífico existem 90 bois. Cada boi traz preso em seu pescoço um cartão contendo seu número de identificação e seu peso. Faça um algoritmo que escreva o número e o peso do boi mais gordo e do boi mais magro (supondo que não haja empates).
5. O Departamento de Trânsito da Pernambuco compilou dados de acidentes de trânsito no Estado no último ano. Para cada motorista envolvido num acidente, os seguintes dados foram registrados: ano de nascimento do motorista, sexo ('M' ou 'F'), código de registro (1 para Pernambuco e 0 para outros Estados). Escrever um algoritmo para ler este conjunto de dados para vários motoristas e imprimir as seguintes estatísticas:
 - a) Percentagem de motoristas com menos de 25 anos
 - b) Percentagem de mulheres
 - c) Percentagem de motoristas maiores de 18 anos, mas menores de 25 anos
 - d) Percentagem de motoristas com registro feito fora de Pernambuco
6. Um ano de nascimento igual a 0 indica o fim dos dados para o algoritmo.
7. Faça um algoritmo que leia a idade de uma pessoa expressa em anos, meses e dias e mostre-a expressa apenas em dias.

8. Faça um algoritmo que leia a idade de uma pessoa expressa em dias e mostre-a expressa em anos, meses e dias.
9. Faça um algoritmo que leia as 3 notas de um aluno e calcule a média final deste aluno. Considerar que a média é ponderada e que o peso das notas é: 2,3 e 5, respectivamente.
10. Faça um algoritmo que leia o tempo de duração de um evento em uma fábrica expressa em segundos e mostre-o expresso em horas, minutos e segundos.
11. Calcule a média aritmética das 3 notas de um aluno e mostre, além do valor da média, uma mensagem de "Aprovado", caso a média seja igual ou superior a 6, ou a mensagem "reprovado", caso contrário.
12. Elaborar um algoritmo que lê 3 valores a,b,c e os escreve. A seguir, encontre o maior dos 3 valores e o escreva com a mensagem : "É o maior ".
13. Elabore um algoritmo que dada a idade de um nadador classifica-o em uma das seguintes categorias:
 - infantil A = 5 - 7 anos
 - infantil B = 8-10 anos
 - juvenil A = 11-13 anos
 - juvenil B = 14-17 anos
 - adulto = maiores de 18 anos
14. Faça um algoritmo que leia um n° inteiro e mostre uma mensagem indicando se este número é par ou ímpar, e se é positivo ou negativo.
15. Tendo como dados de entrada a altura e o sexo de uma pessoa (?M? masculino e ?F? feminino), construa um algoritmo que calcule seu peso ideal, utilizando as seguintes fórmulas:
 - para homens: $(72.7 * h) - 58$
 - para mulheres: $(62.1 * h) - 44.7$
16. Um banco concederá um crédito especial aos seus clientes, variável com o saldo médio no último ano. Faça um algoritmo que leia o saldo médio de um cliente e calcule o valor do crédito de acordo com a tabela abaixo. Mostre uma mensagem informando o saldo médio e o valor do crédito. (use o comando caso-de e não faça repetições)

Saldo médio	Percentual
de 0 a 200	nenhum crédito
de 201 a 400	20% do valor do saldo médio
de 401 a 600	30% do valor do saldo médio
acima de 601	40% do valor do saldo médio
17. Uma empresa concederá um aumento de salário aos seus funcionários, variável de acordo com o cargo, conforme a tabela abaixo. Faça um algoritmo que leia o salário e

o cargo de um funcionário e calcule o novo salário. Se o cargo do funcionário não estiver na tabela, ele deverá, então, receber 40% de aumento. Mostre o salário antigo, o novo salário e a diferença.

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

18. Escrever um algoritmo que lê a hora de início e hora de término de um jogo, ambas subdivididas em dois valores distintos : horas e minutos. Calcular e escrever a duração do jogo, também em horas e minutos, considerando que o tempo máximo de duração de um jogo é de 24 horas e que o jogo pode iniciar em um dia e terminar no dia seguinte.
19. O departamento que controla o índice de poluição do meio ambiente mantém 3 grupos de indústrias que são altamente poluentes do meio ambiente. O índice de poluição aceitável varia de 0,05 até 0,25. Se o índice sobe para 0,3 as indústrias do 1o grupo são intimadas a suspenderem suas atividades, se o índice cresce para 0,4 as do 1o e 2o grupo são intimadas a suspenderem suas atividades e se o índice atingir 0,5 todos os 3 grupos devem ser notificados a paralisarem suas atividades. Escrever um algoritmo que lê o índice de poluição medido e emite a notificação adequada aos diferentes grupos de empresas.
20. Faça um algoritmo que mostre os 100 primeiros números pares.
21. Faça um algoritmo que mostre todos os múltiplos de 5 no intervalo de 1 ate 5235.
22. A prefeitura de Rio Branco realizou uma pesquisa entre vários habitantes da cidade, coletando dados sobre o rendimento familiar e o número de filhos menores de cada família. A prefeitura deseja saber:
A média dos rendimentos da população;
A média do número dos filhos;
O percentual de famílias com rendimento igual ou superior a \$100 dólares.
23. Escreva um algoritmo que lê um vetor A(10) e escreva a posição de cada elemento igual a 10 deste vetor.
24. Faça um algoritmo que leia uma matriz 5x5, some todos os elementos da matriz, depois onde a linha for igual a coluna coloque o valor 5 dentro da matriz e depois mostre a nova soma.
25. Faça um algoritmo que leia uma temperatura dada na escala Celsius (C) e imprimir a equivalente em Fahrenheit (F) (Fórmula de conversão: $F = 9/5 * C + 32$);
26. Faça um algoritmo que leia uma quantidade de chuva dada em polegadas e imprimir a equivalente em milímetros (25,4 mm = 1 polegada).

27. Faça um algoritmo para ler os comprimentos dos três lados de um triângulo (L1, L2 e L3) e calcular a área do triângulo de acordo com a fórmula:

$$\text{área} = T (T - L1) (T - L2) (T - L3)$$
$$\text{onde } T = (L1 + L2 + L3) / 2$$

28. Uma companhia de carros paga a seus empregados um salário de R\$ 500,00 por mês mais uma comissão de R\$ 50,00 para cada carro vendido e mais 5% do valor da venda. Todo mês a companhia prepara os seguintes dados para cada vendedor: nome, número de carros vendidos e o valor total das vendas. Elabore um algoritmo para calcular e imprimir o salário do vendedor num dado mês.
29. Faça um algoritmo que leia o nome e o salário bruto de um funcionário e calcule o salário líquido. Sabendo que o imposto a ser descontado é de 5% sobre o salário bruto, calcule o salário líquido. O algoritmo deve escrever o nome do funcionário, o salário bruto, o valor do desconto e o salário líquido.
30. O preço de um automóvel é calculado pela soma do preço de fábrica com o preço dos impostos (45% do preço de fábrica) e a percentagem do revendedor (28% do preço de fábrica). Faça um algoritmo que leia o nome do automóvel e o preço de fábrica e, calcule e imprima o nome do automóvel e o preço final.
31. Desenvolva um algoritmo que leia um número N, o primeiro termo A1 e a razão q de uma Progressão Geométrica (PG), calcule e imprima o N-ésimo termo desta PG.
32. Escreva um algoritmo para escrever a palavra PROGRAMACAO 5 vezes utilizando uma estrutura de repetição e um contador.
33. Escreva um algoritmo para ler um valor N (validar para aceitar apenas valores positivos) e imprimir os N primeiros números inteiros.
34. Ler 2 valores A e B. Se A for igual a B devem ser lidos novos valores para A e B. Se A for menor que B calcular e imprimir a soma dos números ímpares existentes entre A(inclusive) e B(inclusive). Se A for maior que B calcular e imprimir a média aritmética dos múltiplos de 3 existentes entre A(inclusive) e B(inclusive).
- OBS: Considere que só serão informados valores inteiros positivos.
35. Suponha que exista um prédio de 20 andares onde existam três elevadores, identificados pelos números 1, 2 e 3. Para otimizar o sistema de controle dos elevadores, foi realizado um levantamento no qual cada usuário respondia qual o elevador que utilizava com mais frequência. Escreva um algoritmo que leia o número de usuários do prédio, leia as respostas de cada usuário calcule e imprima qual o elevador mais freqüentado.
36. Ler um número indeterminado de dados, contendo cada um o peso de um indivíduo. O último dado que não entrará nos cálculos, contém um valor negativo. Calcular e imprimir:

-A média aritmética das pessoas que possuem mais de 60 Kg.

- O peso do mais pesado entre aqueles que possuem menos de 60 Kg.
37. Faça um algoritmo que calcule o faturamento de um cinema a cada sessão. Devemos considerar que os menores de 18 anos pagam meia, devido à carteirinha de estudante e que os maiores de 65 anos também pagam meia, devido à carteirinha de aposentado. O preço normal do ingresso irá variar conforme o dia da semana, portanto deve ser solicitada estas informações. A quantidade de pessoas no cinema irá variar a cada sessão, portanto deve haver esta informação também.

Além disto, o algoritmo pode ser executado mais de uma vez, ou seja, deve-se verificar ao final do cálculo de uma sessão se o usuário deseja verificar o faturamento de outra sessão.
 38. Escreva Algoritmo que apresente todos fatoriais cujo resultado seja inferior a um dado valor que é lido pelo teclado.
 39. Escreva um Algoritmo que apresente em ordem decrescente, os fatoriais desde um dado valor, até o fatorial de \pm . □Elabore um algoritmo que obtenha o número inteiro que é mais próximo da raiz quadrada de um número fornecido pelo usuário.
 40. Elabore um algoritmo que obtenha o número inteiro que é mais próximo da raiz quadrada de um número fornecido pelo usuário.
 41. Escreva um algoritmo que leia e mostre um vetor de 20 números. A seguir, conte quantos valores diferentes existem no vetor.
 42. Escrever um algoritmo que leia 2 vetores de tamanho 10. Crie, a seguir, um vetor S de 20 posições que contenha os elementos dos 2 vetores em ordem crescente. Obs.: copie primeiro os valores para o vetor S para depois ordená-los
 43. Escrever um algoritmo que leia um vetor X[20]. Escreva, a seguir, cada um dos valores distintos que aparecem em X dizendo quantas vezes cada valor aparece em X.
 44. Faça um algoritmo que leia um código numérico inteiro e um vetor de 50 posições de números. Se o código for zero, termine o algoritmo. Se o código for 1, mostre o vetor na ordem como ele foi lido. Se o código for 2, mostre o vetor na ordem inversa, do último até o primeiro.
 45. Escreva um algoritmo que leia uma matriz M[6,6]. A seguir, troque os elementos da primeira coluna com os elementos da segunda coluna, os da terceira coluna com a quarta coluna e os elementos da quinta coluna com os elementos da sexta coluna.
 46. Faça um algoritmo que leia um vetor de 10 posições. Mostre então os 3 menores valores do vetor.
 47. Faça um algoritmo que leia 4 variáveis A,B,C e D. A seguir, se B for maior do que C e se D for maior do que A e a soma de C com D for maior que a soma de A e B e se C e D, ambos, forem positivos e se a variável A for par escrever a mensagem “valores aceitos”, senão escrever “valores não aceitos”.
 48. Escrever um algoritmo para determinar o consumo médio de um automóvel sendo

- fornecidos a distância total percorrida pelo automóvel e o total de combustível gasto.
49. Elabore um algoritmo que converta um valor em dólar (US\$) para real (R\$). O algoritmo deverá solicitar o valor da cotação do dólar e também a quantidade de dólares a ser convertida.
 50. Faça um algoritmo que leia a idade em anos de 3 pessoas e determine se a soma das idades é ou não maior ou igual a 100 anos.