Apostila de Técnicas de Programação

Profa. Elaine J.C.Brito

Conceito de Algoritmo

A automação é o processo pelo qual uma tarefa deixa de ser executada pelo homem para ser realizada por máquinas. Para que a automação seja bem sucedida é necessário que a máquina seja capaz de desempenhar cada uma das etapas constituintes do processo a ser automatizado com eficiência, de modo a garantir a repetibilidade do mesmo. Portanto é necessário que seja especificado com clareza e exatidão o que deve ser realizado em cada uma das fases do processo a ser automatizado, bem como a seqüência em que estas fases devem ser realizadas.

À especificação da seqüência ordenada de passos que deve ser seguida para a realização de uma tarefa, garantindo a sua repetibilidade, dá-se o nome **algoritmo**.

Algoritmo não é a solução de um problema, pois, se assim fosse, cada problema teria um único algoritmo. Algoritmo é um caminho para a solução de um problema, e em geral, os caminhos que levam a uma solução são muitos.

O aprendizado de algoritmos não se consegue a não ser através de muitos exercícios.

Algoritmos não se aprende:
Copiando Algoritmos
Estudando Algoritmos
Algoritmos só se aprendem:
Construindo Algoritmos
Testando Algoritmos

Exemplo de Algoritmo

Imagine o seguinte problema: Calcular a média final dos alunos da 1ª Série. Os alunos realizarão quatro provas: P1, P2, P3 e P4.

Onde: $$\text{Média Final} = \frac{P1+P2+P3+P4}{4}$$

Para montar o algoritmo proposto, faremos três perguntas:

a) Quais são os dados de entrada?

R: Os dados de entrada são P1, P2, P3 e P4

b) Qual será o processamento a ser utilizado?

R: O procedimento será somar todos os dados de entrada e dividi-los por 4 (quatro)

c) Quais serão os dados de saída?

R: O dado de saída será a média final

Algoritmo

Receba a nota da prova1

Receba a nota de prova2

Receba a nota de prova3

Receba a nota da prova4

Calcule Média Final somando todas as notas e dividindo o resultado por 4

Mostre a Média Final

Variáveis

O computador possui uma área de armazenamento conhecida como memória. Todas as informações existentes no computador estão ou na memória primária (memória RAM), ou na memória secundária (discos, fitas, CD-ROM etc). Nós iremos trabalhar, neste curso, somente com a memória primária, especificamente com as informações armazenadas na RAM (memória de acesso aleatório).

A memória do computador pode ser entendida como uma seqüência finita de caixas, que

num dado momento, guardam algum tipo de informação, como um número, uma letra, uma palavra, uma frase etc, não importa, basta saber que lá sempre existe alguma informação.

O computador, para poder trabalhar com alguma destas informações, precisa saber onde, na memória, o dado está localizado. Fisicamente, cada caixa, ou cada posição de memória, possui um endereço, ou seja, um número, que indica onde cada informação está localizada. Este número é representado através da notação hexadecimal, tendo seu tamanho definido em bytes, pois se quisermos escrever (ou ler) algum dado da (na) memória do computador, teremos que faze-lo byte a byte. Abaixo segue alguns exemplos:

Endereço Físico	Informação
3000: B712	'João'
2000: 12EC	12345
3000: 0004	'H'

Como pode ser observado, o endereçamento das posições de memória através de números hexadecimais é perfeitamente compreendido pela máquina, mas para nós humanos torna-se uma tarefa complicada. Pensando nisto, as linguagens de computador facilitaram o manuseio, por parte dos usuários, das posições de memória da máquina, permitindo que, ao invés de trabalhar diretamente com o número hexadecimal, fosse possível dar nomes diferentes a cada posição de memória. Tais nomes seriam de livre escolha do usuário. Com este recurso, os usuários ficaram livres dos endereços físicos (números hexadecimais) e passaram a trabalhar com endereços lógicos (nomes dados pelos próprios usuários). Desta forma, o Exemplo acima, poderia ser alterado para ter o seguinte aspecto:

Endereço Lógico	**Informação**
Nome	'João'
Número	12345
Letra	'H'

Como tínhamos falado, os endereços lógicos são como caixas, que num dado instante guardam algum tipo de informação. Mas é importante saber que o conteúdo desta caixa não é algo fixo, permanente, na verdade, uma caixa pode conter diversas informações, ou seja, como no Exemplo acima, a caixa (Endereço Lógico) rotulada de "Nome" num dado momento contém a informação "João", mas em um outro momento, poderá conter uma outra informação, por Exemplo "Pedro". Com isto queremos dizer que o conteúdo de uma destas caixas (endereço lógico) podem variar, isto é podem sofrer alterações em seu conteúdo. Tendo este conceito em mente, a partir de agora iremos chamar de forma genérica, as caixas ou endereços lógicos, de **variáveis**.

Desta forma podemos dizer que uma variável é uma posição de memória, representada por um Nome simbólico (atribuído pelo usuário), a qual contém, num dado instante, uma informação.

Uma variável possui três atributos: um nome, um tipo de dado associado à mesma e a informação por ela guardada.

Para o algoritmo da página 1 desta apostila, temos 5 variáveis:

Nota1, Nota2, Nota3, Nota4 e Média

O nome de uma variável é formado por uma letra ou então por uma letra seguida de letras ou dígitos, em qualquer número. Não é permitido o uso de espaços em branco ou de qualquer outro caractere, que não seja letra ou dígito, na formação de um identificador.

Exemplo:

SALÁRIO	correto
SAL/HORA	errado (contém o caractere “/”)
SAL HORA	errado (contém o caractere em branco)
SAL_HORA	correto
1ORÇAMENTO	errado (não começa com caractere)

Na formação do nome da variável dê um nome significativo, para que se possa ter idéia do seu conteúdo sem abri-la. Se utilizar mais de uma palavra para compor o nome da variável utilize o “_ “ underline para separar as palavras.

Tipos de Dados

Vimos que uma variável é como uma caixa que se pode armazenar “coisas” definidas dentro dela. Por que coisas **definidas**? Pegue uma caixa de papel e tente guardar água dentro dela. Depois de um curto espaço de tempo, ela se desmanchará. Deste mesmo modo são os nossos repositórios, é necessário definir claramente o que ele devera conter: valores numéricos, caracteres, datas.

É necessário que indiquemos o tipo de informação que uma variável pode conter. Para isto, a linguagem Pascal possui definido alguns tipos que deverão ser usados quando formos trabalhar com variáveis.

Algoritmo	Pascal	Descrição	Espaço
Inteiro	INTEGER	Representa números entre -32768 até +32767.	2 bytes
Real	REAL	Representa os números entre 2.9×10^{-39} até 1.7×10^{38}.	6 bytes
Caracter	CHAR	Representa um dos caracteres, da tabela ASCII.	1 byte
Cadeia	STRING	Conjunto de caracteres (CHAR).	1 a 255 bytes
Lógica	BOOLEAN	Valor lógico. Assume TRUE (Verdade) ou FALSE (Falso).	1 byte
	WORD	Números de 0 até 65535.	2 bytes
	BYTE	Números de 0 até 255.	1 byte
	Shor Int	Representa os números entre -128 até 128 .	1 byte
	LongInt	Representa os n^{os} entre -2.147.483.648 até 2.147.483.648.	4 bytes
	Single	Representa os números entre 1.5×10^{-45} até 3.4×10^{38}.	4 bytes
	Double	Representa os números entre 5×10^{-324} até 1.7×10^{308}.	8 bytes

Exercícios:

1) Sendo:

a: inteiro

x, y: real

n, t: string

Quais as atribuições a variáveis estão incorretas? Justifique.

a) a ← 11 b) t ← "Certo" c) y ← 17.3 d) x ← 11

e)n ← COTIL f) n ← “t” g) n ← g h) n ← t

2) Supondo que as variáveis Nota, Nome, NMat, SX sejam utilizadas para armazenar a nota, o nome do aluno, o número de matrícula e o sexo, declare-as corretamente, associando o tipo primitivo adequado ao dado que será armazenado.

3) Encontre os erros da seguinte declaração de variáveis:

Endereço, NFilhos, 5quantidade:inteiro

Idade, C: Caracter

C, Peso, R$: Real

Formas de representação de Algoritmos

Antes de utilizarmos uma linguagem de computador, é necessário organizar as ações a serem tomadas pela máquina de forma organizada e lógica, sem nos atermos às regras rígidas da Sintaxe de uma linguagem. Faremos isso através dos algoritmos.

Existem diversas formas de representação de algoritmos, dentre as mais conhecidas estão:

Descrição Narrativa;

Diagrama de Bloco ou Fluxograma;

Pseudocódigo, também conhecido como Linguagem Estruturada.

1 Descrição Narrativa

Nesta forma de representação, os algoritmos são expressos diretamente em linguagem natural.

Exemplo:

TROCA DE UM PNEU FURADO:

Afrouxar ligeiramente as porcas

Suspender o carro

Retirar as porcas e o pneu

Colocar o pneu reserva

Apertar as porcas

Abaixar o carro

Dar o aperto final nas porcas

Esta representação é pouco usada na prática porque o uso da linguagem natural muitas vezes dá oportunidade a más interpretações, ambigüidades e imprecisões.

Por exemplo, a instrução “afrouxar ligeiramente a porca” no algoritmo da troca de pneus está sujeita a interpretações diferentes por pessoas distintas. Uma instrução mais precisa seria: “afrouxar a porca girando-a 30º no sentido anti-horário”.

2 Diagrama de Bloco ou Fluxograma

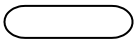
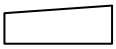
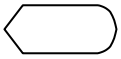
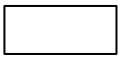
O que é um diagrama de bloco ou fluxograma?

O diagrama de blocos ou fluxograma é uma representação gráfica de algoritmos onde formas geométricas diferentes implicam ações (instruções, comandos) distintos. Tal propriedade facilita o entendimento das idéias contidas nos algoritmos e justifica sua popularidade. Com o diagrama podemos definir uma seqüência de símbolos, com significado bem definido, portanto, sua principal função é a de facilitar a visualização dos passos de um processamento.

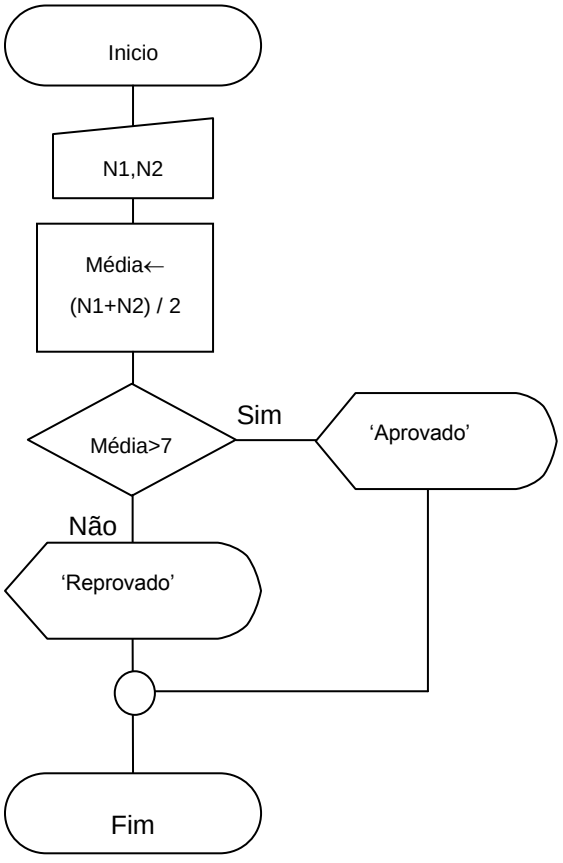
Simbologia

Existem diversos símbolos em um diagrama de bloco. No decorrer do curso apresentaremos os mais utilizados.

Veja no quadro a seguir alguns dos símbolos que iremos utilizar:

 Terminal	Início e final de um processamento Exemplo: Inicio do algoritmo
 Entrada de dado manual	Indica entrada de dados através do teclado Exemplo: Digite seu nome
 Exibir	Mostra informações ou resultados Exemplo: Exiba a média
 Processamento	Processamento geral Exemplo : Cálculo da média
Decisão	Decisão Exemplo : Se média >= 5

Dentro do símbolo sempre terá algo escrito, pois somente os símbolos não nos dizem nada. Veja no exemplo a seguir:



Exercícios:

1 Faça um algoritmo que receba pelo teclado dois números inteiros, calcule e mostre o dobro do primeiro e o triplo do segundo.

2 Faça um algoritmo que receba pelo teclado as notas de duas provas, dois trabalhos e uma lista de exercícios. Calcule e mostre a média, sabendo que as provas têm peso 6, os trabalhos, peso 3 e as listas, peso 1.

3 Faça um algoritmo que receba pelo teclado um número inteiro, calcule e mostre o quadrado deste número.

4 Faça um algoritmo que receba o valor do raio de uma esfera e calcule a sua área e o seu volume, baseando-se nas fórmulas dadas. Mostre o valor da área e o valor do volume da esfera. Utilize para π o valor de 3,14.

$$área = 4\pi R^2 \qquad volume = \frac{4}{3}\pi R^3$$

5 Elabore um algoritmo para ler dois valores inteiros, efetuar e mostrar o resultado das operações de adição, subtração e multiplicação de um número pelo outro.

6 Segundo a Organização Mundial de Saúde (O.M.S.) o Índice de Massa Corporal (IMC) normal é entre 18.5 e 25 para uma pessoa adulta. Para calcular esse índice utiliza-se a seguinte fórmula:

IMC = peso/(altura)^2

Faça um algoritmo que solicite o peso em Kg e a altura em m de uma pessoa, calcule e exiba o IMC.

7 Faça um algoritmo que calcule e exiba o troco a ser dado ao cliente. É necessário informar o valor a ser cobrado e o valor pago.

Troco = valor pago – valor a ser cobrado

3 Pseudocódigo

Esta representação de algoritmos é rica em detalhes, como definição dos tipos das variáveis usadas no algoritmo e, por assemelhar-se bastante à forma como os programas são escritos, encontra muita aceitação.

Exemplo:

ALGORITMO Média_Calc

Var N1,N2,Média: **real**

INÍCIO

Escreva (‘ Digite as duas notas:’)

Leia (N1,N2)

Média ← (N1+N2)/2

Escreva (‘ Sua média é ‘, Média)

FIM

Definição de Variáveis

Para definir uma ou mais variáveis vamos utilizar a palavra VAR. No algoritmo anterior, declaramos as variáveis N1,N2 e Média, todas do tipo Real. Abaixo segue Exemplo de algumas definições de variáveis:

ALGORITMO Teste

VAR

Nome : STRING

Letra, Identificador: CHAR

Soma: INTEIRO

INICIO

<comandos>

FIM

Comando de atribuição

Quando definimos uma variável é natural atribuirmos a ela uma informação. Uma das formas de colocar um valor dentro de uma variável, consequentemente colocar este dado na memória do computador, é através da atribuição direta, do valor desejado que a variável armazena. Para isto utilizaremos o símbolo

: = (Pascal)

← (Algoritmo)

que significa: recebe, ou seja, a posição de memória que uma variável representa, receberá uma informação, a qual será armazenada no interior desta variável.

Exemplo:

```
ALGORITMO Teste
VAR
        Valor: INTEIRO
INICIO
        Valor ←10
FIM
```

O Exemplo acima nos informa que:

a) Foi definida uma variável, a qual demos o Nome de “Valor”, e informamos que esta variável, ou posição de memória, só poderá aceitar dados, que sejam numéricos e que estejam entre -32768 a +32767 (tipo INTEGER).

b) Atribuímos à variável “Valor” o valor 10

A memória se comportaria da seguinte forma, de acordo com os itens acima:

a)

Variável	Conteúdo
Valor	Indefinido

b)

Variável	Conteúdo
Valor	10

Teste de Mesa

O teste de mesa é uma ferramenta de verificação dos algoritmos. É dividido em: área de memória e Área de tela.

Devemos seguir linha por linha do algoritmo lançando os dados simulando a execução do mesmo. Exemplo de teste de mesa do Algoritmo Média

Etapa1:

Área de Memória

(O primeiro passo é fazer a alocação das variáveis declaras na memória)

N1

N2

Média

(Observe que elas não apresentam valores)

Área de Tela

(O segundo passo é percorrer o algoritmo, linha por linha de comando. Na primeira linha após o begin temos o comando Escreva, que exibirá na tela uma mensagem, portanto escrevemos a mensagem na área de tela)

Digite as duas notas:

Etapa2:

Área de Memória

N1 **6.0**

N2 **7.0**

Média

Área de Tela

(Na linha seguinte temos o comando Leia, que recebe o que o usuário digitou na tela para atribuir às variáveis, ou seja armazena nas variáveis já alocadas os valores lidos)

Digite as duas notas: **6.0** **7.0**

Etapa3:

Na terceira linha, temos um cálculo (Média ←(N1+N2)/2) que é feito pelo programa, não sendo exibido na tela, mas ele atribui o resultado da fórmula à variável média

Área de Memória

N1 6.0

N2 7.0

Média **6.5**

Área de Tela

(Nessa etapa a área de tela não se altera)

Digite as duas notas: 6.0 7.0

Etapa4:

Área de Memória

N1 6.0

N2 7.0

Média 6.5

Área de Tela

(Na quarta linha, temos o comando escreva que vai exibir uma mensagem na tela, e exibir o valor armazenado na variável média)

Digite as duas notas: 6.0 7.0

Sua média é 6.5

Através dessa simulação pudemos verificar que o programa vai funcionar corretamente. Caso tivéssemos algum erro no algoritmo ele seria identificado durante a execução do teste de mesa.

Exercícios:

1) Identifique os dados de entrada, processamento e saída no algoritmo abaixo

- Receba código da peça
- Receba valor da peça
- Receba Quantidade de peças
- Calcule o valor total da peça (Quantidade * Valor da peça)
- Mostre o código da peça e seu valor total

2) Faça um algoritmo para "Calcular o estoque médio de uma peça", sendo que
ESTOQUEMÉDIO = (QUANTIDADE MÍNIMA + QUANTIDADE MÁXIMA) /2

3) Faça o Teste de mesa do algoritmo anterior com dados definidos por você.

4) Construa um algoritmo que :
Leia a cotação do dólar
Leia um valor em dólares
Converta esse valor para Real
Mostre o resultado

5) Desenvolva um algoritmo que:
Leia 4 (quatro) números
Calcule o quadrado para cada um
Somem todos e
Mostre o resultado

6) Construa um algoritmo para pagamento de comissão de vendedores de peças, levando-se em consideração que sua comissão será de 5% do total da venda e que você tem os seguintes dados:
Identificação do vendedor
Código da peça
Preço unitário da peça
Quantidade vendida

E depois faça um teste de mesa.

Variáveis do tipo CHAR(Caracter) e STRING(Cadeia)

As definições de variáveis como sendo do tipo CHAR e STRING, possuem algumas curiosidades que merecem um cuidado especial por parte do usuário.

Uso das aspas (‘)

Quando estivermos fazendo a atribuição de um valor para uma variável do tipo CHAR (Caracter) ou STRING (Cadeia), temos que ter o cuidado de colocar o valor (dado) entre aspas (‘), pois esta é a forma de informar que a informação é caracter.

Exemplo:

```
ALGORITMO Teste
VAR
        Letra : CHAR
        Nome : STRING
INICIO
        Letra ←‘A’
        Nome ←‘João’
FIM
```

Quando definimos uma variável como sendo do tipo STRING não estamos alocando 1 posição de memória apenas, mas na verdade, estamos alocando até 255 caixas, uma para cada caracter da STRING (lembre-se que uma STRING pode ter no máximo 255 caracteres). É possível definir variáveis do tipo STRING(Cadeia) com menos de 255 caracteres. Para isto, basta colocar, após a palavra STRING(Cadeia), o número de caracteres desejados entre colchetes [ ].

Exemplo:

```
ALGORITMO Define
VAR
        Nome: STRING[80]
INICIO
FIM
```

Desta forma, o espaço ocupado por uma variável STRING(Cadeia) passa de 255 bytes para apenas 80 bytes, na memória.

Operadores aritméticos

Os operadores aritméticos são os seguintes:

+	Soma.
-	Diferença.
/	Divisão
*	Multiplicação
mod	Retorna o resto da divisão de um número **inteiro** por outro. Ex: 10 div 3 retorna o resto da divisão de 10 por 3, ou seja, 1.
div	Retorna o quociente da divisão **inteira** de um número por outro. Ex: 5 div 2 retorna o quociente da operação 5/2, ou seja, 2.

Temos também algumas funções matemáticas e de conversão pré definidas no Pascal.

sqr(x)	Retorna o valor do quadrado de x.
sqrt(x)	Retorna o valor da raiz quadrada de x.
sin(x)	Retorna o valor do seno de x em radianos
cos(x)	Retorna o valor do cosseno de x. em radianos
abs(x)	Valor absoluto de um número x.
arctan(x)	Retorna o valor do arco tangente de x em radianos
ln(x)	Retorna o logaritmo neperiano de x.
trunc(x)	Retorna o valor de x com a parte decimal removida.
exp(x)	Retorna o exponencial de x.
Exp(ln(x)*y)	Retorna x elevado a y
Pi	Retorna o valor de π (3,1415...)
Random(x)	Retorna um número aleatório entre 0 e x
Int(x)	Retorna a parte inteira de x
Frac(x)	Retorna a parte fracionária de x
Round(x)	Arredonda x para um inteiro

Comandos de Entrada e Saída

O computador não é uma máquina isolada, pois ele precisa se comunicar com o mundo exterior, com vídeo, impressora, teclado, discos, fitas etc. Para realiza esta comunicação existem comandos que permitem que informações sejam exibidas, por exemplo, no vídeo, como também existem comandos que permitem que informações sejam colocadas na memória do computador através do teclado do PC.

Os comandos que iremos ver são os comandos LEIA e ESCREVA, respectivamente, comando de entrada e de saída (em Pascal escreve-se READ e WRITE).

Exemplo 1: Escrever um algoritmo para ler um valor numérico do teclado e atribuí-lo a uma variável do tipo numérica.

```
ALGORITMO LeNúmero
VARIÁVEIS
        Num : INTEIRO
INICIO
        LEIA(Num)
FIM
```

O algoritmo acima executará os seguintes comandos:

a) Define uma variável rotulada " Num", a qual só poderá armazenar dados numéricos, que estejam compreendidos entre -32768 a +32767 (INTEIRO), sendo que inicialmente o conteúdo desta variável está indefinido.

b) interrompe o processamento até que uma informação seja digitada, seguida do pressionamento da tecla ENTER. Caso a informação digitada seja compatível com o tipo INTEGER (INTEIRO), este valor será armazenado dentro da variável " Num".

Exemplo 2: Fazer um algoritmo para escrever no vídeo do PC uma mensagem qualquer.

```
ALGORITMO EscreveMsg
INICIO
    ESCREVA ('Alo Mundo' )
FIM
```

Obs.: A mensagem está entre aspas porque representa uma cadeia de caracteres. É perfeitamente possível mandar escrever o conteúdo de variáveis no vídeo. Desta forma, o Exemplo acima poderia ser escrito do seguinte modo:

```
ALGORITMO EscreveMsg
VARIÁVEIS
        Msg : CADEIA
INICIO
        Msg ←'Alo Mundo'
        ESCREVA (Msg)
FIM
```

Agora em Pascal:

Todos os programas terão a primeira linha indicando que iremos digitar um programa:

```
program Primeiro_Programa;
```

... na segunda linha indicamos quais unidades serão usadas (no nosso caso é importante sempre definir o uso da unidade crt que traz variáveis de controle de som, vídeo e teclado)...

```
uses crt;
```

... na terceira linha indicamos o inicio do programa...

```
begin
```

... após o begin vem o corpo do programa com todas as ações que devem ser feitas; por exemplo, escrever Bem vindo ...

```
  writeln ('Bem vindo');
```

... para que o programa aguarde um enter antes de encerrar usamos o comando readln(assim dá tempo de ler a mensagem)...

```
  readln;
```

... e, finalmente indicamos que o programa acabou...

```
end.
```

Temos então:

```
program Primeiro_Programa;
uses crt;
begin
  writeln ('Bem vindo');
  readln;
end.
```

Para que o computador escreva também o seu nome, é só acrescentar mais um comando *writeln*:

```
program Segundo_Programa;
uses crt;
begin
  writeln ('Bem vindo');
  writeln ('seu nome');
  readln;
end.
```

Observação Importante:

Begin não termina com ";" porque representa o começo de um bloco de comandos, que termina com o comando *end*.

Todo programa Pascal termina com *end* seguido de "." (ponto final) Quando o end não indica o fim do programa ele não deve ser seguido do ponto final.

O nome de um programa (que vem depois do comando program) segue a regra de nomenclatura de variáveis

WRITE / WRITELN

Com o exemplo anterior podemos concluir que o comando *writeln* escreve algo na tela e se posiciona na linha seguinte. O comando write também escreve na tela, mas permanece na mesma linha. Observe:

```
write('Nome da escola:');
writeln('COTIL');
```

exibe na tela:

```
Nome da escola:COTIL
```

A frase "Nome da escola::" foi escrita pelo *write*. Na mesma linha o comando writeln escreveu COTIL e em seguida foi para a linha de baixo.

Para escrever ao mesmo tempo vários itens com um único *writeln* ou com um único *write*. colocamos tudo o que queremos escrever entre os parênteses do comando, separando cada item por vírgulas:

```
writeln('Eu tenho ' , 2008-1992 , ' anos.');
```

Com esse exemplo pudemos verificar que é possível fazer operações aritméticas no Pascal. Veja alguns exemplos:

```
Program Calculos;
Uses crt;
Begin
        Writeln (4+2);
        Writeln (4-2);
        Writeln (4*2);
        Writeln (4/2);
        Writeln (5 div 2);
        Writeln (5 mod 2);
        Writeln ((2+2)/(3-1)*3);
End.
```

Utilizando variáveis no Pascal

O programa a seguir armazena o valor 6 na variável "numero" e depois mostra o valor dessa variável:

```
program Exibe_Numero;
uses crt;
var
  numero: integer;
begin
  numero:=6;
  writeln(numero);
  readln;
end.
```

Como já explicado, o valor de uma variável pode **variar** de acordo com os comandos que executarmos. Veja:

```
program Exibe_Numero_2;
uses crt;
var  numero: integer;
begin
  numero:= 8;
  writeln(numero);
  numero:= 4+2;
  writeln(numero);
  numero:= numero*2;
  writeln(numero);
  readln;
end.
```

Ao ser executado, esse programa faz os seguintes passos:

Atribui à variável "numero" o valor 8

Mostra na tela o valor da variável "numero", ou seja, 8.

Atribui à variável "numero" o valor 6(resultado da expressão 4+2)., que substitui o valor anterior (ou seja, o computador perde o valor 8).

Mostra na tela o valor da variável "numero", ou seja, 6.

Atribui à variável “número” o valor da variável número multiplicado por 2. Antes da nova atribuição o valor armazenado em número é 6. No momento da nova atribuição esse valor será multiplicado por 2 e só então atribuído o resultado à variável , ou seja, 12. À partir desse momento o valor anterior 6 é perdido.

Mostra na tela o valor da variável "numero", ou seja, 12.

Quando não queremos perder o valor armazenado, não podemos utilizar a mesma variável, devemos utilizar mais de uma variável em um mesmo programa. Aliás, a maior parte dos programas precisa usar várias delas.

```
program aposentadoria;
uses crt;
var
  idade: integer;
  idade_maxima: integer;
begin
  idade:=55;
  idade_maxima:=70;
  writeln('A idade do funcionário é ',idade, 'anos.');
  write('Faltam ');
  write(idade_maxima - idade);
  writeln(' anos para ele se aposentar.');
  readln;
end.
```

Neste exemplo o valor da variável "idade" se mantém na memória do computador até o fim da execução do programa, perceba que não foi atribuído um segundo valor à essa variável.

Variáveis tipo real

Quando utilizamos variáveis do tipo real, o resultado é exibido em notação exponencial, uma forma de representação de números muito útil quando trabalhamos com valores muito grandes ou muito pequenos, e freqüentemente utilizada em Física e Química. O valor 2.3050000000E+02 quer dizer 2.3050000000(ou seja, 2.305) vezes 10 Elevado à 2ª. potência, ou seja, 230,5.

Acredito que seja mais interessante exibir o número em um formato com duas casas decimais, como "230.50". Para fazer isso, precisamos formatar a saída. Veja um exemplo:

```
writeln(preco:6:2);
```

O valor 6 representa quantos caracteres serão necessários na tela para representar o número a ser exibido e o valor 2 representa o número de casas decimais.

Exemplo:

```
program troco;
uses crt;
var
  preço, recebido,troco: real;
begin
  preco:=21.5;
  recebido:=50;
  writeln('O valor a ser cobrado é R$',preco:4:2);
  writeln('O valor recebido foi de R$',recebido:4:2);
  troco:=recebido-preco;
  writeln(' Seu troco é de R$',troco:4:2);
end.
```

Se aumentarmos o valor 4 para 5, perceberemos que aparece um espaço entre o $ e o número exibido. Isso acontece porque o número tende a ser alinhado à direita dentro do espaço reservado

para ele; como o número exibido tem 4 caracteres, um espaço em branco é inserido no início do valor para preencher todo o espaço reservado. Experimente modificar tanto o número de casas decimais reservado quanto o número de caracteres reservados (2 e 4, respectivamente) para perceber como eles funcionam.

Regra importante: Inicializar variáveis

Quando declaramos uma variável, é reservado um espaço na memória do computador para armazenar informações. Esse espaço de memória pode ter sido usado anteriormente por outro programa e ter qualquer valor armazenado nessa posição de memória. Os compiladores Pascal mais antigos, não limpam esses espaços de memória por isso devemos ter o hábito de inicializar as variáveis que serão exibidas ou utilizadas para cálculos, garantindo assim que não vamos trabalhar com lixo de memória.

Recebendo valores

Os comandos write e writeln são utilizados para exibir valores na tela, para ler valores digitados pelo usuário utilizamos READLN.

Exemplo:

```
program eco;
uses crt;
var
  nome: string;
begin
  writeln('Digite seu nome:');
  readln(nome);
  writeln('Olá ', nome);
  readln;
end.
```

Se quisermos fazer um programa que calcula a média entre dois números digitados pelo usuário, então podemos digitar algo como:

```
program boletim;
uses crt;
var
  nota1, nota2, media: real;

begin
  writeln('Digite a primeira nota:');
  readln(nota1);
  writeln('Digite a segunda nota:');
  readln(nota2);
  write('As notas digitadas foram: ');
  writeln(nota1:4:1, ' e ', nota2:4:1);
  media:= (nota1 + nota2) / 2;
  writeln('A média é: ', media:4:1);
  readln;
end.
```

Sua tela deve ter ficado assim:

```
Digite a primeira nota:
5.5
Digite a segunda nota:
8.5
As notas digitadas foram:  5.5 e  8.5
A média das é:  7.0
```

Faça as seguintes alterações:

```
program boletim_2;
```

```
uses crt;
var
  nota1, nota2, media: real;
begin
  writeln('Digite duas notas:');
  readln(nota1, nota2);
  write('As notas digitadas foram: ');
  writeln(nota1:4:1, ' e ', nota2:4:1);
  media:= (nota1 + nota2) / 2;
  writeln('A média é: ', media:4:1);
  readln;
end.
```

Ao executar esse programa, digite as duas notas separando-as por pelo menos um espaço em branco, ou então por ENTER. Pressione ENTER quando terminar a segunda nota. Sua tela deve ter ficado mais ou menos assim:

```
Digite duas notas:
5.5 8.5
As notas digitadas foram:  5.5 e  8.5
A média é:  7.0
```

Agora vamos fazer uma pequena experiência. Ao executar o programa boletim, o que acontece se você digitar dois valores seguidos na mesma linha, separados por espaços?

```
Digite uma nota:
5.5 8.5
Digite outra nota:
7.5
As notas que você digitou foram:  5.5 e  7.5
A média das notas é:  6.5
```

O valor 8.5 foi ignorado. Na verdade, o comando *readln* lê a linha inteira que você digitou e depois olha a lista de variáveis nos parênteses após o comando, para associar cada valor digitado a uma variável. Quando o programa executou a linha de comando *readln(nota1)* digitamos dois valores na mesma linha, mas o comando só esperava por um único valor na linha; logo, ele jogou fora o segundo valor.

Read em vez de Readln

O comando *read* também permite que o usuário digite um ou mais valores e os armazena em variáveis, com algumas diferenças em relação ao *readln*.

O comando *read*, verifica tudo o que o usuário digitou em uma linha, mas **não joga fora o restante da linha digitada**; ele guarda o resto da linha na memória, para que possa ser usada pelo próximo comando *read* ou *readln*.

Um exemplo pode esclarecer melhor:

```
program tres_valores;
uses crt;
var
  a,b,c: integer;
begin
  writeln('Digite três números:');
  read(a);
  writeln('a:',a);
  read(b);
  writeln('b:',b);
  read(c);
  writeln('c:',c);
end.
```

Ao executar o programa, experimente digitar valores na mesma linha, ou em linhas diferentes, ou mesmo dois valores numa linha e um em outra. O *read* não se importa com a forma como você passou os dados, se existem dados que o usuário digitou o computador usa aquilo que

precisar desses dados e só pede mais dados se for necessário.

Exemplo:

```
program boletim_3;
uses crt;
var
  nota1, nota2, media: real;
begin
  writeln('Digite uma nota:');
  read(nota1);
  writeln('Digite outra nota:');
  read(nota2);
  write('As notas digitadas foram: ');
  writeln(nota1:4:1, ' e ', nota2:4:1);
  media:= (nota1 + nota2) / 2;
  writeln('A média é: ', media:4:1);
  readln;
end.
```

Ao executar o programa, quando o computador digitar "Digite uma nota:", experimente digitar duas, separadas por espaço. Veja o que acontece:

```
Digite uma nota:
5.5 8.5
Digite outra nota:
As notas digitadas foram:  5.5 e  8.5
A média é:  7.0
```

O computador armazenou tudo o que foi digitado; quando executou *read(nota1)* ele verificou o que tinha sido digitado e pegou apenas um valor - 5.5 - e o atribuiu à variável "nota1". O valor 8.5 ficou reservado para um próximo comando de entrada de dados. Então, ele executou o *writeln('Digite outra nota:');* e quando apareceu outro comando *read, c*omo ele ainda tinha dados que tínhamos digitado mas ele não havia usado, ele pegou o valor que estava guardado 8.5 - e atribuiu-o à variável "nota2".

Exercícios:

Escreva um algoritmo (fluxograma e pseudocódigo) para cada um dos exercícios a seguir:

1.Calcular a média entre dois números quaisquer.

2. 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)

4.Cálculo do quadrado de um número, ou seja, o produto de um número por si mesmo.

5.Ler uma temperatura em graus centígrados e apresenta-la convertida em graus Fahrenheit. A fórmula de conversão é : F ← (9 * C + 160) / 5. Onde F é a temperatura em Farenheit e C é a temperatura em Centígrados.

6. Ler uma temperatura em graus Fahrenheit e apresenta-la convertida em graus centígrados. A fórmula de conversão é : C ← (F - 32) * (5 / 9). Onde F é a temperatura em Farenheit e C é a temperatura em Centígrados.

7.Calcular e apresentar o valor do volume de uma lata de óleo, utilizando a fórmula:

VOLUME ←3.14159 * R^2 * ALTURA.

8.Ler dois valores para as variáveis A e B, efetuar a troca dos valores de forma que a variável A passe a possuir o valor da variável B e que a variável B passe a possuir o valor da variável A.

Operadores Relacionais

Os operadores relacionais são utilizados para comparações. Estes operadores sempre retornam valores lógicos (verdadeiro ou falso/ True ou False). Para estabelecer prioridades no que diz respeito a qual operação executar primeiro, utilize os parênteses.

Os operadores relacionais são:

a) Igual a =

b)Diferente de <>

c) Maior que >

d) Menor que <

e) Menor Igual <=

f) Maior Igual >=

Exemplo: Tendo duas variáveis A=5 e B=3. Os resultados das expressões seriam:

Expressão	Resultado
A = B	Falso
A <> B	Verdadeiro
A > B	Verdadeiro
A < B	Falso
A >= B	Verdadeiro
A <= B	Falso

Operadores Lógicos

Os operadores lógicos realizam as operações da álgebra booleana. São eles:

a) AND (E) b) OR (OU) c) NOT (NÃO)

Hierarquia das Operações Lógicas

1 º Not

2 º And

3 º Or

Hierarquia entre os Operadores:

1º Aritméticos e Literais

2º Relacionais

3º Lógicos

Exemplo:

a) Operador AND:

TRUE AND TRUE => TRUE

TRUE AND FALSE => FALSE

FALSE AND TRUE => FALSE

FALSE AND FALSE => FALSE

b) Operador OR:

TRUE OR TRUE => TRUE

TRUE OR FALSE => TRUE

FALSE OR TRUE => TRUE

FALSE OR FALSE => FALSE

c) Operador NOT:

NOT TRUE => FALSE

NOT FALSE => TRUE

Exemplo: Suponha que temos três variáveis A = 5, B = 8 e C =1 Os resultados das expressões seriam:

Expressões			Resultado
A = B	AND	B > C	Falso
A <> B	OR	B < C	Verdadeiro
A > B	NOT		Verdadeiro
A < B	AND	B > C	Verdadeiro
A >= B	OR	B = C	Falso
A <= B	NOT		Falso

Exercícios:

1) Tendo as variáveis SALARIO, IR e SALLIQ, e considerando os valores abaixo. Informe se as expressões são verdadeiras ou falsas.

SALARIO	IR	SALLIQ	EXPRESSÃO	V ou F
100,00	0,00	100	(SALLIQ >= 100,00)	
200,00	10,00	190,00	(SALLIQ < 190,00)	
300,00	15,00	285,00	SALLIQ = SALARIO - IR	

2) Sabendo que A=3, B=7 e C=4, informe se as expressões abaixo são verdadeiras ou falsas.

a) (A+C) > B b) B >= (A + 2) c) C = (B –A) d) (B + A) <= C e) (C+A) > B

3) Sabendo que A=5, B=4 e C=3 e D=6, informe se as expressões abaixo são verdadeiras ou falsas.

a) (A > C) **AND** (C <= D) b) (A+B) > 10 **OR** (A+B) = (C+D) c) (A>=C) **AND** (D >= C)

4) Suponha que X, Y e Z são variáveis reais, e A e B variáveis lógicas, com os respectivos valores:

X = 2.0, Y = 3.0, Z=0.5, A = .V. e B =.F.

Calcule o resultado das seguintes expressões:

B .E. A .OU. X <> Y/2 B .E. (A .OU. X <> Y/2)

5) Dada a declaração de variáveis:

```
VAR     A, B, C      : inteiro
        X, Y, Z      : real
        NOME, RUA    : literal[20]
        L1, L2 : lógico
```

São dados os valores seguintes:

A = 1 B = 2 C = 3 X = 2.0 Y = 10.0 Z = -1.0

L 1= .V. L2 = .F. NOME = “PEDRO” RUA = “PEDRINHO”

Determine o resultado da avaliação das expressões a seguir:

a) NOME = RUA b) L1 .OU. L2

c) (L1 .E. (.NÃO. L2)) d) (L2 .E. (.NÃO. L1))

e) X>Y .E. C <=B f) (C – 3 * A) < (X + 2 * Z)

g) (L1 .E.(.NÃO. L2)).OU. (L2 .E. (.NÃO. L1))

6) De acordo com a tabela de operadores lógicos estudada no item 7.2 da apostila, determine o resultado lógico das expressões mencionadas, assinalando se são verdadeiras ou falsas. Considere para as respostas os seguintes valores: X=1, A=3, B=5, C=8 e D=7

a)	.NÃO. (X>3)	Verdadeiro ( )	Falso ( )
b)	(X < 1) .E. .NÃO. (B > D)	Verdadeiro ( )	Falso ( )
c)	.NÃO. (D < 0) .E. (C > 5)	Verdadeiro ( )	Falso ( )
d)	.NÃO. (X > 3) .OU. (C < 7)	Verdadeiro ( )	Falso ( )
e)	(A > B) .OU. (C > B)	Verdadeiro ( )	Falso ( )
f)	(X >= 2)	Verdadeiro ( )	Falso ( )
g)	(X < 1) .E. (B >= D)	Verdadeiro ( )	Falso ( )
h)	(D < 0) .OU. (C > 5)	Verdadeiro ( )	Falso ( )
i)	.NÃO. (D > 3) .OU. .NÃO. (B < 7)	Verdadeiro ( )	Falso ( )
j)	(A >B) .OU. .NÃO. (C> B)	Verdadeiro ( )	Falso ( )

Estrutura de decisão

Os comandos de decisão ou desvio fazem parte das técnicas de programação que conduzem a estruturas de programas que não são totalmente seqüenciais. Com as instruções de

SALTO ou DESVIO pode-se fazer com que o programa proceda de uma ou outra maneira, de acordo com as decisões lógicas tomadas em função dos dados ou resultados anteriores. As principais estruturas de decisão são: “Se Então”, “Se então Senão” e “Caso Selecione”

Estrutura SE…ENTÃO…SENÃO (IF…THEN…ELSE)

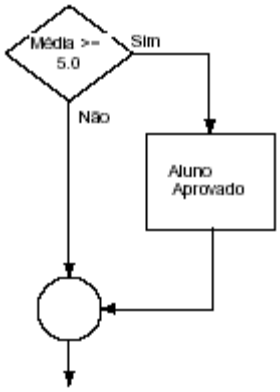
A estrutura de decisão “SE/IF” normalmente vem acompanhada de um comando, ou seja, se determinada condição for satisfeita pelo comando SE/IF então execute determinado comando.

SE <Condição FOR verdade> ENTÃO

<Comandos>

[SENÃO

<Comandos>] Colchete indica que o comando é opcional



Imagine um algoritmo que determinado aluno somente estará aprovado se sua média for maior ou igual a 5.0, veja no exemplo de algoritmo como ficaria.

SE MEDIA >= 5.0 ENTÃO ALUNO APROVADO

A estrutura de decisão “SE/ENTÃO/SENÃO”, funciona exatamente como a estrutura “SE”, com apenas uma diferença, em “SE” somente podemos executar comandos caso a condição seja verdadeira, diferente de “SE/SENÃO” pois sempre um comando será executado independente da condição, ou seja, caso a condição seja “verdadeira” o comando da condição será executado, caso contrário o comando da condição “falsa” será executado.

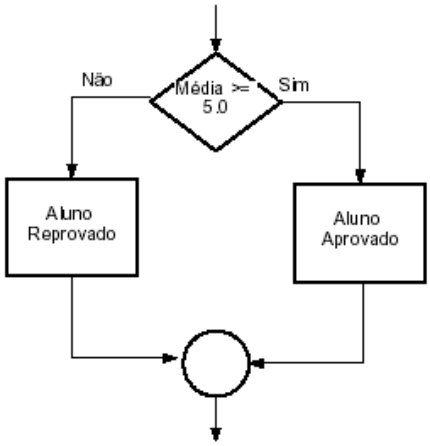
Em algoritmo ficaria assim:

SE MÉDIA >= 5.0 ENTÃO

ALUNO APROVADO

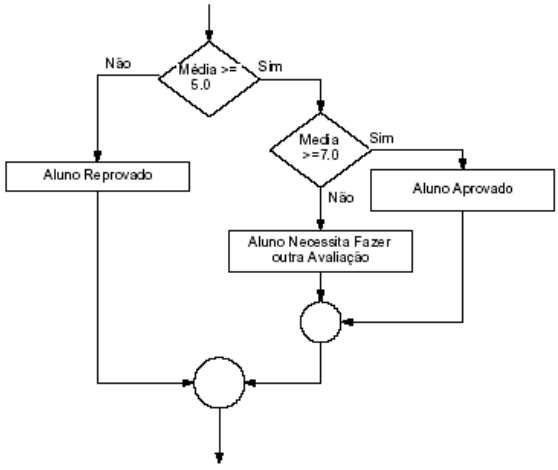
SENÃO

ALUNO REPROVADO



Em diagrama, observe a figura ao lado.

No exemplo anterior está sendo executada uma condição que, se for verdadeira, executa o comando “APROVADO”, caso contrário executa o segundo comando “REPROVADO”. Podemos também dentro de uma mesma condição testar outras condições. Como no exemplo a seguir:

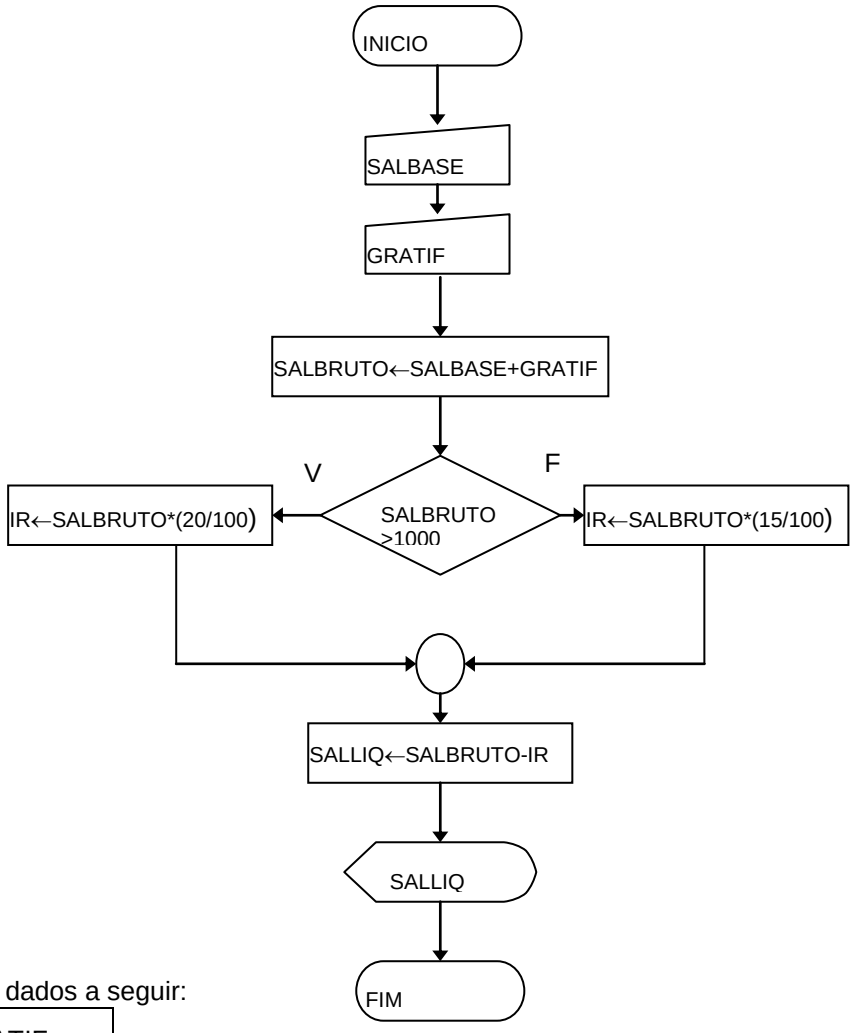


Outro Exemplo: Dados dois valores A e B quaisquer, faça um algoritmo que imprima se A > B, A < B, ou A=B

```
ALGORITMO Maior
VARIÁVEIS
        A,B : INTEIRO
INICIO
        ESCREVA('Digite os valores A e B' );
        LEIA (A,B)
        SE A > B ENTÃO
            ESCREVA('A é maior que B' )
        SENÃO
           SE A < B ENTÃO
              ESCREVA('A é menor que B' )
           SENÃO
              ESCREVA('A é igual a B' )
FIM
```

Exercícios:

1) Faça um teste de mesa do diagrama apresentado abaixo, de acordo com os dados fornecidos:



Teste o diagrama com os dados a seguir:

SALBASE	GRATIF

3000,00	1200,00
1200,00	400,00
500,00	100,00

Memória

SALBASE	GRATIF	SALBRUTO	IR	SALLIQ

Dados de Saída

SALLIQ

2. Indique a saída dos trechos de programa em português estruturado, mostrado a seguir. Para os cálculos considere os seguintes valores: A = 2, B = 3, C= 5 e D = 9.

a)
```
se .não. (D>5) então
        X ← (A+B) * D
senão
        X ← (A – B) / C
escreva X
```
Resposta:_____

b)
```
se  (A > 2) .e. (B < 7)  então
        X ← (A+2) * (B - 2)
senão
        X ← (A + B) / D * (C + D)
escreva X
```
Resposta:_____

c)
```
se  (A = 2) .ou. (B < 7)  então
        X ← (A+2) * (B - 2)
senão
        X ← (A + B) / D * (C + D)
escreva X
```
Resposta:_____

d)
```
se  (A > 2) .ou. .não.  (B < 7)  então
        X ← A + B – 2
senão
        X ← A – B
escreva X
```
Resposta:_____

e)
```
se  .não. (A > 2)  .ou. .não.  (B < 7)  então
        X ← A + B
senão
        X ← A  / B
escreva X
```
Resposta:_____

3. Seja o seguinte algoritmo:

```
x,y,z: inteiro
resposta: string;
```

```
INICIO
   leia (x);
   leia (y);
   z ← (x*y) + 5;
   se z  <= 0
        então resposta ← 'A'
   senão  se z <= 100
                então resposta ← 'B'
           senão resposta ← 'C';
                escreva (z,resposta);
FIM
```

Faça um teste de mesa e complete o quadro a seguir para os seguintes valores:

x	y	z	resposta	saída
3	2			
150	3			
7	-1			
-2	5			
50	3			

Decisão no Pascal

Estrutura condicional simples

Exemplo:

```
program ex_1;
uses crt;
var  A,B:integer;
Begin
  writeln('Digite dois números:');
  readln(A);
  readln(B);
  IF A=B THEN
       writeln('Números iguais');
end.
```

Nesse exemplo, a frase números iguais só será exibida se a condição a=b for verdadeira.

Observe que na estrutura if then só é usado o **;**(ponto e vírgula) após o comando a ser executado caso a condição seja verdadeira e esse ponto e virgula encerra a estrutura. Nesse exemplo, o comando writeln que vem logo após o then será executado e em seguida a estrutura condicional é encerrada, assim, o comando que vem depois do ponto e virgula é executado independente da condição testada, pois ele se encontra fora da estrutura de decisão.

Vamos observar esse outro exemplo, que contém um erro que iremos corrigir em seguida:

```
program ex_2;
uses crt;
var   N1,N2,N3,M:real;
Begin
  writeln('Digite três notas:');
  readln(N1,N2,N3);
  M:=(N1+N2+N3)/3;
  IF M>=7 THEN
       writeln('Média = ',M:4:1);
       writeln('Aprovado');
end.
```

No exemplo, o comando writeln('Média = ',M:4:1) está dentro da estrutura condicional ou seja será executado quando a média for 7 ou mais. O ponto e virgula encerra a estrutura, então de qualquer forma é executado o comando seguinte, ou seja, independente da média do aluno, será

exibida a mensagem aprovado. Para corrigir esse problema devemos colocar todos os comandos a serem executados se a condição for verdadeira entre um begin e um end. Observe a seguir:

```
program ex_2;
uses crt;
var
  N1,N2,N3,M:real;
Begin
  Clrscr;
  writeln('Digite três notas:');
  readln(N1,N2,N3);
  M:=(N1+N2+N3)/3;
  IF M>=7 THEN
       begin
               writeln('Média = ',M:4:1);
               writeln('Aprovado');
       end;
  readln;
end.
```

Agora os dois comandos serão executados apenas se a condição testada na estrutura if then for verdadeira (ou seja, se Média for maior ou igual a sete)

Vamos então definir a seguinte regra: quando forem necessários mais de um comando dentro da estrutura condicional devemos colocar todos eles entre um **begin** e um **end**.

Exercícios:

1) Escreva um algoritmo (fluxograma e pseudocódigo) para cada um dos exercícios a seguir:

a) Efetuar a leitura da idade de uma pessoa e se ela tiver 18 anos ou mais escrever a mensagem que ela é maior de idade.

b) Altere o programa anterior para que além de escrever a mensagem que ela é maior, escrever uma segunda mensagem dizendo que ela pode dirigir.

Estrutura condicional composta

Exemplo:

```
program ex_3;
uses crt;
var
  A,B:integer;
Begin
  Clrscr;
  writeln('Digite dois números:');
  readln(A);
  readln(B);
  IF A=B THEN
       writeln('Números iguais')
  ELSE
       writeln('Números diferentes');
  readln;
end.
```

Nesse exemplo, a frase Números iguais só será exibida se a condição A=B for verdadeira e a frase Números diferentes só será exibida se a condição A=B for falsa ou seja não for verdadeira.

Obs.: Não se usa ;(ponto e vírgula) antes do else.

A mesma regra do begin e end valem para a estrutura composta, veja exemplo:

```
program ex_4;
uses crt;
```

```
var
  N1,N2,N3,M:real;
Begin
  writeln('Digite três notas:');
  readln(N1,N2,N3);
  M:=(N1+N2+N3)/3;
  IF M>=7 THEN
       begin
               writeln('Média = ',M:4:1);
               writeln('Aprovado');
       end
  ELSE
       begin
               writeln('Média = ',M:4:1);
               writeln('Reprovado');
       end;
end.
```

Podemos melhorar esse algoritmo. Veja que nas duas condições ou seja no if e no else temos o comando writeln('Média = ',M:4:1).Se desejamos exibir a média de qualquer forma, esse comando deve ser colocado fora da estrutura de decisão.

```
program ex_5;
uses crt;
var
  N1,N2,N3,M:real;
Begin
  writeln('Digite três notas:');
  readln(N1,N2,N3);
  M:=(N1+N2+N3)/3;
  writeln('Média = ',M:4:1);
  IF M>=7 THEN
       writeln('Aprovado')
  ELSE
       writeln('Reprovado');
end.
```

Exercícios:

1) Escreva um algoritmo para cada um dos exercícios a seguir:

a) Ler 2 números e apresentar a diferença do maior pelo menor.

b) Ler um número e dizer se o número digitado é par ou ímpar

c) Em um curso, a nota final do aluno é calculada a partir do seu desempenho em 3 aspectos do trabalho semestral. Existe um exame intermediário que vale 30% da nota final. Ao trabalho de laboratório é dada uma nota que vale 20% da nota final. O exame final completa o restante. Faça um algoritmo que leia o nome do estudante e suas notas em cada item, calcule e informe o nome do estudante, suas notas intermediárias, sua nota final e uma indicação sobre a sua aprovação ou não, sabendo que a nota final mínima para aprovação é 6.

d) Faça um algoritmo que leia o nome de um funcionário, as horas trabalhadas no dia e o seu salário/hora. Calcule o que ele tem a receber e escreva o nome do funcionário, as horas trabalhadas e o salário a receber. OBS. Se a jornada de trabalho for maior que 8 horas, esta diferença será considerada hora-extra, a qual tem um acréscimo de 50%.

Estrutura condicional composta aninhada

Exemplo:

```
program ex_6;
uses crt;
var
  N1,N2,N3,M:real;
Begin
  Clrscr;
  writeln('Digite três notas:');
  readln(N1,N2,N3);
  M:=(N1+N2+N3)/3;
  writeln('Média = ',M:4:1);
  IF M>=7 THEN
              writeln('Aprovado');
  ELSE
       IF (M>=4) THEN
          begin
               writeln('Exame');
               writeln('Você precisa tirar',10-M, 'no Exame');
          end
       ELSE
          begin
               writeln('Você deveria ter estudado mais.');
               writeln('Reprovado');
          end;
  readln;
end.
```

Observe que embora o conteúdo do primeiro else seja extenso, ele é composto de um único comando if, portanto não precisa de begin end.

Exercícios:

1) Escreva um algoritmo (fluxograma e pseudocódigo) para cada um dos exercícios a seguir:

a)Efetuar a leitura da idade de uma pessoa e se ela tiver 18 anos ou mais escrever duas mensagens: “Maior de idade” e “Já pode dirigir” alem disso, solicite que ela digite seu sexo e caso seja do sexo masculino, escreva a mensagem “Alistamento militar obrigatório”.

b) Desenvolva um algoritmo que recebe dois valores a e b, verifique se a é divisível por b e informe os valores recebidos e o resultado da verificação. Lembre-se que nenhum número é divisível por 0.

c) Em certa instituição de ensino, a média final é calculada de duas formas, valendo a que apresentar resultado mais interessante para o aluno. Durante o semestre, três notas são obtidas. Uma forma de definir a média final é calcular a média aritmética. A outra é uma média ponderada, com pesos 3, 2 e 4, respectivamente. Desenvolva um algoritmo que informe o nome do aluno, suas notas, a média aritmética, a média ponderada e qual delas foi considerada.

Combinando condições

Exemplo:

```
program ex_7;
uses crt;
var
  N1,N2,N3,M,Limite:real;
  Aulas_dadas,Faltas:integer;
Begin
  Clrscr;
  writeln('Digite o número de aulas dadas:');
  readln(aulas_dadas);
  Limite:=aulas_dadas*0,25;
  writeln('Digite o número de faltas:');
  readln(faltas);
  writeln('Digite três notas:');
  readln(N1,N2,N3);
  M:=(N1+N2+N3)/3;
  writeln('Média = ',M:4:1);
  IF (M>=7) AND (faltas<=Limite) THEN
       writeln('Aprovado')
  ELSE
       IF (M>=4) AND (faltas<=Limite)  THEN
          begin
               writeln('Exame');
               writeln('Você precisa tirar',10-M, 'no Exame');
          end
       ELSE
          begin
               writeln('Você deveria ter estudado mais e/ou faltado
menos.');
               writeln('Reprovado');
          end;
  readln;
end.
```

Obs:Quando existe mais de uma condição, essas devem estar entre parênteses.

Exercícios: Escreva um algoritmo para cada um dos exercícios a seguir:

a) Dados três valores X, Y e Z, verificar se eles podem ser os comprimentos dos lados de um triângulo e, se forem, verificar se é um triângulo eqüilátero, isósceles ou escaleno. Se não forem um triângulo escrever uma mensagem. Considerar que:

- O comprimento de cada lado de um triângulo é menor que a soma dos outros dois lados;
- Chama-se triângulo eqüilátero o que tem 3 lados iguais;
- Chama-se triângulo isosceles o que tem o comprimento de 2 lados iguais
- Chama-se triângulo escaleno o que tem 3 lados diferentes

b) Efetuar a leitura de três valores (variáveis A,B e C) e apresentar os valores em ordem crescente.

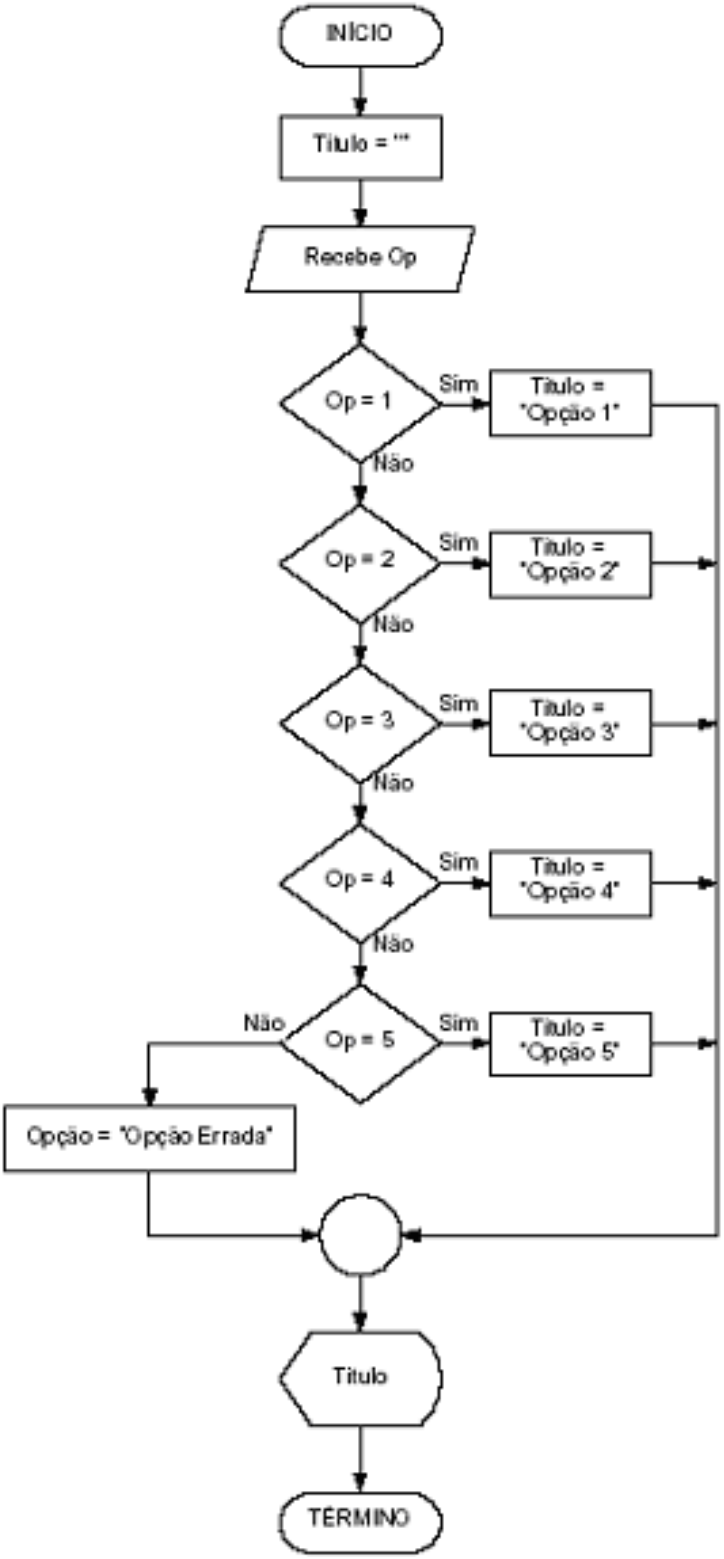
c) Efetuar a leitura de três valores (variáveis A,B e C) e efetuar o cálculo da equação do segundo grau, apresentando as duas raízes, se para os valores informados for possível efetuar o referido cálculo.

d) Certa máquina realiza a cobrança automática de valores até R$ 50,00. Ela é informada que valor deve ser cobrado, recebe uma nota de R$ 50,00 e precisa devolver o troco. As notas disponíveis para o troco são de R$ 10,00, R 5,00 e R$ 1,00. Desenvolva um algoritmo que calcule o troco e informe quantas notas de cada valor devem ser usadas. Os valores não necessários não devem ser informados.

e) Faça um programa que determine a data cronologicamente maior de duas datas fornecidas pelo usuário. Cada data deverá ser formada por três números inteiros (1º para dia, 2º para mês e 3º para ano).

Múltiplas escolhas Estrutura CASE

Em alguns programas existem comandos mutuamente exclusivos, isto é, se um comando for executado, os demais não serão. Quando este for o caso, um comando seletivo é o mais indicado, e este comando seletivo em Pascal é o CASE OF, veja exemplos:



No exemplo do diagrama de blocos anterior, é recebido uma variável “**Op**” e testado seu conteúdo, caso uma das condições seja satisfeita, é atribuído para a variável Titulo a String “Opção X”, caso contrário é atribuído a string “Opção Errada”.

Exemplo em Pascal:

```
program ex_8;
uses crt;
var
  idade:integer;
Begin
  Clrscr;
  writeln('Digite a idade:');
  readln(idade);
  CASE idade OF
       0..3:writeln('Bebê');
       4..10:Begin
               writeln('Criança');
               writeln('Aproveite para brincar bastante');
             end;
       11..18: Begin
               writeln('Adolescente');
               writeln('Acabou a brincadeira, é hora de estudar');
             end
  ELSE
       writeln('Adulto');
  end;
  readln;
end.
```

Note que na estrutura CASE temos a mesma regra: para mais de um comando dentro da condição temos que utilizar begin e end.

A condição pode ser uma lista de valores aceitáveis:

CASE Letra OF

‘A’ ,’E’ ,’I ‘’O’ ,’U’ : writeln(‘Vogais’);

FIM;

Obs.: O comando CASO só aceita valores do tipo CHAR,INTEGER ou BOOLEAN.

Exercícios:

1.Escrever um algoritmo que dada uma variável inteira DiaSem, com valores entre 1 e 7, escreva o correspondente dia da semana.

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

Trabalho de laboratório	peso 2
Avaliação semestral	peso 3
Exame final	peso 5

Faça um programa que receba as três notas, calcule e mostre a média ponderada e o conceito que segue a tabela abaixo:

MÉDIA PONDERADA	***CONCEITO***
9,0 - 10,0	A
7,0 - 8,0	B
5,0 - 6,0	C
3,0 - 4,0	D
0,0 - 2,0	E

Obs: O Case não aceita comparação com variável real, apenas inteira, então use o comando trunc para utilizar apenas a parte inteira da média.

3.Faça um programa que receba a média do primeiro semestre e a média do segundo semestre de um aluno, calcule e mostre a média anual (sabendo que a média do segundo semestre tem peso dois e a do primeiro semestre tem peso um) e a mensagem que segue a tabela abaixo. Para alunos de exames, calcule e mostre a nota que deverá ser tirada no exame para a aprovação, considerando a seguinte fórmula: Exame = 12 - Média Anual

MÉDIA ARITMÉTICA	**MENSAGEM**
00 ● 3,0	Reprovado
3,0 ● 6,0	Exame
6,0 ● 10,0	Aprovado

4.Faça um programas que exiba as informações abaixo:

VALOR DE Opção	**FORMA DE ESCREVER**
1	A,B e C em ordem crescente.
2	A, B e C em ordem decrescente.
3	O maior fica entre os outros dois números.

Em seguida, solicite ao usuário que digite a opção desejada (Opção é um valor inteiro entre 1 e 3) e os valores para A,B e C. Escreva os números A,B e C obedecendo a ordem escolhida pelo usuário.

Observação: Caso o valor digitado para opção seja inválido, ou seja, diferente de 1, 2 e 3, exibir mensagem de opção inválida.

5.Faça um programa que mostre o menu de opções a seguir, receba a opção do usuário e os dados necessários para cada opção e mostre o resultado
Menu de opções:
Somar os dois números
Raiz quadrada de um número
Digite a opção desejada:

Processos de Repetição

Muitas vezes temos a necessidade de repetir um trecho de programa. Neste caso, será criado o chamado looping ou laço de repetição, que executa um determinado conjunto de comandos tantas vezes quantas forem necessárias.

Looping com variável de controle

Os loopings que possuem um número pré-determinado de execuções poderão ser processados através da estrutura PARA.

Estrutura de repetição PARA-FAÇA (FOR ... TO... DO)

Essa estrutura tem o seu funcionamento controlado por uma variável denominada contador. Sendo assim, poderá executar um determinado conjunto de instruções um determinado número de vezes.

Exemplo em Pascal

```
program ex_9;
uses crt;
var
  N:integer;
Begin
  Clrscr;
  writeln('Escrevendo os números de 1 a 100');
  FOR N:=1 TO 100 DO
       write(N,' - ');
  readln;
end.
```

Nesse exemplo, inicializamos a variável N com o valor 1 e esse valor será incrementado de 1 em 1 até alcançar o limite superior de 100. Para cada incremento será escrito o valor de I e um hífen para separar os valores, como pede o comando writeln, (como esse looping é composto de um único comando, não usamos begin e end.)

Se quisermos que a contagem seja regressiva devemos utilizar FOR DOWNTO DO

Exemplo em Pascal

```
program ex_10;
uses crt;
var
  N:integer;
Begin
  Clrscr;
  writeln('Escrevendo os números de 10 a 0');
  FOR N:=10 DOWNTO 0 DO
       begin
          clrscr;
          write(N);
       end;
  readln;
end.
```

Exercícios:

Escreva um algoritmo (fluxograma e pseudocódigo) para cada um dos exercícios a seguir:

a) Escrever os números pares entre 0 e 50.

b) Escrever um algoritmo/programa em Pascal que escreve a soma dos números múltiplos de 7 entre 100 e 200.

c) Escrever um algoritmo/programa em Pascal que escreve a soma dos números que não são múltiplos de 13 entre 100 e 200.

d) Apresentar o total da soma obtida dos cem primeiros números inteiros (1 + 2 + ... + 99 + 100)

e) Calcular a soma de dez números quaisquer fornecidos pelo usuário.

f) Apresentar os resultado de um tabuada de um número qualquer. Esta deverá ser impressa no seguinte formato:

Considerando o número 2

2 X 1 = 2

2 X 2 = 4

2 X 3 = 6

(...)

2 X 10 = 20

g) Calcular o Fatorial de um número qualquer solicitado pelo usuário. Sabendo que: 5! = 5 * 4 * 3 * 2 * 1 = 120

h) Faça um programa que escreva todas as letras do alfabeto em ordem . Obs. A variável de controle deve ser ordinal (ou seja, pode ser integer, mas também char).

i) Faça um programa que simule um relógio digital no meio da tela com hora,minuto e segundo.

j) Ler 5 valores para a, um de cada vez, e conta quantos destes valores são negativos, escrevendo esta informação.

k) Escreva um programa que apresente a série e Fibonacci até o n-ésimo termo. A série de

Fibonacci é formada pela sequência: 1, 1, 2, 3, 5, 8, 13,21,34,...etc. Esta série caracteriza-se pela soma de um termo posterior com seu anterior subsequente.

l) Uma empresa decide dar um aumento de 30% aos funcionários cujo salário é inferior a 500. O algoritmo deve permitir o cálculo do salário reajustado de 10 funcionários segundo o mesmo critério de reajuste.

2) Escreva um programa usando um for dentro do outro que peça a altura e a largura de um retângulo e desenhe-o totalmente preenchido e formado por asteriscos.

Ex. : **********

****

****

3) Faça um programa que cadastre 10 pessoas, onde será perguntado o sexo e a idade da pessoa. Ao final será exibida a média de idade do sexo masculino e feminino, e a maior e menor idade cadastrada.

4) Escrever um algoritmo/programa em Pascal que lê 20 valores, um de cada vez, e conta quantos deles estão em cada um dos intervalos [0, 25], (25, 50], (50, 75], (75, 100], escrevendo esta informação.

5) Escrever um algoritmo/programa em Pascal semelhante ao anterior que calcula as médias aritméticas de cada intervalo e as escreve, juntamente com o número de valores de cada intervalo.

6) Escrever um algoritmo/programa em Pascal que lê um número e calcula e escreve quantos divisores ele possui.

7) Escrever um algoritmo/programa em Pascal que lê um número e escreva se ele "é primo" ou "não é primo"

8) Escrever um algoritmo/programa em Pascal que lê um valor X e calcula e escreve os 20 primeiros termos da série: $1 + 1/x^2 + 1/x^3 + 1/x^4 + ...$ Considere que para calcular x^i no pascal utiliza-se a seguinte sintaxe: exp(ln(x)*i)

9) Escrever um algoritmo/programa em Pascal que lê N, inteiro e positivo e calcula e escreve o termo de ordem N da sucessão abaixo:

ordem:	1	2	3	4	5	6	7	8	...
sucessão:	-1	0	5	6	11	12	17	18	...

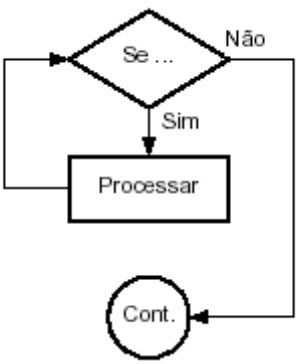
Looping com teste lógico no início

Estrutura de Repetição ENQUANTO-FAÇA (WHILE DO)

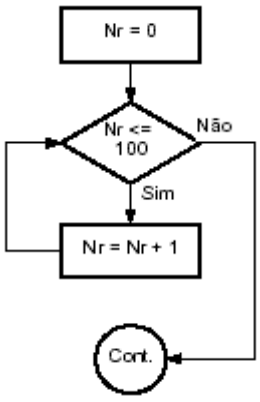
Neste caso, o bloco de operações será executado enquanto a condição x for verdadeira. O teste da condição será sempre realizado antes de qualquer operação.

Enquanto a condição for verdadeira o processo se repete. Podemos utilizar essa estrutura para trabalharmos com contadores.

Em diagrama de bloco a estrutura é a seguinte:



Exemplo de Contador:



Algoritmo

ALGORITMO ExEnquanto
ENQUANTO <Condição for verdadeira> FAÇA
    <Comandos>
FIM ENQUANTO

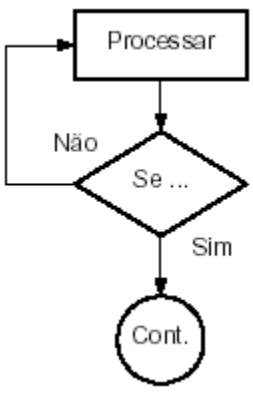
Exemplo em Pascal

```
program ex_enquanto;
uses crt;
var
  N1,N2,N3,M:real;
  Cont:integer;
Begin
  Clrscr;
  Cont:=1;
  While cont<=40 do
       begin
         writeln('Digite três notas:');
         readln(N1,N2,N3);
         M:=(N1+N2+N3)/3;
         writeln('Média = ',M:4:1);
         cont:=cont+1;
        end;
  readln;
end.
```

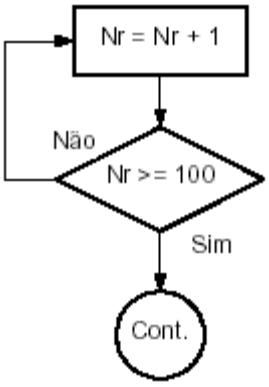
Esse programa vai calcular a média de 40 alunos. Perceba que antes da estrutura While, é inicializado o contador com valor 1. O programa verifica que o valor do contador (1) é menor que 40, então ele executa os comandos do loop, definidos entre o begin e o end. Depois de calcular e exibir a média do 1º aluno, executa o comando cont:=cont+1 que faz com que o contador seja incrementado em 1 unidade e retorna para fazer a verificação. Perceba que depois de calcular e exibir a média do 40º aluno, o contador passa para 41 e retorna para fazer a verificação; como cont não é menor ou igual a 40 ele não executa os comandos do loop e pula para o próximo comando após o end; que encerra o loop.

Estrutura de repetição REPITA-ATÉ (Repeat Until)

Neste caso, executa-se primeiro o bloco de operações e somente depois é realizado o teste de condição. Se a condição for verdadeira, o fluxo do programa continua normalmente. Caso contrário são processados novamente os comandos antes do teste da condição.



Exemplo:



Algoritmo

REPITA

<Comandos>

ATE <Condição for verdadeira>

Exemplo:

Esse programa calcula a média de três notas e verifica se o usuário deseja calcular outra média, se sim, ele retorna, se não ele encerra o programa.

```
program ex_repita;
uses crt;
var
  N1,N2,N3,M:real;
  Resposta:char;
Begin
  Clrscr;
  REPEAT
        writeln('Digite três notas:');
        readln(N1,N2,N3);
        M:=(N1+N2+N3)/3;
        writeln('Média = ',M:4:1);
        writeln('Deseja calcular nova media? (S para sim N para Não)');
        readln(resposta);
        resposta:=upcase(resposta);
  UNTIL resposta='N';
  readln;
end.
```

Exercícios:

1) Quais são as estruturas de repetição existentes no Pascal ?

2) Qual a principal diferença entre o WHILE-DO e o REPEAT-UNTIL ?

3) Em que situações é mais indicado o uso da estrutura FOR ?

4) Em que situações não podemos utilizar a estrutura FOR ?

5) Escreva um programa Pascal que leia um conjunto de 100 números inteiros positivos e determine a quantidade de números pares e números ímpares contidos no mesmo.

6) Dado o trecho de programa abaixo:

```
readln(N)
R := 1;
I := 2;
while I <= N-1 do
begin
  R := R * 2;
  I := I + 1;
end;
write(R);
```

Reescreva-o utilizando: a) o comando FOR b) o comando REPEAT

7) Escreva um programa Pascal que apresente o menu de opções abaixo:

OPÇÕES:
1 -SAUDAÇÃO
2 -BRONCA
3 -FELICITAÇÃO
0 -FIM

O programa deve ler a opção do usuário e exibir, para cada opção, a respectiva mensagem:

1 -Olá. Como vai ?
2 -Vamos estudar mais.
3 -Meus Parabéns !
0 -Fim de serviço.
Enquanto a opção for diferente de 0 (zero) deve-se continuar apresentando as opções.
Obs: use como estrutura de repetição o comando REPEAT e como estrutura condicional o comando CASE.

8) Faça um programa que leia 3 valores inteiros (N, X, Y) e mostre todos os números múltiplos de N entre X e Y.

9) Um programa que receba três notas de um aluno, calcule e mostre a média aritmética e a mensagem que segue a tabela abaixo. Para alunos de exame, calcule e mostre a nota que deverá ser tirada no exame para aprovação, considerando que a média no exame é 6.0.

Média	Mensagem
0,0 até 3,0	Reprovado
3,0 (inclusive) até 7,0	Exame
7,0 (inclusive) até 10,0 (inclusive)	Aprovado

10) Faça um programa que receba o código correspondente ao cargo de um funcionário e o seu salário atual e mostre o cargo, o valor do aumento e seu novo salário. Os cargos na tabela a seguir:

Código	Cargo	Percentual
1	Escriturário	50%
2	Secretário	35%
3	Caixa	20%
4	Gerente	10%
5	Diretor	Não tem aumento

11) Faça um programa que receba a idade de um nadador e mostre a sua categoria usando as regras a seguir:

a. Infantil de 5 a 7;
b. Juvenil de 8 a 10;
c. Adolescente de 11 a 15;
d. Adulto de 16 a 30;
e. Sênior acima de 30.

12) Faça um programa que leia o número de termos, determine e mostre os valores de acordo com a série abaixo:

Série 2, 7, 3, 4, 21, 12, 8, 63, 48, 16, 189, 192, 32, 567, 768, 64.....

13) Faça um programa que receba vários números, calcule e mostre:

a) a soma dos números digitados;
b) a quantidade de números digitado;
c) a média dos números digitados;
d) o maior número digitado;

e) o menor número digitado;
f) a média dos números pares.

14) Faça um programa que leia um valor n, inteiro e positivo, calcule e mostre a soma S da seguinte forma: S= 1 + 1/2 +1/3+1/4+...+1/n

15) Uma loja utiliza o código V para transação a Vista e P para transação a prazo. Faça um programa que receba o código e o valor das transações e mostre:
o valor total das vendas a vista;
o valor total das vendas a prazo;
o valor total das vendas efetuadas.

16) A prefeitura de uma cidade fez uma pesquisa entre seus habitantes, coletando dados sobre o salário e o número de filhos. A prefeitura deseja saber:
a média dos salários da população;
a média dos número de filhos;
o maior salário
a porcentagem de pessoas com salários até R$ 320,00

17) Cada espectador de um cinema respondeu a um questionário no qual constava sua idade e a sua opinião em relação ao filme: ótimo, bom e regular. Faça um programa que recebe a idade e a opinião das pessoas, calcule e mostre ao final:
a média das idades das pessoas que responderam ótimo;
a quantidade de pessoas que respondeu regular;
a porcentagem de pessoas que respondeu bom entre os entrevistados.

18) Foi feita uma estatística em cinco cidades brasileiras para coletar dados sobre acidentes de trânsito. Foram obtidos os seguintes dados:

Código da cidade;

Número de veículos de passeio em 2016;

Número de acidentes de trânsito com vítimas em 2016.

Deseja-se saber:

qual o maior e o menor índice de acidentes de trânsito;

qual a média de veículos nas cinco cidades juntas;

qual a média de acidentes de transito nas cidades com menos de 2000 veículos de passeio.

19) Uma empresa possui 10 funcionários com as seguintes características: código, número de horas trabalhadas no mês, turno de trabalho(M-Matutino, V- Vespertino ou N-Noturno), categoria (O-Operário ou G-Gerente), valor da hora trabalhada. Sabendo-se que essa empresa deseja informar sua folha de pagamento, faça um programa que:

Leia as informações dos funcionários, exceto o valor da hora trabalhada, não permitindo que sejam informados turnos nem categorias inexistentes. Trabalhar sempre com a digitação de letras maiúsculas;

Calcule o valor da hora trabalhada de acordo com a tabela s seguir:

Categoria	Turno	Valor da Hora trabalhada
G	N	18% do salário mínimo
G	M ou V	15% do salário mínimo
O	N	13% do salário mínimo
O	M ou V	10% do salário mínimo

Adote o valor de R$410,00 para o salário mínimo.

Calcule o salário inicial dos funcionários com base no valor da hora trabalhada e no número de horas trabalhadas.

Calcule o valor do auxílio-alimentação recebido por funcionário de acordo com o seu salário inicial, conforme a tabela a seguir:

Salário Inicial	Auxílio Alimentação
Até R$500,00	20% do salário inicial
Entre R$500,00 e R$800,00	15% do salário inicial
Acima de R$800,00	5% do salário inicial

Mostre o código, número de horas trabalhadas, valor da hora trabalhada, salário inicial, auxílio-alimentação e o salário final(salário inicial + auxálio-alimentação)

20) Uma empresa contratou 15 funcionários temporários. De acordo com o valor de vendas mensais, os funcionários adquirem pontos que determinarão seus salários ao final de cada mês. Sabe-se que esses funcionários trabalharão nos meses de novembro a janeiro do ano subseqüente.

Faça um programa que:

Leia as pontuações nos três meses de cada funcionário;

Calcule e mostre a pontuação geral de cada funcionário nos três meses;

Calcule e mostre a média das pontuações de cada funcionário nos três meses

Determine e mostre a maior pontuação atingida entre os funcionários nos três meses.

Bibliografia:

Saliba, Walter Luiz Caram

Técnicas de Programação

Makron Books

Manzano,José A e Oliveira Jayr F

Algoritmos

Editora Érica

Apostila do Curso Básico de Lógica de Programação

Paulo Sérgio de Moraes

Centro de Computação - Unicamp