

Índice

| | | |
|-------|--|----|
| 1 | Introdução à lógica de programação..... | 4 |
| 1.1 | Lógica de programação..... | 4 |
| 1.2 | Sequência Lógica..... | 4 |
| 1.3 | Instruções..... | 4 |
| | Algoritmo..... | 4 |
| 1.5 | Programas | 5 |
| 1.6 | Exercícios..... | 6 |
| | Desenvolvendo algoritmos..... | 7 |
| 2.1 | Pseudocódigo..... | 7 |
| 2.2 | Regras para escrever um Algoritmo..... | 7 |
| 2.3 | Fases..... | 7 |
| 2.4 | Exemplo de algoritmo | 8 |
| 2.5 | Teste de mesa..... | 9 |
| 2.6 | Exercícios..... | 10 |
| | Diagrama de blocos..... | 11 |
| | O que é?..... | 11 |
| | Exercícios..... | 13 |
| | Constantes e variáveis..... | 14 |
| 4.1 | Constantes..... | 14 |
| 4.2 | Variáveis..... | 15 |
| 4.3 | Tipos..... | 16 |
| 4.4 | Exercícios..... | 17 |
| | Operadores..... | 19 |
| 5.1 | Operadores Aritméticos..... | 19 |
| 5.2 | Operadores Relacionais..... | 20 |
| 5.3 | Operadores Lógicos | 22 |
| 5.4 | Exercícios..... | 24 |
| 6 | Operações lógicas | 25 |
| 6.1 | Exercícios..... | 27 |
| 7 | Estrutura de Decisão e Repetição..... | 30 |
| 7.1 | Comandos de Decisão..... | 30 |
| 7.1.1 | SE ENTÃO ou IF..... | 31 |
| 7.1.2 | SE ENTÃO SENAO ou IF ... ELSE..... | 32 |
| 7.1.3 | CASO SELECIONE ou SE ... ELSE ... SE ... ELSE..... | 34 |
| 7.1.4 | Exercícios..... | 37 |
| 7.2 | Comandos de Repetição..... | 39 |
| 7.2.1 | Enquanto X, processar..... | 40 |
| 7.2.2 | Até que X, processar | 41 |
| 7.2.3 | Processar ..., enquanto X..... | 42 |
| 7.2.4 | Processar ..., até que X..... | 43 |
| 7.2.5 | Exercícios..... | 44 |
| 8 | Arquivos de Dados..... | 45 |
| 8.1 | Conceitos Básicos | 45 |
| 8.2 | Abertura de arquivos..... | 46 |
| 8.3 | Fechamento de arquivos | 46 |
| 8.4 | Leitura de arquivos..... | 47 |

| | | |
|-----|---|----|
| 8.5 | Movimentação de registros..... | 49 |
| 8.6 | Gravação de arquivos..... | 49 |
| 8.7 | Macro Fluxo | 50 |
| 8.8 | Exercícios..... | 51 |
| 9 | Relatórios..... | 52 |
| 9.1 | Características do Formulário..... | 52 |
| 9.2 | Controle de linhas e salto de páginas..... | 52 |
| 9.3 | Impressão de Cabeçalho e Estética de Página | 53 |
| 9.4 | Exercícios..... | 55 |
| 10 | Simbologia..... | 56 |
| 11 | Vamos brincar com a lógica..... | 57 |

1 Introdução à lógica de programação

1.1 Lógica de programação

A lógica de programação é necessária para pessoas que desejam trabalhar com o desenvolvimento de sistemas e de programas, pois ela permite definir a **seqüência lógica** para o desenvolvimento.

Definição: Lógica de programação é a técnica de encadear pensamentos para atingir determinado objetivo.

1.2 Seqüência Lógica

Esses pensamentos, podem ser descritos como uma seqüência de instruções, que devem ser seguidas para se cumprir uma determinada tarefa.

Definição: Seqüência lógica são os passos executados até atingir um objetivo ou solução de um problema.

1.3 Instruções

Na linguagem comum, instruções são um conjunto de regras ou normas definidas para a realização ou emprego de algo.

Em informática, instrução é a informação que indica a um computador uma ação a executar. Convém ressaltar que uma ordem isolada não permite realizar o processo completo, para isso é necessário um conjunto de instruções colocadas em uma seqüência lógica.

Por exemplo, se quisermos fazer uma batida de frutas, precisamos por em prática uma série de instruções, como, pegar liquidificar, lavar a maçã, descascar a banana etc.

É evidente que estas instruções deverão ser executadas em uma ordem adequada, pois não podemos colocar as frutas no liquidificador se não tirarmos a tampa.

1.4 Algoritmo

Lógica de Programação

Um algoritmo é formalmente uma seqüência finita de passos que levam a execução de uma tarefa. Podemos pensar em algoritmo como uma receita, uma seqüência de instruções que dão cabo de uma meta específica. Estas tarefas não podem ser redundantes nem subjetivas na sua definição, devem ser claras e precisas.

Como exemplos de algoritmos podemos citar os algoritmos das operações básicas (adição, multiplicação, divisão e subtração) de números reais decimais.

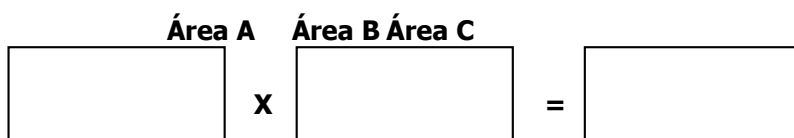
Até mesmo as coisas mais simples, podem ser descritas por seqüências lógicas, por exemplo:

Comer chocolate:

- Pegar o chocolate;
- Retirar a embalagem;
- Comer o chocolate;
- Jogar a embalagem no lixo.

Multiplicar dois números

- Escreva o primeiro número na área A;
- Escreva o segundo número na área B;
- Multiplique o número da área A pelo número da área B e coloque o resultado na área C.



1.5 Programas

Os programas de computadores nada mais são do que algoritmos escritos numa linguagem de computador (Pascal, C, Cobol, Fortran, entre outras) e que são interpretados e executados por uma máquina, no caso um computador.

Notem que dada esta interpretação rigorosa, um programa é por natureza muito específico e rígido em relação aos algoritmos da vida real.

1.6 Exercícios

- 1) Crie uma seqüência lógica para fazer uma ligação telefônica:

- 2) Faça um algoritmo para subtrair dois números e multiplicar o resultado por 1,50;

3) Faça um algoritmo, descrevendo todos os detalhes, para trocar uma lâmpada; e;

4) Descreva com detalhes a seqüência lógica para se trocar um pneu de um carro.

2 Desenvolvendo algoritmos

2.1 Pseudocódigo

Os algoritmos são descritos em uma linguagem chamada pseudocódigo. Este nome é uma alusão à posterior implementação em uma linguagem de programação, ou seja, quando formos programar em uma linguagem, por exemplo **COBOL**, estaremos gerando código em **COBOL**. Por isso os algoritmos são independentes das linguagens de programação. Ao contrário de uma linguagem de programação não existe um formalismo rígido de como deve ser escrito o algoritmo.

O algoritmo deve ser fácil de se interpretar e fácil de codificar. Ou seja, ele deve ser o intermediário entre a linguagem falada e a linguagem de programação.

2.2 Regras para escrever um Algoritmo

Para escrever um algoritmo precisamos descrever a seqüência de instruções, de maneira simples e objetiva. Para isso utilizaremos algumas técnicas:

- ☞ Usar somente um verbo por frase;
- ☞ Imaginar que você está desenvolvendo um algoritmo para pessoas que não trabalham com informática;
- ☞ Usar frases curtas e simples;
- ☞ Ser o mais objetivo possível;
- ☞ Procurar usar palavras que não tenham sentido dúbio.

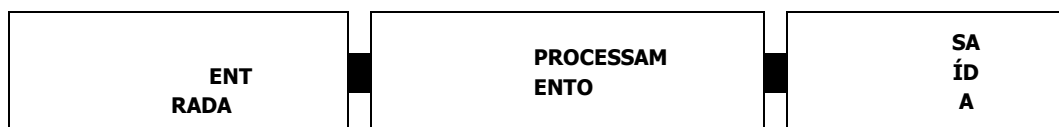
2.3 Fases

Como vimos no capítulo anterior que ALGORITMO é uma seqüência lógica de instruções que podem ser executadas, é importante ressaltar que qualquer tarefa que siga determinado padrão, pode ser descrita por um algoritmo, como por exemplo:

⌚ **Como fazer brigadeiro (doce)** ou ainda;

⌚ **Calcular os juros (simples) de determinada quantia**

Entretanto, para se montar um algoritmo, precisamos primeiramente dividir o problema apresentado em 3 (três) fases fundamentais:



Onde:

ENTRADA são os dados de entrada para o algoritmo;

PROCESSAMENTO são os procedimentos utilizados para chegar ao resultado; e;

SAÍDA são os dados já processados.

2.4 Exemplo de algoritmo

Calcular o juros (simples) de determinada quantia, onde **J = Pin**.

Para que o algoritmo seja montado, faremos 3 (três) perguntas:

- 1) Quais são os dados de entrada?
R.: Os dados de entrada são **P** (valor presente), **i** (taxa real) e **n** (período)
- 2) Qual será o procedimento a ser executado? R.: O procedimento será:
 - a) Dividir taxa por 100;
 - b) Multiplicar valor presente por taxa e posteriormente por período

$P * (Taxa / 100) * período$
- 3) Qual será o dado de saída?
R.: O dado de saída será o valor de juros (simples) calculado

A seguir será mostrado a elaboração do algoritmo:

- Receber o valor de **P** (valor presente);
- Receber o valor de **i** (taxa);
- Receber o valor de **n** (período);
- Dividir **i** por **100**;

- Multiplicar todos os valores ($P * i * n$);
- Mostrar o resultado da multiplicação

Após a elaboração do algoritmo, sempre devemos efetuar um teste, chamado de **TESTE DE MESA**.

2.5 Teste de mesa

Significa seguir as instruções do algoritmo de maneira precisa, para verificar se o procedimento utilizado está correto ou não.

Exemplo:

| P (valor presente) | i (taxa) | n (período) | J (juros simples) |
|--------------------|----------|-------------|-------------------|
| 1000,00 | 10 | 6 | - |
| 1000,00 | 0,10 | 6 | 600,00 |

2.6 Exercícios

- 1) Identifique os dados de entrada, processamento e saída no algoritmo abaixo:
 - Receber o código da peça;
 - Receber a quantidade em estoque;
 - Receber o valor da matéria prima;
 - Receber o valor da mão de obra;
 - Receber o valor das despesas de fábrica;
 - Calcular o custo unitário (valor da matéria prima * valor da mão de obra * valor das despesas de fábrica);
 - Calcular o custo total (custo unitário * quantidade em estoque);
 - Mostrar o código da peça, quantidade em estoque, custo unitário e o custo total.

- 2) Faça um algoritmo para calcular a média entre 2 (dois) valores, sendo que **MEDIA = ((VALOR1 + VALOR2) / 2)**.

- 3) Faça o teste do algoritmo do exercício anterior.

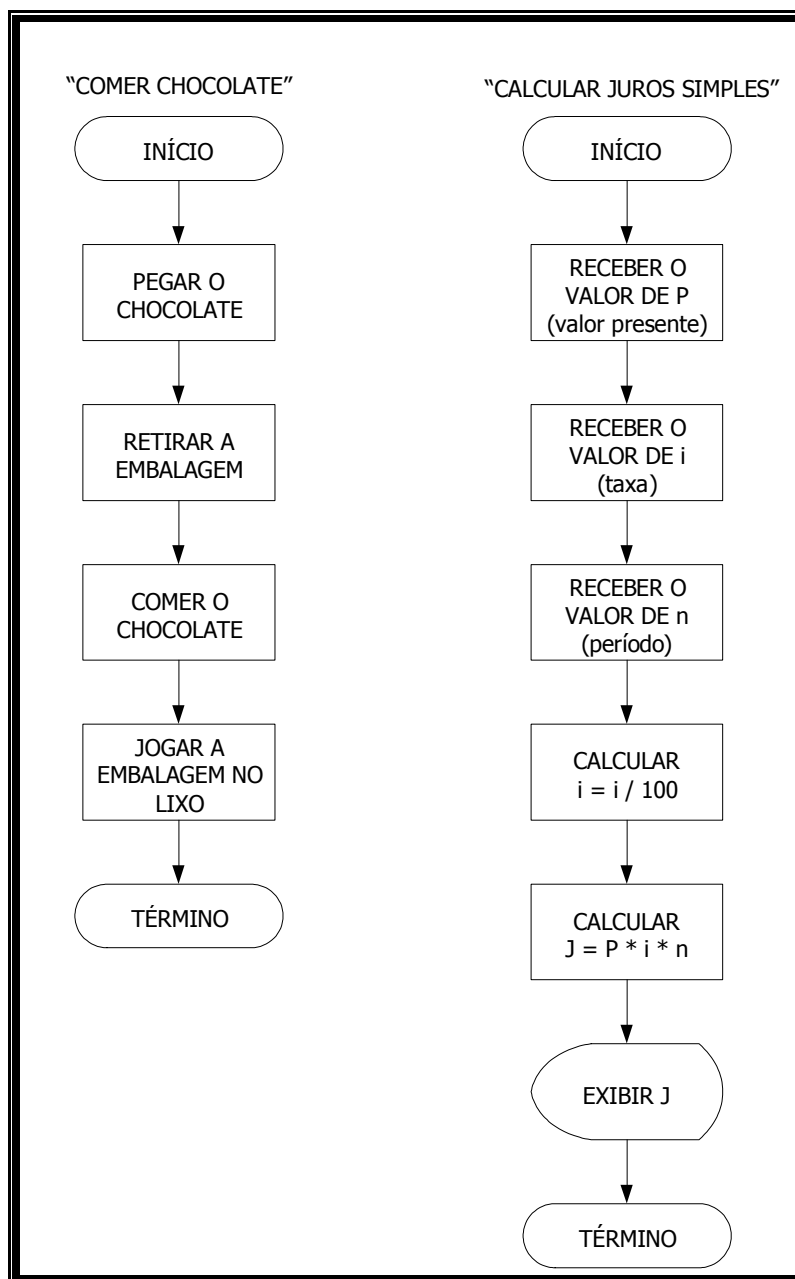
3 Diagrama de blocos

3.1 O que é?

O diagrama de blocos é uma forma padronizada e eficaz para representar os passos lógicos de um determinado processamento.

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.

Exemplos de diagrama de blocos:



Observações

- Podemos observar que no exemplo do chocolate seguimos uma seqüência lógica, somente com informações diretas, enquanto no exemplo dos juros, utilizamos cálculos e exibimos o resultado obtido;
- Vide item 10 para qualquer dúvida a respeito da simbologia.

3.2 Exercícios

1) Faça um diagrama de blocos que:

- Leia a cotação do DÓLAR;
- Leia a cotação do EURO;
- Converta a cotação do DÓLAR em REAIS;

FUTURE SCHOOL – Cursos de Computação

www.osasconamao.com.br/cursos

Fone: (0XX11) 98342.2503

Página 9 de 59

- Converta a cotação do EURO em REAIS; • Calcule a diferença entre as cotações;
 - Mostrar a diferença.
- 2) Faça um diagrama de blocos que:
- Leia 3 (três) números distintos;
 - Eleve esses números ao cubo;
 - Some os valores obtidos;
 - Mostre os números obtidos; • Mostre o valor obtido (soma).
- 3) Faça um algoritmo para pagamento de 10% de comissão a um vendedor de uma loja de auto peças, em relação à sua venda, considerando as informações abaixo:
- Identificação do vendedor;
 - Código da peça; • Preço unitário da peça;
 - Quantidade vendida.
- 4) Faça um diagrama de blocos juntamente com o teste de mesa para o exercício acima.

4 Constantes e variáveis

Constantes e variáveis são os elementos básicos que um programa manipula.

Uma variável é um espaço reservado na memória do computador para armazenar um tipo de dado determinado.

As variáveis devem receber nomes para poderem ser referenciadas e modificadas quando necessário.

Um programa deve conter declarações que especificam de que tipo são as variáveis que ele utilizará e às vezes um valor inicial.

Tipos podem ser por exemplo: inteiros, reais, caracteres etc.

As expressões combinam variáveis e constantes para calcular novos valores.

4.1 Constantes

Constante é um determinado valor fixo que não se modifica ao longo do tempo, durante a execução de um programa.

Conforme o seu tipo, a constante é classificada como sendo numérica, lógica e literal.

Exemplo de constantes:

VAR-01 + VAR-02 + VAR-30

CON10 ←-----→ **CONSTANTE**

4.2 Variáveis

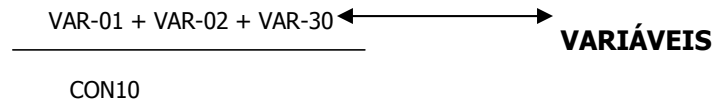
Variável é a representação simbólica dos elementos de um certo conjunto.

Lógica de Programação

Cada variável corresponde a uma posição de memória, cujo conteúdo pode se alterado ao longo do tempo, durante a execução de um programa.

Observação: embora uma variável possa assumir diferentes valores, ela só pode armazenar um valor a cada instante

Exemplos de variáveis



nome da variável conteúdo da variável



NOME = "FUTURE SCHOOL CURSOS DE COMPUTAÇÃO"

4.3 Tipos

As constantes e as variáveis podem ser basicamente de quatro tipos:

- **Numéricas**
Específicas para o armazenamento de números, que posteriormente poderão ser utilizados para cálculos. Podem ser ainda classificadas como **Inteiras** (para armazenamento de números inteiros) ou **Reais** (para armazenamento de números que possuam casas decimais);
- **Caracteres ou alfabéticas**
Específicas para armazenamento de conjunto de caracteres que não contenham números (literais), por exemplo NOME;
- **Alfanuméricas**
Específicas para dados que contenham letras e/ou números. Pode em determinados momentos conter somente dados numéricos ou somente literais. Se usado somente para armazenamento de números, não poderá ser utilizada para operações matemáticas; e;
- **Caracteres ou alfabéticas**
Armazenam somente dados lógicos que podem ser Verdadeiro ou Falso.

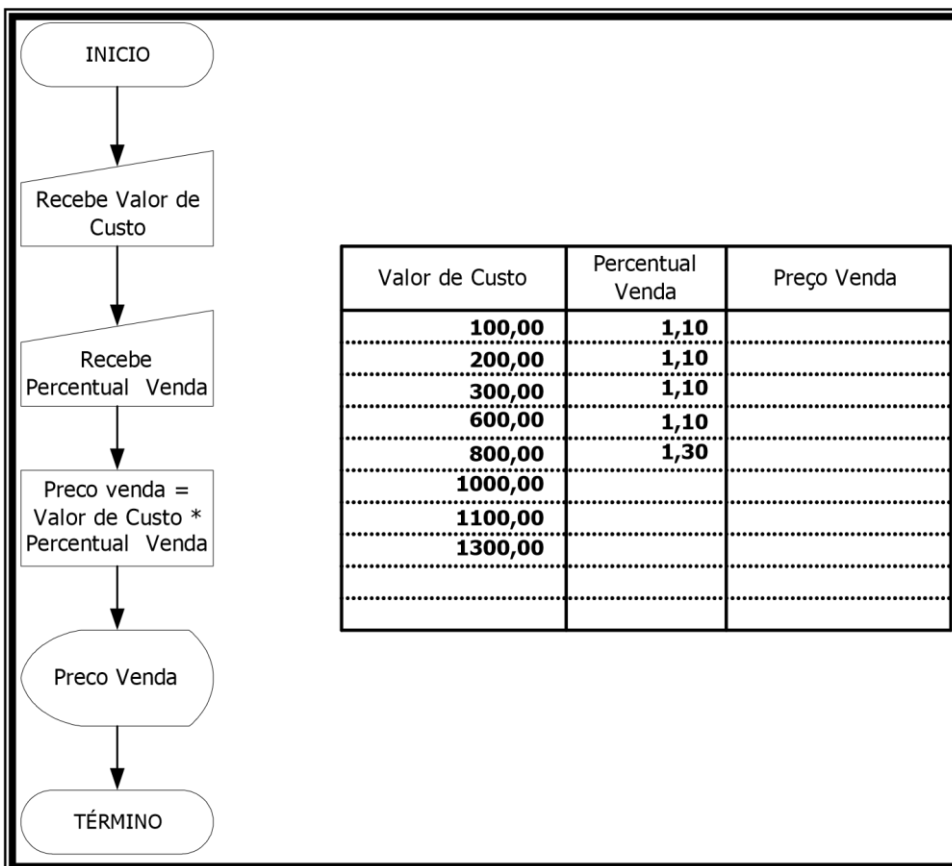
Observação: As variáveis só podem armazenar valores de um mesmo tipo, de maneira que também são classificadas como sendo numéricas, lógicas e literais

4.4 Exercícios

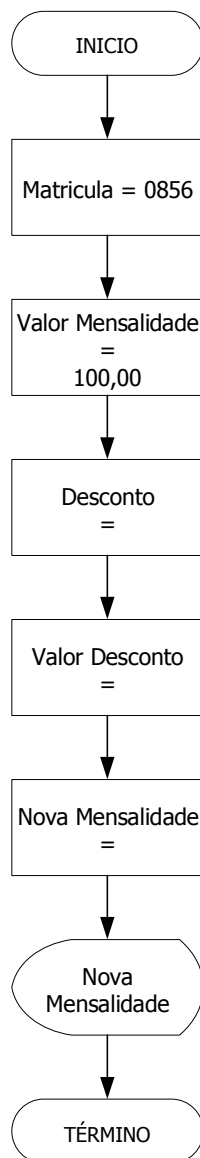
1) O que é uma constante? Cite 3 (três) exemplos.

2) O que é uma variável? Cite 3 (três) exemplos.

3) Faça um teste de mesa no diagrama de blocos abaixo e preencha a tabela ao lado com os dados do teste:



4) Sabendo-se que a escola FUTURE SCHOOL CURSOS DE COMPUTAÇÃO resolveu dar 10% de desconto para o aluno, cujo matricula é 0856, complete o diagrama abaixo:



5 Operadores

São os meios pelo qual incrementamos, subtraímos, comparamos e avaliamos dados dentro do computador.

Temos três tipos de operadores:

- Operadores Aritméticos;
- Operadores Relacionais; e;
- Operadores Lógicos

5.1 Operadores Aritméticos

São os utilizados para obter resultados numéricos.

Além da adição, subtração, multiplicação e divisão, podem utilizar também o operador para exponenciação.

Os símbolos para os operadores aritméticos são:

Lógica de Programação

| Operação | Símbolo |
|---------------|---------|
| Adição | + |
| Subtração | - |
| Multiplicação | * |
| Divisão | / |
| Exponenciação | ** |

Observação: Os operadores aritméticos seguem a hierarquia abaixo:

1º. Parênteses ();

2º. Exponenciação;

3º. Multiplicação ou divisão (o que aparecer primeiro); e; 4º.

Adição ou subtração (o que aparecer primeiro).

5.2 Operadores Relacionais

São utilizados para comparar **String** de caracteres e números.

Os valores a serem comparados podem ser caracteres ou variáveis.

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:

| Descrição | Símbolo |
|------------------|---------|
| Igual a | = |
| Diferente de | <> ou # |
| Maior que | > |
| Menor que | < |
| Maior ou igual a | >= |
| Menor ou igual a | <= |

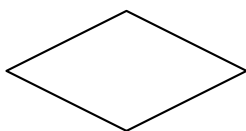
Exemplo: se tivermos a variável A = 5 e a variável B = 3, os resultados da expressão seriam.

| Expressão | Resultado |
|-----------|-----------|
|-----------|-----------|

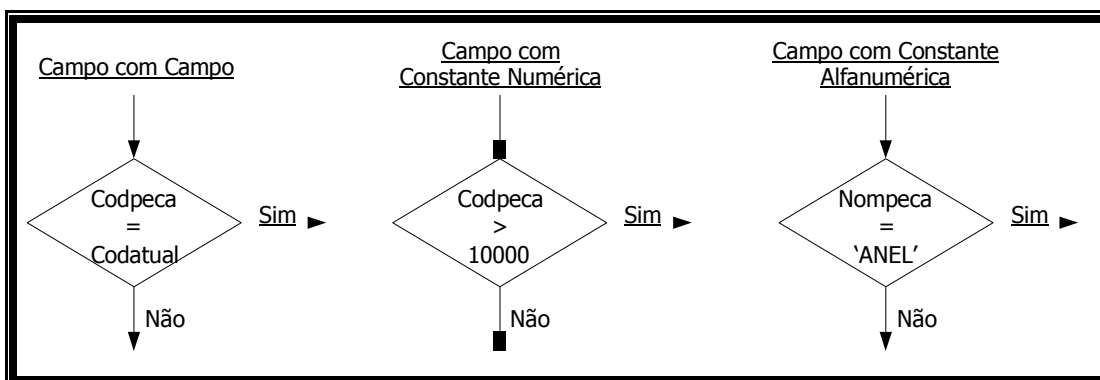
Lógica de Programação

| | |
|---------------------|-------------------|
| A = B | FALSO |
| A <> B | VERDADEIRO |
| A > B | VERDADEIRO |
| A < B | FALSO |
| A >= B | VERDADEIRO |
| A <= B | FALSO |

O símbolo utilizado para tratamento de comparações é:



Quanto à sua utilização, pode ser conforme abaixo:



5.3 Operadores Lógicos

Servem para combinar resultados de expressões, retornando se o resultado final é verdadeiro ou falso.

Os operadores lógicos são:

| | |
|------------|------------|
| E | AND |
| OU | OR |
| NÃO | NOT |

Onde:

- Uma expressão **E (AND)** é verdadeira se todas as condições forem verdadeiras;
- Uma expressão **OU (OR)** é verdadeira se pelo menos uma das condições for verdadeira; e;
- Uma expressão **NÃO (NOT)** inverte o valor da expressão ou condição, ou seja, se verdadeira inverte para falso e vice-versa.

A tabela abaixo mostra os valores possíveis criados pelos operadores lógicos:

| 1º Valor | Operador | 2º Valor | RESULTADO |
|----------|----------|----------|-----------|
| T | AND | T | T |
| T | AND | F | F |
| F | AND | T | F |
| F | AND | F | F |
| T | OR | T | T |
| T | OR | F | T |
| F | OR | T | F |
| F | OR | F | F |
| T | NOT | | F |
| F | NOT | | T |

Exemplos:

Vamos imaginar que temos as seguintes variáveis: **A = 3**, **B = 4** e **C = 2** Os resultados das expressões seriam:

Lógica de Programação

| Expressões | | | RESULTADO |
|------------|-----|---------|------------|
| $A = B$ | AND | $B > C$ | FALSO |
| $A <> B$ | OR | $B < C$ | VERDADEIRO |
| $A > B$ | NOT | | VERDADEIRO |
| $A < B$ | AND | $B > C$ | VERDADEIRO |
| $A \geq B$ | OR | $B = C$ | FALSO |
| $A \leq B$ | NOT | | FALSO |

5.4 Exercícios

- 1) Tendo as variáveis **PREÇO**, **DESCONTO**, **PRECOLIQ**, e considerando os valores abaixo. Informe se as expressões são **Verdadeiras** ou **Falsas**.

| PREÇO | DESCONTO | PRECOLIQ | EXPRESSÃO | V ou F |
|--------|----------|----------|-------------------------------|--------|
| 200,00 | 0 | 200,00 | (PRECOLIQ >= 200,00) | |
| 300,00 | 30,00 | 270,00 | (PRECOLIQ < 270,00) | |
| 500,00 | 50,00 | 450,00 | (PRECOLIQ = PREÇO – DESCONTO) | |

- 2) Sabendo-se que **A = 6**, **B = 14** e **C = 8**, informe se as expressões abaixo são **Verdadeiras** ou **Falsas**:

- a) $(A + C) > B$ ()
 b) $B >= (A + 4)$ ()
 c) $C = (B - A)$ ()
 d) $(B + A) <= C$ ()
 e) $(C + A) > B$ ()

- 3) Sabendo-se que **A = 10**, **B = 8**, **C = 6** e **D = 12**, informe se as expressões abaixo são **Verdadeiras** ou **Falsas**:

- a) $(A > C)$ **AND** $(C <= D)$ ()
 b) $(A + B) > 20$ **OR** $(A + B) = (C + D)$ ()
 c) $(A >= C)$ **AND** $(D >= C)$ ()

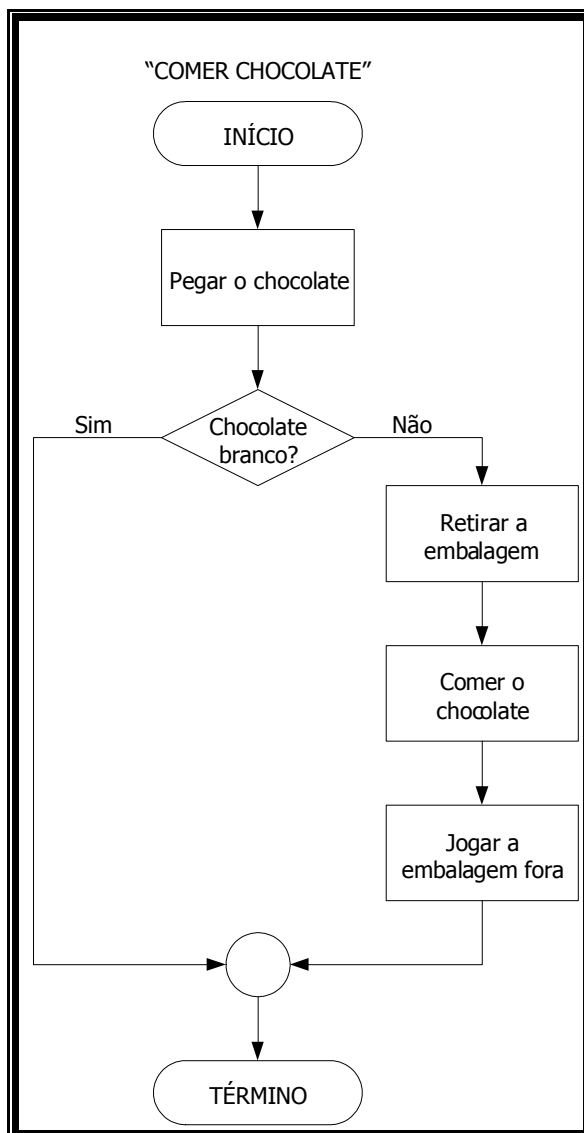
6 Operações lógicas

São utilizadas quando se torna necessário tomar decisões em um diagrama de bloco. Em um diagrama de bloco, toda decisão terá sempre como resposta o resultado **verdadeiro** ou **falso**.

Como no exemplo do algoritmo '**COMER CHOCOLATE**', vamos imaginar que algumas pessoas não gostem do chocolate branco, neste caso teremos que modificar o algoritmo para: **Comer chocolate**:

- Pegar o chocolate;
- Chocolate branco?
- Se **sim**, não coma o chocolate;
- Se **não**, continuar
- Retirar a embalagem;
- Comer o chocolate;
- Jogar a embalagem no lixo.

Com isso o diagrama de blocos passaria a ser conforme abaixo:



6.1 Exercícios

- 1) Tendo como dados de entrada a **ALTURA (h)** e **SEXO (s)** de uma pessoa, faça um algoritmo que calcule o **PESO (p)** utilizando-se das fórmulas abaixo:

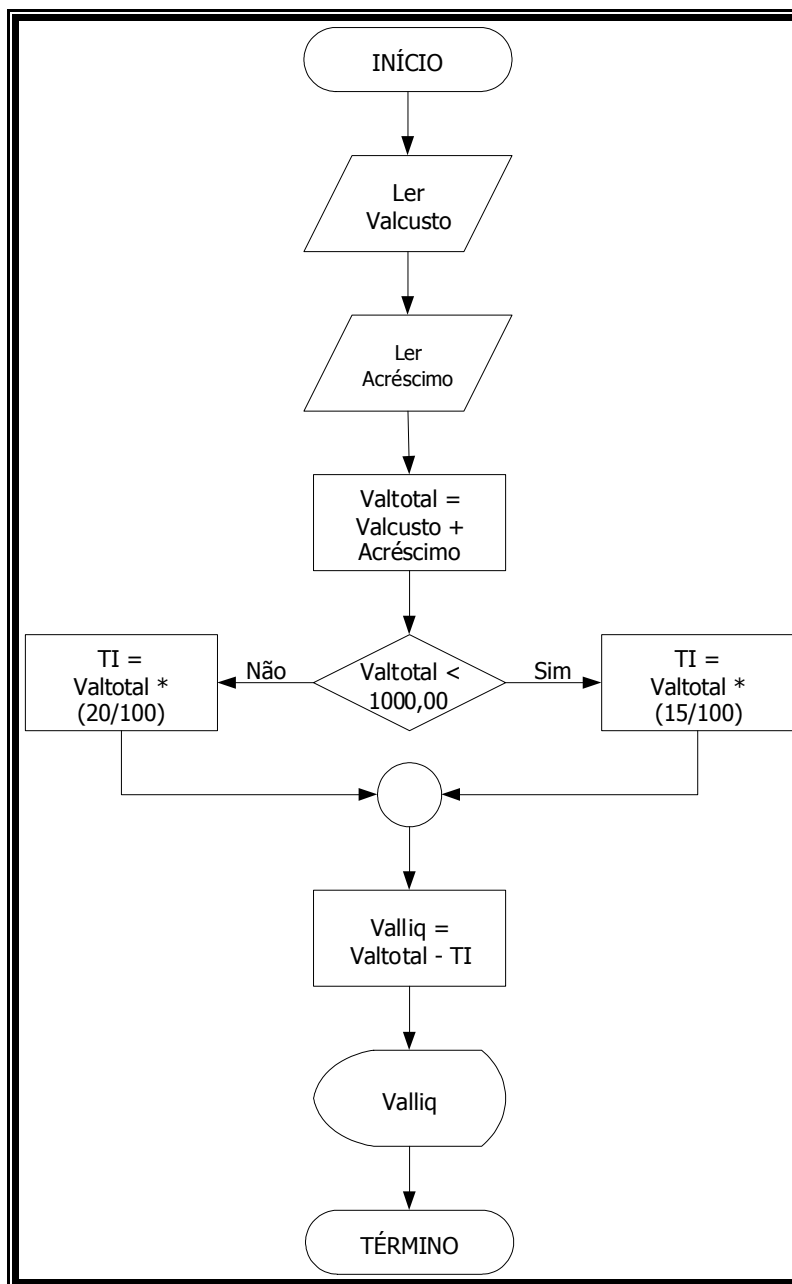
Homens $((72,7 * h) - 58)$

Mulheres $((62,1 * h) - 44,70)$

- 2) Faça um diagrama de blocos que leia um número. Se este número for maior que 20 armazene-o em **X**, caso contrario, em **Z**. No final mostrar resultado.
- 3) Ler um número e verificar se ele é par ou ímpar. Se for par armazenar este número em **NUM-PAR**, caso contrario, armazenar em **NUM-ÍMPAR**. No final exibir **NUM-PAR** e **NUM-ÍMPAR**.
- 4) Faça um diagrama de blocos para ler uma variável numérica **VAL-NUM** e exibi-la, somente se a mesma for maior que 15, caso contrario, exibi-la com valor 0 (zero).

Lógica de Programação

- 5) Faça um teste de mês do diagrama abaixo, de acordo com os valores fornecidos.



Teste o diagrama com as seguintes informações:

| Valcusto | Acrécimo |
|----------|----------|
| 3.000,00 | 1.200,00 |
| 1.200,00 | 400,00 |
| 500,00 | 100,00 |

Atualize as áreas de memória abaixo:

Lógica de Programação

| Valcusto | Acréscimo | Valtotal | TI | Valliq |
|----------|-----------|----------|----|--------|
| | | | | |
| | | | | |
| | | | | |

Atualize os dados de saída abaixo:

| Valliq |
|--------|
| |
| |
| |

Faça um algoritmo levando em conta o diagrama apresentado.

7

Estrutura de Decisão e Repetição

Como vimos no capítulo anterior em "Operações Lógicas", verificamos que na maioria das vezes precisamos tomar decisões no andamento do algoritmo, que interferem diretamente no andamento do programa.

Trabalharemos, então, com dois tipos de estrutura; a estrutura de **Decisão** e a estrutura de **Repetição**.

7.1 Comandos 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**"

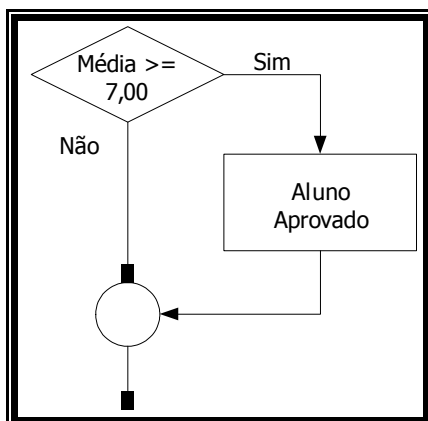
7.1.1 SE ENTÃO ou IF

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.

Imagine um algoritmo que determinado aluno somente estará aprovado se sua média for maior ou igual a 7,00, veja no exemplo de algoritmo como ficaria.

SE MEDIA >= 7,00 ENTÃO ALUNO APROVADO

Em um diagrama de blocos, ficaria assim:



Em **COBOL**, poderíamos usar uma das codificações abaixo:

```
IF MEDIA >= 7,00
  MOVE 'ALUNO APROVADO' TO TEXTO
END-IF
```

Observação: para efetuarmos uma comparação na linguagem **COBOL**, devemos obedecer a seguinte regra, de acordo com os sinais:

| Micro | Mainframe |
|-------|-------------|
| = | EQUAL |
| <> | NOT EQUAL |
| < | LESS |
| > | GREATER |
| <= | NOT GREATER |
| >= | NOT LESS |

7.1.2 SE ENTÃO SENAO ou IF ... ELSE

FUTURE SCHOOL – Cursos de Computação

www.osasconamao.com.br/cursos

Fone: (0XX11) 98342.2503

Página 23 de 59

Lógica de Programação

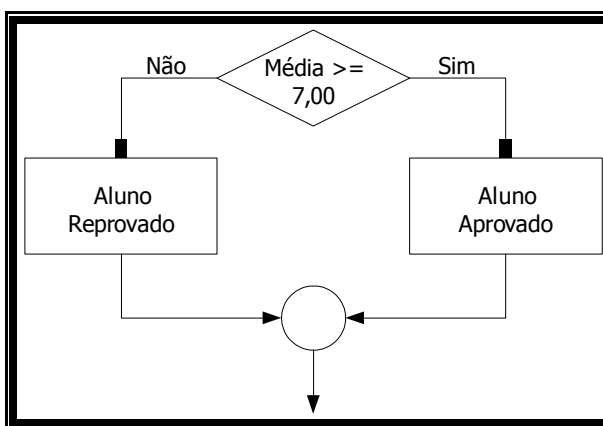
A estrutura de decisão "**SE/ENTÃO/SENÃO**", funciona exatamente como a estrutura "**SE**", com apenas uma diferença, em "**SE**" somente podemos executar comandos caso a condição seja verdadeira, diferente de "**SE/SENÃO**", pois sempre um comando será executado independente da condição, ou seja, caso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

O algoritmo como ficaria.

```

SE MEDIA >= 7,00 ENTÃO
  ALUNO APROVADO
SENÃO
  ALUNO REPROVADO
  
```

Em um diagrama de blocos, ficaria assim:

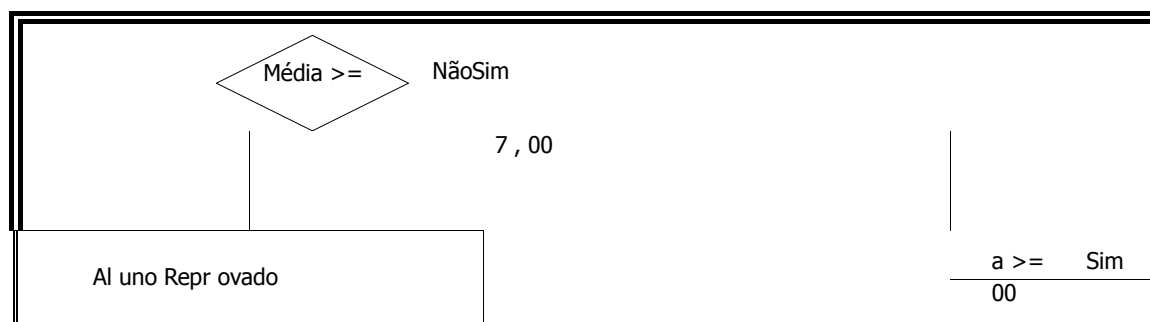


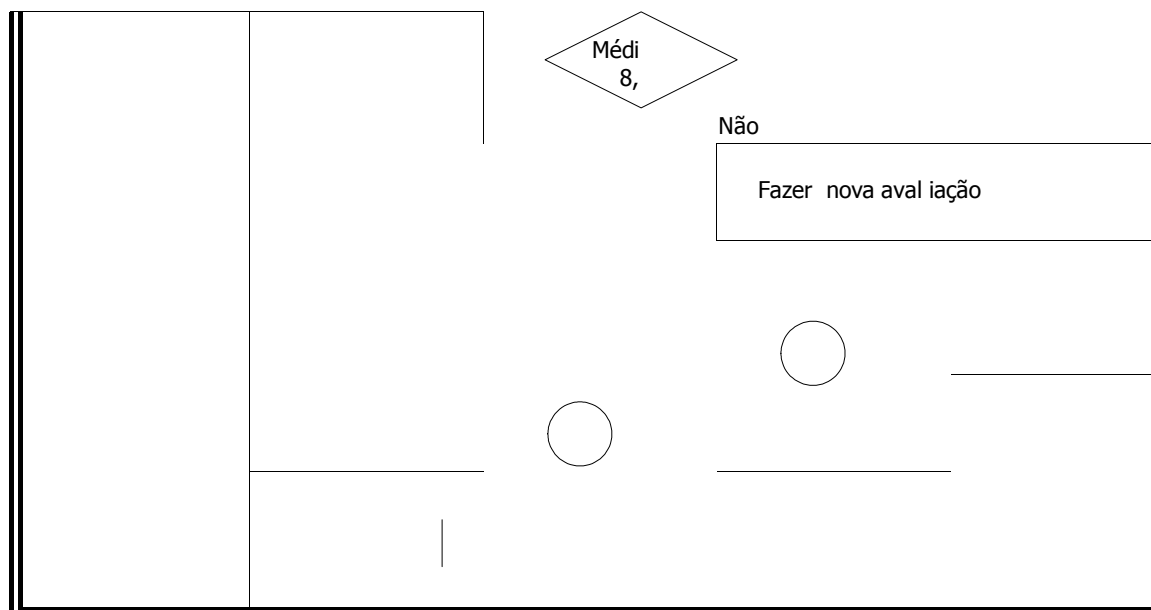
Em **COBOL**, ficaria assim:

```

IF MEDIA >= 7,00
  MOVE 'ALUNO APROVADO' TO TEXTO
ELSE
  MOVE 'ALUNO REPROVADO' TO TEXTO
END-IF
  
```

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 abaixo:





Em **COBOL**, ficaria assim:

```

IF MEDIA >= 7,00
  IF MEDIA >= 8,00
    MOVE 'ALUNO APROVADO' TO TEXTO
  ELSE
    MOVE 'FAZER NOVA AVALIACAO' TO TEXTO
  END-IF
ELSE
  MOVE 'ALUNO REPROVADO' TO TEXTO
END-IF

```

7.1.3 CASO SELECIONE ou SE ... ELSE ... SE ... ELSE

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

A estrutura de decisão "**CASO/SELECIONE**" é utilizada para testar, na condição, uma única expressão, que produz um resultado, ou, então, o valor de uma variável, em que está armazenado um determinado conteúdo.

FUTURE SCHOOL – Cursos de Computação

www.osasconamao.com.br/cursos

Fone: (0XX11) 98342.2503

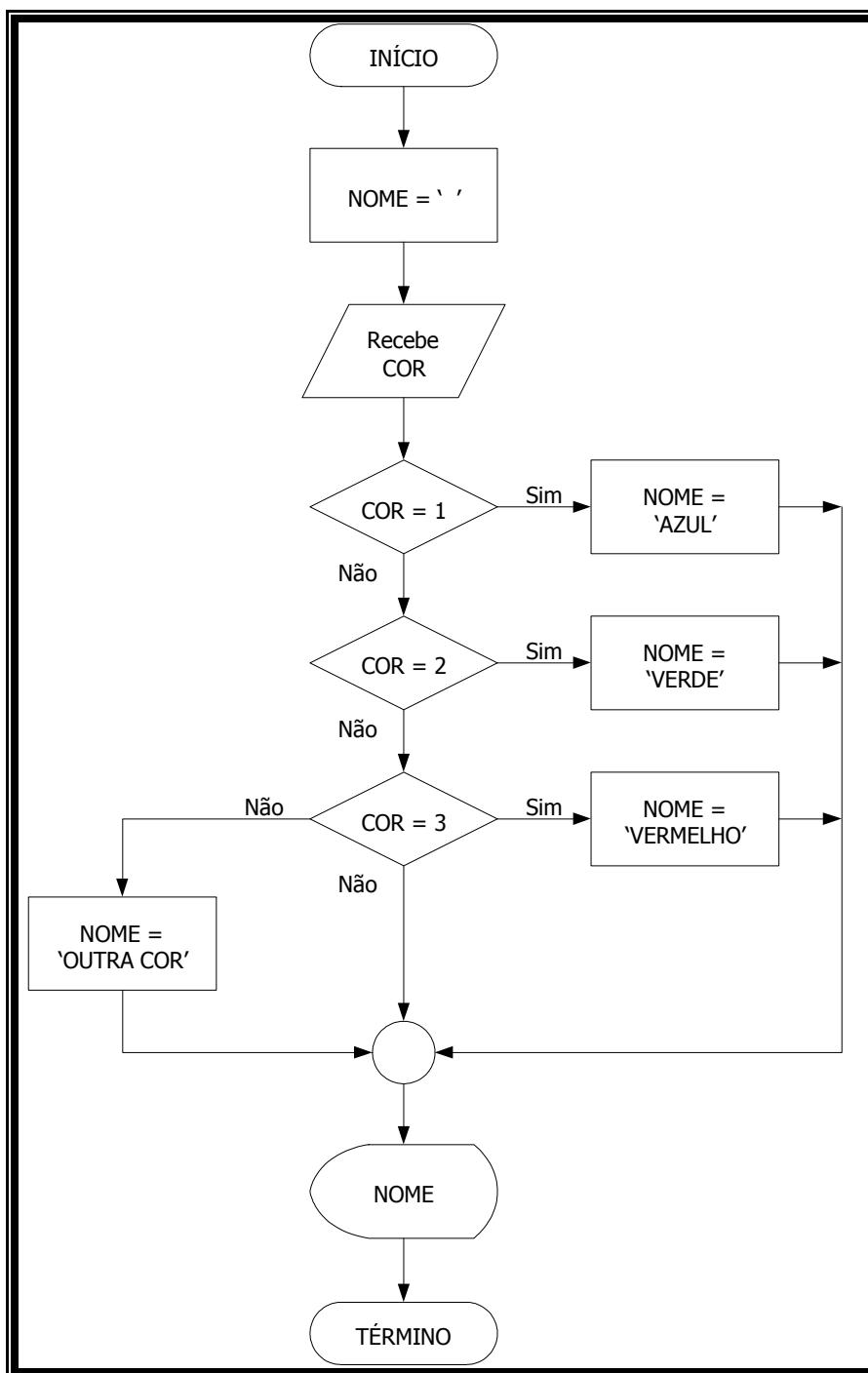
Página 25 de 59

Lógica de Programação

Compara-se, então, o resultado obtido no teste com os valores fornecidos em cada cláusula "Caso".

Lógica de Programação

No exemplo do diagrama de blocos abaixo, é recebido uma variável numérica "COR" e testado seu conteúdo, caso uma das condições seja satisfeita, é atribuído para a variável **NOME** o nome da cor, caso contrário é atribuído a string "OUTRA COR".



Em **COBOL** ficaria assim:

MOVE '' TO NOME

```

DISPLAY 'Digite o código da cor = '
ACCEPT COR FROM CONSOLE
IF COR = 1
    MOVE 'AZUL' TO NOME
ELSE
    IF COR = 2
        MOVE 'VERDE' TO NOME
    ELSE
        IF COR = 3
            MOVE 'VERMELHO' TO NOME
ELSE
    MOVE 'OUTRA COR' TO NOME
END-IF
END-IF
END-IF

```

Observação: A codificação acima recebe o nome de **NINHO de IF**, ou seja, um **IF** dentro de outro **IF**, porém o comando mais utilizado atualmente, para a codificação acima é o **EVALUATE**, que se assemelha ao **CASE** das demais linguagens, por exemplo:

```

MOVE '' TO NOME
DISPLAY 'Digite o código da cor = '
ACCEPT COR FROM CONSOLE
EVALUATE TRUE
    WHEN COR = 1 MOVE 'AZUL' TO NOME
    WHEN COR = 2 MOVE 'VERDE' TO NOME
    WHEN COR = 3 MOVE 'VERMELHO' TO NOME
WHEN OTHER MOVE 'OUTRA COR' TO NOME          END-
EVALUATE.

```

7.1.4 Exercícios

- 1) Faça um diagrama de blocos que leia as variáveis **F** e **U**, respectivamente código e numero de horas trabalhadas de um analista; calcule o salário, sabendo-se que o mesmo ganha R\$ 30,00 por hora. Quando o número de horas exceder 160, calcule a diferença a ser paga e armazene na variável **T**, caso contrário, armazene 0 (zero) nesta variável. A hora excedente de trabalho vale R\$ 100,00. No final do processamento imprimir o salário, salário a ser pago e a diferença encontrada.
- 2) Um profissional, da bolsa de valores, comprou um computador para controlar seus gastos na bolsa. Toda vez que ele ultrapassa a sua cota diária de compra, estabelecida pela bolsa de valores em R\$ 5.000,00 (cinco mil reais), deve pagar 1% de multa a cada R\$ 1.000,00 que ultrapasse esta cota. Este profissional necessita que você faça um diagrama de blocos que leia a variável **U** (compra efetuada) e verifique se o valor de sua compra ultrapassou o limite de sua cota diária. Caso a compra ultrapasse o limite, gravar na variável **R** (diferença) e na variável **E** o valor da multa a ser paga; caso contrário mostrais tais variáveis com 0 (zero).

Lógica de Programação

- 3) Faça um diagrama de blocos, que leia um determinado número inteiro e mostre uma mensagem indicando se este número é positivo ou negativo; caso esse número seja positivo, mostre uma outra mensagem, indicando se é par ou ímpar, caso contrário não precisa mostrar mensagem.
- 4) Faça um diagrama que:
 - a) Leia 5 (cinco) números inteiros e positivos;
 - b) Calcule o quadrado de cada um deles;
 - c) Se o valor do quadrado do quarto número for maior ou igual a 100, imprima o número informado lido no item a, o valor obtido no item b e finalize;
 - d) Caso contrário, imprima todos os números lidos no item a, os números obtidos no item b e uma mensagem indicando sucesso no processamento.
- 5) Faça um algoritmo, que a partir da leitura de uma determinada idade, indique sua respectiva categoria, obedecendo aos critérios abaixo:
 - a) Até 5 anos = INFANTIL A;
 - b) De 5 até 7 anos = INFANTIL B;
 - c) De 8 até 10 anos = INFANTIL B;
 - d) De 11 até 13 anos = INFANTIL C;
 - e) De 14 até 15 anos = JUVENIL A;
 - f) De 16 até 17 anos = JUVENIL B;
 - g) Maiores de 17 anos = ADULTO.
- 6) Faça um algoritmo que imprima todos os números de 0 a 100, que sejam compostos em sua unidade ou dezena do numeral 9 (nove), logo após indique uma mensagem com o total de números obtidos.
- 7) Faça um algoritmo que gere e imprima todos os números pares entre 500 e 600.
- 8) Faça um algoritmo que:
 - a) Leia 3 (três) números positivos e inteiros;
 - b) Encontre o maior número;
 - c) Encontre o menor número;
 - d) Calcule a média entre todos os números lidos;
 - e) Imprima o menor número, com respectiva mensagem;
 - f) Imprima o maior número, com respectiva mensagem; e;
 - g) Imprima a média calculada, com respectiva mensagem.
- 9) Uma companhia, que cuida da água e esgoto de uma determinada cidade, atribui notas de 1 até 5 (cinco) para cada morador, de acordo com o seu consumo. Quanto menor for a nota, menor foi o consumo do morador. Faça um diagrama que leia o código do morador com sua respectiva nota e emita uma notificação adequada para o morador que ultrapassa a nota 3.

7.2 Comandos de Repetição

Utilizamos os comandos de repetição quando desejamos que um determinado conjunto de instruções ou comandos sejam executados um número definido ou indefinido de vezes, ou ainda, enquanto um determinado estado de coisas prevalecer ou até que seja alcançado.

Os modelos de comandos de repetição mais utilizados são:

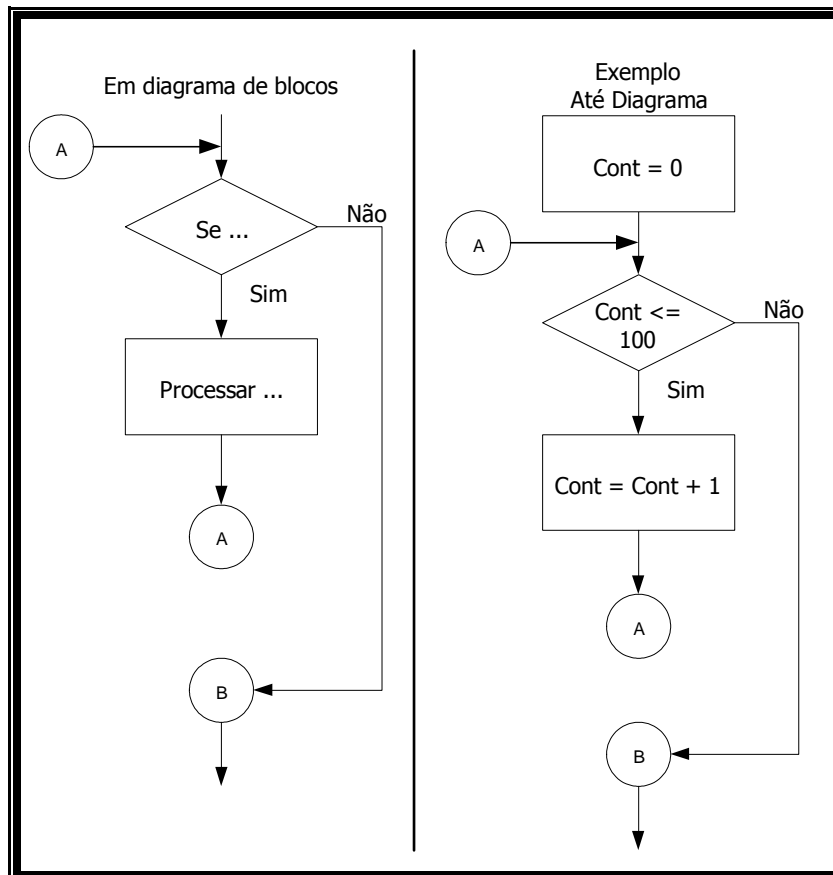
- Enquanto X, processar;
- Até que X, processar ...;
- Processar ..., Enquanto X; e;
- Processar ..., Até que X;

7.2.1 Enquanto X, processar

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 **COBOL**, ficaria assim:

```

A.
MOVE 0 TO CONT.
IF CONT NOT GREATER 100
  ADD 1 TO CONT
  GO TO A
END-IF.

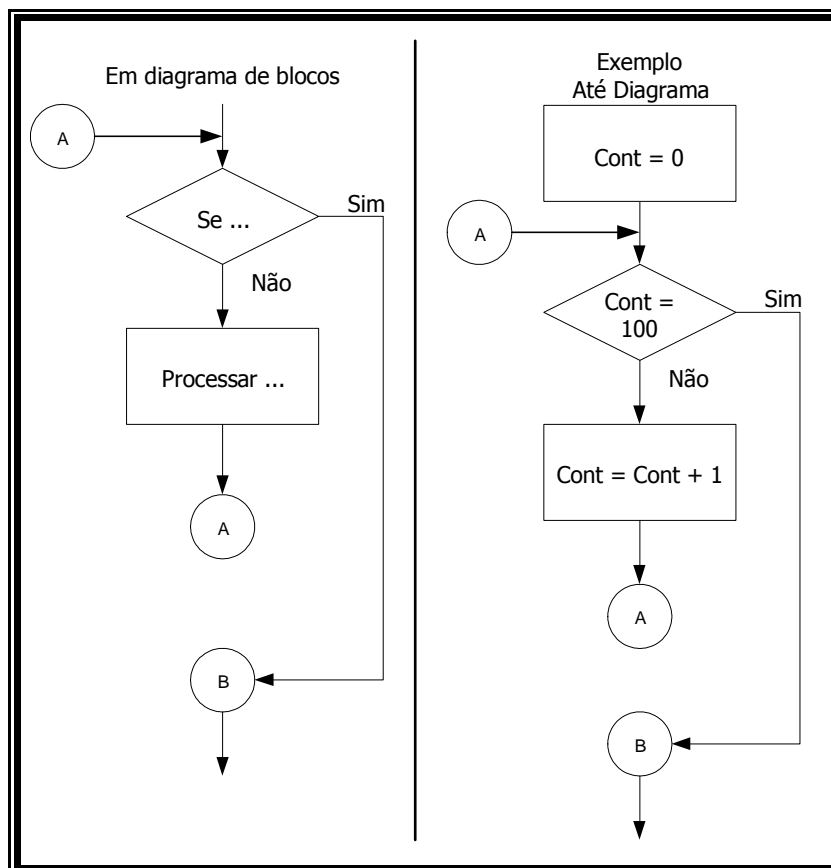
```

B.

...

7.2.2 Até que X, processar

Neste caso, o bloco de operações será executado até que a condição seja satisfeita, ou seja, somente executará os comandos enquanto a condição for falsa.



Em **COBOL**, ficaria assim:

```

A.
MOVE 0 TO CONT.

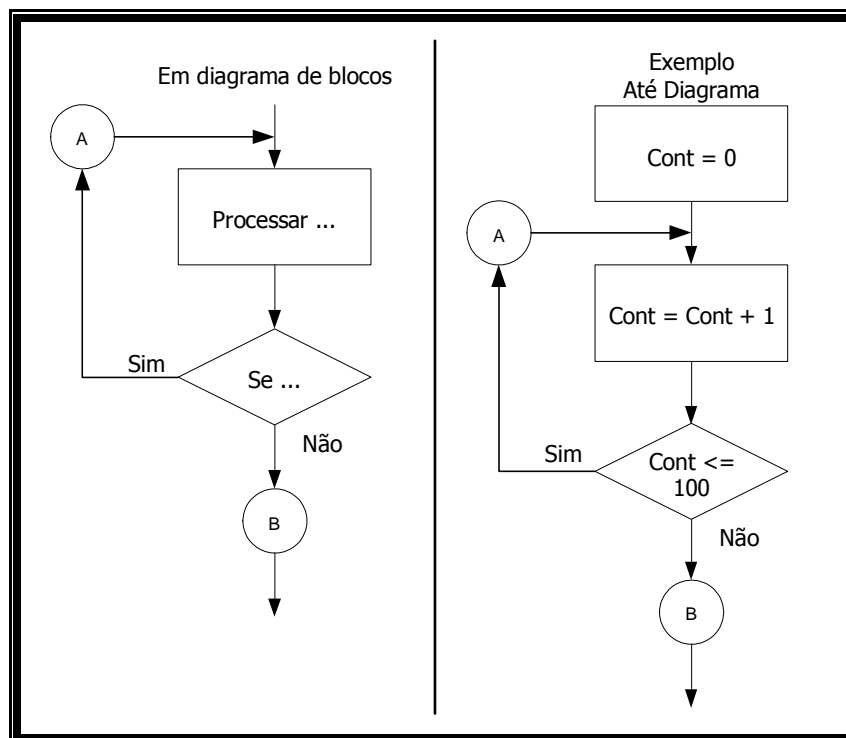
IF CONT EQUAL 100
  GO TO B
END-IF.

ADD 1 TO CONT
GO TO A.

B.
...
  
```

7.2.3 Processar ..., enquanto X

Neste caso primeiro são executados os comandos, e somente depois é realizado o teste da condição. Se a condição for verdadeira, os comandos são executados novamente, caso seja falso é encerrado o comando DO.



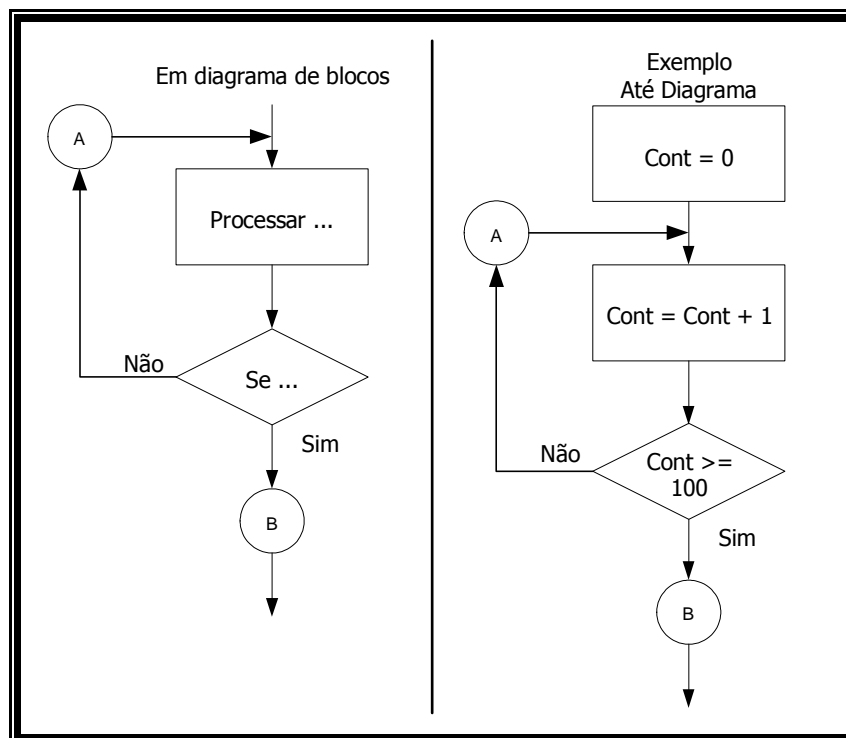
Em **COBOL**, ficaria assim:

```

A.
    MOVE 0 TO CONT.
    ADD 1 TO CONT
    IF CONT NOT GREATER 100
        GO TO A
    END-IF.

B.
    ...
7.2.4 Processar ..., até que X
  
```

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 é processado novamente os comandos antes do teste da condição.



Em **COBOL**, ficaria assim:

```

MOVE 0 TO CONT.
A.
ADD 1 TO CONT
IF CONT NOT LESS 100
GO TO B
END-IF.
GO TO A.

```

B.

7.2.5 Exercícios

- Um rei solicitou os serviços de um monge e disse-lhe que pagaria qualquer preço. O monge, necessitando de dinheiro, indagou ao rei sobre o pagamento, se poderia ser feito com moedas de ouro dispostos em um tabuleiro de xadrez, de tal forma que o primeiro quadro deveria conter apenas uma moeda e os quadros subseqüentes, o dobro do quadro anterior. O rei achou o trabalho barato e pediu que o serviço fosse executado, sem se dar conta de que seria impossível efetuar o pagamento. Faça um algoritmo para calcular o número de moedas de ouro que o monge esperava receber.
- Faça um algoritmo que conte de 1 a 1000 e a cada múltiplo de 10 emita uma mensagem: "Múltiplo de 10".
- Faça um algoritmo que determine o maior entre N números. A condição de parada é a entrada de um valor 0, ou seja, o algoritmo deve ficar calculando o maior até que a entrada seja igual a 0 (ZERO).

8 Arquivos de Dados

Os dados manipulados até o momento, estavam em memória, ou seja, após a execução do diagrama os dados se perdiam.

Para resolver esse problema começaremos a trabalhar com arquivos, onde poderemos guardar os dados e também manipulá-los.

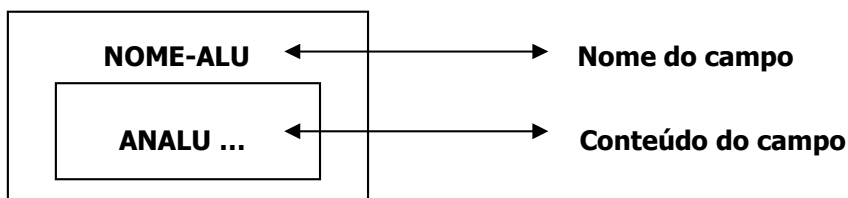
Para isso necessitamos rever alguns conceitos como: campos, registros e arquivos.

8.1 Conceitos Básicos

Campo é um espaço reservado em memória para receber informações.

Exemplo de campos: COD-ALU, NOME-ALU, TURMA-ALU etc

Exemplo de um campo na memória



Registro é um conjunto de campos.

Exemplo: REGISTRO DE ALUNOS

| COD-ALU | NOME-ALU | TURMA-ALU |
|---------|-----------|-----------|
| 00001 | ANALU ... | 001 |

Arquivo é um conjunto de registros.

Exemplo: Arquivo de Alunos (onde estão armazenados os dados de todos alunos)

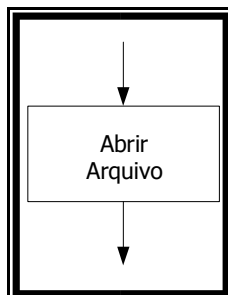


8.2 Abertura de arquivos

Toda vez que for necessário trabalhar com arquivo, primeiramente precisamos **ABRIR** o arquivo.

Abrir o arquivo significa alocar o periférico (disco, disquete) em que o arquivo se encontra, e deixá-lo disponível para leitura e/ou gravação.

Abaixo mostraremos o símbolo para abertura de arquivo:

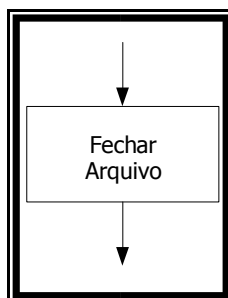


8.3 Fechamento de arquivos

Da mesma maneira que precisamos abrir um arquivo antes do processamento, também se faz necessário o fechamento do mesmo, para que suas informações não possam ser violadas ou danificadas.

Fechar um arquivo significa liberar o periférico que estava sendo utilizado.

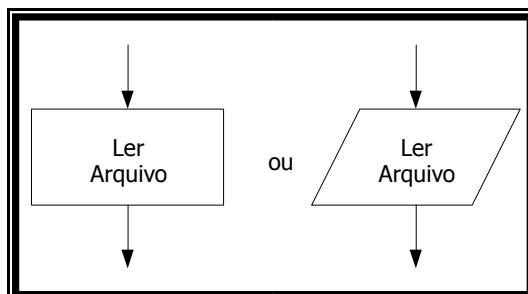
Abaixo mostraremos o símbolo para fechamento de arquivo



8.4 Leitura de arquivos

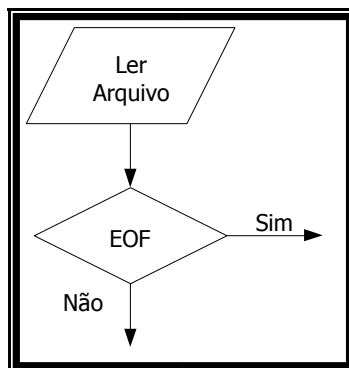
Após abrir um arquivo é necessário **LER** os dados que estão em disco e transferilos para memória. Essa transferência é feita por registro. Esse procedimento é gerenciado pelo próprio sistema operacional.

Abaixo mostraremos o símbolo para leitura de arquivo

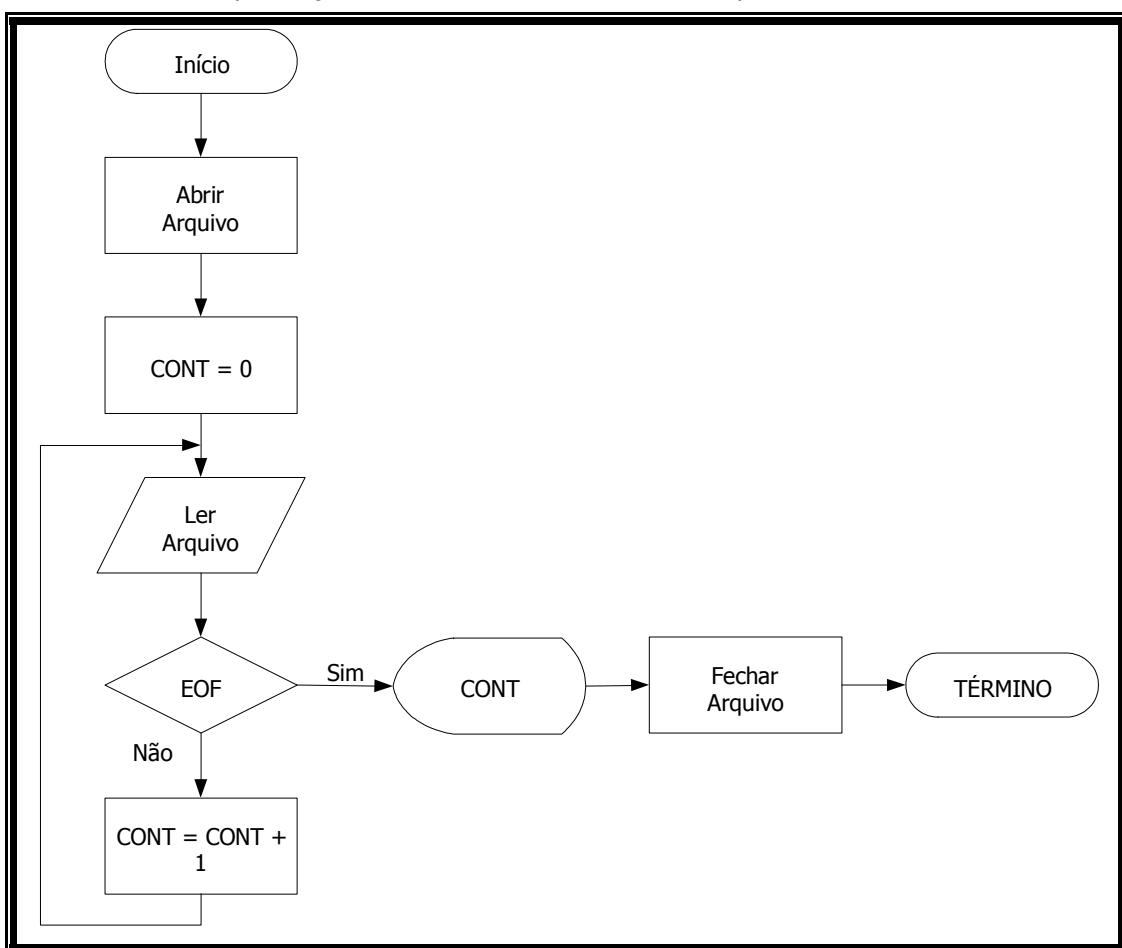


Toda vez que abrimos um arquivo ele posiciona o "**ponteiro**" no primeiro registro, ou seja, no início do arquivo. Para que possamos trabalhar com os dados se torna necessário sabermos onde está o ponteiro do registro. Isso poderemos fazer testando se o ponteiro está no início (**BOF – Bottom Of File**) ou no final do arquivo (**EOF –**

End Of File). Esse é sempre executado após a leitura do registro (mudança da posição do ponteiro). Simbolicamente podemos representar esse passo da seguinte maneira.



fechamento de arquivos, juntamente com teste de fim de arquivos



Abaixo mostraremos o exemplo de um diagrama utilizando abertura, leitura e

8.5 Movimentação de registros

Como dito no item anterior, quando um arquivo é aberto o ponteiro está no primeiro registro. A cada leitura do Arquivo o ponteiro se movimenta para o próximo registro e assim por diante. Como mostra a figura abaixo:

BOF (Início do arquivo)

| COD-ALU | NOME-ALU | TURMA-ALU | |
|---------|---------------|-----------|--------------|
| 00001 | ANALU PIRES | 001 | → Registro 1 |
| 00002 | JOSE DA SILVA | 001 | → Registro 2 |
| 00003 | ANTONIO MARIA | 002 | |
| | | | |

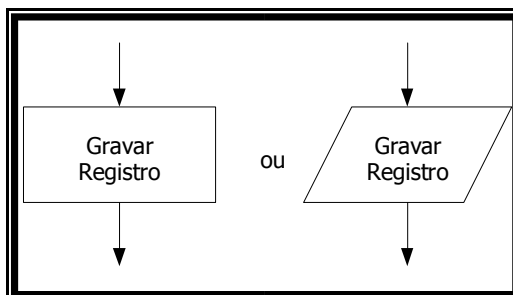
EOF (Fim do arquivo)

8.6 Gravação de arquivos

Da mesma maneira que os registros são lidos de um arquivo, também devemos gravar registros em um arquivo.

A gravação consiste na transferência de um registro da memória, para um periférico (disco, disquete, impressora).

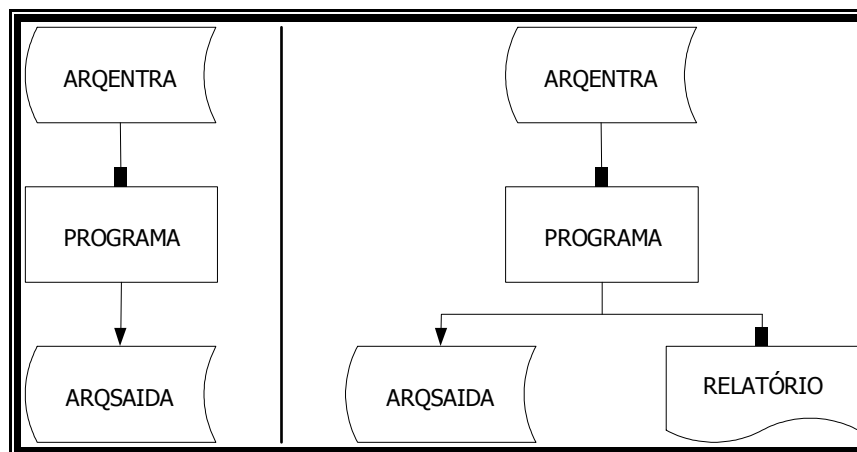
Abaixo mostraremos o símbolo para gravação de arquivo:



8.7 Macro Fluxo

O macro fluxo é a representação gráfica dos arquivos que serão processados em um programa.

Exemplos de macro fluxo:



Estes dois exemplos de Macro fluxo dão uma visão geral de como devemos proceder com cada um dos programas. O primeiro diz que haverá um arquivo de entrada, um processamento e um arquivo de saída. Já o segundo exemplo diz que haverá um arquivo de entrada, um processamento, e dois arquivos de saída (disco e impressão).

8.8 Exercícios

- 1) Foi feita uma pesquisa entre os habitantes de uma região. Foram coletados os dados de idade, sexo (M, F) e salário. Faça um algoritmo que informa:
 - a) A média de salário da região;
 - b) Maior e menor idade da região;
 - c) Quantidade de habitantes;
 - d) Quantidade de mulheres;
 - e) Quantidade de homens;

- 2) Um arquivo de produtos tem os seguintes campos: Código do produto, Descrição, Quantidade em Estoque, Preço de custo, Margem Custo/Venda. Crie um arquivo com os seguintes campos: Código do Produto e Preço de Venda. Utilize o cálculo Preço de Venda = Preço de Custo * Margem Custo/Venda.

- 3) Faça um diagrama de blocos para verificar que produtos precisam ser comprados e a quantidade a ser adquirida, para isso, observe as seguintes informações:
 - Código do produto (CODPROD), Quantidade Mínima (QTDMIN), Quantidade Máxima (QTDMAX) e a quantidade em estoque (QTDEST) de cada produto.;
 - Um produto somente deverá ser comprado quando a quantidade em estoque for menor ou igual a quantidade mínima;
 - $QTDCOMPRAR = (QTDMAX - QTDEST)$
 - Gravar em outro arquivo: Código do Produto e Quantidade a Comprar

9

Relatórios

A impressão de relatórios é o registro de informações processadas pelo computador em um meio de armazenamento de dados chamado de formulário.

Para efetuarmos a impressão de relatórios devemos nos preocupar com os seguintes aspectos:

- Características do formulário;

- Controle de linhas e salto de página;
- Impressão de cabeçalho e estética da página; • Impressão de rodapé; e;
- Numeração de páginas.

9.1 Características do Formulário

A maioria dos formulários possui um formato padrão, isto é, a quantidade de linhas por página e de caracteres por linha são constantes.

9.2 Controle de linhas e salto de páginas

Uma preocupação com impressão de relatórios é não permitir que a impressora imprima fora do papel, pois além de esteticamente não ficar bom, haveria perda de informações.

Para controlarmos o número de linhas impressas, devemos criar um contador de linha e não deixar o valor desses contadores ultrapassarem o número desejado de linhas por páginas.

9.3 Impressão de Cabeçalho e Estética de Página

Para termos uma idéia melhor da estética do formulário, veja o exemplo abaixo:

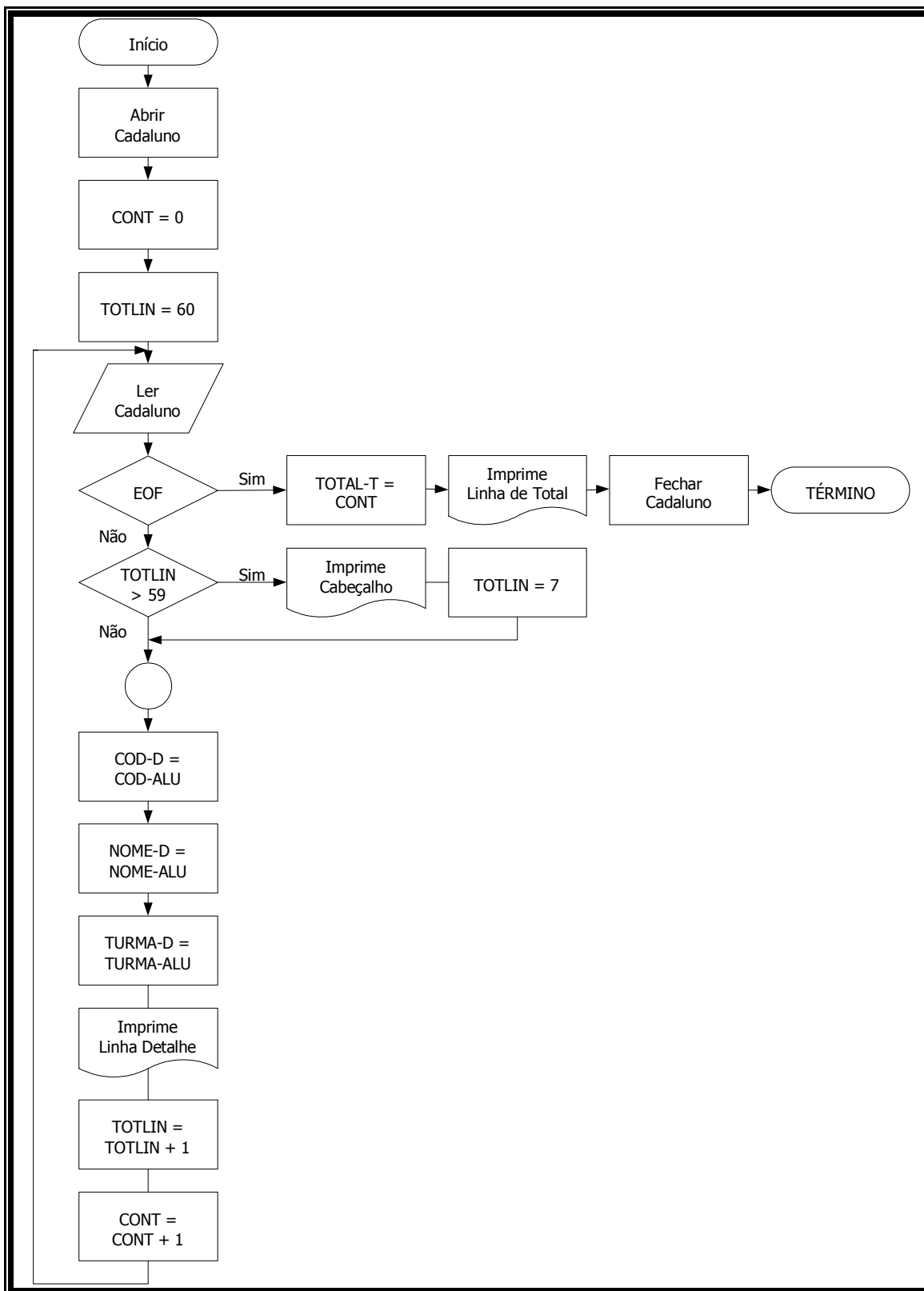
| | | | | |
|-------------------|---------------------------------|---------------|------------|----------|
| Cabeçalho | FUTURE SCHOOL CURSOS COMPUTAÇÃO | PAG. | 1 | LINCAB01 |
| | PROGRAMA = TESTE001 | | 01/02/2006 | LINCAB02 |
| | LISTAGEM DOS ALUNOS | | | LINCAB03 |
| | CODIGO ALUNO | NOME DO ALUNO | TURMA | LINCAB04 |
| | 00001 | ANALU PIRES | 001 | LINDET01 |
| | 00002 | JOSE DA SILVA | 001 | LINDET02 |
| | 00003 | ANTONIO MARIA | 002 | LINDET03 |
| | | | | |
| | | | | |
| | | | | |
| Linhas de detalhe | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| Rodapé | | | | |
| | | | | |
| | | | | |
| | | | | |

Conforme a figura acima, podemos definir o relatório com 3 áreas:

- **Área de cabeçalho:** local onde devemos colocar um cabeçalho para identificarmos o assunto a que se refere o conteúdo da página como um todo, e um cabeçalho indicando o conteúdo de cada coluna de informações; pode haver outras linhas de cabeçalho, de acordo com a necessidade;
- **Área de detalhe:** São as linhas geradas a partir de dados lidos de um arquivo; e;

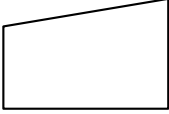
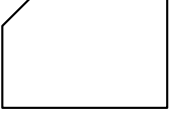
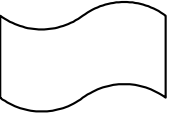
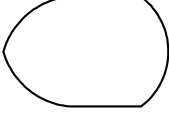




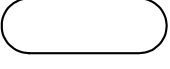
- **Área de rodapé:** Pode haver linhas contendo valores de totalizações de determinadas colunas e / ou linhas de identificação da empresa ou outras informações.

Veja abaixo um exemplo de diagrama de bloco para impressão de relatório:



9.4

Exercícios

| | | |
|--|---|--|
| ARMAZENAMENTO INTERNO | | |
|  ENTRADA MANUAL |  CARTÃO |  FITA DE PAPEL |
|  EXIBIÇÃO |  OPERAÇÃO manualmente MANUAL |  PREPARAÇÃO |
|  REFERÊNCIA NA PÁGINA |  REFERÊNCIA FORA DA PÁGINA |  processamento TERMINAL |

11

Vamos brincar com a lógica

- 1) Você está numa cela onde existem duas portas, cada uma vigiada por um guarda. Existe uma porta que dá para a liberdade, e outra para a morte. Você está livre para escolher a porta que quiser e por ela sair. Poderá fazer apenas uma pergunta a um dos dois guardas que vigiam as portas. Um dos guardas sempre fala a verdade, e o outro sempre mente e você não sabe quem é o mentiroso e quem fala a verdade. Que pergunta você faria?
- 2) Você é prisioneiro de uma tribo indígena que conhece todos os segredos do Universo e portanto sabem de tudo. Você está para receber sua sentença de morte. O cacique o desafia: "Faça uma afirmação qualquer. Se o que você falar for mentira você morrerá na fogueira, se falar uma verdade você será afogado. Se não pudermos definir sua afirmação como verdade ou mentira, nós te libertaremos. O que você diria?
- 3) Um grande empresário na necessidade de ir a São Paulo, chegou a seu guarda noturno e ordenou que ele o acordasse às 6 horas da manhã em ponto. Exatamente às 6:00 da manhã o guarda acordou o empresário e disse: - Patrão, estou com um mal pressentimento: sonhei esta noite que o senhor teria um acidente com o avião e

Lógica de Programação

me permita sugerir que não viaje. O empresário não deu ouvidos ao guarda. Sem incidentes, chegou a São Paulo e por telefone mandou demitir o guarda. Por quê?

- 4) Um pastor diz para outro: "Dê um de seus carneiros que ficamos com igual número de carneiros." O outro responde: "Nada disso, dê-me um de seus carneiros que ficarei com o dobro dos seus". Quantos carneiros têm cada um?
- 5) Uma lesma deve subir um poste de 10 metros de altura. De dia sobe 2m e à noite desce 1m. Em quantos dias atingirá o topo do poste?
- 6) Três gatos comem três ratos em três minutos. Cem gatos comem cem ratos em quantos minutos?
- 7) O pai do padre é filho do meu pai. O que eu sou do Padre?
- 8) Qual é o dobro da metade de dois?
- 9) Se um bezerro pesa 75 kg mais meio bezerro, quanto pesa um bezerro inteiro?
- 10) Um avião lotado de passageiros parte do Rio de Janeiro em direção a Buenos Aires. Por uma fatalidade cai na fronteira Brasil-Argentina. Onde serão enterrados os sobreviventes?
- 11) Uma pata nascida no Chile bota um ovo na divisa Brasil-Chile. Segundo o Itamaraty, a quem pertence o ovo?
- 12) Um senhor de 80kg e suas 2 filhas cada uma com 40kg precisam atravessar uma ilha com um barco. Só que há um problema, o barco só suporta 80kg. Como farão para atravessar?
- 13) O meu pato botou um ovo no quintal do meu vizinho, segundo o IBAMA de quem é o ovo?
- 14) 200 burros estão andando em fila, um burro cai ele olha para trás, quantos burros ele vai contar?
- 15) Um pescador esta do lado de um rio, ele tem um barco e precisa levar um saco de milho, uma galinha e uma raposa para o outro lado, o barco só agüenta ele e mais alguma coisa (milho ou a galinha ou a raposa). Ele não pode deixar a galinha com o milho, porque a galinha comeria o milho, e nem pode deixar a galinha com a raposa, se não a raposa comeria a galinha... O que ele deve fazer?
- 16) O que é preto e branco, preto e branco, preto e branco...?
- 17) Que horas são quando um elefante senta em cima do seu carro?
- 18) Qual é a metade de dois mais dois?

Respostas:

1) Segundo o outro guarda, qual a porta que dá para a liberdade? E saia pela outra porta...

Após a porta ser indicada saia pela outra porta, porque se o guarda for o mentiroso, este indicaria a porta que leva para a morte, e se for o que conta a verdade, este sabendo que o outro sempre mente, também indicaria a porta que leva à morte.

2) Eu vou morrer na fogueira.

Porque se você realmente morrer na fogueira, isto é uma verdade, então você deveria morrer afogado, mas se você for afogado a afirmação seria uma mentira, e você teria que morrer na fogueira. Mesmo que eles pudessem prever o futuro, cairiam neste impasse

3) Guardas noturno não devem dormir em serviço.

4) 5 (cinco) e 7 (sete)

5) 9 (nove) dias.

No nono dia a lesma sobe 2 (dois) metros, atinge o topo e evidentemente não desce 1 (um) metro.

6) 3 (três) minutos

7) Tio

8) 2 (dois)

9) 150 Kg

10) Os sobreviventes ainda estão vivos.

11) A ninguém, pois o Brasil não faz divisa com o Chile.

12) Vão as 2 (duas) filhas, volta 1 (uma) filha, ele vai sozinho, volta a outra filha, e as duas vão novamente.

13) De ninguém, pois o pato não bota ovo.

14) Nenhum, pois burros não contam.

15) Ele leva a galinha, volta, leva a raposa e trás de volta a galinha, leva o milho, volta e leva a galinha novamente.

16) Uma zebra rolando de uma montanha.

17) Hora de comprar um carro novo.

18) 3 (três)