

A collection of historical artifacts is arranged on a light-colored surface. In the top left, a portion of a wooden chessboard with a checkered pattern and several chess pieces is visible. Below the chessboard, there are two medals: one with a red ribbon and a star-shaped emblem, and another with a blue ribbon and a star-shaped emblem. A small, ornate compass is located in the bottom left corner. A pair of round, gold-rimmed glasses with thin temples is positioned in the center of the image, with the temples extending towards the right. The text 'Disciplina: Algoritmos I' is written in a large, black, serif font on the right side of the image. Below it, '1º Período – 2020' is written in a smaller, brown, serif font. At the bottom right, 'Prof. Cristiano Vieira' is written in the same brown, serif font.

Disciplina: Algoritmos I

1º Período – 2020

Prof. Cristiano Vieira

Apresentação





Cristiano Vieira

- Técnico em Processamento pela FEPI - 1995
- Graduado em Sistemas de Informação pela FEPI -2002
- Especialista em MBA pela UNIFEI - 2005
- Consultor Certificado SAP - 2011
- Mestre em Desenvolvimento e Tecnologias pela UNIFEI - 2014
- Especialista em Desenvolvimento Mobile pelo INATEL - 2017
- Professor no Curso de Sistemas de Informação desde 2004



Plano de Ensino

EMENTA:

Noções básicas sobre sistemas de computação: Hardware e software; Conceitos básicos sobre Sistemas Operacionais; Algoritmos, tipos de dados básicos, programação de computadores.

OBJETIVO:

Apresentar aos alunos os conceitos elementares da lógica para a construção de algoritmos, seu uso no dia a dia aplicando as principais formas de representação de algoritmos, suas estruturas e aplicações.



Plano de Ensino

RECURSOS METODOLÓGICOS:

Aulas teóricas expositivas em sala e práticas no LTI, apresentação de modelos práticos e aplicáveis, exercícios propostos e dinâmicas em grupos.

CONSIDERAÇÕES:

1. Tolerância de atrasos máximo em até 10 minutos, salvo justificção;
2. Chamadas de presença são efetuadas apenas 1 vez, no final da aula;
3. Não é permitido conversas paralelas;
4. Não é permitido alimentar-se em horário de aula;
5. Proibido uso de Celulares, MP3-MP4 ou qualquer outro equip. eletrônico
6. Para celulares, se extremamente necessário deixar em silencioso;
7. Participação conta ponto adicional;



Programação da Disciplina

1. Introdução a Lógica;
2. Definição de Algoritmos;
3. Tipos de dados;
4. Expressões Aritméticas e lógicas;
5. Comandos de Entrada e Saída;
6. Estruturas de Controle (Seleção e Repetição);
7. Variáveis multivaloradas e unidimensionais;



Avaliação da Aprendizagem

- 2 Provas práticas (30 pontos);
- 2 Trabalhos individuais (10 pontos);
- 1 Trabalho em grupo (20 pontos);



Referência bibliográficas

DROZDEK, A. Estruturas de Dados e Algoritmos em C++ , Ed.Thomson, 2002.

GOODRICH, M. T. & TAMASSIA, R. Estruturas de Dados e Algoritmos em Java , Ed. Bookman, 2007.

LAFORE, R. Estruturas de Dados e Algoritmos em Java, Ed.Ciência Moderna, 2004.

LAUREANO, M. Estrutura de Dados com Algoritmos em C, Ed.Brasport, 2008.

LORENZI, F. & MATTOS, P. N. & CARVALHO, T., Estruturas de Dados , Ed. Cengage Learning, 2006.

PEREIRA, Estruturas de Dados Fundamentais: Conceitos e Aplicações, Ed. Érica, 2006.

MANZANO & OLIVEIRA. Algoritmos Lógica para Desenvolvimento de Programação de Computadores. São Paulo: Ed. Erica, 2009.



Capítulo I

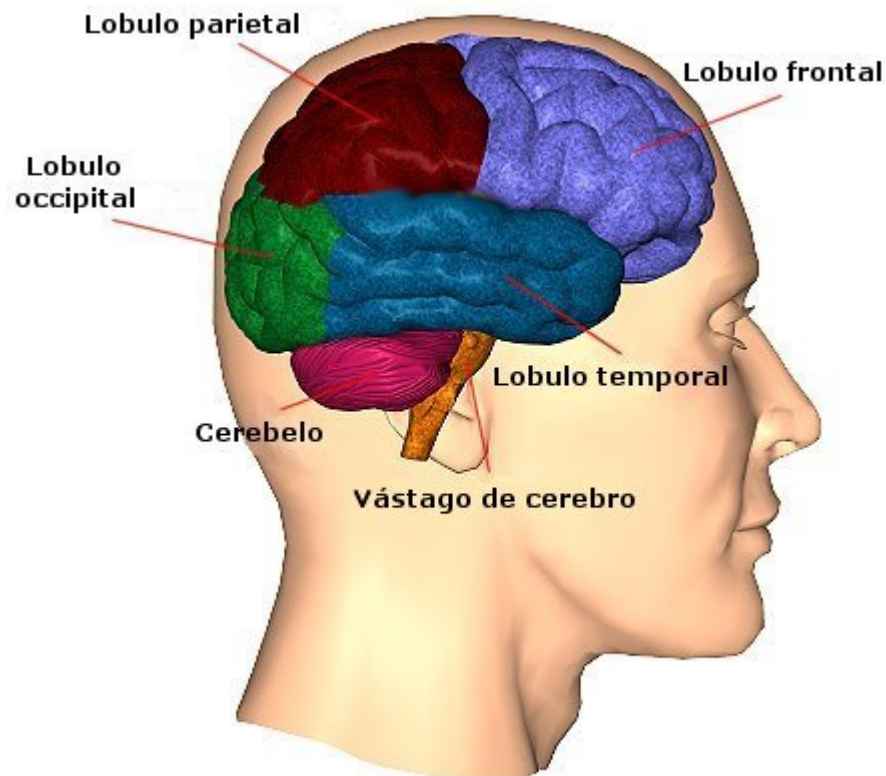
Introdução e Definição

- Lógica
- Lógica de Programação
- Algoritmos



1.1 INTRODUÇÃO A LÓGICA

Lógica?



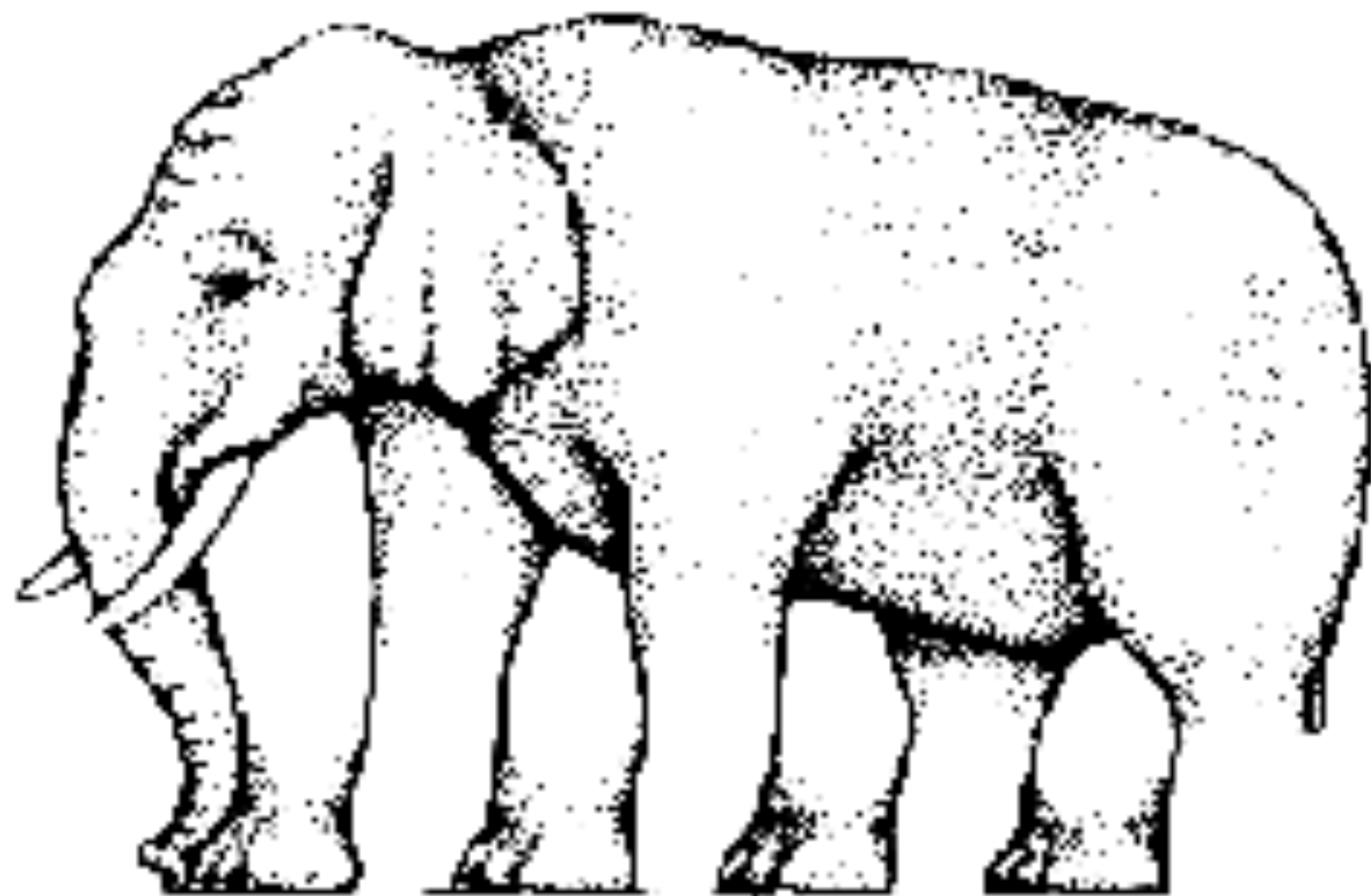


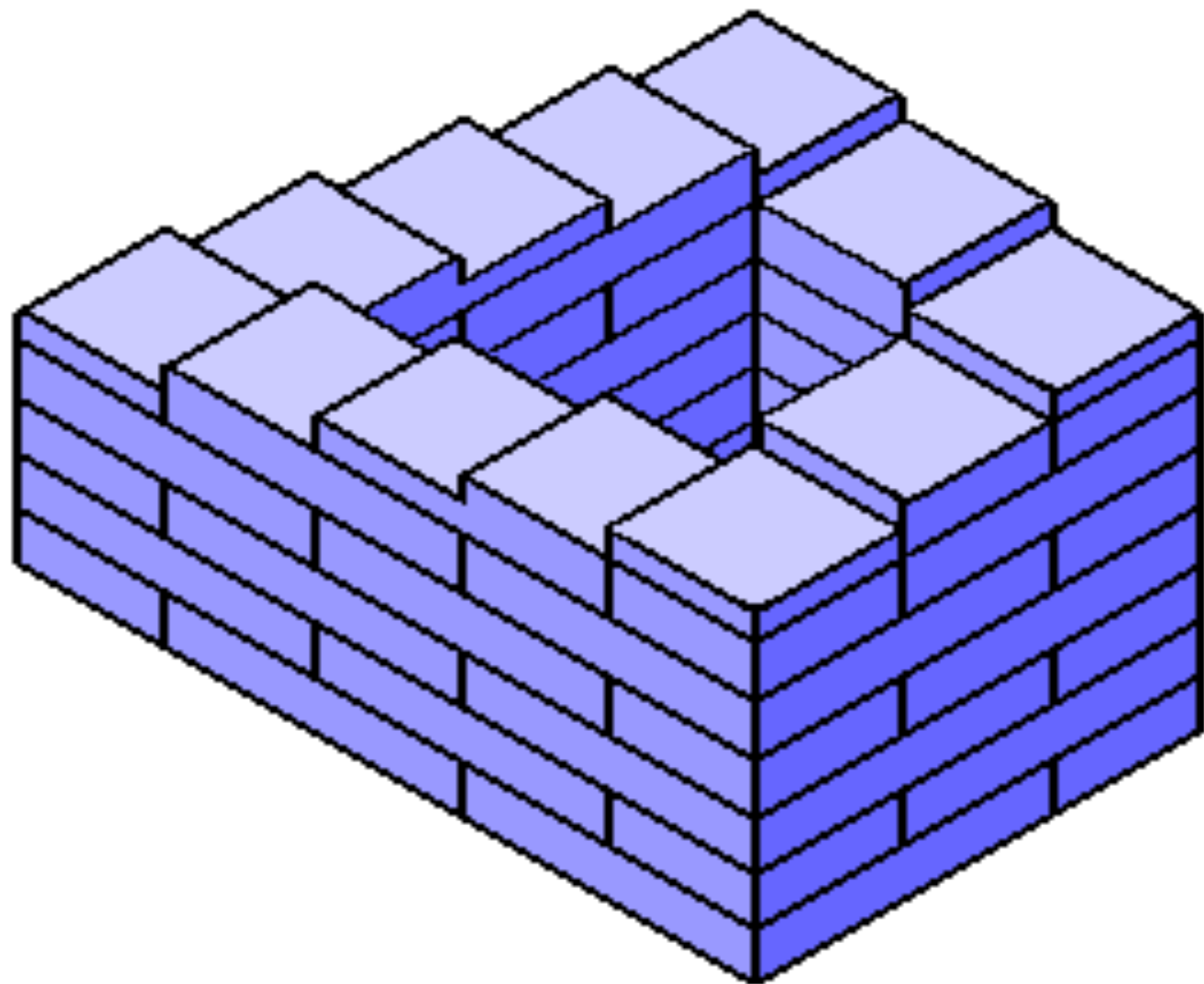
Por definição (segundo MICHAELIS)

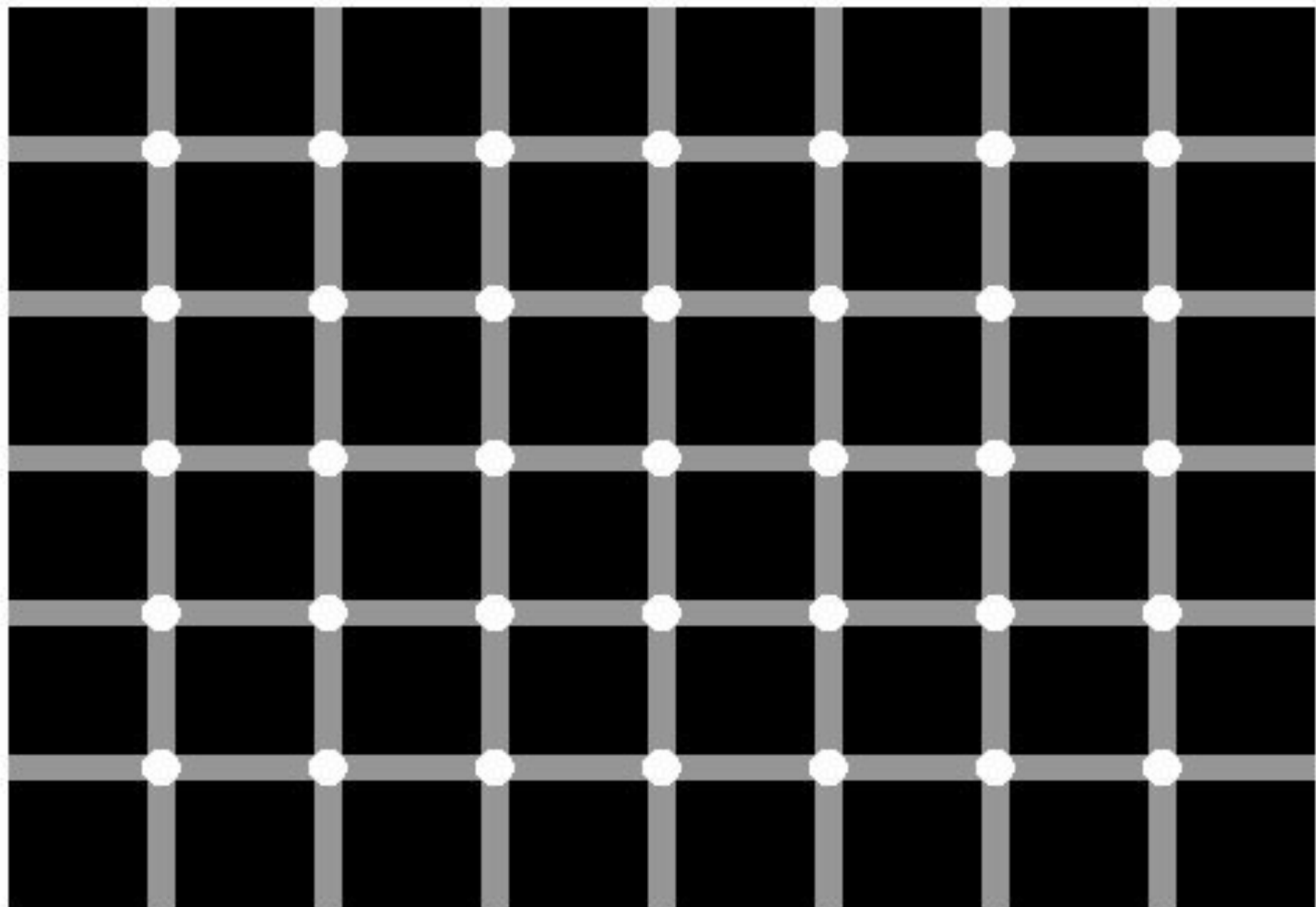
Ló.gi.ca *do grego: Algo*

- modo de raciocinar tal como de fato se exerce;
- estudo que tem por objeto determinar quais as operações que são válidas e quais as que não são;
- arte de bem pensar;
- ciência do desenvolvimento e representação de princípios lógicos mediante símbolos, a de constituir um modelo exato de dedução, baseado em idéias primitivas, postulados e regras de formação e transformação; também chamada *lógica matemática*.












Mais definições de Lógica

- Coerência de raciocínio e de idéias.
- Correção do pensamento
- Tudo aquilo que o cérebro pode entender ou interpretar
- Parte importante da criatividade
- Forma mais abrangente do "Pensar Criativo"



O nosso cérebro é doido!!!

De acordo com uma pesquisa e uma inversão da ordem das letras de uma palavra, a única coisa importante é que a ordem e a posição das letras estejam no lugar certo. O resto pode ser uma bagueta mágica, que você ainda pode ler sem problema. Isso é porque nós não lemos cada letra isoladamente, mas a palavra como um todo.



Fixe os seus olhos no texto abaixo e deixe que a sua mente
leia correctamente o que está escrito.

35T3 P3QU3N0 T3XTO 53RV3 4P3N45
P4R4 M05TR4R COMO NO554 C4B3Ç4
CONS3GU3 F4Z3R CO1545
1MPR3551ON4ANT35! R3P4R3 N155O!
NO COM3ÇO 35T4V4 M310
COMPL1C4DO, M45 N3ST4 L1NH4 SU4
M3NT3 V41 D3C1FR4NDO O CÓD1GO
QU453 4UTOM4T1C4M3NT3, S3M
PR3C1S4R P3N54R MU1TO, C3RTO?
POD3 F1C4R B3M ORGULHO5O D155O!
SU4 C4P4C1D4D3 M3R3C3! P4R4BÉN5!



Aplicação da Lógica

No pensar criativo, na criatividade, na obtenção de resultados, nas tomadas de decisões.





Lógica de Programação

É um método pelo qual se aplica os fundamentos do Raciocínio Lógico no desenvolvimento de programas de computador, fazendo uso ordenado dos elementos básicos suportados por um padrão conhecido como programação.



1.2 INTRODUÇÃO A LÓGICA DE PROGRAMAÇÃO



Raciocínio Lógico

- Ler atentamente o enunciado
- Retirar do enunciado a relação das entradas de dados
- Retirar do enunciado a relação das saídas de dados
- Determinar as ações que levarão a atingir o resultado desejado



Boa Lógica de Programação

- ◆ Organização
- ◆ Criatividade
- ◆ Perseverança
- ◆ Padronização
- ◆ Otimização



Exercício de raciocínio

1. Um homem viajava transportando junto a si um balde de leite, um cão e um gato. Um certo momento da viagem precisou atravessar um rio. Ali encontrava a sua disposição um barco que possui capacidade de transporte apenas dele mesmo e mais uma de suas três cargas. O que o homem deve fazer para conseguir atravessar o rio sem perder suas cargas?

Nota: O gato bebe o leite
O cão come o gato

Informações: homem, cão, gato, leite e barco.

Ação: transportar homem mais carga com segurança.

Resultado: Atravessar as cargas para a outra margem do rio.



Resposta do exercício 1

1. Atravessa homem + gato;
2. Volta o homem;
3. Atravessa o homem + leite;
4. Volta homem + gato;
5. Atravessa homem + cão;
6. Volta o homem;
7. Atravessa homem + gato.



Exercício de raciocínio

2. Três Jesuítas e três canibais precisam atravessar um rio, para tal, dispõe de um barco com capacidade para duas pessoas. Por medidas de segurança não se permite que em alguma margem a quantidade de jesuítas seja inferior a de canibais. Qual a seqüência de passos que permitirá atravessar com os jesuítas e canibais com segurança?

Informações: Três jesuítas, Três canibais, Um barco com capacidade para duas pessoas.

Ações: Atravessar para a outra margem do rio

Resultado: Chegar com segurança



Resposta do exercício 2

1. Atravessar **um** Jesuíta e **um** Canibal
2. Voltar **um** Canibal
3. Atravessar **dois** Canibais
4. Voltar **um** Canibal
5. Atravessar **um** Jesuíta e **um** Canibal
6. Voltar **um** Canibal
7. Atravessar **dois** Canibais
8. Voltar **um** Canibal
9. Atravessar **um** Jesuíta e **um** Canibal



1.3 ALGORITMO



Origem da palavra

ALGO → Lógica

RITMO → Estudo

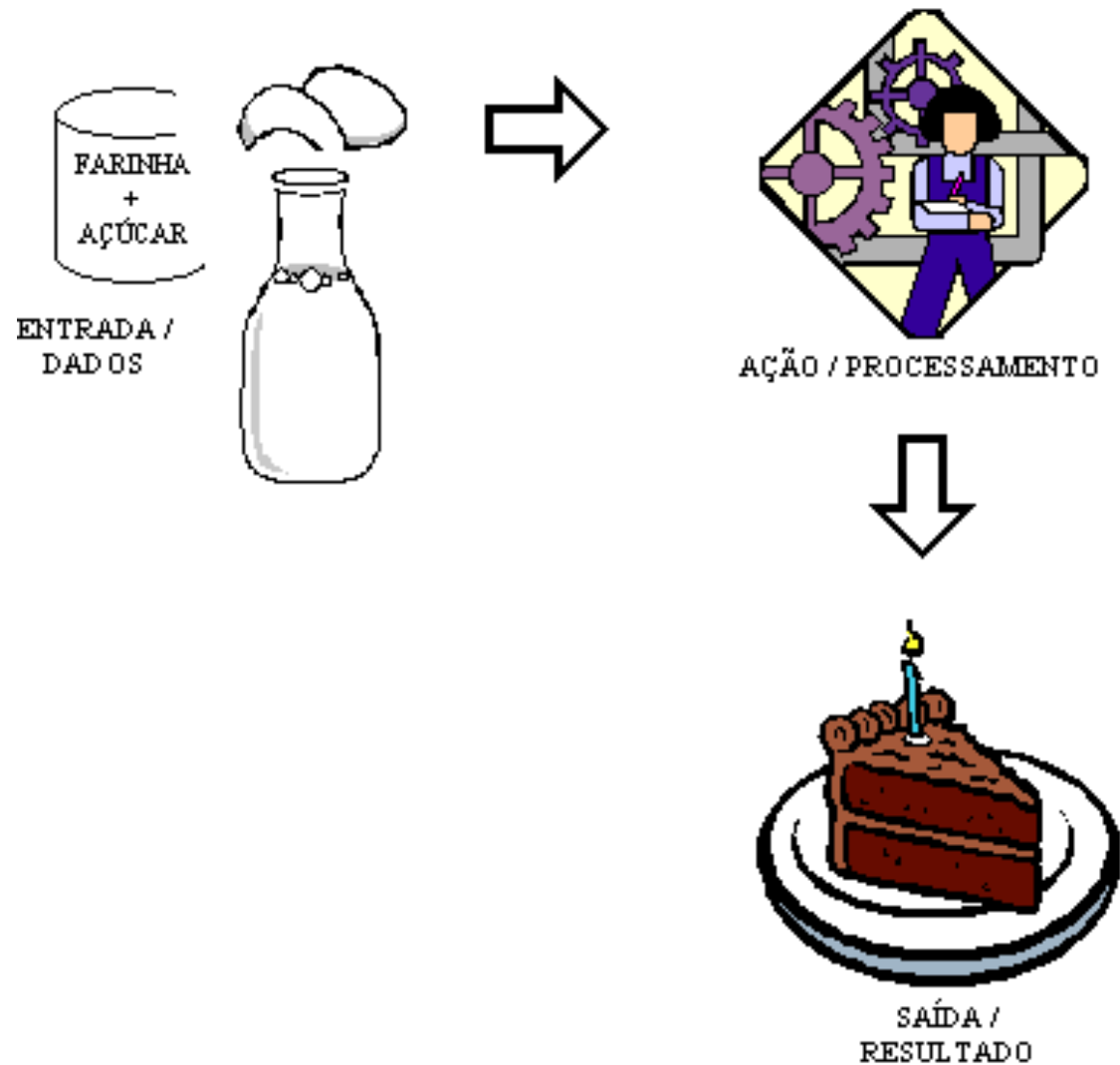
Algoritmo = Estudo da Lógica



Defini-se Algoritmo em:

- Uma seqüência de passos (ações) que devem ser executados para se alcançar um determinado objetivo.
- Aplicação computacional do Estudo da Lógica para se alcançar um objetivo.
- Linguagem em alto nível que permite ao usuário/programador preparar e programar o computador para resolver problemas através de uma seqüência lógica de ações que deverão ser executados passo-a-passo e que seguirão os conceitos da programação estruturada.

Aplicação no mundo real





Iniciando um algoritmo

1. Faça uma leitura de todo o problema até o final, a fim de formar a primeira impressão. A seguir, releia o problema e faça anotações sobre os pontos principais.
2. Verifique se o problema foi bem entendido;
3. Extraia do problema todas as suas entradas (informações, dados).
4. Extraia do problema todas as suas saídas (resultados).



Iniciando um algoritmo

5. Identifique qual é o processamento principal.
6. Verifique se será necessário algum valor intermediário que auxilie a transformação das entradas em saídas;
7. Teste cada passo do algoritmo;
8. Crie valores de teste;
9. Reveja o algoritmo, checando as normas.




Solucionando problemas por meio de algoritmos

Problema em questão?

Ponto de Partida?

Solução em linguagem natural (alto nível)



Problema: Ir ao trabalho

Ponto de partida: Estar dormindo

1. ACORDAR ← Início
 2. SAIR DA CAMA
 3. IR AO BANHEIRO
 4. LAVAR O ROSTO E ESCOVAR OS DENTES
 5. COLOCAR A ROUPA
 6. SE CHOVER, ENTÃO PEGAR O GUARDA-CHUVA ← Tomada de decisão
 7. IR AO PONTO DE ONIBUS
 8. ANDAR DE ONIBUS ATÉ ELE CHEGAR AO TRABALHO
 9. BATER O CARTÃO DE PONTO
 10. COMEÇAR O EXPEDIENTE
- ↑
Obtenção da Solução



Formas de apresentação de Algoritmos

- Descrição Narrativa ou Linguagem de Alto nível (natural)
- Pseudocódigo (Linguagem Estruturada ou Portugol)
- Fluxograma Convencional ou Diagrama de Bloco



Linguagem Natural

- ◆ É o modo pelo qual nos comunicamos (linguagem de alto nível) e é ponto partida para a abstração dos problemas em fatos solucionáveis.
- ◆ É pouco utilizada, porque o uso desta, muitas vezes, da oportunidade a más interpretações, ambigüidade e imprecisões.

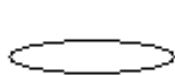


Pseudocódigo ou Portugol

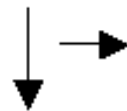
- ◆ Muito utilizado por projetistas de software e programadores para expressar sua lógica no idioma nativo de origem portuguesa.
- ◆ Forma de representação padrão para algoritmos
- ◆ Estruturas básicas de controle: sequência, seleção e repetição

Fluxograma

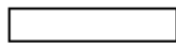
- ♦ Muito utilizado pela área de Engenharia e Informática
- ♦ Utiliza símbolos específicos para representar a execução dos procedimentos



Início
e Fim



Fluxo de
dados



Sentenças e
Comandos



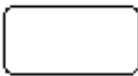
Conectores
de fluxos



Estrutura de
Seleção



Estruturas de
Repetição



Processo
Alternativo



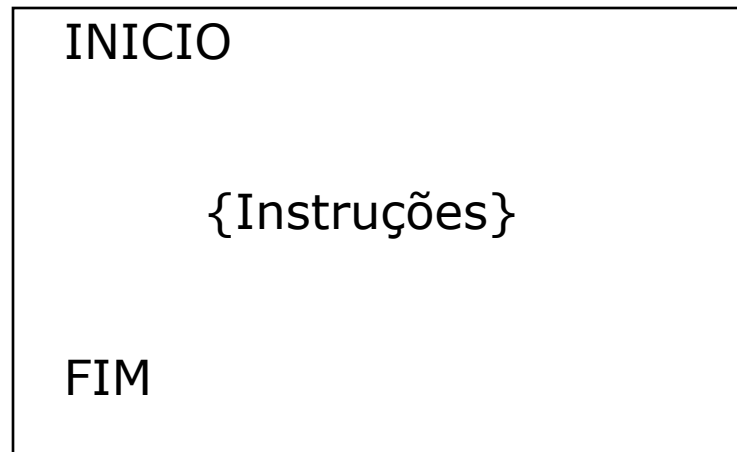
1.4 Estruturas de Controle

- Estrutura Seqüencial
 - CONSTANTES
 - VARIÁVEIS
 - ENTRADA, PROCESSAMENTO E SAÍDA
- Estrutura Condicional ou de Seleção
 - SE-ENTÃO
 - SE-ENTÃO-SENÃO
- Estruturas de Repetição
 - PARA-FAÇA
 - ENQUANTO-FAÇA
 - REPITA-ATÉ

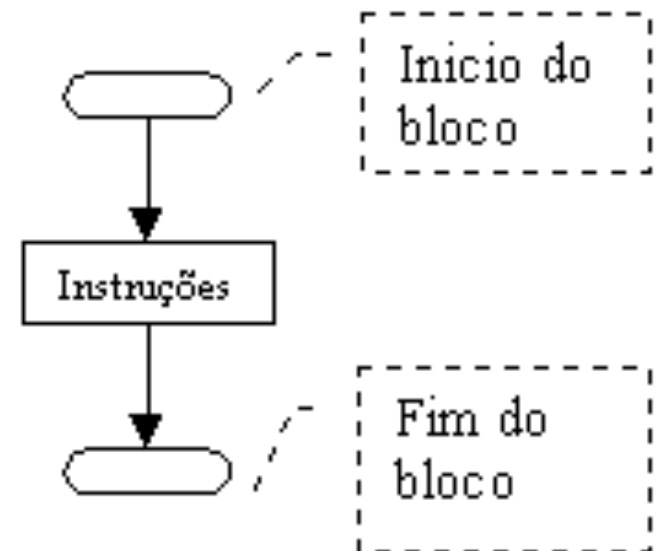
Estrutura de Controle Sequencial

Uma programação seqüencial é aquela cujas ações são executadas uma após a outra (TOP-DOWN) e identificadas pelo início e fim de cada bloco de instruções.

Exemplo :



FLUXOGRAMA





Estrutura de Controle Seqüencial

A Programação Estruturada é definida por uma seqüência de ações que seguem a seguinte composição:

- Constantes e Variáveis
- Símbolos
- Operadores
- Funções
- Comandos
- Sentenças

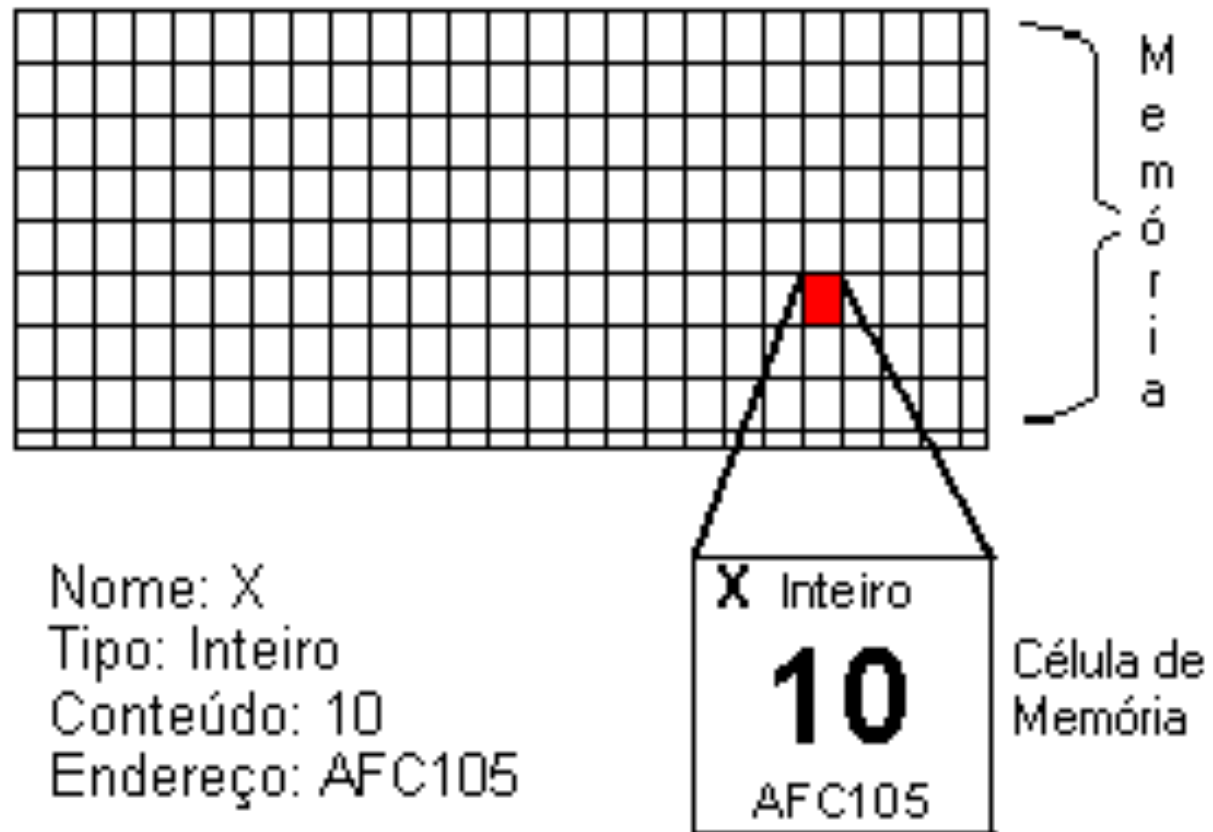


Constantes e Variáveis

Constantes - Vaz referências a dados que não sofrem alterações ao longo do algoritmo.

Variáveis - Vaz referências aos dados do problema e que deverão ser armazenadas na memória do computador e sofrem alterações ao longo do algoritmo. Toda variável deve ser classificada a um tipo exclusivo de informação pela qual a memória da máquina será preparada para armazenar o dado, e este tipo será incluído já no início dos procedimentos.

Alocação de variáveis na memória do computador





Elementos que compõem uma variável

Nome da Variável: Nome que será utilizado para identificar a informação e determinar onde a mesma se encontra na memória.

Tipo da Variável: É determinado pelo conteúdo da variável (informação a ser armazenada), e pode ser classificada nos seguintes modelos: INTEIRO e REAL para valores numéricos, CHARACTER para valores alfanuméricos, LÓGICO para valores verdadeiros ou falsos e MULTIDIMENSIONAIS para matrizes e vetores.



Elementos que compõem uma variável

Conteúdo: Valor atribuído à variável e que será colocado na memória.

Endereço: Endereço físico de armazenamento que será gerenciado pelo sistema operacional do computador e este evitará que outras informações invadam a área restrita a uma variável



Toda variável deve ser declarada

A declaração é dada sempre no topo do algoritmo.

Exemplo:

INICIO

INTEIRO A, X, idade;

REAL peso, L;

CARACTER nome, cep, K;

LOGICO resposta;

VETOR notas[10];

FIM.



Símbolos

Representação por símbolos que o algoritmo identifica e interpreta a fim de satisfazer um procedimento.

Os principais símbolos são:

- ← atribuição de dados a variáveis
- (início de um conjunto dados
-) fim de um conjunto de dados
- " início e fim de uma expressão caracter
- // comentário
- ; terminador de linhas de instruções
- , separador de conjunto de dados
- . fim do algoritmo



Símbolos

Exemplo:

INICIO

// declaração de variáveis

INTEIRO A, X, idade;

REAL peso, L;

CHARACTER nome, cep, K;

LOGICO resposta;

// atribuição de valores a variáveis

A ← 5;

Peso ← 65.4;

nome ← "FEPI" ;

resposta ← falso;

FIM.



Operadores

Símbolos aritméticos ou relacionais que geram processamento e retornam resultados.

- Aritméticos (cálculos)

**Adição(+), Subtração(-),
Multiplicação(*), Divisão(/) e
Exponenciação (^)**

- Relacionais (equivalência ou igualdade entre dois ou mais valores)

**Igual(=), maior(>), maior igual(>=),
menor(<), menor igual(<=) e
diferente(<>)**



Operadores

Exemplos de uso:

INICIO

// variáveis

INTEIRO A, B, C;

// principal

A \leftarrow 2; B \leftarrow 3;

C \leftarrow A + B;

//(C = 5)

B \leftarrow C * A;

//(B = 10)

C > A;

//(condição verdadeira)

B = A;

//(condição falsa)

A <> B;

//(condição verdadeira)

A \leftarrow C ^ B + C / A;

FIM.



Funções

Rotinas prontas no processador que geram operações sobre um determinado valor especificado e retornam resultados.

Funções numéricas

SEN(x) → seno

COS(x) → cosseno

ABS(x) → valor absoluto

SQRT(x) → raiz quadrada

INT(x) → parte inteira de um valor flutuante

ARRED(x,y) arredondamento de y casas decimais

MOD(x,y) → resto da divisão de x por y em inteiro binário

Funções para manipulação de string

ASC(x) → retorna o código ASCII do caracter especificado

CHR(x) → retorna o dígito correspondente ao código ASCII especificado

LEN(x) → número de dígitos de uma string

MID(x,m,n) → n dígitos de uma string a partir da posição m



Funções

Exemplo de uso:

INICIO

// variáveis

INTEIRO A, B, C, M;

CHARACTER: G, H;

// principal

A ← 25;

B ← -4;

G ← "UNIVERSITAS FEPI";

C ← **SQRT**(A); (C = 5)

C ← **ABS**(B); (C = 4)

M ← **LEN**(G); (M = 16)

H ← **ASC**(65); (H = "A")

FIM.



Comandos

Palavras chaves ou instruções pré-definidas que são interpretadas pelo processador e passam produzir iterações entre o usuário e a máquina.

Principais comandos:

LEIA() permite ao usuário informar um valor para a variável durante a execução do algoritmo.

ESCREVA() mostra na tela do computador o valor da variável em questão.

IMPRIMA() envia para impressora o valor da variável em questão.



Comandos

Exemplo de uso:

INICIO

INTEIRO: A, B;

REAL: C;

Leia(A);

Escreva(" Informe um valor para variável B: ");

Leia(B);

$C \leftarrow A / B$;

Escreva(A);

Escreva(" Resultado ", C);


Imprima(" A divisão de ", A, " por ", B, " é ", C);

FIM.



Sentenças

É a mistura das todas as estruturas afim de se obter um resultado ou uma solução para um problema. Formam uma sentença o conjunto de regras, comandos, variáveis, funções e símbolos, agrupados de forma lógica permitindo que o computador possa processar e interagir com os dados.



Apreender não é somente
entender, mas sim, saber
colocar em prática tudo
aquilo que entendeu.

1ª lista de exercícios sobre algoritmos



Estrutura de Controle Condicional ou de Seleção

Na Estrutura de Controle Condicional uma condição será posta em questão e o computador através de um processamento de dados contidos na memória reconhecerá se a condição é verdadeira ou falsa e assim determinará quais ações serão executadas.

Existem 3 tipos de estruturas condicional:

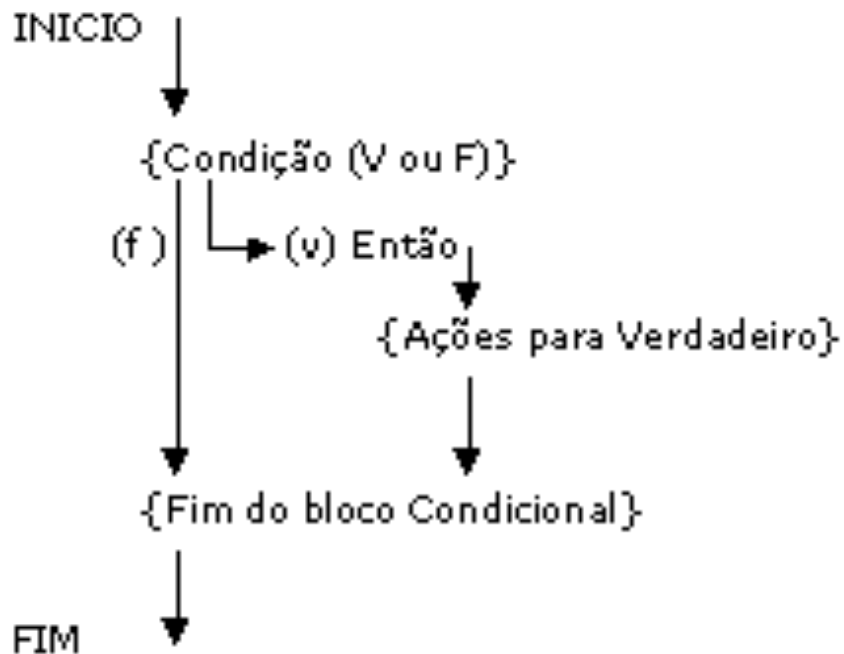
- Seleção Simples (SE-ENTÃO)
- Seleção Composta (SE-ENTÃO-SENÃO)
- Seleção Múltipla (CASO)

Seleção Simples

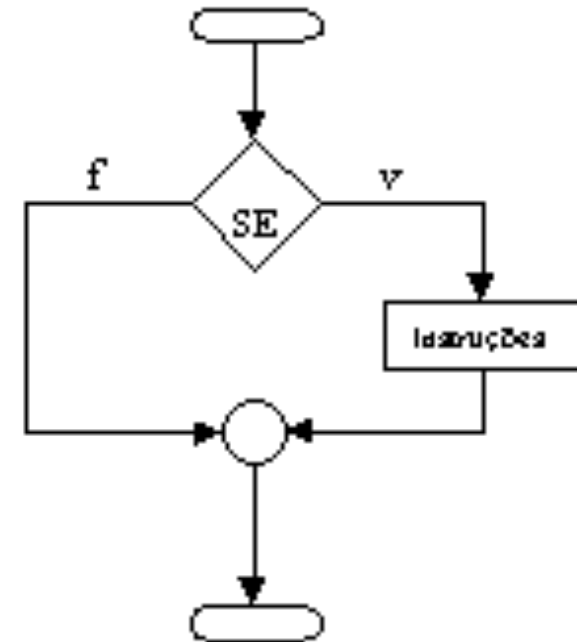
* SE-ENTÃO *

O bloco de instruções somente será executado se a verificação de entrada for verdadeira, caso isto não ocorra, o bloco condicional não será executado.

Pseudocódigo:



Fluxograma:





Exemplo utilizando o português estruturado:

a. Ir para escola

Procurar minha agenda

Se (encontrar a agenda)

Então

Levá-la para escola

Fim da condição

Pegar o ônibus

Chegar na escola

b. Ir a um baile

Chegar a portaria do clube

Mostrar a identidade

Se (idade menor que 18 anos)

Então

Apresentar acompanhante

Entrar no baile

Curtir a festa

Fim da condição

Exemplo em algoritmo:

Início

```
inteiro idade;  
caracter fase;  
Escreva("Informe a idade");  
Leia(idade);  
Se idade < 10 Então  
    fase ← "Criança";  
Fim se;  
Escreva(fase);
```

Fim.



Exemplo em algoritmo:

Inicio

```
inteiro idade;  
caracter resposta;  
Escreva("Informe a idade");  
Leia(idade);  
Se idade > 1 e idade < 18 Então  
    resposta ← "Jovem";  
Fim se;  
Se idade >=18 e idade <60 Então  
    resposta ← "Adulto";  
Fim se;  
Se idade >=60 Então  
    resposta ← "Velho";  
Fim se;  
Escreva(resposta);
```

Fim.



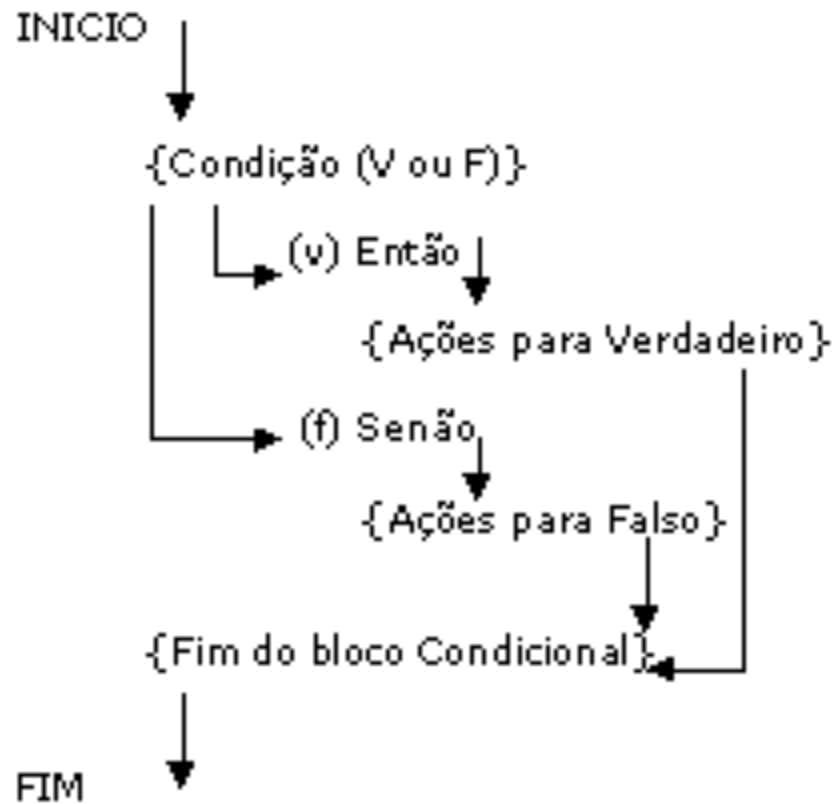
Seleção Composta

* SE-ENTÃO-SENÃO *

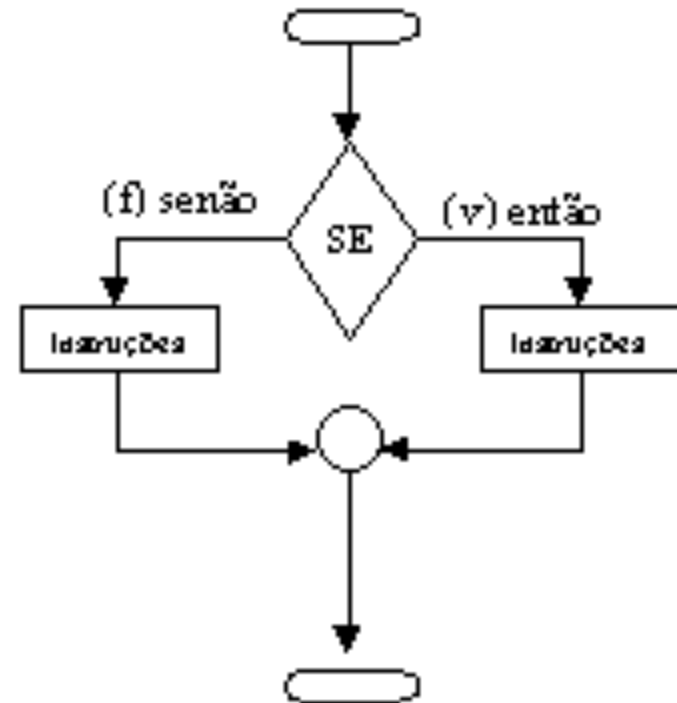
Neste caso passa a existir mais de um bloco de instruções porém apenas um será executado, e isto se dará pelo resultado obtido na verificação de entrada. Quando esta verificação resultar em caminhos distintos para verdadeiro ou para falso, chamamos de Seleção Dupla, mas quando na verificação existir várias possibilidades de respostas então a chamamos de seleção aninhada.

Modelo de Seleção Dupla

Pseudocódigo:



Fluxograma:





Exemplo utilizando o português estruturado:

Ir para escola

Se (fazer Frio) Então

Vestir blusa

Senão

Vestir Camiseta

Fim se

Chegar a escola

Exemplo em algoritmo:

INICIO

Inteiro: A, B, C;

Leia(A,B);

Se $A > B$ Então

$C \leftarrow A$;

Senão

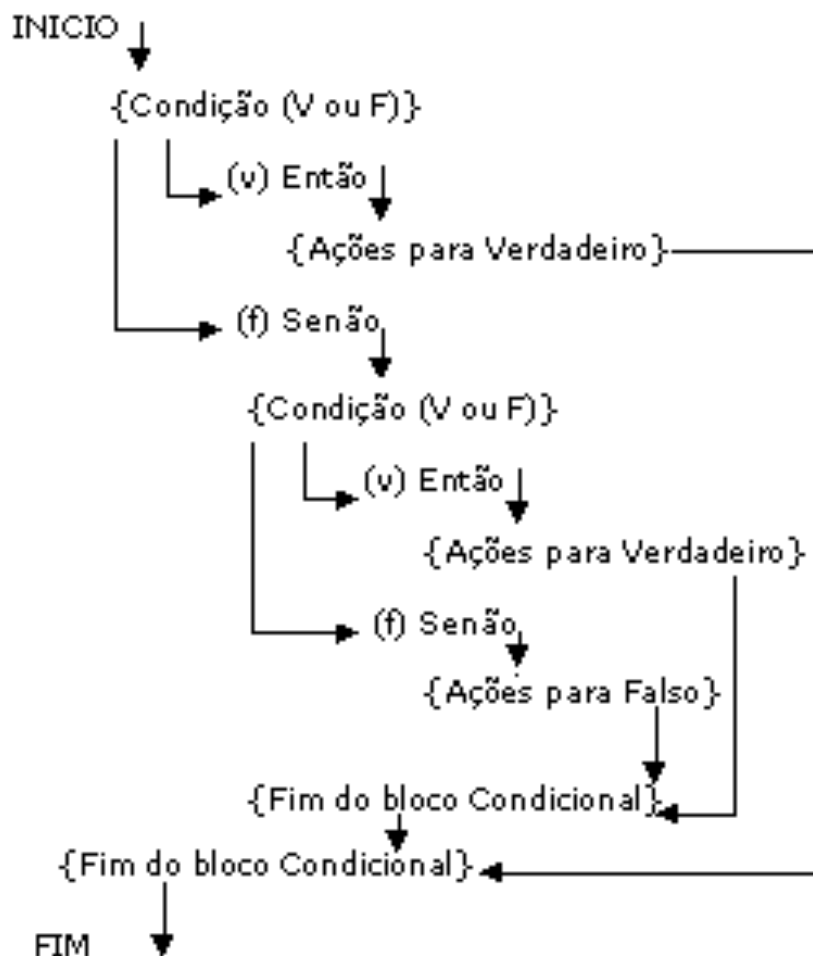
$C \leftarrow B$;

Fim se;

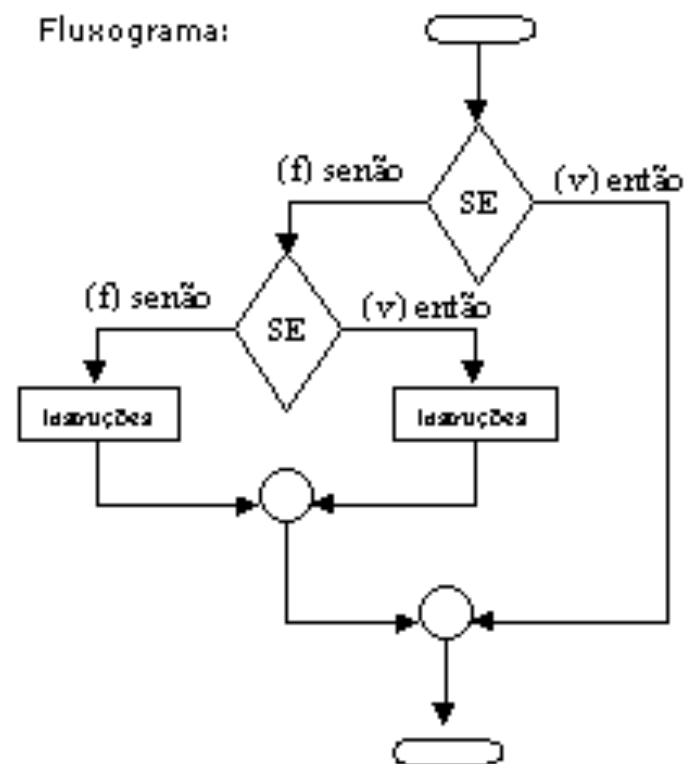
FIM.

Modelo de Seleção Aninhada

Pseudocódigo:



Fluxograma:



Exemplo utilizando o português estruturado:

Fazer a prova

Obter a nota

Se (nota < 30) Então

Reprovado

Senão

Se (nota \geq 30) e (nota < 60) Então

Recuperação

Senão

Aprovado

Fim se

Fim se



Exemplo em algoritmo:

Início

```
inteiro hora;  
caracter período;  
leia(hora);  
Se hora > 6 e hora < 12 então  
    período = manha;
```

Senão

```
    Se hora > 12 e hora < 18 então  
        período = tarde;
```

Senão

```
    Se hora > 18 e hora < 24 então  
        período = noite;
```

Senão

```
    Se hora > 0 e hora < 6 então  
        período = madrugada;
```

Fim se;

Fim se;

Fim se;

Fim se;

Fim.

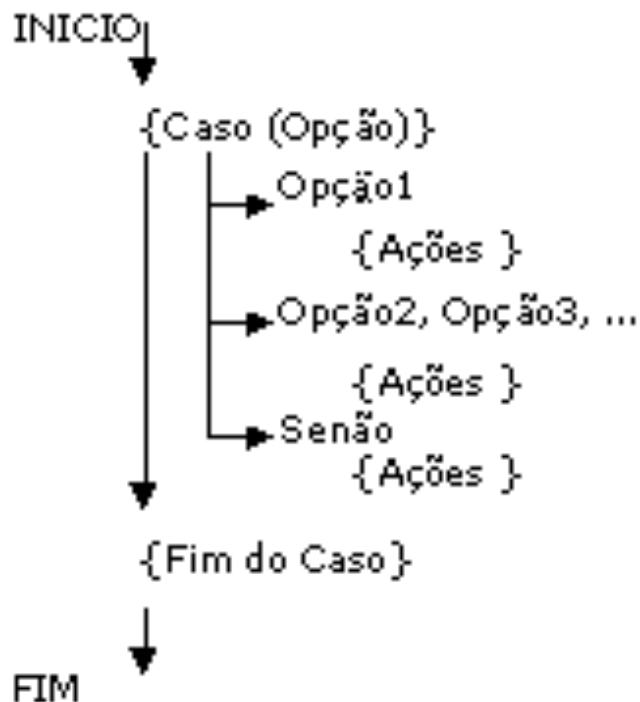


Seleção Múltipla

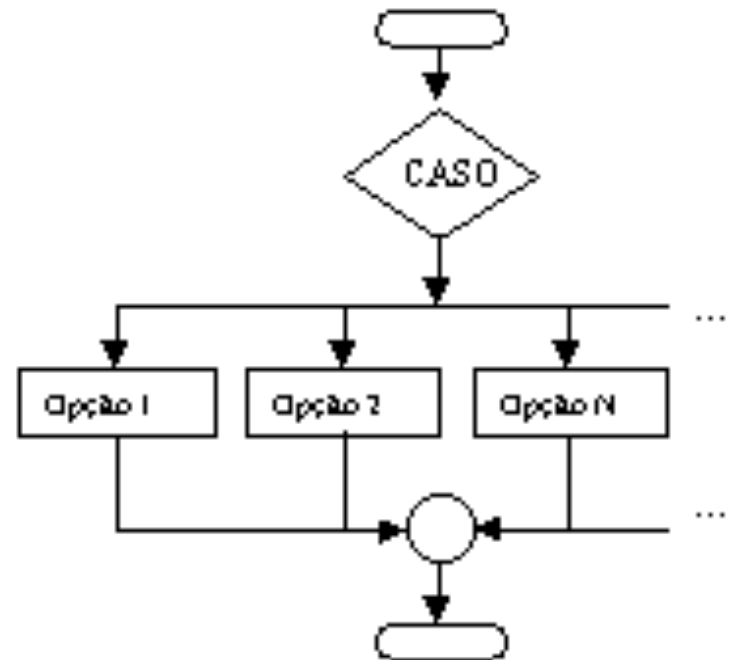
* CASO *

Utilizada quando temos muitas possibilidades para uma determinada situação, onde a aplicação da estrutura se...então...senão, tornaria o algoritmo muito complexo.

Pseudocódigo:



Fluxograma:





Exemplo em Português Estruturado: Cumprimentar alguém

Olhe as Horas

Caso(Horas)

≥ 6 e < 11 : Bom Dia

≥ 12 e < 18 : Boa Tarde

≥ 18 e < 24 : Boa Noite

Fim do caso

Exemplo em Algoritmo:

Inicio

Inteiro Numero;

Caracter Extenso;

leia(Numero);

caso(Numero)

1: Extenso \leftarrow 'Um';

2: Extenso \leftarrow 'Dois';


3: Extenso \leftarrow 'Três';

4: Extenso \leftarrow 'Quatro';

senão: Extenso \leftarrow 'Erro';

fim-caso;

Fim.



Você não precisa saber
tudo, basta saber onde
encontrar tudo o que
precisa.

2ª lista de exercícios sobre algoritmos



Programação Estruturada de Repetição

Uma estrutura de Repetição é aquela cujas ações são executadas repetidamente, enquanto uma determinada condição permanece válida. O fator que diferencia os vários modelos de estrutura de repetição é dado pelo conhecimento do número de loops ou pelo acondicionamento da estrutura por meio de testes e condições.

Existem 3 tipos de estruturas de repetição:

- Para-Faça
- Enquanto-Faça
- Repita-Até

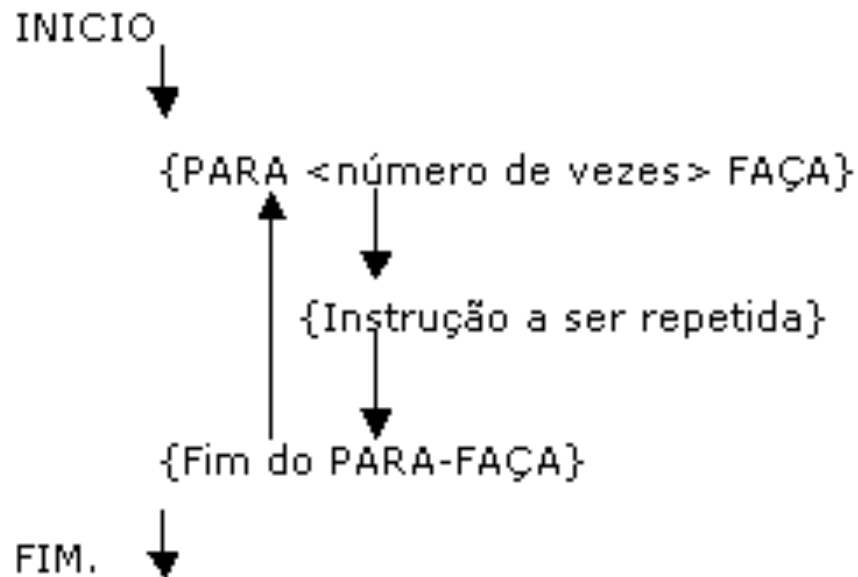


Repetição por Laços definidos

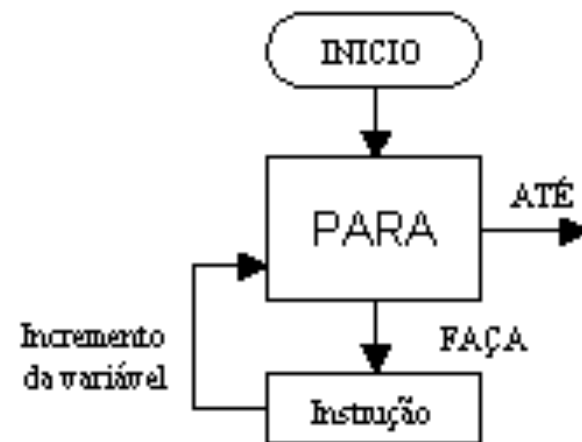
* PARA-FAÇA *

Usamos a estrutura Para-Faça quando precisamos repetir um conjunto de comandos em um número pré-definido de vezes. É conhecida como estrutura de repetição por laços (loops) definidos, pois se utiliza uma variável de controle que é incrementada em um número determinado de unidades de um valor inicial até um valor final.

Pseudocódigo:



Fluxograma:



Quando o algoritmo encontra a instrução fim-para, incrementa a variável INICIO em 1 unidade (default) ou mais. Cada vez que é completada o bloco de instrução, ele retorna a linha da estrutura PARA e testa se INICIO é menor ou igual a FIM, se for menor ou igual o processo continua no laço (loop), caso não, o processo é abandonado.

Obs: O valor da variável INICIO não pode ser alterado no interior da estrutura.



Exemplo Prático: (utilizando o português estruturado)

Ir de elevador do primeiro ao quinto Andar
Chamar o elevador
Entrar no elevador
Informar o andar
Para andar do primeiro até o quinto faça
 mover ao próximo andar
Fim do Movimento
Sair do elevador

Exemplo Prático: (utilizando o algoritmo)

a.

```
Inteiro var, resultado;  
para var ← 1 até 10 faça  
    resultado ← 2 * var;  
fim-para;  
escreva(resultado);
```




Exemplo Prático: (utilizando o algoritmo)

b.

```
Inteiro var, resultado;  
para var ← 1 até 10 faça  
    resultado ← 2 * var;  
    escreva(resultado);  
fim-para;
```

c.

```
Inteiro var, resultado;  
para var←1 até 5 passo 2 faça  
    resultado ← 2 * var;  
    escreva(resultado);  
fim-para;
```



Séries finitas e infinitas através de estrutura de repetição

Dadas as informações a seguir escreva a série

Variavel `x`

Valor inicial `1`

Valor final `30`

Regra de ciclo `x<-x+1`

Formula padrao `x+(x*2)/(x^2)`

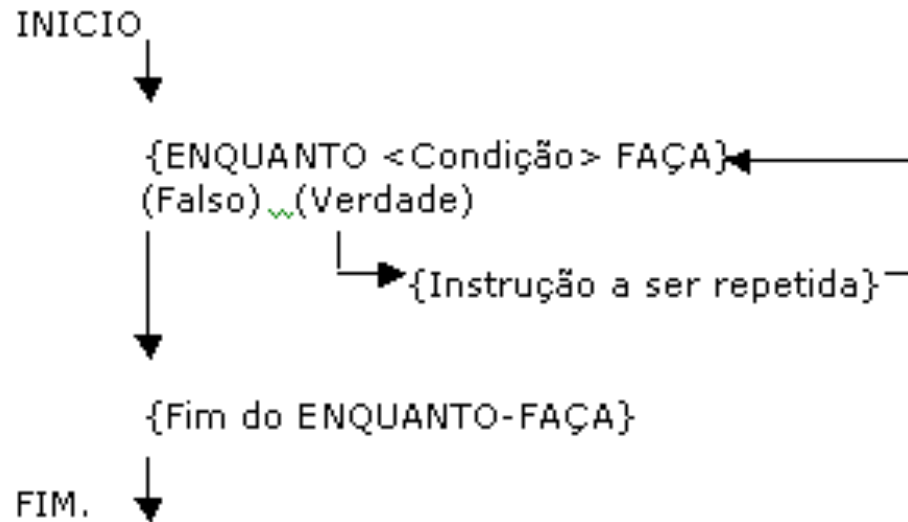


Repetição por Condição

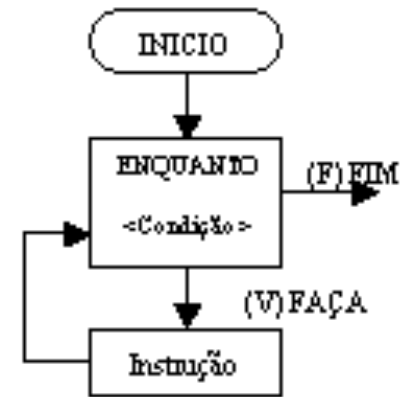
* ENQUANTO-FAÇA *

Utilizada quando não sabemos o número de repetições e quando possuímos uma expressão que deve ser avaliada para que os comandos da estrutura sejam executados. Assim, enquanto o valor da <condição> for verdadeiro, as ações dos comandos são executadas. Quando for falso, a estrutura é abandonada, passando a execução para a próxima linha após o comando FIM-ENQUANTO. Se já da primeira vez o resultado for falso, os comandos não serão executados.

Pseudocódigo:



Fluxograma:



É sempre importante observar que primeiro se analisa a condição para depois dependendo do resultado obtido executar o bloco a ser repetido. Caso a condição não seja satisfeita nada será feito e a próxima linha após o fim-enquanto será requisitada. Também é necessário caso a condição seja verdadeira permitir o incremento para a variável em condição (se necessário) para que a estrutura não entre em loop infinito.

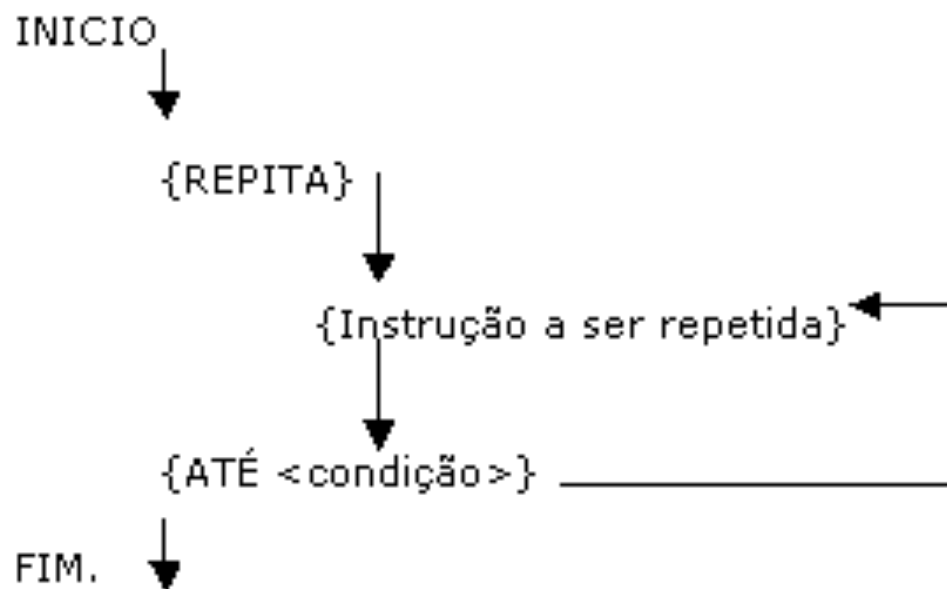


Repetição por Condição

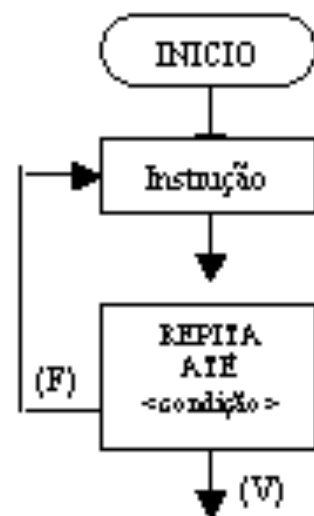
* REPITA-ATÉ *


Utilizada quando não sabemos o número de repetições e quando os comandos devem ser executados pelo menos uma vez, antes da expressão ser avaliada. Assim, o programa entra na estrutura Repita...Até que executa seus comandos pelo menos uma vez. Ao chegar no fim da estrutura, a expressão será avaliada. Se o resultado da expressão for verdadeiro, então o comando é abandonado.

Pseudocódigo:



Fluxograma:





Entender não é a mesma
coisa que aprender.
Somente a prática leva a
perfeição.

3ª lista de exercícios sobre algoritmos



Variável Dimensionais

- Variáveis Unidimensionais: Vetores
- Variáveis Bidimensionais: Matrizes
- Mecanismos de Busca e Ordenação



Capítulo II

Programando com Visualg

- Estrutura da Programação em C
- Estrutura de Programação em Java