

## S.I. LÓGICA DE PROGRAMAÇÃO

Prof.: Marcos Henrique Gomes

### Orientações de estudo:

A principal fonte de consulta e estudo deverá ser a bibliografia recomendada, este material deve ser trabalhado como notas de aula.

#### Bibliografia Básica

MANZANO, José Augusto. *Algoritmos - Lógica para Desenvolvimento de Programação*. Rio de Janeiro: Érica, 1996.

WIRTH, Niklaus. Algoritmos e estruturas de dados. ETH Zurich: LTC Editora, 1999.

SALVETTI, Dirceu Douglas. Algoritmos. São Paulo: Pearson/Makron Books, 2004.

KERNIGHAN, Brian W. *A linguagem de programação*. Porto Alegre: Editora Campus Ltda, 1987.

#### Bibliografia Complementar

CELES, Waldemar, . *Estruturas de Dados*. Rio de Janeiro: PUC-RIO, 2002.

BROOKSHEAR, J. Glenn. *Ciência da Computação é Uma visão abrangente*. Porto Alegre: Bookman, 2005.

PUGA, Sandra. *Lógica de programação e Estrutura de Dados*. : Pearson - Prentice Hall, .

SEBESTA, Robert W. *Concepts of Programming Languages*. : Addison-Wesley, 1996.

As listas de exercícios e respectivas resoluções serão postadas no portal Moodle e no endereço: [www.profmarcoshg@webs.com](mailto:www.profmarcoshg@webs.com)

## CONCEITO DE LÓGICA

A lógica trata da elaboração do pensamento. Da filosofia, procura saber por que pensamos assim e não de outra maneira, questionamentos quanto a incertezas tem sido foco de pesquisador ao longo da história da humanidade. Com arte ou técnica, nos, ensinando a usar corretamente as leis do pensamento.

Poderíamos dizer também que a lógica é a arte de pensar corretamente e, visto que a forma mais complexa do pensamento é o raciocínio, a lógica estuda ou tem em vista a “correção do raciocínio”. Dizemos então que a lógica tem em vista a “ordem da razão”.

Isto dá a entender que a nossa razão pode funcionar desordenadamente. Por isso a lógica ensina a colocar Ordem no Pensamento.

Uma parte importante da lógica é o estudo do erro (falácia) ou métodos incorretos do raciocínio. Então podemos afirmar que o indivíduo pode cometer erros lógicos e é com o intuito de evitá-los que estudamos técnicas que possibilitam evitar o erro.

## LÓGICA NO COTIDIANO

Os conceitos relacionados à lógica não estão somente atrelados a matemática, seu significado está corretamente associado à resolução de problemas sejam eles de qualquer ordem. Ao resolver problemas relacionados à fala ou a coordenação motora a criança apresenta soluções que lhe permitirão superar os obstáculos, o que podemos ter como exemplo é o de caminhar, comunicar-se, enfim o sujeito evoluindo.

Da mesma maneira que um mestre de obras decide em construir determinada estrutura a partir do empírico realizando seu trabalho, sem a necessidade de realizar cálculos.

## LÓGICA DE PROGRAMAÇÃO

Definimos por lógica de programação o conjunto das técnicas de encadeamento do pensamento para atingir determinados objetivos, isto é, encapsular um processo através de comandos lógicos.

Pensamentos que devem obedecer a uma seqüência lógica que corresponde ao conjunto de *instruções* que corresponde às regras ou normas definidas para o emprego ou realização de uma determinada ação. Em programação dizemos que a instrução é a informação que indica a máquina (computador) uma ação elementar a ser executada.

Para que uma ordem seja executada de forma a dizermos que um processo foi realizado de maneira completa não basta apenas uma instrução, mas um conjunto de instruções obedecendo a uma seqüência lógica chamado de programa ou sistemas.

## CONCEITO DE SISTEMA

Compreende-se por sistema a agremiação de determinadas funções ou informações que em conjuntos realizam determinada tarefa. Nas organizações empresariais os sistemas são interdependentes formando um grupo de ações que compõe o sistema maior (geral) da empresa.

Os homens encarregados de fazer o planejamento estratégico de um sistema são chamados de generalistas ou desenhistas de sistemas, sua principal competência é de sintetizar problemas.

Tendo os sistemas surgidos do conceito de natureza e da decomposição dos organismos humano interdependentes podemos citar como exemplo os seguintes exemplos.

- Sistema solar;
- Sistema sanguíneo;
- Sistema cardiovascular;
- Sistema de transportes;

## DIFERENTES TIPOS DE LÓGICA

### *Lógica proposicional*

Verifica-se a validade do argumento. Um argumento é considerado válido se todas as suas instancias são válidas. Entretanto no silogismo disjuntivo podemos afirmar que mesmo quando os argumentos são verdadeiros se as premissas não forem verdadeiras a conclusão pode não ser verdadeira.

Ou seja, a partir do silogismo disjuntivo podemos ter argumentos válidos que só podem conduzir a uma conclusão válida se as premissas forem verdadeiras.

Exemplo

Os gatos e os cães são mamíferos.

Eu sou um mamífero.

Eu não sou um cão.

Logo eu sou um gato.

Utilizando uma forma de argumento válida podemos ter uma conclusão falsa. Pois a primeira premissa é falsa (existem muitos outros seres que são mamíferos além de um cão e um gato). Entretanto nada impede que, chegemos a conclusões verdadeiras a partir de premissas falsas.

Exemplo:

Nenhum mamífero nasce de ovos.

Gatos são mamíferos.

Portanto gatos não nascem de ovos.

Note que tanto a forma de argumento quanto a conclusão são verdadeiras, entretanto a conclusão foi deduzida a partir de uma premissa falsa, pois a afirmação “Nenhum mamífero nasce de ovos” não é verdadeira o ornitorrinco (animal australiano) é mamífero e nasce de ovos.

Há ainda a possibilidade de encontrar argumentos que não levam a conclusões verdadeiras, neste caso dizemos que as premissas são verdadeiras apensar da conclusão ser falsa. Logo o argumento é inválido.

Exemplo:

Se você está estudando em marte, então você está vivo.

Você está vivo.

Portanto você está em marte.

Note: as premissas são verdadeiras, entretanto a conclusão é falsa. Portanto o argumento é inválido.

### *Lógica probabilística*

Em grande destaque na modelagem computacional está à lógica probabilística que associação a lógica matemática as inferências probabilísticas, os cálculos são baseados no conceito de 0 ou 1 e não no conceito de 0 e 1 como no tradicional processamento de dados característico no tratamento quantitativo.

Utilizando-se do conceito probabilístico engenheiros planejam criar um chip capaz de trabalhar realizar os processos de maneira mais dinâmica gerando menor ruído e menor consumo de energia.

## **Lógica Fuzzy**

Diferentemente da aleatoriedade, certas variáveis utilizadas em nosso cotidiano, transmitidas e perfeitamente compreendidas linguisticamente entre interlocutores, têm invariavelmente permanecido fora do tratamento matemático tradicional. Este é o caso de variáveis lingüísticas oriundas da necessidade de se distinguir qualificações por meio de graduações.

Para descrever certos fenômenos relacionados ao mundo sensível, temos utilizado graus que representam qualidades ou verdades parciais ou ainda padrões do melhor (na linguagem sofista). Este é o caso do conceito de alto, fumante, infeccioso, presa, etc.

É nesse tipo de incerteza que a lógica Fuzzy tem dado suas principais contribuições. Usando o conceito de conjunto das pessoas, altas ou das doenças infecciosas, isto é conjunto definido por propriedades subjetivas ou atributos imprecisos.

Na modelagem Fuzzy é associado um valor  $x(p)$  a uma proposição  $p$ , indicando o grau de validade desta proposição, cujo conjunto imagem está entre 0% e 100%, permitindo uma gama de infinitos valores possíveis.

Um exemplo de proposições baseadas na Lógica Fuzzy é o seguinte:

Suponha que se deseja representar de forma Fuzzy a altura de quatro indivíduos, e 'p' será alto.

Milena 1,65m  
Matheus 1,75m  
João 2,0 m  
Nadine 1,45m

1° Milena é alta,  $x(p) = 55\%$   
2° Matheus é alto,  $x(p) = 75\%$   
3° João é alto,  $x(p) = 100\%$   
4° Nadine é alta,  $x(p) = 0\%$

Percebe-se que a lógica Fuzzy está associada à teoria dos conjuntos. E que cada afirmação representa o grau de pertinência ao conjunto das pessoas altas.

Apesar das diferentes versões de lógica todas estão ligadas a determinação da verdade de uma afirmação de certeza quanto a um conjunto de proposições ou afirmações que podem compor um teorema matemático ou a base de um sistema de computação. Estes dois têm suas principais diferenças em:

- Um programa lida tipicamente com a lógica de predicados ou proposicional, enquanto o matemático usa a língua corrente associada a símbolos típicos da matemática.
- Enquanto um matemático utiliza-se de intuição, raciocínio e estratégias pessoais, o programa se utiliza de uma instrução algorítmica fixa.

## **ALGORÍTMO NO COTIDIANO**

O pensamento encadeado que leva a uma ação ou o processo organizado pela lógica que nos permite realizar nossas tarefas de maneira simples, pode ser descrito como uma seqüência de instruções, que devem ser obedecidas para se cumprir às tarefas. Esta seqüência chamada de algoritmo está presente no nosso dia a dia. Por exemplo, há algoritmos para a construção de maquetes (expressos na forma de instruções de montagem) ou para tocar música (na forma de partituras). O mais conhecido algoritmo é o da divisão, para encontrar o quociente de dois números ou para calcular o máximo divisor comum de dois inteiros positivos<sup>1</sup>.

Exemplos de algoritmos (não computacionais)

Objetivo: Usar um telefone público.

Início

1. tirar o fone do gancho;
2. ouvir o sinal de linha;
3. introduzir o cartão;
4. digitar o número pretendido;
5. se der o sinal de chamar
  - 5.1 conversar
  - 5.2 desligar
  - 5.3 retirar o cartão
6. se der o sinal ocupado
  - 6.1 colocar o fone no gancho
  - 6.2 repetir o processo à partir da ação 1.

Fim

exemplo 02

Objetivo: Somar dois números quaisquer.

Início.

1. Escreva o primeiro número no retângulo A;
2. Escreva o segundo número no retângulo B;
3. Some o número do retângulo A com o número do retângulo B e coloque o resultado no retângulo C.

Fim.

## TÓPICO 04

### USO DE LÓGICA APLICADA A INFORMÁTICA

Na informática a lógica é tratada como “lógica de programação” e seu trabalho é automatizar o pensamento de tal maneira que as atividades possam ser realizadas por uma máquina.

O algoritmo utilizado na programação tem sua sub-divisão lógica na instrução. Em linguagem comum entendemos por instrução o conjunto de regras, normas ou comandos elementares pré-definidas com fins de realizar determinada tarefa.

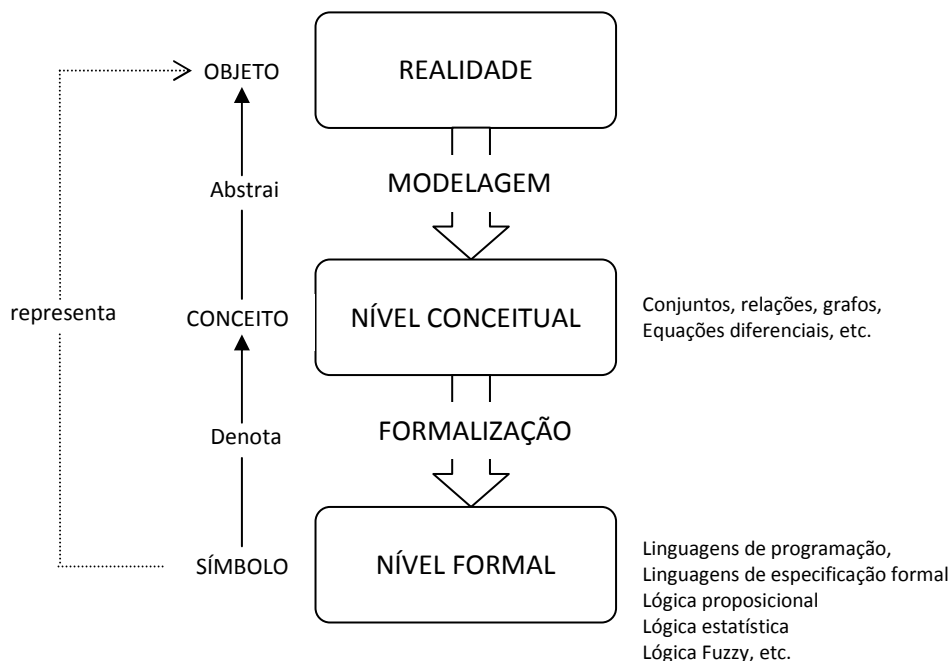
É importante ressaltar que uma ordem isolada não consiste na realização de um processo, mas o conjunto de instruções (sistema) encadeadas obedecendo a uma seqüência lógica é que realiza um processo.

### PROBLEMAS COMPUTACIONAIS

Os problemas computacionais estão relacionados a manipulação da informação, ou seja sempre haverá “uma pergunta de caráter geral a ser respondida”. Onde um problema é descrito especificando-se a entrada (quais são os possíveis dados) e a condição (ou relações) que a saída deve satisfazer para uma determinada entrada.

O generalista, desenhista de sistema ou programador de computador, quando encarregado de construir um algoritmo ou uma base de dados para certa aplicação tem a problemática de relacionar as informações que deverão ser tratadas com limitações impostas desde espaço físico quando um profissional programa um manipulador robótico, as relações ditadas pela interação empregados e empregador, clientes e fornecedores questões sociais e econômicas até limitações relacionadas ao sentido, raciocínio e intuição. Tal processo de construção é denominado *modelagem*.

Durante a análise e construção do algoritmo computacional que deverá conter entidades matemáticas tais como conjuntos, funções, grafos, relações, equações diferenciais, etc., o profissional deve construir um modelo conceitual que, não apenas inclua todos os detalhes necessários a solução do problema, mas também não inclua detalhes em excesso, que poderão comprometer a eficiência do processo computacional.



## ALGORÍTMO APLICADA A INFORMÁTICA

O processo de cálculo matemático ou de resolução de problemas semelhantes é conhecido por programa que é composto por um conjunto de algoritmos interdependentes trabalhando de maneira compreensível para a máquina, proporcionando um passo a passo na solução de um problema.

No momento em que a tarefa passa a ser realizada seja parcial ou totalmente pela máquina está sendo realizado um processo de automação. Este para obter êxito em seu trabalho necessita ser instruído em todas as etapas do processo que realizará, esta instrução por sua vez deverá ser dada de maneira a realizar a tarefa com a maior eficiência e eficácia possíveis.

O tratamento destas instruções é dado pela capacidade de captar e transferir inteligência mediante os algoritmos que são construídas as máquinas que exibem comportamento inteligente. Por conseguinte, o nível de inteligência demonstrada pela máquina fica limitada pela inteligência que um algoritmo é capaz de transportar. Por outro lado, se não houver algoritmo capaz de executar tal tarefa, então a sua realização excede a capacidade da máquina.

Estudos fundamentados em um conhecimento lingüístico e gramatical com o objetivo de criar algoritmos capazes de criar máquinas consideradas inteligentes conduziram a uma grande diversidade de esquemas para a representação de algoritmos conhecidos como **linguagem de programação** ou paradigmas de programação.

A busca de algoritmos para controlar as tarefas cada vez mais complexas que possibilitem atribuir inteligência a máquina conduziu a situações relacionadas as limitações do algoritmo. A inexistência de algoritmos que não tem solução implica que o problema não pode ser resolvido por uma máquina. Isto é, as máquinas são capazes de resolver apenas problemas passíveis de solução algorítmica.

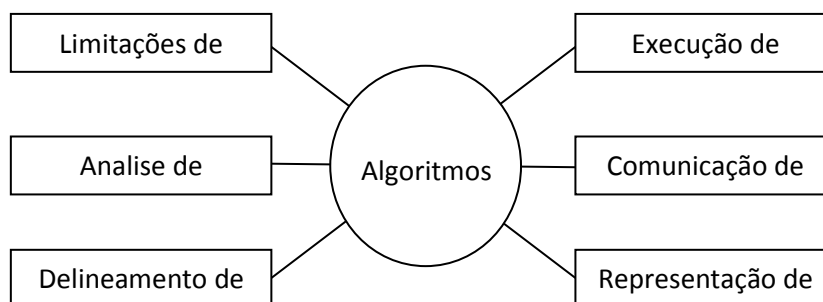
Para representar um algoritmo, são empregados diversos tipos ou técnicas e cabe ao profissional, utilizar aquele que se adéqua as suas necessidades.

Existem algoritmos que apresentam passo a passo do processo apenas a nível lógico, ou seja, não especificam em detalhes a linguagem de programação que se utilizada, e outros que tratam os passos do programa com maior riqueza de detalhes.

Na medida em que se avançou no estudo de algoritmos tornou-se necessário normatizar métodos e ou técnicas que possibilitaram um enfoque epistemológico que principiam nos seguintes questionamentos.

- Quais problemas poderão ser resolvidos por meio de processos algorítmicos?
- De que modo o processo de descoberta de algoritmos podem ser facilitados?
- Como se pode melhorar as técnicas de representação e comunicação de algoritmos?
- Como o nosso conhecimento de algoritmos e da tecnologia pode ser aplicado na obtenção de máquinas algorítmicas melhores?
- De que maneira as características de diferentes algoritmos podem ser analisadas e comparadas?

Que analiticamente são respondidos através do diagrama abaixo:



## TÓPICO 06

### PRINCIPAIS MÉTODOS DE REPRESENTAÇÃO DE UM ALGORITMO

#### *Fluxograma e diagrama de blocos*

Representação gráfica do processo, ou seja, das instruções e ou módulos do processamento, que compõem o algoritmo e que também é conhecido como diagrama de blocos.

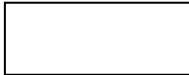
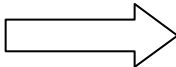
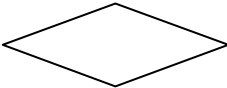


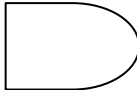
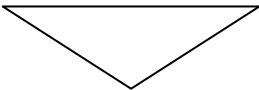


No planejamento o “generalista” ou “desenhista de sistema” tem uma poderosa ferramenta de representação gráfica que possibilita:

- Rápida análise durante o processo de planejamento e construção do sistema.
- Identificação das atividades críticas para cada um dos processos.
- Identificação dos subsistemas e suas ligações que permitem a interdependência.
- Conhecimento da seqüência encadeada das atividades dando uma visão de luxo do processo.
- Documentação do processo para análises futuras, adequação a normas e certificações e esclarecimento sobre o funcionamento para novos colaboradores no sistema.

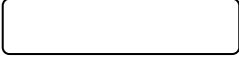
#### *Diagrama de blocos*

- Indica as atividades realizadas sem diferenciá-las por tipo.
- Visualização rápida do processo.
- Desenho horizontal ou vertical
- Utiliza frases curtas que identifiquem as atividades realizadas.

#### **Símbolos de fluxogramas (padrão ANSI)<sup>1</sup>**

SÍMBOLO	SIGNIFICADO
	Operação
	Movimento/transporte
	Ponto de decisão
	Inspeção
	Documento impresso
	Espera
	Armazenagem
	Sentido do fluxo
	Conexão

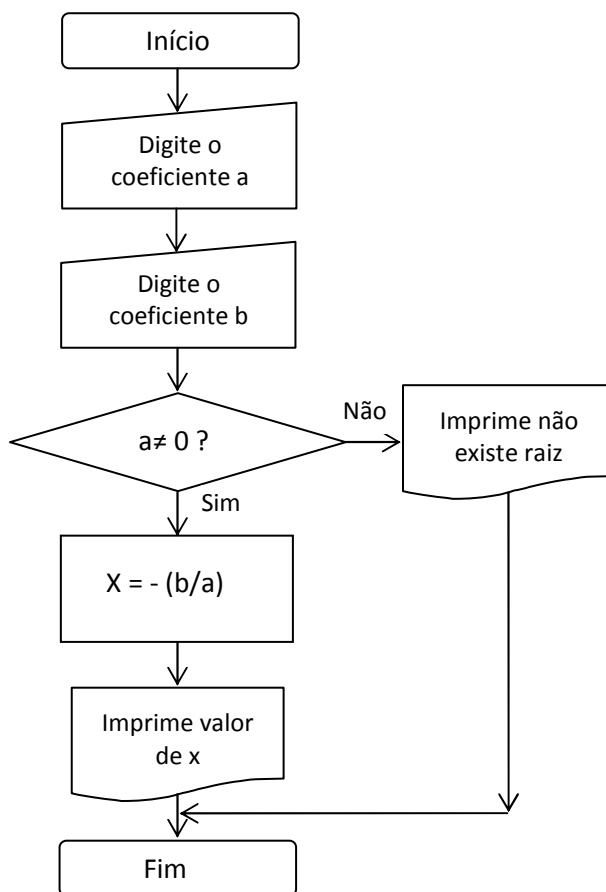


	Limites (início, pare, fim)
	Entrada de dados teclado

### Observações

- 1° - o uso de abreviaturas e siglas deve ser evitado.
- 2° - o tamanho dos retângulos para cada órgão pode variar conforme o nível de hierarquia.
- 3° - podem-se variar cores, com fins de destacar determinado órgão da estrutura.

### EXEMPLOS DE FLUXOGRAMA



## DESCRIÇÃO NARRATIVA OU SEQUENCIA LÓGICA

Largamente utilizado em manuais de instrução, por sua simplicidade de compreensão e linguagem acessível a todo usuário.

Exemplo.

1. Aperte o botão "ligar" do seu controle remoto;
2. No quadro canais, pressione a tecla correspondente ao canal que quer assistir;
3. Pressione seta para a se quer ou para a direita para selecionar o volume de seu aparelho;
4. ...
5. ...

O método de descrição narrativo é pouco usado na construção de algoritmos computacionais devido a sua natureza lingüística, que pode proporcionar em diversos casos interpretações erradas do objetivo, pois na construção de um algoritmo (conjunto de regras) ele deve ser claro e preciso, evitando tarefas que incorram em redundância ou comandos subjetivos em sua definição. Exemplo.

- Acabamento final (O acabamento vem sempre no fim)
- A avaliação foi antecipada para antes da data marcada (se foi antecipado foi para antes)
- Ganhei inteiramente grátis (se ganhou foi grátis)

## PORTUGOL (PSEUDO CÓDIGO)

Muito semelhante às linhas de código de uma linguagem de programação, a técnica apresenta um algoritmo rico em detalhes, onde temos de definir variáveis, rotinas, sub-rotinas, etc... Ao contrário do fluxograma que se utiliza de símbolos utilizamos comandos para solicitar uma determinada tarefa. Sua sintaxe básica é:

Algoritmo <nome do algoritmo>

<declaração das variáveis>

Início da rotina

<instruções a serem seguidas>

Fim da rotina

Exemplo:

```
Algoritmo Valor_dia
Var Salariominimo,media:real
```

```
Início
SalarioMinino = 510,00
Media=Salariominimo/30
Envie para impressora "Media"
```

```
Fim
```

Note que a declaração da variável Var é do tipo Real representando uma variável que pode assumir valores pertencentes ao conjunto dos números reais.

---

<sup>1</sup> – Este algoritmo pressupõe que sejam fornecidos dois inteiros positivos e calcula o seu máximo divisor comum.

Procedimento:

**Passo 1.** Atribuir, inicialmente, a M e N os valores correspondentes ao maior e menor dos dois números inteiros positivos fornecidos, respectivamente.

**Passo 2.** Dividir M por N, e chamar de R o resto da divisão.

**Passo 3.** Se R não for 0, atribua a M o valor de N, a N o valor de R e retorne ao passo 2; caso contrário, o máximo divisor comum será o valor correspondente de N.

<sup>2</sup>- **American National Standards Institute** ("Instituto Nacional Americano de Padronização"), também conhecido por sua sigla **ANSI**, é uma organização particular sem fins lucrativos que tem por objetivo facilitar a padronização dos trabalhos de seus membros.

Segundo a própria organização, o objetivo é melhorar a qualidade de vida e dos negócios nos Estados Unidos. São conhecidos por terem inúmeros padrões, entre eles o ANSI C que serve como guia na escrita de compiladores e de programas nesta linguagem de programação.

## TIPOS DE DADOS

### *Estrutura básica de dados*

A grande massa de informações a serem processadas representa de certo modo a abstração de uma parte da realidade. Enquanto a informação disponível ao computador consiste de um conjunto selecionado de *dados* a cerca do problema a ser resolvido. A partir dos quais é possível obter-se os resultados desejados. (Polya – Método para solução de problemas)

Os dados têm certas propriedades e características do objeto real que são desprezadas, por serem inexpressivas ou irrelevantes para a solução do problema corrente em particular. Desta maneira, a abstração pode ser também, visualizada como uma simplificação dos fatos. Exemplo a altura de um indivíduo em um determinada cadastro de crédito bancário.

Na solução de um problema, com ou sem o auxílio do computador, é necessário escolher uma abstração da realidade, isto é, definir um conjunto de dados que irão representar uma situação real. Esta escolha deve ser orientada segundo as características do problema a ser resolvido.

A escolha da forma de representar os dados é uma tarefa que não depende apenas do problema a ser resolvido, mas dependerá dos recursos disponíveis do ambiente interno e externo e das operações a serem realizadas.

### *Tipos de dados*

Matematicamente é fácil classificar as variáveis que envolvem um problema com características claras como variáveis reais, complexas, lógicas, que representam valores unitários ou conjuntos de valores. No processamento de dados se faz necessário que inclua ainda características incorporadas às linguagens de programação, que são:

- Tipo de dados que determina o conjunto de valores assumidos por uma constante ou variável, gerados por uma função ou um operador; (Exemplo  $\pi$ )
- Operador ou função que exigem argumentos de um dado tipo fixado e produz um resultado. Não necessariamente do mesmo tipo.

### *Temos em C os seguintes tipos de dados:*

1. **char** “Caractere”: O valor armazenado é um caractere, geralmente são armazenados em códigos, com tamanho de 8 bits.
2. **int** “Número inteiro”: É o tipo padrão e o tamanho do conjunto que pode ser representado normalmente depende da máquina em que o programa está rodando.
3. **float** “Número em ponto flutuante”: De precisão simples. São conhecidos normalmente como números reais.
4. **double** “Número em ponto flutuante de precisão dupla”

### *Indicador de função:*

**void** “Indica que não há tipo definido”: Este tipo serve para indicar que um resultado não tem um tipo definido. Uma das aplicações deste tipo é criar um tipo vazio que pode posteriormente ser modificado para um dos tipos anteriores.

### ***Modificadores dos Tipos Básicos:***

Podem ser aplicados modificadores aos tipos básicos, estes modificadores correspondem ao mecanismo de alteração de tamanho do conjunto de valores que o tipo pode representar. Estes modificadores são:

unsigned, short, long

Note: quando um destes modificadores é utilizado sozinho os compiladores interpretam que é do tipo 'int'. uma das aplicações está no armazenamento de números inteiros maiores (nº de caracteres) ou armazenar um número sem sinal.

Tipos de dados (padrão ANSI C)

Tamanho (bits)	Tipo	Faixa
1	char	-127 a 127
1	unsigned char	0 a 255
4	int	-2.147.483.648 a 2.147.483.647
4	unsigned int	0 a 4.294.967.295
2	short int	-32.768 a 32.767
2	unsigned short int	0 a 65.535
4	long int	-2.147.483.648 a 2.147.483.647
4	unsigned long int	0 a 4.294.967.295
4	float	Seis dígitos de precisão
8	double	Dez dígitos de precisão
10	long double	Dez dígitos de precisão

## TÓPICO 08

### *Variáveis*

Uma variável é a representação dos elementos de um conjunto que ocupa um espaço na memória do computador dedicado ao armazenamento de um tipo de dado determinado. Em um programa as variáveis devem ser declaradas, especificando seu tipo que pode ser inteiro, real ou caractere, devendo receber nomes para poderem ser referenciadas e alteradas quando necessário.

Cada variável corresponde a uma posição de memória, cujo conteúdo é alterado durante o processo e execução de um programa.

```
main()
int a; // a é a variável
a = 12/2; // o resultado será 6
a = 12 % 5; // o resultado será 2
```

### *Constantes*

Constante é um determinado valor que não se modifica ao longo da execução de um programa, podendo ser classificada como numérica, lógica ou literal conforme o tipo.

EXEMPLO.: Na expressão matemática para obtenção da nota final em determinada organização educacional temos:

$$N_f = \frac{((N_1 * 4) + (N_2 * 6))}{10}$$

Onde:

$N_f$  → nota final

$N_1$  → avaliação do 1º semestre

$N_2$  → avaliação do 2º semestre

(4 e 6) → **constantes** multiplicativas (que correspondem ao peso de cada uma das avaliações)

10 → **constante** (que corresponde a soma dos pesos das avaliações)

### *Tipos de variáveis e constantes*

Classificam-se as variáveis e as constantes por quatro tipos, numéricas, caracteres, alfanuméricas e lógicas.

- Numéricas: armazenamento numérico, que podem ou não representar números que serão utilizados em cálculos, podendo ser classificada como inteiras ou reais.
- Caracteres: utilizando na representação de conjuntos não numéricos. (por exemplo nomes)
- Alfanuméricos: Dados que contenham letras e ou números. Quando utilizando para armazenar números não poderão ser utilizados em cálculos.

- Lógicos: Baseado na lógica booleana só pode armazenar dados do tipo *true* (verdadeiro) 1 ou *false* (falso) 0.

Nota: observe que 0 “ou” 1 significa exclusivo.

### ***Bit e seu armazenamento***

O armazenamento da informação em um computador é feito no padrão binário, este representado por 0 e 1, os quais, não necessariamente representam valores numéricos mas podem representar símbolos, caracteres, imagens e sons, ou seja representam dados armazenados em um computador. Estes dados são gerados e administrados a partir da lógica booleana que consiste em uma matriz binária que é acessada (operações) de maneira semelhante à aritmética de produto e de soma.

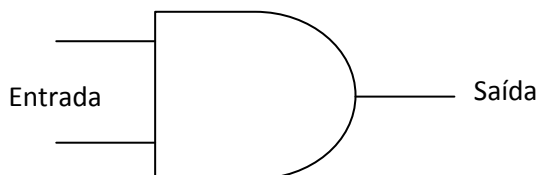
Nota: Observe que 0 “e” 1 significa inclusivo.

### ***Portas lógicas***

Baseada na lógica booleana de *true* (verdadeiro) e *false* (falso), 1 para verdadeiro e 0 para falso, estudamos os **operadores booleanos** que nos permite agir conceitualmente sobre os valores de *true* e *false* manipulando-os. É importante notar que todo sistema de processamento binário dos computadores é binário.

Os operadores AND (e), OR (ou) e XOR (ou exclusivo), apresentam semelhança em por operarem de forma semelhante a aritmética que toma dois valores para produzir um terceiro como resultado, enquanto o operador NOT (inversão) utiliza apenas uma entrada para produzir uma saída, que consiste no inverso da entrada. Os quatro operadores são representados por símbolos que facilitam o entendimento de projetos.

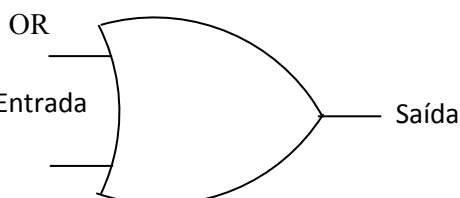
AND



AND de significa em português E, e da mesma maneira que no português, na lógica booleana faz a junção de idéias.

Essa porta lógica possui dois bits de entrada e um de saída. Para que o bit de saída seja verdadeiro (valor 1) ambos os bits de entrada devem ser verdadeiros.

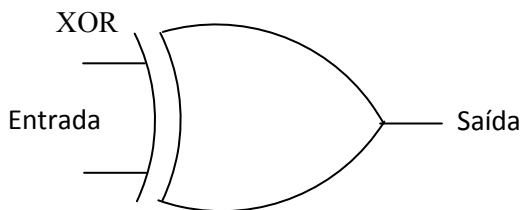
ENTRADAS		SAÍDA
0	0	0
0	1	0
1	0	0
1	1	1



OR de significado em português OU, e da mesma maneira que no português indica uma escolha.

Essa porta lógica possui dois bits de entrada e um de saída. Para que o bit de saída seja verdadeiro (valor 1) pelo menos um dos bits de entrada precisa ser verdadeiro.

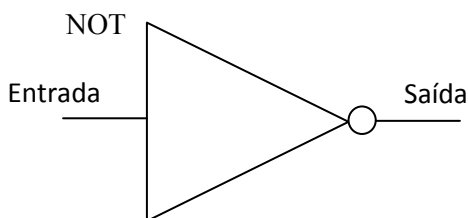
ENTRADAS		SAÍDA
0	0	0
0	1	1
1	0	1
1	1	1



XOR de significado em português OU exclusivo indica que há uma exigência de oposição.

A porta lógica XOR possui dois bits de entrada e um de saída. Para que o bit de saída seja verdadeiro (valor 1) os bits de entrada devem ser diferentes ou seja um verdadeiro outro falso.

ENTRADAS		SAÍDA
0	0	0
0	1	1
1	0	1
1	1	0



NOT de significado em português negação, e da mesma maneira que no português indica a negação a uma afirmação.

A porta lógica NOT é também conhecida como inversor por (ação de negação a uma determinada afirmação), literalmente, inverte o bit de entrada. Se o bit de entrada for um, por exemplo, o bit de saída será zero, e vice-versa.

ENTRADA	SAÍDA
0	1
1	0

### ***Circuito flip-flop***

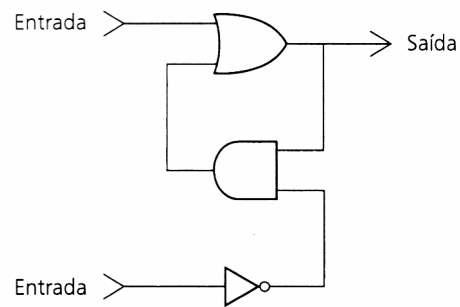
Corresponde ao circuito que apresenta um dos dois valores binários 0 e 1, permanecendo assim até que um pulso temporário em sua entrada modifique sua saída.



Tal procedimento permite aos operadores lógicos que sejam agrupados em blocos, conceito do qual é constituído o computador.

O valor da saída alterna entre dois valores, conforme os estímulos externos provenientes de outro circuito.

Exemplo:



Enquanto as duas entradas do circuito permanecerem com valores em 0, sua saída não se alterará:

No entanto se variarmos as entradas ocorrerá que poderemos obter a saída igual a 1 ou a 0 dependendo dos valores das entradas.

## TÓPICO 09

### ***Comandos para atribuição de valor***

Existem dois tipos de atribuição de valor um quando se impõe uma igualdade, ou seja:

```
int a; /* atribuição de tipo para a variável "a" */
```

Logo depois da declaração do tipo de variável é necessário que se indique qual o valor inicial, para evitar que algo que esteja carregado na memória seja carregado como “sujeira”.

```
a = 0;
```

Uma característica importante do comando de atribuição é que o mesmo deve ser expresso sempre da esquerda para direita.

Um erro clássico em programação é confundir o operador lógico de igualdade “= =” com o operador de atribuição “=”.

Exemplo:

```
int x = 0; // atribuição de valor para a variável x  
if (x==4); //Teste lógico if (se) que indica a verificação se x tem valor igual a 4  
  
if (x = 0); /* não convêm fazer uma atribuição em um teste lógico */
```

## TÓPICO 10

### *Operadores Matemáticos*

Também chamados de operadores aritméticos são os responsáveis pelo processo matemática a ser realizado pelo computador. Podendo ser separados em dois tipos:

**Binário:** quando utilizados em operações matemáticas de radiciação, exponenciação, divisão, multiplicação, adição e subtração.

**Unário:** quando atua na inversão do estado de um valor numérico, ou seja de positivo para negativos e de negativo para positivo.

A tabela a seguir apresenta os operadores matemáticos para uso em lógica descrevendo seu tipo.

Tabela de operadores aritméticos					
Operador	Operação	Descrição	Tipo	Prioridade	Resultado
+	"+" ou "n"	Manutenção de sinal	Unário	-	Positivo
-	-n	Inversão de sinal	Unário	-	Negativo
←	x ← n	Atribuição do valor "n" a "x"	Binário	-	Positivo ou Negativo
↑	x ↑ n	Exponenciação de x <sup>n</sup>	Unário	1	Inteiro ou Real
↑ (1 / n)	x ↑ (1 / n)	Radiciação de <sup>n</sup> √x	Unário	1	Real
/	x / n	Divisão de "x" por "n"	Binário	2	Real
*	x * n	Multiplicação de "x" por "n"	Binário	2	Inteiro ou Real
+	x + n	Adição de "x" com "n"	Binário	3	Inteiro ou Real
-	x - n	Subtração de "n" de "x"	Binário	3	Inteiro ou Real
div	x div n	Divisão de "x" por "n"	Binário	4	Inteiro

Algoritmos – lógica para desenvolvimento de programas de computadores - Manzano – Erica:2010 – pag 47

Para variar o nível de prioridade requerido ao um determinado operador é necessário o uso de parênteses, considerando-se regras de ordem para resolução matemática.

As cinco operações aritméticas suportadas pela linguagem C estão descritos abaixo com seus respectivos símbolos em português e no padrão ANSI

SIGNIFICADO	LINGUAGEM C	PORTUGOL	SIMB. NO PADRO ANSI
Adição	+	+	+
Subtração	-	-	-
Multiplicação	*	*	⊗
Divisão (resto real)	/	/	/
Divisão (resto inteiro)		div	div
Resto da divisão	%		

Exemplo:

Se escrevermos a função:

`a=11%4;` // a variável a irá conter 3 como resultado já que 4 é o resto da divisão de 11 por 4.

## Tópico 11

### *Operadores relacionais*

Os operadores relacionais definem a maneira como as proposições podem ser conectadas, podendo a proposição composta assumir um dos dois valores verdadeiro ou falso.

Tabela de operadores

Operador	Símb	Exemp	Resultado	
			Literal	Binário
Maior	>	$a > b$	V se a maior que b, senão F	1 se a maior que b, senão 0
Menor	<	$a < b$	V se a menor que b, senão F	1 se a menor que b, senão 0
Maior ou igual	>=	$a >= b$	V se a maior ou igual a b, senão F	1 se a maior ou igual a b, senão 0
Menor ou igual	<=	$a <= b$	V se a menor ou igual a b, senão F	1 se a menor ou igual a b, senão 0
Igual	=	$a = b$	V se a igual a b, senão F	1 se a igual a b, senão 0
Diferente	<>	$a <> b$	V se a diferentes de b, senão F	1 se a diferentes de b, senão 0

## TÓPICO 12

### **Operadores lógicos**

Também referenciados como operadores booleanos, os operadores lógicos são utilizados na programação quando se deseja realizar uma tomada de decisão com varias condições, estas condições exigem uma ordem de precedência que corresponde a:

1º negação	.não.
2º conjunção	.e.
3º disjunção inclusiva	.ou.
4º disjunção exclusiva	.xou.

#### **- Operador lógico de negação.**

Corresponde a operação lógica de negação de rejeição ou a contradição do todo ou de parte do todo.

O operador lógico de negação utiliza-se em Português estruturado dos comandos:

**se .não.** (<condição 1>) **então**

[ ação para condição não verdadeira]

**fim\_se**

### ***Operador lógico de conjunção.***

Corresponde a operação lógica entre duas ou mais condições (proposições), representado na teoria de conjuntos como a intersecção entre conjuntos.

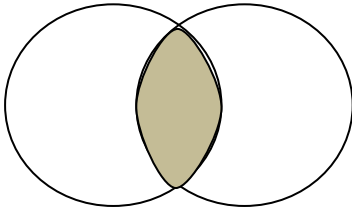


Diagrama de Venn para o operador de conjunção.

O operador de conjunção *.e.* corresponde a intersecção de dois conjuntos (condição 1 e 2)

O operador lógico de conjunção utiliza-se em Português estruturado dos comandos:

**se** (<condição 1>) **.e.** (<condição 2>) **então**

[ ação para condição 1 e condição 2 verdadeiras]

**fim\_se**

### ***Operador lógico de disjunção inclusiva.***

Da lógica clássica se diz que o operador lógico de disjunção inclusiva corresponde a relação entre duas proposição de tal modo que seu resultado lógico será verdadeiro quando pelo menos uma das duas proposições for verdadeira.

O operador lógico *.ou.* corresponde a operação entre duas ou mais condições (proposições), representado na teoria de conjuntos como a união entre conjuntos.

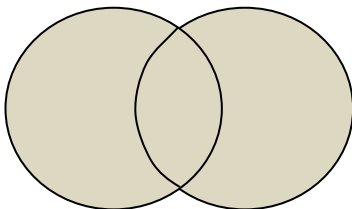


Diagrama de Venn para o operador de disjunção inclusiva.

O operador de disjunção inclusiva *.ou.* corresponde a união de dois conjuntos (condição 1 e 2)

O operador lógico de disjunção inclusiva utiliza-se em Português estruturado dos comandos:

**se** (<condição 1>) **.ou.** (<condição 2>) **então**

[ ação para condição 1 e/ou condição 2 verdadeiras]

**fim\_se**

### ***Operador lógico de disjunção exclusiva.***

Da lógica clássica se diz que o operador lógico de disjunção exclusiva corresponde a relação entre duas proposição de tal modo que seu resultado lógico será verdadeiro quando uma e apenas uma das duas proposições for verdadeira.

O operador lógico `.xou.` corresponde a operação entre duas condições (proposições), representado na teoria de conjuntos como a subtração da união de dois conjuntos pela intersecção dos mesmos conjuntos.

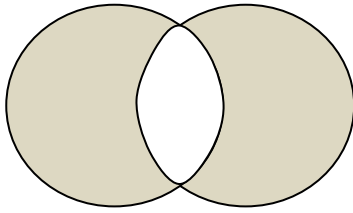


Diagrama de Venn para o operador de disjunção exclusiva.

O operador de disjunção exclusiva `.xou.` corresponde subtração da união de dois conjuntos por sua intersecção. (condição 1 e 2)

O operador lógico de disjunção inclusiva utiliza-se em Português estruturado dos comandos:

**se** (<condição 1>) **.xou.** (<condição 2>) **então**

[ ação para condição 1 verdadeira e condição 2 falsa ou condição 1 falsa e condição 2 verdadeira]

**fim\_se**

## Tabela verdade

### Proposição:

Chama-se proposição ou sentença toda oração declarativa que pode ser classificada em verdadeira ou em falsa.

Observa-se que toda proposição apresenta três características obrigatórias:

1° - sendo oração, tem sujeito e predicado;

2° - é declarativa (não é exclamativa nem interrogativa);

3° - tem um, e somente um, dos dois valores lógicos: verdadeiro (v) ou falso(f).

### Algumas Leis Fundamentais

**Lei do meio excluído:** Uma proposição é falsa (F) ou verdadeira (V).

**Lei da contradição:** Uma proposição não pode ser, simultaneamente V e F.

**Lei da Funcionalidade:** O valor ( V ou F) de uma proposição composta é unicamente determinada pelos valores lógicos de suas proposições constituintes.

Toda proposição deve ter um valor verdade (isto é, deve ser verdadeira ou falsa). Desta maneira, proposições podem ser combinadas através de operadores lógicos. Os operadores lógicos são representados da seguinte forma:

TIPO	OPERADOR	SÍMBOLO
Proposição de negação	Não é verdade	$\neg$ ou $\sim$
Conectivo de conjunção	E	$\wedge$
Conectivo de disjunção	Ou	$\vee$
Implicação	Se ... então	$\rightarrow$
Bi-condicional	Se e somente se	$\leftrightarrow$
Disjunção exclusiva	Ou exclusivo	$\underline{\vee}$

### Negação ~

A partir de uma proposição  $p$  qualquer, sempre será possível construir outra proposição, denominada negação de  $p$  e identificada pelo símbolo  $\sim p$ .

$p$	$\sim p$
V	F
F	V

A negação se classifica pelo seguinte postulado

A proposição  $\sim p$  tem sempre o valor oposto de  $p$ , isto é,  $\sim p$  é verdade quando  $p$  é falsa e  $\sim p$  é falsa quando  $p$  é verdadeira.

### Proposição composta – Conectivos

A partir de proposições dadas podemos construir novas proposições mediante a utilização de dois símbolos lógicos chamados conectivos:

$\wedge$  (**lê-se: e**), conectivo que quando utilizado entre duas proposições  $p \wedge q$ , denominada conjunção das sentenças.

$p$	$q$	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

A conjunção  $p \wedge q$  é verdadeira se  $p$  e  $q$  são ambas verdadeiras; se ao menos uma delas for falsa, então  $p \wedge q$  é falsa.

$\vee$  (**lê-se: ou**), conectivo que quando utilizado entre duas proposições  $p \vee q$ , denominada disjunção das sentenças.

$p$	$q$	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

A disjunção  $p \vee q$  é verdadeira se ao menos uma das proposições  $p$  ou  $q$  é verdadeira; se  $p$  e  $q$  são ambas falsas, então  $p \vee q$  é falsa.



$\underline{\vee}$  (**lê-se: ou exclusivo**) XOR, Disjunção exclusiva que quando utilizada entre duas proposições  $p \underline{\vee} q$ , onde

A conjunção  $p \underline{\vee} q$  é verdadeira se, e somente se, apenas um dos operandos for verdadeiro

p	q	$p \underline{\vee} q$
V	V	F
V	F	V
F	V	V
F	F	F

### Condicionais

$\rightarrow$  (**lê-se: se ... então ...**), condicional que quando utilizando entre duas proposições  $p \rightarrow q$ , onde p é denominado antecedente e q é conseqüente. (implicação)

O condicional  $p \rightarrow q$  é falso somente quando p é verdadeiro e q é falsa; caso contrário  $p \rightarrow q$  é verdadeiro.

P	Q	$p \rightarrow q$
V	V	V
V	F	F
F	V	V
F	F	V

P	q	$p \leftrightarrow q$
V	V	V
V	F	F
F	V	F
F	F	V

$\leftrightarrow$  (**lê-se: se, e somente se**), condicional (ou bi-condicional) que quando utilizando entre duas proposições  $p \leftrightarrow q$ , onde p é condição necessária e suficiente para q e reciprocamente. (Equivalência)

O condicional  $\leftrightarrow$  é verdadeiro somente quando p e q são ambas verdadeiras ou ambas falsas; se isso não acontecer, o condicional  $\leftrightarrow$  é falso.

### Adaga de Quine (NOR)

A conjunção é verdadeira se e somente se os operandos são falsos

A	B	$A \downarrow B$	$A \downarrow A$
V	V	F	F
V	F	F	F
F	V	F	F
F	F	V	V

### ***Tautologias ou proposição logicamente verdadeira.***

Seja uma proposição formada a partir de outras (p, q, r, ...) mediante o emprego de conectivos ( $\vee$  ou  $\wedge$ ) ou do modificador ( $\sim$ ) ou de condicionais ( $\rightarrow$  ou  $\leftrightarrow$ ). Dizemos que v é uma tautologia ou proposição logicamente verdadeira quando v tem o valor V (verdadeiro) independentemente dos valores lógicos de p, q, etc.

Assim a tabela-verdade de uma tautologia v apresenta só V na coluna de v.

#### **Exemplo**

1°  $(p \wedge \sim q) \rightarrow (q \vee p)$  é uma tautologia, pois:

p	q	$\sim p$	$p \wedge \sim q$	$q \vee p$	$(p \wedge \sim q) \rightarrow (q \vee p)$
V	V	F	F	V	V
V	F	F	F	V	V
F	V	V	F	V	V
F	F	V	F	F	V

### ***Proposições logicamente falsas***

Seja f uma proposição formada a partir de outras (p, q, r, ...) mediante o emprego de conectivos ( $\vee$  ou  $\wedge$ ) ou do modificador ( $\sim$ ) ou de condicionais ( $\rightarrow$  ou  $\leftrightarrow$ ). Dizemos que f é uma proposição logicamente falsa quando f tem o valor lógico F (falso) independentemente dos valores lógicos de p, q, etc.

Assim, a tabela-verdade de uma proposição logicamente falsa f apresenta só F na coluna de f.

#### **Exemplo**

$p \wedge \sim p$  é proposição logicamente falsa, pois:

P	$\sim p$	$p \wedge \sim p$
V	F	F
F	V	F

## Tópico 14

### **PROGRAMAÇÃO ESTRUTURADA OU PROGRAMAÇÃO MODULAR**

A técnica mais importante no projeto da lógica de programação em algoritmos denomina-se programação estruturada ou programação modular, estando em consonância com o pensamento, que é estruturado e serve como base e fundamentação para o uso e estudo de outras técnicas, como a técnica de programação orientada a objetos.

#### ***A técnica utiliza a seguinte metodologia:***

- Escrever instruções ligadas entre si apenas por estruturas seqüenciais, tomadas de decisão, laços de repetição e de selecionamento.
- Escrever instruções em grupos pequenos e combiná-las na forma de sub-rotinas ou de módulos estruturados ou orientados a objeto.
- Distribuir módulos do programa entre os diferentes programadores que trabalharão sob a supervisão de um programador sênior, chefe de programação ou analista de sistemas de informação.
- Revisar o trabalho executado em reuniões regulares e previamente programado.

#### ***Objetivando a:***

1. Agilizar a codificação da escrita da programação;
2. Facilitar a depuração da leitura;
3. Permite a verificação de possíveis falhas apresentadas pelos programas.

#### ***Procedimentos.***

A ordem ideal de procedimentos corresponde a:

- |   |   |
|---|---|
| 1º - Estabelecer a seqüência lógica do projeto;   | { Durante esta fase de projeto o engenheiro deverá ter em mente que em geral é necessário três processos. <ul style="list-style-type: none"><li>• Entrada ou leitura de dados.</li><li>• Processamento dos dados.</li><li>• Saídas dos dados.</li></ul> |
| 2º - Construir o diagrama de blocos (fluxograma) que represente o projeto;                | { Etapa em que se pode exigir retorno e reestruturação do processo anterior.  |
| 3º - Escrever em linguagem de programação ou em LPP.(linguagem de projeto de programação) | { Finalização, que exige rigor no atendimento do planejado anteriormente.   |

### **LINGUAGEM DE PROJETO DE PROGRAMAÇÃO - PSEUDOCÓDIGO**

Para facilitar o trabalho com português estruturado adotamos o padrão de linguagem de projeto de programação, com seus comandos descritos abaixo. Descendente da técnica PDL (Program Design Language).

O português estruturado ou LPP, por ser uma linguagem de comunicação humano-máquina hipotética, está formada com estruturas de verbos, substantivos, conjunções, interjeições, preposições adjetivos, com fins de associar-se a outras linguagens (LUA, PASCAL, C, C++, etc) estas de programação de alto nível.

**TABELA DE COMANDOS ADOTADOS EM LPP**

<b>LPP</b>	<b>CLASSIFICAÇÃO SINTÁTICA</b>
ATÉ	Preposição
ATÉ QUE	Conjunção (de acordo com seu equivalente em inglês- <i>until</i> )
ATÉ SEJA	Preposição com interjeição
CADEIA	Substantivo feminino
CARACTERE	Substantivo feminino
CASO	Substantivo masculino
CLASSE	Substantivo feminino
CONJUNTO	Adjetivo
CONST (constante)	Adjetivo
CONTINUA	Verbo (imperativo afirmativo)
DE	Preposição
EFETUE	Verbo (imperativo afirmativo)
ENQUANTO	Conjunção
ENQUANTO SEJA	Conjunção com verbo
ENTÃO	Advérbio
ESCREVA	Verbo (imperativo afirmativo)
FAÇA	Verbo (imperativo afirmativo)
FIM	Substantivo masculino
FIM ATÉ SEJA	Substantivo masculino com preposição e com interjeição)
FIM CASO	Substantivo masculino com substantivo masculino
FIM CLASSE	Substantivo masculino com substantivo feminino
FIM ENQUANTO	Substantivo masculino com conjunção
FIM FAÇA	Substantivo masculino com verbo
FIM LAÇO	Substantivo masculino com substantivo masculino
FIM PARA	Substantivo masculino com preposição
FIM REGISTRO	Substantivo masculino com substantivo masculino
FIM SE	Substantivo masculino com conjunção
FUNÇÃO	Substantivo masculino
HERANÇA	Substantivo feminino
INÍCIO	Substantivo masculino
INTEIRO	Adjetivo
LAÇO	Substantivo masculino
LEIA	Verbo (imperativo afirmativo)
LÓGICO	Adjetivo
OBJETO	Substantivo masculino
PARA	Preposição
PASSO	Substantivo masculino
PRIVADA	Substantivo feminino
PROCEDIMENTO	Substantivo masculino
PROGRAMA	Substantivo masculino
PROTEGIDA	Adjetivo
PÚBLICA	Adjetivo
REAL	Substantivo masculino
REGISTRO	Substantivo masculino
REPITA	Verbo (imperativo afirmativo)
SAIA CASO	Verbo imperativo afirmativo com substantivo masculino
SE	Conjunção
SEÇÃO PRIVADA	Substantivo feminino com substantivo masculino
SEÇÃO PROTEGIDA	Substantivo feminino com adjetivo
SEÇÃO PÚBLICA	Substantivo feminino com adjetivo
SEJA	Interjeição
SENÃO	Conjunção
TIPO	Substantivo masculino
VAR (variável)	Substantivo feminino
VIRTUAL	Adjetivo

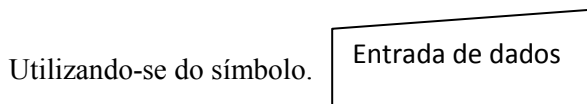
## Tópico 15 -17

### **Comandos de entrada e saída**

Os procedimentos de entrada e saída correspondem a execuções que ocorrem em quase todos os tipos de programa (normalmente os programas são divididos em três grades blocos “entrada de dados↔processamento/procedimentos↔saída de dados”).

Exemplo:

Uma entrada poderá ser através de teclado, sensor, modem, leitores ópticos, disco e etc.



Uma saída poderá ser através de vídeo ou formulário (impresso).



Sintaxe em portugol.

Entrada comando **leia**

Saída comando **escreva**

### **Estrutura de controles**

Controle corresponde a ação de tomada de decisão devida a uma condição, sendo que uma condição pode ser entendida como uma obrigação que se impõe e se aceita, enquanto decisão corresponde ao ato ou efeito de decidir.

As estruturas de controle são formadas por um cabeçalho seguido por um bloco de comandos que indique seu objetivo e delimitado por um comando que indique o fim\_<nome da estrutura>.

Tais condições correspondem a expressões booleanas cujo resultado é um valor lógico falso ou verdadeiro, assim o estabelecimento de uma condição, ou seja, uma relação lógica entre dois elementos é feita a partir de operadores relacionais.

#### **Características das estruturas se seleção:**

- Adota-se a sinalização das linhas de fluxo indicando a direção de processamento;
- Utiliza-se o símbolo de decisão *decision* com os rótulos S (sim) e N (não) independentemente a esquerda ou a direita respectivamente;
- Ao fim do processo de decisão utiliza-se o símbolo de conector *connector*;

## Tópico 18

### *Estrutura de seleção simples*

Corresponde a tomada de simples (desvio condicional), a principal característica de um bloco de decisão (seleção) simples corresponde ao fato de existir um bloco de operações somente se a condição for verdadeira.

Ao lado do rótulo S executam-se as instruções subordinadas para depois direcionar o fluxo ao conector.

Na linguagem de projeto (português estruturado) utiliza-se apresentar as condições entre parênteses (condição). Outra característica é o deslocamento de dois ou três espaços a direita a fim de evidenciar qual de fato é o bloco subordinado e uma determinada condição e quais são as instruções subordinadas.

A estrutura de seleção simples utiliza-se dos comandos **se.....então** e **fim\_se**, que na linguagem de programação C, corresponde aos comandos **if ... then**.

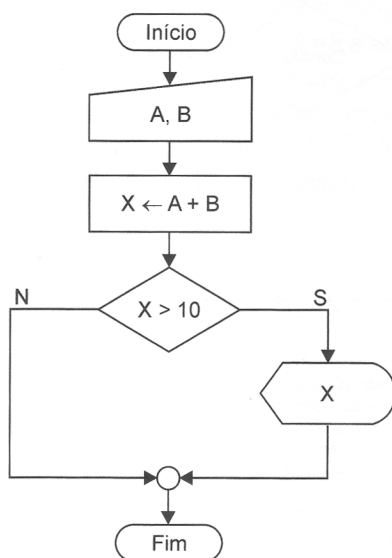
#### **Exemplo:**

Elaborar um programa de computador que leia dois valores numéricos reais desconhecidos. Em seguida o programa deve efetuar a soma entre os dois valores e caso o resultado seja maior que 10 apresentá-lo em vídeo.

#### **a) Seqüência lógica (entendimento)**

1. Definir a entrada de dois valores incógnitos (variáveis A e B), note que apesar do enunciado indicar a leitura de dois números, podemos entender que não são dados armazenados, pois são desconhecidos, assim devem ser inseridos.
2. Efetuar a soma dos valores A e B e atribuindo o resultado à variável X.
3. Apresentar o resultado armazenado na variável X, caso a variável X tenha seu valor maior que 10.

#### **b) Diagrama de blocos.**



#### **c) Código**

```
programa ADIÇÃO_DE_NÚMEROS_1
var
    A, B, X : real
início
    leia A, B
    X ← A + B
    se (X > 10) então
        escreva X
    fim_se
fim
```

## Tópico 19

### ***Estrutura de seleção composta.***

Corresponde a tomada de decisão composta, que semelhantemente a seleção simples refere-se à operação booleana de verdadeiro ou falso, no entanto a tomada de decisão composta (desvio condicional composto) corresponde a tomada de decisão que impõe ações (instruções subordinadas), tanto para a condição lógica S, quanto para a condição N, isso implica que sua principal característica é a existência de um bloco de operações para cada um dos lados da condição (decisão, *decision*).

Na estrutura de seleção composta utiliza-se dos comandos, **se ....então/senão/fim\_se.**, que na linguagem de programação C, corresponde a **if... then...else** .

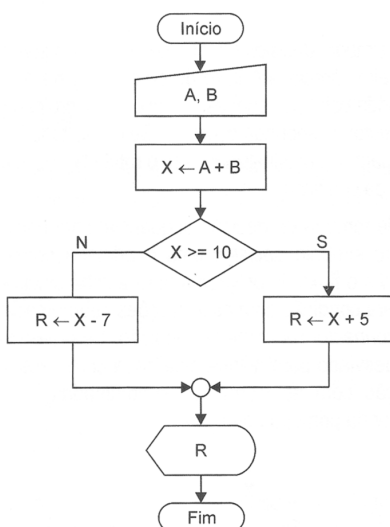
Exemplo:

Elaborar um programa que leia dois valores numéricos reais desconhecidos. Em seguida o programa deve efetuar a soma dos dois valores lidos e caso seja o resultado maior ou igual a 10, deve ser somado a 5. Caso contrário, o valor do resultado deve ser subtraído de 7. Após a obtenção de um dos novos resultados o novo resultado deve ser apresentado em vídeo.

#### **a) Seqüência lógica (Entendimento)**

1. Definir a entrada dos dois valores incógnitos (variáveis A e B), note que apesar do enunciado indicar a leitura de dois números, podemos entender que não são dados armazenados, pois são desconhecidos, assim devem ser inseridos.
2. Efetuar a adição dos valores A e B e atribuir o resultado da adição à variável X.
3. Verificar se o valor da variável X é maior ou igual a 10; caso seja maior ou igual a 10, proceder ao cálculo de  $X + 5$ , atribuindo seu resultado a variável R. Se o valor da variável X não for maior ou igual a 10, proceder ao cálculo de  $X - 7$ , atribuindo seu resultado a variável R.

#### **b) Diagrama de blocos**



#### **c) Código**

```
programa ADIÇÃO_DE_NÚMEROS_2
```

```
var
```

```
    A, B, X, R : real
```

```
início
```

```
    leia A, B
```

```
    X ← A + B
```

```
    se (X >= 10) então
```

```
        R ← X + 5
```

```
    senão
```

```
        R ← X - 7
```

```
    fim_se
```

```
    escreva R
```

```
fim
```