

2. ESTRUTURA DE UM ALGORITMO

Na criação de algoritmos, utilizam-se os conceitos de bloco lógico, entrada e saída de dados, variáveis, constantes, atribuições, expressões, relacionais e aritméticas, bem como comandos que traduzem estes conceitos de forma a representar o conjunto de ações.

Para que esse conjunto de ações se torne viável, deve existir uma perfeita relação lógica ligada ao modo pelo qual essas ações são executadas, pelo qual é regido o fluxo de execução. Através das estruturas básicas de controle de fluxo de execução-seqüenciação, seleção, repetição e da combinação delas, pode-se criar um algoritmo para solucionar qualquer problema.

2.1 ALGORITMOS ESTRUTURADOS

Muitos programadores preparam um programa iniciando com um diagrama de blocos para demonstrar sua linha de raciocínio, tendo como objetivo estabelecer uma seqüência de operações a serem efetuadas em um programa.

Esta técnica permite posteriormente uma codificação, praticamente em qualquer linguagem de programação.

A técnica mais importante no projeto de lógica de programa é chamada de programação estruturada, a qual consiste em uma metodologia de projeto, objetivando:

1. Agilizar a codificação da escrita da programação;
2. Permite a verificação de possíveis falhas apresentadas pelos programas;
3. Facilitar as alterações e atualizações dos programas.

A programação estruturada deve ser composta de quatro passos fundamentais:


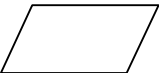


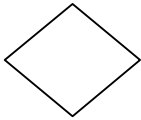
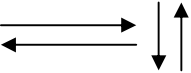

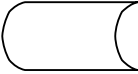
1. Escrever as instruções em seqüências ligadas entre si apenas por estruturas seqüenciais, respectivas ou de selecionamento.
2. Escrever instruções em grupos pequenos e combiná-los.
3. Distribuir módulos do programa entre os diferentes programadores que trabalharão sob a supervisão de um programador sênior, ou chefe de programação.
4. Revisar o trabalho executado em reuniões regulares e previamente programadas, em que compareçam apenas programadores de um mesmo nível.

2.2 REPRESENTAÇÃO DE ALGORITMOS

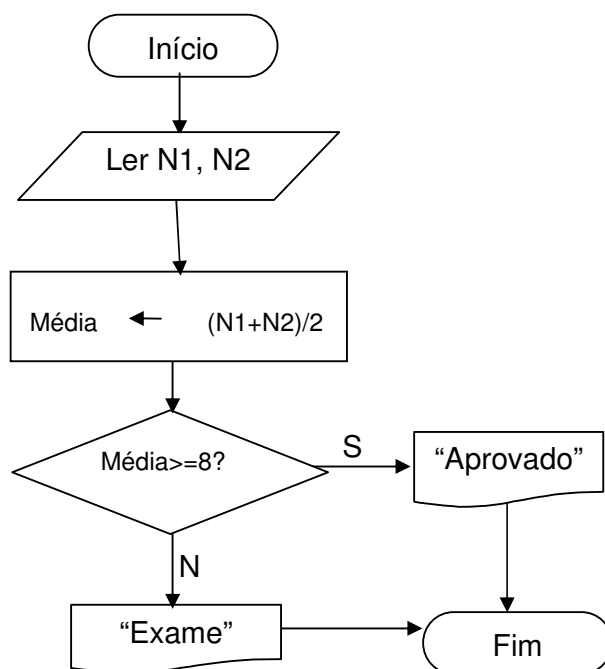
É comum os profissionais de desenvolvimento da área de TI denominarem símbolos que representam as linhas do raciocínio lógico. A representação desses símbolos se dá através de quatro formas, a saber:

2.2.1 FLUXOGRAMA

É uma ferramenta usada pelo profissional de sistemas. Tem como finalidade descrever o fluxo, seja manual ou mecânico, especificando os suportes usados para os dados e as informações. Usa símbolos convencionais, permitindo poucas variações. Representando alguns dados, do processamento de dados e da saída de dados, acompanhados dos procedimentos requeridos pelo analista de sistemas, e a serem realizados pelo programador através do desenvolvimento do raciocínio lógico, o qual deverá solucionar o problema do programa a ser processado pelo computador.

Simbologias Básicas	
	= Início e final do fluxograma
	= Entrada e saída de dados, representa um dispositivo qualquer
	= Operação de saída de dados, impressora
	= Operações de atribuição e
	= Decisão
	= Direção do fluxo de dados ou de processamento
	= Conector, utilizado quando é preciso particionar o diagrama. Quando houver mais de uma partição, é colocado uma letra ou um número dentro do símbolo de conexão para identificar os pares de ligação.
	= Saída de Dados em vídeo

Exemplo:



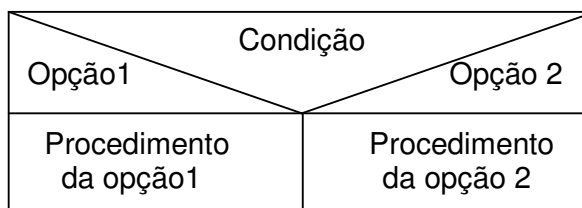
2.2.2 Diagrama de Chapin

É uma forma de especificação gráfica que permite o desenvolvimento de algoritmos estruturados.

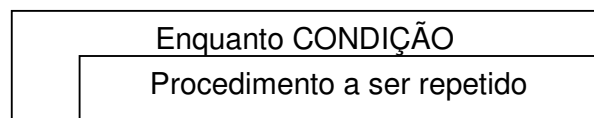
Simbologia básica:

Comando 1
Comando 2
Comando 3

= Seqüência

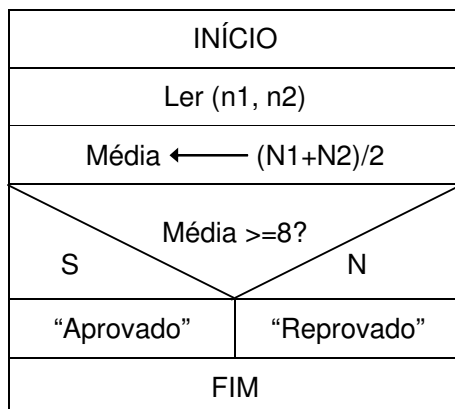


= Seleção



= Repetição

Exemplo:



2.2.3 Português Estruturado

Tendo estabelecido os passos anteriores (fluxograma, diagrama de Chapin), será efetuada a codificação. Esta fase obedece ao que está definido no fluxograma ou no diagrama, pois ela é a representação gráfica da lógica de um programa, porém sempre deverá ser relacionado com todas as variáveis que serão utilizadas dentro do programa. Este relacionamento, além de definir tipos de dados que serão utilizados, define também o espaço de memória que será necessário para manipular as informações fornecidas durante a execução do programa.

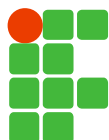
O português estruturado é uma linguagem de programação de computadores baseada na categoria pseudocódigo. Assim sendo, o computador que vai utilizar essa linguagem de programação é imaginário, ou seja, a própria mente do desenvolvedor.

Programa nome_do_programa
Var declaração de variáveis
Subalgoritmos
Início
 Corpo do algoritmo
Fim

Português estruturado é formado pelas instruções: **programa, var, inteiro, real, caractere, lógico, início, leia, escreva, fim, se, então, senão, fim-se, enquanto, faça, fim-enquanto, repita, até-que, para, de, até, passo, fim-para, conjunto, tipo, registro, fim-registro, procedimento, caso, seja, fim-caso e função**, além de operadores aritméticos (adição, subtração, multiplicação, divisão, exponenciação e atribuição), operadores relacionais (igual, diferente, maior, menor, maior ou igual, menor ou igual) e operadores lógicos (**.e.**, **.ou.**, **.não.**).

O conceito de comando advém do fato de se passar uma ordem a ser realizada pelo computador. Assim sendo, um comando pode ser escrito em uma ou mais linhas e ser formado por uma ou mais instruções. Por exemplo, o programa representado abaixo possui:

- Na primeira linha de código ocorre a definição de um comando representado por uma instrução escrita em apenas uma linha: **Programa** Calcula_Media.
- Da segunda até a quarta linha de código está a definição de um único comando escrito em três linhas, a partir da segunda linha, representado pela instrução **VAR** e seguido das demais linhas, respectivamente, com as instruções **caractere** e **real**.



- Da quinta até a oitava linha e da décima terceira até a vigésima linha de código ocorre o uso de um comando por linha.
- A sétima e oitava linhas não possuem instruções escritas, mas possuem a definição de expressões matemáticas. A décima e décima primeira linhas também não possuem instruções escritas, mas têm a definição de expressões lógicas usadas para atribuir um determinado valor à variável RESULTADO.
- Da nona até a décima segunda linha acontece a definição de um único comando escrito em 4 linhas.

Exemplo:

```
1  Programa Calcula_Media
2  Var
3      RESULTADO: caractere
4      N1, N2, N3, N4, SOMA, MEDIA: real
5  Início
6      Leia (N1, N2, N3, N4)
7      SOMA ← N1 + N2 + N3 + N4
8      MEDIA ← SOMA / 4
9      Se (MÉDIA >= 7)
10         Então RESULTADO ← "Aprovado"
11         Senão RESULTADO ← "Reprovado"
12  Fim-Se
13  Escreva ("Nota 1: ", N1)
14  Escreva ("Nota 2: ", N2)
15  Escreva ("Nota 3: ", N3)
16  Escreva ("Nota 4: ", N4)
17  Escreva ("Soma: ", SOMA)
18  Escreva ("Média: ", MEDIA)
19  Escreva ("Resultado: ", RESULTADO)
20 Fim
```

3. TIPOS PRIMITIVOS DE DADOS

Antes de iniciar o estudo, é necessário considerar que um computador nada mais é do que uma ferramenta utilizada para solucionar problemas que envolvam a manipulação de informações, as quais se classificam, a grosso modo, em dois tipos básicos: **dados** e **instruções**.

Os **dados** são representados por elementos advindos do mundo externo, os quais representam as informações que os seres humanos manipulam. Eles devem ser abstraídos para serem processados em um computador. Os dados podem ser categorizados em três tipos: numéricos, caracteres e lógicos. Os tipos de dados numérico inteiro, numérico real, caracteres e lógico são também referenciados como **tipos de dados primitivos** ou **tipos de dados básicos**.

3.1. INTEIROS

Toda informação numérica que pertença ao conjunto dos números inteiros relativos (negativos, nulo, positivos), excluindo-se destes qualquer número fracionário.

Exemplos:

- a) Ele tem 15 irmãos.
- b) Ela tem 20 anos de idade.
- c) -9.

3.2. REAIS

Toda a informação numérica que pertença ao conjunto dos números reais (negativos, positivos, nulo).

Exemplos:

- a) Ela tem 1,73 metro de altura.
- b) Meu saldo bancário é de R\$ 125,07.

3.3. CARACTERES

Toda a informação composta por um único caracter alfanumérico (0..9 / A..Z / a..z) e/ou caracteres especiais (#, /, \$, %, *, ?, ~, >, !, @, ...). OBS.: o espaço é considerado caracter especial.

Exemplos:

- a) 0 1 2 3 4 5 6 7 8 9
- b) A B C D E F ...

3.4. STRING

Toda a informação composta por um conjunto de caracteres (cadeia) alfanuméricos (0..9 / A..Z / a..z) e/ou caracteres especiais (#, /, \$, %, *, ?, ~, >, !, @, ...). OBS.: o espaço é considerado caracter especial.

Exemplos:

- c) Constava na prova: "Use somente caneta".
- d) O endereço é: "Rua 24 de Outubro, número 1215 – Apto 101 – POA/RS".
- e) O nome do escritor é: "Mário Quintana".

3.5. LÓGICOS

Toda a informação que pode apenas assumir duas situações (biestável, ou seja, dois estados).

Exemplos:

- a) Verdadeiro ou Falso.
- b) Aberto ou Fechado.
- c) A porta pode estar aberta ou fechada.
- d) A lâmpada pode estar acesa ou apagada.

4. TIPO DE INFORMAÇÕES

As informações tratadas em um algoritmo podem ser classificadas em dois tipos:

4.1. CONSTANTES

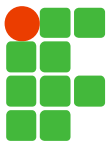
São informações que não sofrem alterações, que não muda com o decorrer do tempo. Dentro do algoritmo, constantes são valores que permanecem os mesmos do início ao fim. As informações constantes do tipo CARACTERE devem ser representadas entre aspas.

Exemplos:

- a) 'Não Fume'
- b) 5
- c) Falso
- d) -71

4.2. VARIÁVEIS

Informações que têm a possibilidade de serem alteradas em algum instante no decorrer do tempo. Tem-se como definição de variável tudo aquilo que está sujeito a variações, que é incerto, instável ou inconstante. E quando se fala de computadores, é preciso ter em mente que o volume de



informações a serem tratadas é grande e diversificado. Desta forma, os dados a serem processados serão bastante variáveis.

Todo dado a ser armazenado na memória de um computador deve ser previamente identificado, ou seja, primeiro é necessário saber o seu tipo para depois fazer o seu armazenamento adequado. Armazenado o dado, ele pode ser utilizado e manipulado a qualquer momento.

Exemplos:

- a) Temperatura
- b) Índice da inflação
- c) Alíquotas do imposto
- d) Comprimento
- e) Altura
- f) Peso

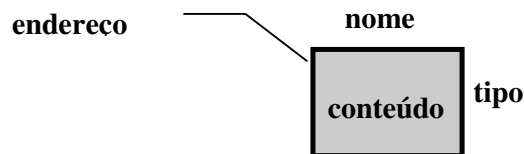
4.2.1 As variáveis de memória

No ambiente computacional, as informações variáveis são guardadas em dispositivos eletrônicos analogamente chamados de “memória”. Podemos imaginar essa “memória” como sendo um armário cheio de gavetas, no qual as gavetas seriam os locais físicos responsáveis por armazenar objetos; os objetos (que podem ser substituídos) seriam as informações e as gavetas, as variáveis.

Visto que, na memória (armário) existem inúmeras variáveis (gavetas), precisamos diferenciá-las, o que é feito por meio de identificadores (etiquetas). Cada variável (gaveta), no entanto, pode guardar apenas uma informação (objeto) de cada vez, sendo sempre de mesmo tipo (material). Portanto, precisamos definir nomes para determinadas gavetas especificando qual o material dos objetos que lá podem ser armazenados.

Variáveis são posições de memória com as seguintes características:

- Tem um nome, que a diferencia das demais;
- Tem um tipo de dado associado, que indica o tipo de informação que poderá ser armazenada na variável;
- Tem um conteúdo, que é o dado guardado na variável;
- Tem um endereço, que a localiza na memória.



Observações:

1. Nome de uma variável é único em um algoritmo e deve seguir as regras de formação de identificadores;
2. Uma variável somente pode receber como conteúdo um dado do tipo que foi definido para ela;
3. O conteúdo de uma variável é **SUBSTITUÍDO** por outro conteúdo que venha a ser colocado na variável;
4. O nome e o tipo de dado de uma variável, uma vez definidos, não mudam por todo o algoritmo;
5. O uso do nome de uma variável em uma expressão significa o uso do seu conteúdo (naquele momento) dentro da expressão;
6. O uso de um conteúdo de variável em uma expressão não modifica o seu valor.

4.3 Regras para Formação de Identificadores

Identificadores são quaisquer nomes que sejam usados para identificar informações variáveis ou constantes dentro de um algoritmo. A criação de um nome deve obedecer algumas regras de utilização das mesmas:

1. Devem começar com um caracter alfabético;
2. Podem ser seguidas por mais caracteres alfabéticos e/ou numéricos;
3. Não é permitido o uso de outros caracteres especiais;
4. Não poderá ser nome de uma variável, uma palavra reservada a uma instrução do programa;
5. O nome de uma variável não poderá possuir espaços em branco;
6. Não poderão ser utilizados outros caracteres a não ser letras e números, exceto o uso do caracter especial 'sublinha' (-), e/ou underline;

Exemplos:

1. Identificadores **NÃO** permitidos: _XPTO, 1ABC3, EF*GH, 5X, NOTA/2, String, REAL.
2. Identificadores permitidos: X, NOME_VAR, A12B, MEDIA.