4 Introdução ao Algoritmo

4.1 Conceito de Algoritmo

A automação é o processo em que uma tarefa deixa de ser desempenhada pelo homem e passa a ser realizada por máquinas, sejam estes dispositivos mecânicos, eletrônicos (como os computadores) ou de natureza mista.

Para que a automação de uma tarefa seja bem sucedida é necessário que a máquina que passará a realizá-la seja capaz de desempenhar cada uma das etapas constituintes do processo a ser automatizado com eficiência, de modo a garantir a repetibilidade do mesmo. Assim, é necessário que seja especificado com clareza e exatidão o que deve ser realizado em cada uma das fases do processo a ser automatizado, bem como a seqüência em que estas fases devem ser realizadas.

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

Ao contrário do que se pode pensar, o conceito de algoritmo não foi criado para satisfazer às necessidades da computação. Pelo contrário, a programação de computadores é apenas um dos campos de aplicação dos algoritmos. Na verdade, há inúmeros casos que podem exemplificar o uso (involuntário ou não) de algoritmos para a padronização do exercício de tarefas rotineiras (vide exemplos da Seção 2.1). No entanto, daqui adiante a atenção desta apostila estará voltada à automação de tarefas utilizando computadores.

Para que um computador possa desempenhar uma tarefa é necessário que esta seja detalhada passo-a-passo, numa forma compreensível pela máquina, utilizando aquilo que se chama de programa. Neste sentido, um programa de computador nada mais é que um algoritmo escrito numa forma compreensível pelo computador (linguagem de programação).

4.1.1 Algumas Definições de Algoritmo

"Serve como modelo para programas, pois sua linguagem é intermediária à linguagem humana e às linguagens de programação, sendo então, uma boa ferramenta na validação da lógica de tarefas a serem automatizadas."

"Os algoritmos, servem para representar a solução de qualquer problema, mas no caso do Processamento de Dados, eles devem seguir as regras básicas de programação para que sejam compatíveis com as linguagens de programação."

4.2 Formas de Representação de Algoritmos

Existem diversas formas de representação de algoritmos, mas não há um consenso com relação à melhor delas.

O critério usado para classificar hierarquicamente estas formas está diretamente ligado ao nível de detalhe ou, inversamente, ao grau de abstração oferecido.

Algumas formas de representação de algoritmos tratam os problemas apenas em nível lógico, abstraindo-se de detalhes de implementação muitas vezes relacionados com alguma linguagem de programação específica. Por outro lado existem formas de representação de algoritmos que possuem uma maior riqueza de detalhes e muitas vezes acabam por obscurecer as idéias principais do algoritmo, dificultando seu entendimento.

Dentre as formas de representação de algoritmos mais conhecidas podemos citar:

- Descrição Narrativa;
- Fluxograma Convencional;
- Pseudocódigo, também conhecido como Português Estruturado ou Portugol.

4.2.1 Descrição Narrativa

Nesta forma de representação os algoritmos são expressos diretamente em linguagem natural. Como exemplo, têm-se os algoritmos seguintes:

- Receita de bolo:

  Misture os ingredientes
  Unte a forma com manteiga
  Despeje a mistura na forma
  Se houver coco ralado
  então despeje sobre a mistura
  Leve a forma ao forno
  Enquanto não corar
  deixe a forma no forno
  Retire do forno
  Deixe esfriar

- Troca de um pneu furado:

  Afrouxar ligeiramente as porcas
  Suspender o carro
  Retirar as porcas e o pneu
  Colocar o pneu reserva
  Apertar as porcas
  Abaixar o carro
  Dar o aperto final nas porcas

- Tomando um banho:

  Entrar no banheiro
  Tirar a roupa
  Abrir a torneira do chuveiro
  Entrar na água
  Ensaboar-se
  Sair da água
  Fechar a torneira
  Enxugar-se Vestir-se

- Cálculo da média de um aluno:

  Obter as suas 2 notas de provas
  Calcular a média aritmética
  Se a média for maior que 7,
  o aluno foi aprovado,
  senão ele foi reprovado

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

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

4.2.2 Fluxograma Convencional

É uma representação gráfica de algoritmos onde formas geométricas diferentes implicam ações (instruções, comandos) distintos. Tal propriedade facilita o entendimento das

idéias contidas nos algoritmos e justifica sua popularidade.

Esta forma é aproximadamente intermediária à descrição narrativa e ao pseudocódigo (subitem seguinte), pois é menos imprecisa que a primeira e, no entanto, não se preocupa com detalhes de implementação do programa, como o tipo das variáveis usadas.

Nota-se que os fluxogramas convencionais preocupam-se com detalhes de nível físico da implementação do algoritmo. Por exemplo, figuras geométricas diferentes são adotadas para representar operações de saída de dados realizadas em dispositivos distintos, como uma fita magnética ou um monitor de vídeo. Como esta apostila não está interessada em detalhes físicos da implementação, mas tão somente com o nível lógico das instruções do algoritmo, será adotada a notação simplificada da Fig. 28 para os fluxogramas. De qualquer modo, o Apêndice A contém uma tabela com os símbolos mais comuns nos fluxogramas convencionais.

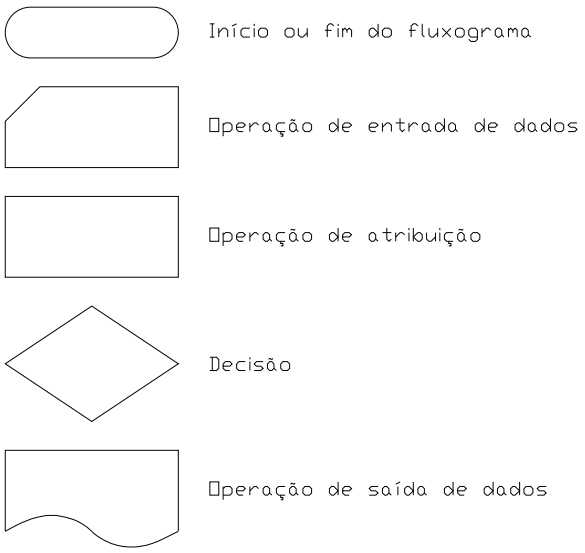

Figura 28: Principais formas geométricas usadas em fluxogramas.

De modo geral, um fluxograma se resume a um único símbolo inicial por onde a execução do algoritmo começa, e um ou mais símbolos finais, que são pontos onde a execução do algoritmo se encerra. Partindo do símbolo inicial, há sempre um único caminho orientado a ser seguido, representando a existência de uma única seqüência de execução das instruções. Isto pode ser melhor visualizado pelo fato de que, apesar de vários caminhos poderem convergir para uma mesma figura do diagrama, há sempre um único caminho saindo desta. Exceções a esta regra são os símbolos finais, dos quais não há nenhum fluxo saindo, e os símbolos de decisão, de onde pode haver mais de um caminho

de saída (usualmente dois caminhos), representando uma bifurcação no fluxo. A Figura 29 mostra a representação do algoritmo de cálculo da média de um aluno sob a forma de um fluxograma.

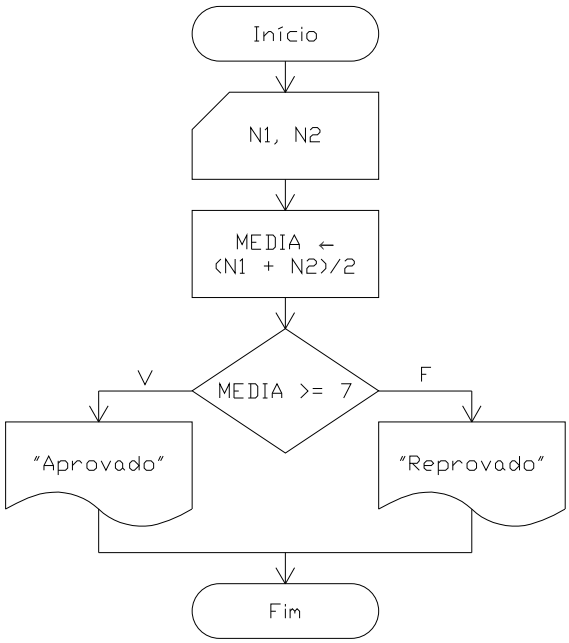

Figura 29: Exemplo de um fluxograma convencional.

4.2.3 Português Estruturado ou Pseudocódigo

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

Na verdade, esta representação é suficientemente geral para permitir a tradução de um algoritmo nela representado para uma linguagem de programação específica seja praticamente direta.

A forma geral da representação de um algoritmo na forma de pseudocódigo é a seguinte:

```
Programa <nome_do_programa>;
{1. Denominação do programa}
   Variáveis
   {2. Declaração e classificação de variáveis}
   Constantes
   {3. Declaração de constantes}
Início {4. Início do bloco principal}
{5. Iniciar as variáveis}
{6. Solicitar a entrada de dados ao usuário}
{7.Entrada de dados}
{8. Processamento/Cálculos}
{9. Saída de informações}
Fim {10. Final do bloco principal}
```

Algoritmo é uma palavra que indica o início da definição de um algoritmo em forma de pseudocódigo.

nome do programa é um nome simbólico dado ao algoritmo com a finalidade de distinguí-los dos demais;

Variáveis consiste em uma porção opcional onde são declaradas as variáveis globais usadas no algoritmo principal;

Início e Fim são respectivamente as palavras que delimitam o início e o término do conjunto de instruções do corpo do algoritmo.

O algoritmo do cálculo da média de um aluno, na forma de um pseudocódigo, fica da seguinte forma:

```
Programa Calculo_Media
  N1, N2, MEDIA: Real;
Início
  Leia(N1, N2);
  MEDIA ← (N1 + N2)/2;
  Se(MEDIA >= 7) então
    Escreva("Aprovado");
  Senão
    Escreva("Reprovado");
  Fim Se
Fim
```

4.3 Tipos de Dados

Todo o trabalho realizado por um computador é baseado na manipulação das informações contidas em sua memória. Grosso modo, estas informações podem ser classificadas em dois tipos:

- As instruções, que comandam o funcionamento da máquina e determinam a maneira como devem ser tratados os dados. As instruções são específicas para cada modelo de computador, pois são funções do tipo particular de processador utilizado em sua implementação;
- Os dados propriamente ditos, que correspondem à porção das informações a serem processadas pelo computador.

O objetivo deste capítulo é justamente o de classificar os dados de acordo com o tipo de informação contida neles. A classificação apresentada não se aplica a nenhuma linguagem de programação específica; pelo contrário, ela sintetiza os padrões utilizados na maioria das linguagens.

4.3.1 Dados Numéricos

Antes de apresentar formalmente os tipos de dados numéricos, é conveniente recordar alguns conceitos básicos relacionados à teoria dos números e conjuntos.

O conjunto dos números naturais é representado por N e é dado por:

$$N = 0, 1, 2, 3, 4, \cdots \tag{4.1}$$

Algumas correntes de matemáticos teóricos convencionam que o número 0 está contido neste conjunto; contudo, não convém perder tempo em tais discussões filosóficas, uma vez que isto não influenciará de forma alguma este estudo.

Na seqüência, encontramos o conjunto dos números inteiros:

$$Z = \cdots, -3, -2, -1, 0, 1, 2, 3, \cdots \tag{4.2}$$

O conjunto Z contém todos os elementos de N, bem como alguns números que não pertencem a N (os números negativos e o zero). Portanto, dizemos que N está contido em Z.

Englobando o conjunto dos números inteiros, existe o conjunto dos números fracionários (Q), dado pelo universo dos números que podem ser expressos na forma de uma fração, isto é, um quociente onde o numerador e o denominador são números inteiros. Mais formalmente:

$$Q = p/q \mid (p, q) \in Z \tag{4.3}$$

Por último, surge o conjunto dos números reais (R), formado pela união do conjunto dos números fracionários Q com o conjunto dos números que não podem ser expressos na forma de uma fração (os números irracionais). Ex.: $\sqrt{2} = 1,41421356\ldots$, $\pi = 3,14159265\ldots$.

4.3.1.1 Dados numéricos inteiros

Os números inteiros são aqueles que não possuem componentes decimais ou fracionários, podendo ser positivos ou negativos.

Os elementos pertencentes aos conjuntos N e Z, apesar de serem representáveis na classe dos números reais, são classificados como dados do tipo inteiro, por não possuírem parte fracionária. Esta possibilidade é interessante por permitir uma economia do espaço de memória, como veremos adiante.

Por sua vez, os elementos dos conjuntos Q e R, por possuírem parte fracionária, não podem ser representados na classe inteira, pertencendo necessariamente aos tipos de dados ditos reais.

Tabela 3: Exemplos de números inteiros.

Exemplo	**Descrição**
13	Número inteiro positivo
0	Número inteiro
-397	Número inteiro negativo

4.3.1.2 Dados numéricos reais

Os dados de tipo real são aqueles que podem possuir componentes decimais ou fracionários, e podem também ser positivos ou negativos.

Como dito anteriormente, os elementos dos conjuntos de números fracionários e reais são necessariamente representados no computador por dados do tipo real.

Observe que há uma diferença entre "0", que é um dado do tipo inteiro "0.0" (ou 0.)

Tabela 4: Exemplos de números reais.

Exemplo	Descrição
13.01	Número real positivo com duas casas decimais
144	Número real positivo
-1003.3	Número real negativo com uma casa decimal
0.0	Número real com uma casa decimal

que é um dado do tipo real. Portanto, a simples existência do ponto decimal serve para diferenciar um dado numérico do tipo inteiro de um do tipo real.

4.3.2 Dados Literais (Textos ou Letras)

O tipo de dados literal é constituído por uma seqüência de caracteres contendo letras, dígitos e/ou símbolos especiais. Este tipo de dados é também muitas vezes chamado de alfanumérico, cadeia (ou cordão) de caracteres, *string*.

Usualmente, os dados literais são representados nos algoritmos pela coleção de caracteres, delimitada em seu início e término com o caractere aspas ("*texto*").

Tabela 5: Exemplos de dados do tipo literal.

Exemplo	Descrição
`"QUAL ?"`	Literal de comprimento 6
`" "`	Literal de comprimento 1
`"qUaL ?!$"`	Literal de comprimento 8
`" AbCdefGHi"`	Literal de comprimento 9
`"1-2+3="`	Literal de comprimento 6
`"0"`	Literal de comprimento 1

Note que, por exemplo, "1.2" representa um dado do tipo literal de comprimento 3, constituído pelos caracteres '1', '.' e '2', diferindo de 1.2 que é um dado do tipo real.

4.3.3 Dados Lógicos

A existência deste tipo de dado é, de certo modo, um reflexo da maneira como os computadores funcionam. Muitas vezes, estes tipos de dados são chamados de booleanos, devido à significativa contribuição de **Boole** à área da lógica matemática.

O tipo de dados lógico é usado para representar dois únicos valores lógicos possíveis: verdadeiro e falso. É comum encontrar-se em outras referências outros tipos de pares de

valores lógicos como `sim/não`, `1/0` ou `true/false`.

Nos algoritmos apresentados nesta apostila os valores lógicos serão delimitados pelo caractere ponto (.).

Tabela 6: Exemplos de dados do tipo lógico.

Exemplo	**Descrição**
V	Valor lógico (verdadeiro)
F	Valor lógico (falso)

4.3.4 Exercícios Propostos

Questão 01: Classifique os dados especificados abaixo de acordo com seu tipo, assinalando com 'I' os dados do tipo inteiro, com 'R' os reais, com 'L' os literais, com 'B' os lógicos (booleanos), e com 'N' aqueles para os quais não é possível definir a priori um tipo de dado.

( ) 0.21	( ) T	( ) V	( ) 0,35
( ) 1	( ) +3257	( ) .V	( ) `"F"`
( ) W	( ) `"a"`	( ) `"abc"`	( ) +36
( ) `"0."`	( ) `"+3257"`	( ) F	( ) q
( ) 1%	( ) +3257.	( ) C	( ) ±3
( ) `"José"`	( ) `"-0.0"`	( ) Maria	( ) -0.001