

Lógica de Programação e Algoritmos

Profa. Eliane Oliveira Santiago

Lógica de Programação

É uma técnica de desenvolver sequências lógicas para atingir um determinado objetivo.

Plano de Ensino: conteúdo programático (1)

Módulo 1:

- Conceitos Básicos
- Conceito de Algoritmo
- Método para a construção de algoritmos
- Tipos de Algoritmos
 - ▣ o Descrição Narrativa
 - ▣ o Fluxograma
 - ▣ o Pseudocódigo ou português
- • Exemplos de Algoritmos.

Módulo 2:

- Conceitos de Variáveis
- Tipos de Dados (Numérico, Lógico, Literal ou caractere)
- Formação de Identificadores. Exemplos de Identificadores

Processamento de Dados



Algoritmo

Um processo que reúne um conjunto de ações que são necessárias para tratar os dados de entrada e transformá-los em resultados para um determinado objetivo.

Exemplo de Algoritmo

Receita de Bolo

Entrada

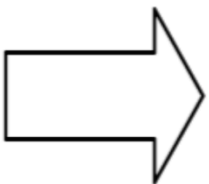


Processamento

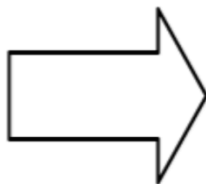


Saída

Lista de
Ingredientes



Misturar todos
os ingredientes



Bolo

Método para desenvolver um bom algoritmo

- ❑ ler e compreender o problema para o qual será construído um algoritmo;
- ❑ determinar quais serão a entrada de dados do seu algoritmo;
- ❑ determinar quais as ações, lógicas e/ou aritméticas, que deverão ser realizadas no seu algoritmo, bem como, as restrições, se houver, para cada uma dessas ações;
- ❑ determinar qual será a saída de resultados do seu algoritmo;
- ❑ construir o algoritmo em um dos tipos de algoritmo descritos na próxima seção;
- ❑ testar o algoritmo usando o teste de mesa descrito no final deste capítulo.

Torres de Hanoi

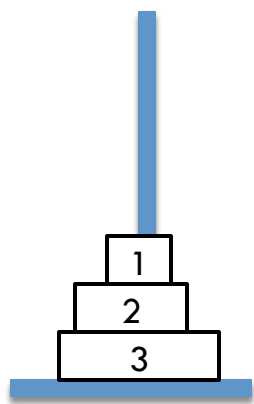
Considere as Torres de Hanói como sendo três hastes e três discos.

Uma das hastes serve de suporte para três discos de tamanhos diferentes de forma que os menores estão sobre os maiores.

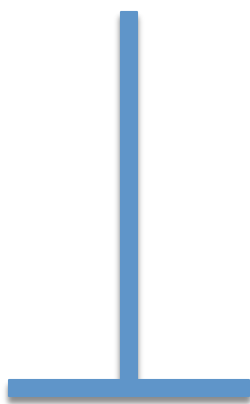
Desenvolva um algoritmo que mova todos os discos da haste onde se encontram para uma das outras duas hastes da Torre de Hanói, seguindo as seguintes regras: pode-se mover um disco de cada vez para qualquer haste, contanto que um disco maior nunca seja colocado sobre um disco menor.

Torres de Hanoi

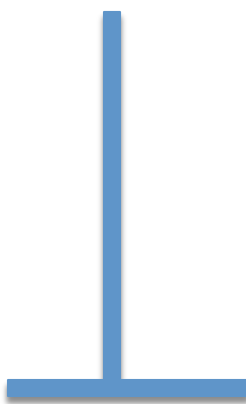
1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



Torre A



Torre B



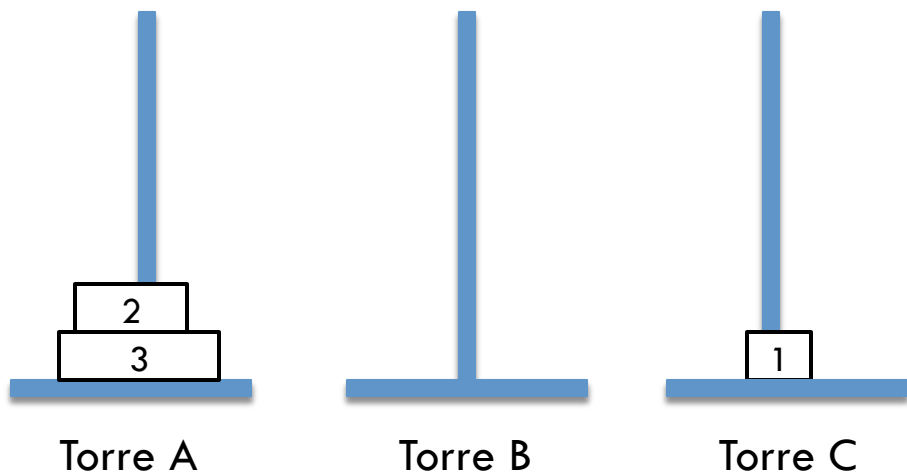
Torre C

Entradas: 3 discos na Torre A

Processamento:

Torres de Hanoi

1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



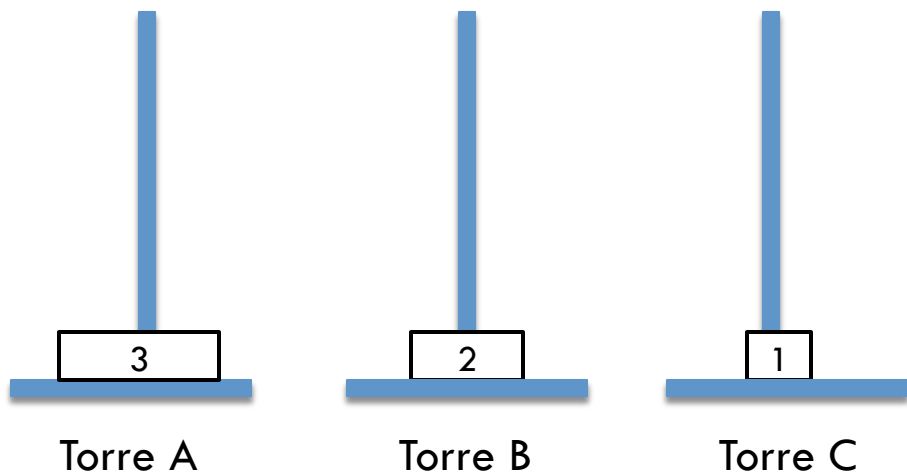
Entradas: 3 discos na Torre A

Processamento:

1. Mova o disco 1 para a haste C

Torres de Hanoi

1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



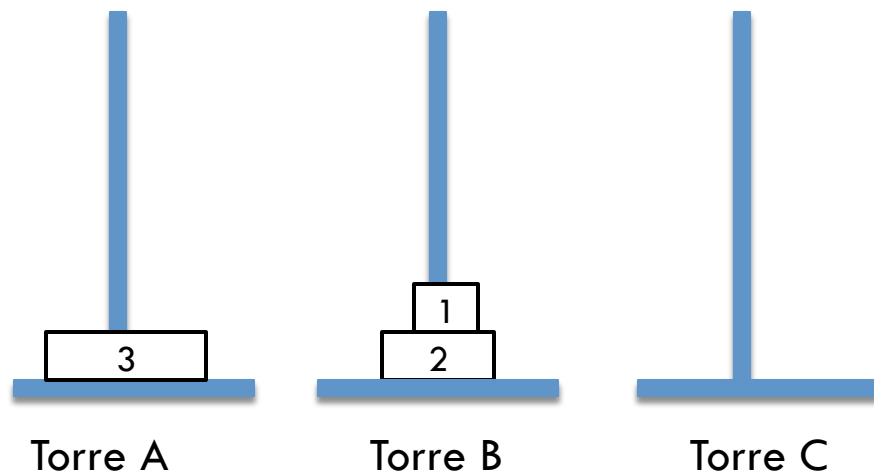
Entradas: 3 discos na Torre A

Processamento:

1. Mova o disco 1 para a haste C
2. Mova o disco 2 para a haste B

Torres de Hanoi

1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



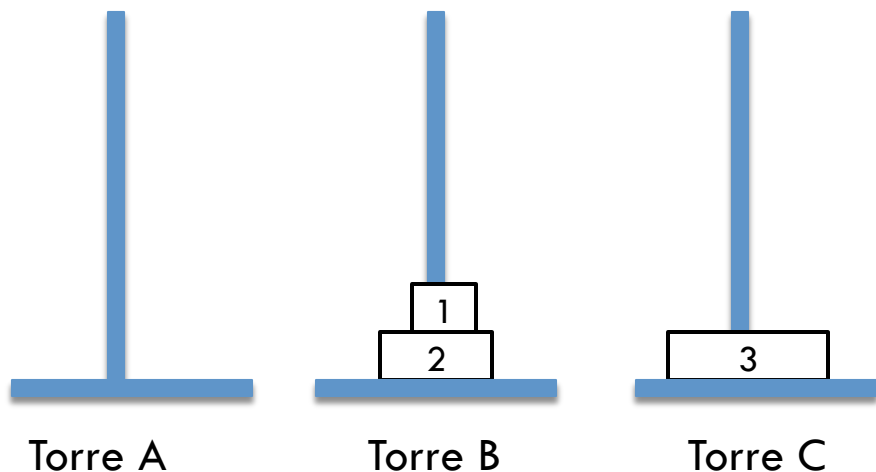
Entradas: 3 discos na Torre A

Processamento:

1. Mova o disco 1 para a haste C
2. Mova o disco 2 para a haste B
3. Mova o disco 1 para a haste B

Torres de Hanoi

1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



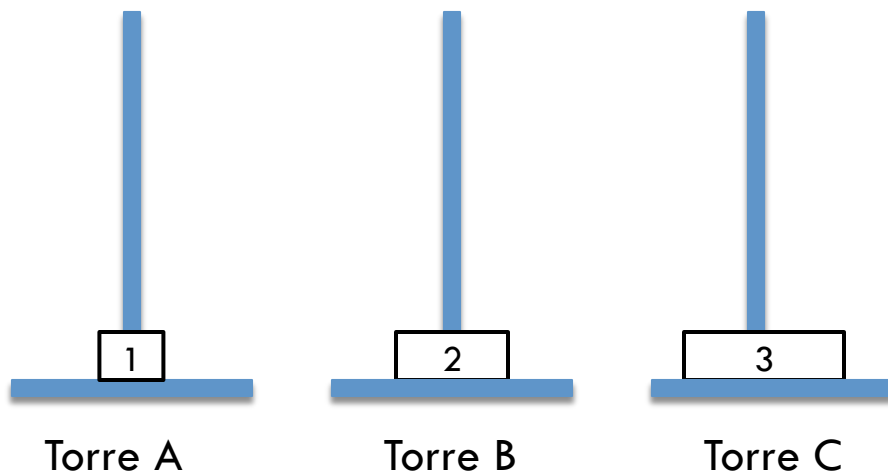
Entradas: 3 discos na Torre A

Processamento:

1. Mova o disco 1 para a haste C
2. Mova o disco 2 para a haste B
3. Mova o disco 1 para a haste B
4. Mova o disco 3 para a haste C

Torres de Hanoi

1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



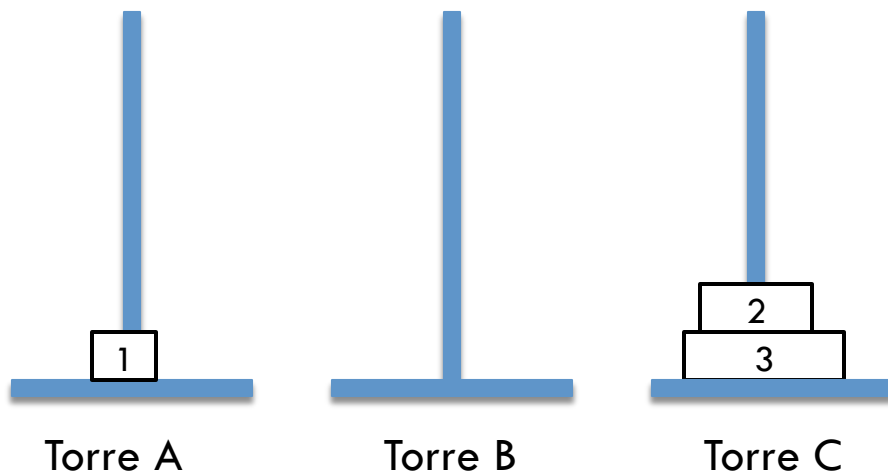
Entradas: 3 discos na Torre A

Processamento:

1. Mova o disco 1 para a haste C
2. Mova o disco 2 para a haste B
3. Mova o disco 1 para a haste B
4. Mova o disco 3 para a haste C
5. Mova o disco 1 para a haste A

Torres de Hanoi

1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



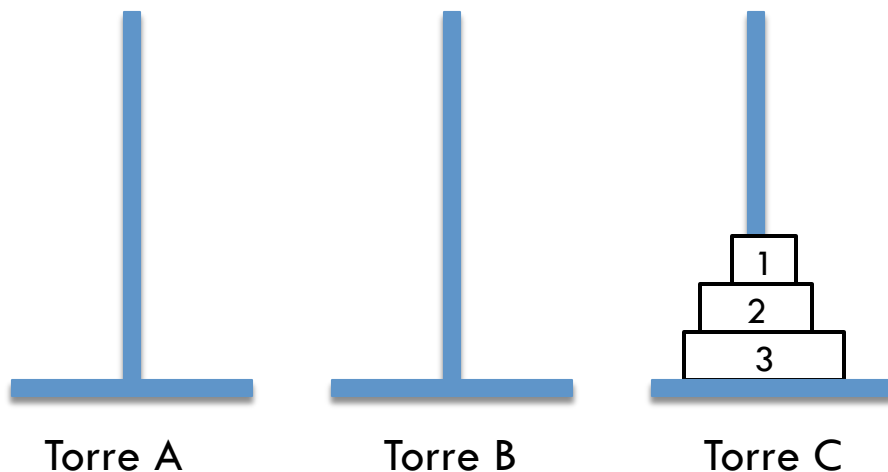
Entradas: 3 discos na Torre A

Processamento:

1. Mova o disco 1 para a haste C
2. Mova o disco 2 para a haste B
3. Mova o disco 1 para a haste B
4. Mova o disco 3 para a haste C
5. Mova o disco 1 para a haste A
6. Mova o disco 2 para a haste C

Torres de Hanoi

1. Transferir os 3 discos da Torre A para a Torre C
2. É proibido colocar um disco maior sobre um menor.
3. Objetivo: transferir com a menor movimentação necessária.



Número de movimentos = $2^n - 1$,
onde n é o número de discos.

Entradas: 3 discos na Torre A

Processamento:

1. Mova o disco 1 para a haste C
2. Mova o disco 2 para a haste B
3. Mova o disco 1 para a haste B
4. Mova o disco 3 para a haste C
5. Mova o disco 1 para a haste A
6. Mova o disco 2 para a haste C
7. Mova o disco 1 para a haste C

Saída: 3 discos na Torre C

Tipos de algoritmos

- ❑ descrição narrativa
- ❑ Fluxograma
- ❑ pseudocódigo (ou português estruturado ou portugol).

Tipos de algoritmos

Descrição Narrativa

Descrição narrativa. A descrição narrativa nada mais é do que descrever, utilizando uma linguagem natural (no nosso caso, a língua portuguesa), as ações a serem realizadas no tratamento dos dados de entrada para os resultados de saída na resolução do problema proposto.

A vantagem desse tipo de algoritmo é que a linguagem natural não oferece o esforço do aprendizado, pois ela já é bem conhecida. A desvantagem é que a linguagem natural pode ter interpretações diferentes dependendo do ponto de vista de cada pessoa, podendo dificultar a transcrição para o programa.

Tipos de algoritmos

Descrição Narrativa – Exemplo

1. Iniciar o algoritmo;
2. Receber dois números;
3. Somar esses dois números;
4. Apresentar a soma dos dois números;
5. Finalizar o algoritmo.

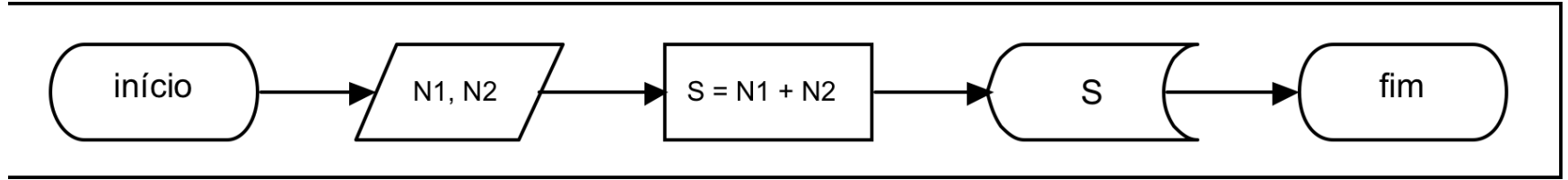
Tipos de algoritmos

Fluxograma

- ❑ **Fluxograma.** O fluxograma é a forma de descrever as ações a serem realizadas no tratamento dos dados de entrada para os resultados de saída usando uma representação simbólica preestabelecida, por exemplo, como os símbolos na Figura 3.
- ❑
- ❑ A vantagem desse tipo de algoritmo é o entendimento. Qualquer representação gráfica sempre é mais bem entendida do que uma representação por escrito. A desvantagem desse tipo de algoritmo é a necessidade do aprendizado de cada símbolo bem como seu significado. Além disso, um algoritmo descrito por um fluxograma não apresenta detalhes, podendo dificultar a transcrição para o programa.






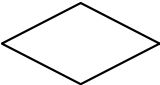

Tipos de algoritmos

Fluxograma - Exemplo



Tipos de algoritmos

Fluxograma - Exemplo

	Símbolo utilizado para determinar o início e o término do algoritmo.
	Símbolo utilizado para determinar o sentido do fluxo de dados do algoritmo e utilizado para conectar os símbolos existentes.
	Símbolo utilizado para determinar uma ação ou comando que pode ser um cálculo ou uma atribuição de valores.
	Símbolo utilizado para determinar a saída de dados do algoritmo. Essa entrada de dados pode ser via dispositivos de entrada.
	Símbolo utilizado para determinar a saída de dados do algoritmo. Essa saída de dados pode ser via monitor.
	Símbolo utilizado para determinar uma decisão que indicará qual caminho será seguido no algoritmo.
	Símbolo utilizado para determinar uma conexão entre partes de um mesmo algoritmo.

Tipos de algoritmos

Pseudocódigo

Pseudocódigo. O pseudocódigo é a forma de descrever as ações para a resolução de um problema proposto por meio de regras preestabelecidas.

A vantagem desse tipo de algoritmo é a possibilidade da transcrição quase que imediata desse algoritmo para uma linguagem de programação, bastando saber a sintaxe e a semântica dessa linguagem de programação. A desvantagem deste tipo de algoritmo é a necessidade do aprendizado de cada uma das regras do pseudocódigo bem como seu significado. A seguir, apresentaremos cada uma dessas regras do pseudocódigo para o melhor aprendizado.

Tipos de algoritmos

Pseudocódigo - exemplo

Algoritmo Soma

início_algoritmo

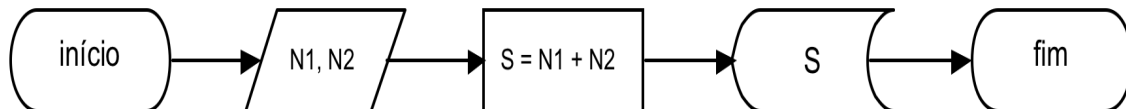
declare N1, N2, S: **inteiro**;

leia (N1, N2);

$S \leftarrow N1 + N2$;

escreva(S);

fim_algoritmo.



Tipos de algoritmos

Pseudocódigo - exemplo

Algoritmo Soma

início_algoritmo

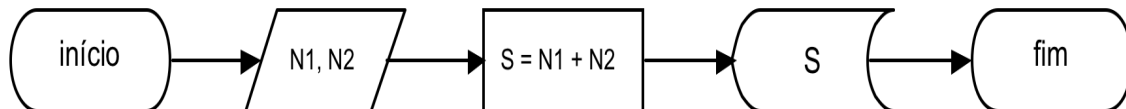
declare N1, N2, S :inteiro;

leia (N1, N2);

$S \leftarrow N1 + N2$;

escreva(S);

fim_algoritmo.



Teste de mesa

Testando o algoritmo construído

Para realizar o teste de mesa em um algoritmo é muito simples, basta seguir, passo-a-passo, cada comando do algoritmo como se fosse o computador. Assim, no final do teste de mesa, você pode concluir se o algoritmo está correto ou não.

Se não estiver, basta corrigir o algoritmo, refazer o teste de mesa e repetir esse processo até que você certifique-se de que o algoritmo esteja realmente correto.

Mesmo que você suponha que seu algoritmo esteja correto, você deve fazer o teste de mesa, pois erros imperceptíveis podem ser detectados. Não se corre o risco de ter um algoritmo que não retorna sempre o resultado esperado.

Comentários em pseudocódigo

Os comentários são representados por `//` seguidas do comentário.

Linhas de comentários são ignoradas e servem para explicar alguma lógica de negócios ou regras da aplicação.

início_algoritmo

```
declare N1, N2, S: inteiro    //declaração das variáveis
```

```
//entrada de dados: leitura de 2 números inteiros
```

```
leia(a, b, c)
```

```
//processamento: cálculo da soma dos 2 números
```

```
S ← N1 + N2
```

```
//saída: resultado da soma dos 2 números
```

```
escreva(delta);
```

fim_algoritmo.

Como construir um algoritmo (1)

Exemplo de como fazer um café

Descrição Narrativa Seqüencial

iniciar algoritmo;

pegar bule;

colocar coador de plástico sobre o bule;

colocar coador de papel sobre o coador de plástico;

colocar café tostado e moído sobre o coador de papel;

colocar água sobre o café;

finalizar algoritmo.

Como construir um algoritmo (2)

Exemplo de como fazer um café

Descrição Narrativa de Seleção

iniciar algoritmo;

pegar bule;

colocar coador de plástico sobre o bule;

colocar coador de papel sobre o coador de plástico;

colocar café tostado e moído sobre o coador de papel;

se a água estiver fervente, **então**

colocar água sobre o café;

finalizar algoritmo.

Como construir um algoritmo (3)

Exemplo de como fazer um café

Descrição Narrativa de Repetição sem a estrutura de repetição

iniciar algoritmo;
pegar bule;
colocar coador de plástico sobre o bule;
colocar coador de papel sobre o coador de plástico;
colocar café tostado e moído sobre o coador de papel;
se a água não estiver fervente, **então**
aquecer a água;
se a água não estiver fervente, **então**
continuar aquecendo a água;
se a água não estiver fervente, **então**
continuar aquecendo a água;
...
continua até quando?
colocar água sobre o café;
finalizar algoritmo.

Como construir um algoritmo (4)

Exemplo de como fazer um café

Descrição Narrativa de Repetição usando a estrutura de repetição

iniciar algoritmo;

pegar bule;

colocar coador de plástico sobre o bule;

colocar coador de papel sobre o coador de plástico;

colocar café tostado e moído sobre o coador de papel;

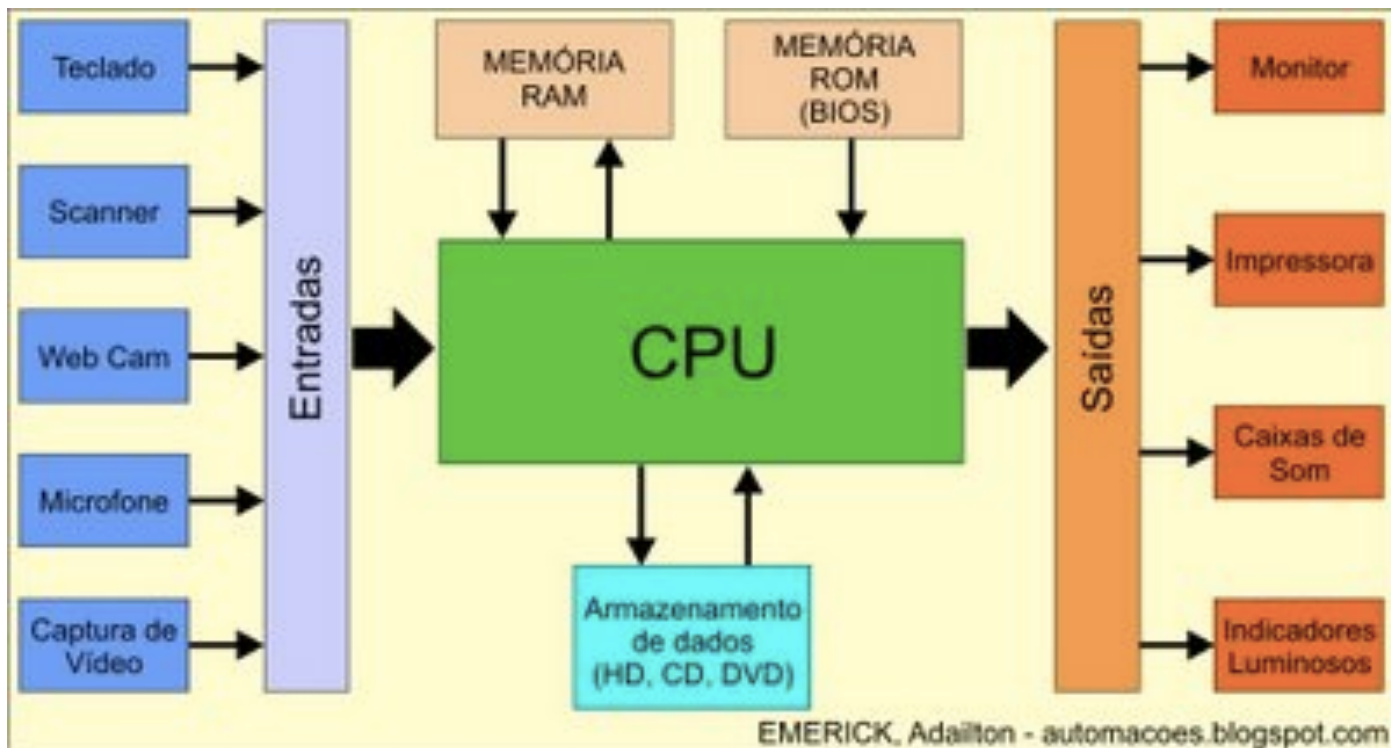
enquanto a água não estiver fervente, **faça**

aquecer a água;

colocar água sobre o café;

finalizar algoritmo.

Arquitetura dos Computadores



Unidades de Medida de Dados

Unidade	Símbolo	Valor Equivalente
Bit	0 ou 1	
Byte	B	8 bits
Kilobyte	KB	1024 B
Megabyte	MB	1024 KB
Gigabyte	GB	1024 MB
Terabyte	TB	1024 GB
Petabyte	PB	1024 TB
Exabyte	EB	1024 PB
Zettabyte	ZB	1024 EB
Yottabyte	YB	1024 ZB



Tipos de Dados

Os tipos de dados são as características comuns dos dados a serem manipulados.

Podemos considerar quatro classificações para os tipos de dados:

- Inteiro
- Real
- Caracter
- Lógico

Tipos de Dados

- O **tipo inteiro** caracteriza qualquer dado numérico que pertença ao conjunto dos números inteiros. Por exemplo: -5, 0, 32. O tipo inteiro quando armazenado no computador ocupa 4 bytes, o que corresponde a 65536 possibilidades, ou seja, de -32767 até 32767.
- O **tipo real** caracteriza qualquer dado numérico que pertença ao conjunto dos números reais. Por exemplo: -9.0, 0, 29.45. O tipo real quando armazenado no computador ocupa 8 bytes, o que corresponde a 2^{32} possibilidades, podendo ter de 6 a 11 dígitos significativos com sinal.
- O **tipo caracter** caracteriza qualquer dado que pertença a um conjunto de caracteres alfanuméricos e são simbolizados por entre aspas duplas (""). Por exemplo: "15", "Eu", "Pare!", "?%@" . O tipo caracter quando armazenado no computador ocupa 1 byte para cada caracter.
- O **tipo lógico** caracteriza qualquer dado que possa assumir somente duas situações: verdadeiro ou falso. Por exemplo: feminino ou masculino, loja aberta ou fechada, brasileiro ou estrangeiro. O tipo lógico quando armazenado no computador ocupa 1 byte, pois possui apenas duas formas de representação.

Operador de atribuição (\leftarrow) e comandos de atribuição

Atribui um determinado valor para uma variável, tendo o cuidado de verificar se o valor que está sendo atribuído à variável tem o tipo de dado compatível, ou seja, se uma variável **x** foi declarada como inteiro, só é permitido atribuir valores inteiros à variável **x**.

Sintaxe:

<nome da variável> \leftarrow <expressão> ;

onde **expressão** pode ser uma expressão lógica ou aritmética

Exemplo de uso do operador de atribuição (\leftarrow)

Por exemplo, vamos considerar que as variáveis **x**, **y** e **z** foram declaradas como do tipo numérico **inteiro**:

$x \leftarrow 25;$

$y \leftarrow x + 15 - 3;$

$z \leftarrow y - x + \text{rad}(x) - \text{pot}(y, 2);$

Exemplo: equação de 2º. grau

Equação: $2x^2 + 3x + 5$

Variáveis do problema: $ax^2 + bx + c$

$$\Delta = b^2 - 4.a.c$$

$$x = \frac{-b \pm \sqrt{\Delta}}{2.a}$$

Algoritmo para Resolver parte da Equação do 2o. grau

Algoritmo Calculo Delta para calcular a Equação 2º. grau

início_algoritmo

declare a, b, c, delta: inteiro

 x1, x2 : **real**

leia (a, b, c);

 delta \leftarrow b*b - 4 * a * c;

escreva (delta);

fim_algoritmo.

Equação: $2x^2 + 3x + 5$

Variáveis do problema: $ax^2 + bx + c$

Entradas: a, b, c

Processamento: calcular o delta

$$\Delta = b^2 - 4.a.c$$

Saída: o valor do delta

Algoritmo para calcular o delta

início_algoritmo

declare

a, b, c , delta: **inteiro**

x1, x2 : **real**

//entrada

leia (a, b, c);

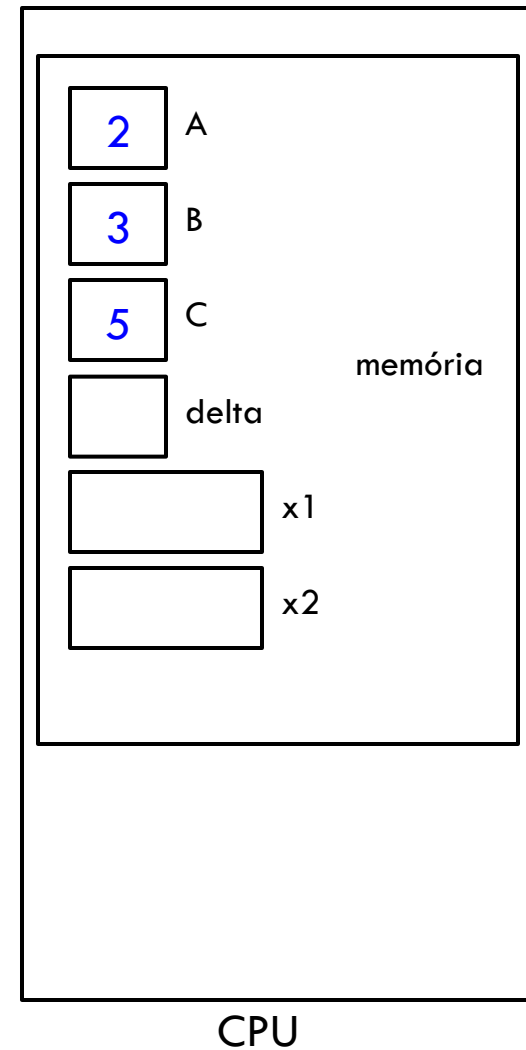
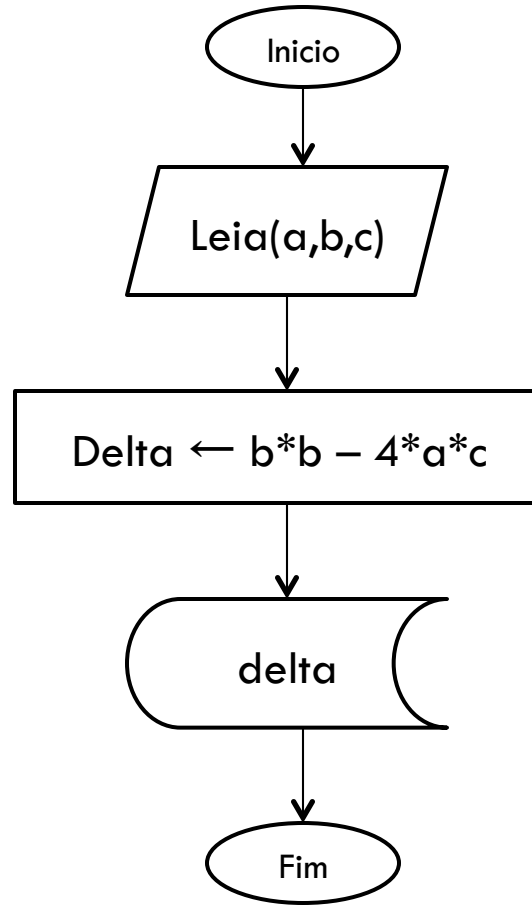
//processamento

delta \leftarrow b*b - 4 * a * c;

//saída

escreva (delta);

fim_algoritmo.



Instância do problema: $2x^2 + 3x + 5$

Bibliografias

BÁSICA

- GOMES, Ana Fernanda A. Campos, Edilene Aparecida V. Fundamentos da Programação de Computadores – Algoritmos, Pascal e C/C++. Prentice Hall, 2007.
- CARBONI, Irenice de Fátima. Lógica de Programação. Thomson.
- XAVIER, Gley Fabiano Cardoso. Lógica de Programação - Cd-rom. Senac São Paulo – 2007.

COMPLEMENTAR

- FORBELLONE, André Luiz Villar. Eberspache, Henri Frederico. Lógica de Programação – A construção de Algoritmos e Estrutura de Dados. Makron Books, 2005.
- LEITE, Mário - Técnicas de Programação – Brasport - 2006.
- PAIVA, Severino – Introdução à Programação – Ed. Ciência Moderna – 2008.
- PAULA, Everaldo Antonio de. SILVA, Camila Ceccatto da. Lógica de Programação –Viena – 2007.
- CARVALHO, Fábio Romeu, ABE, Jair Minoru. Tomadas de decisão com ferramentas da lógica paraconsistente anotada: Método Paraconsistente de Decisão (MPD), Editora Edgard Blucher Ltda. - 2012.