



# **Universidade Federal do Piauí**

## **Centro de Ensino Aberto e a Distância**

### **Curso de Sistemas de Informação**

# **Algoritmos e Programação I**

## **Conceitos Básicos**

Prof. Arlino Magalhães  
[arlino@ufpi.edu.br](mailto:arlino@ufpi.edu.br)

# Agenda

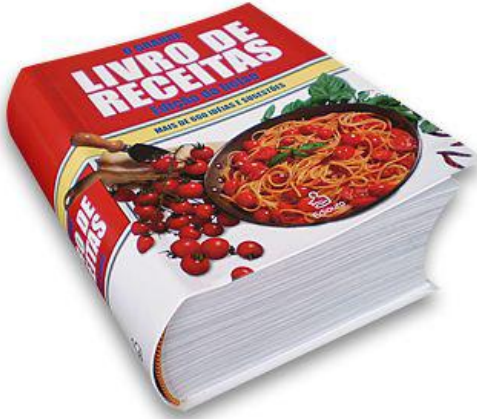
---

1. O que é um Algoritmo?
2. Algoritmos
  1. Variável
  2. Constante
  3. Operadores
3. Comandos
  1. Atribuição
  2. Entrada de Dados
  3. Saída de Dados
4. Exemplos
5. Compilador
6. Exercícios

# O que é um Algoritmo?

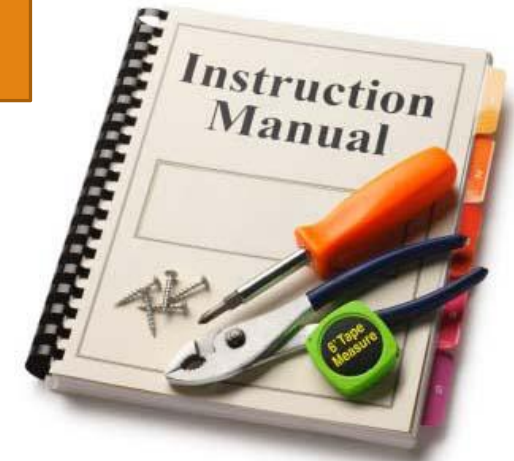
---

**Algoritmo:** É uma sequência não ambígua e finita de instruções, cuja a execução, em tempo finito, resolve um problema computacional (SEBESTA, 2003).

A screenshot of a Notepad window titled "hello.c - Notepad". The window has a menu bar with "File", "Edit", "Format", "View", and "Help". The text area contains the following C code:

```
#include <stdio.h>

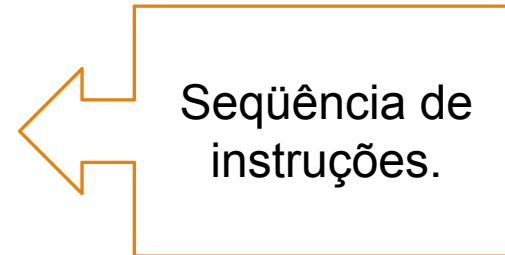
main()
{
    printf("Hello World!\n");
}
```



# Algoritmos - Fritar um Ovo

---

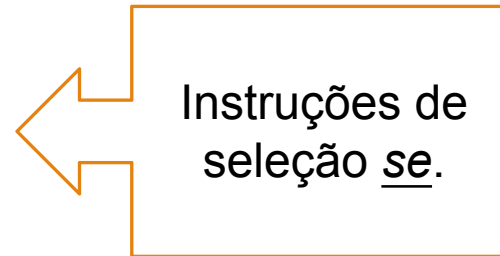
1. Pegar frigideira, ovo, óleo e sal;
2. Colocar óleo na frigideira;
3. Acender o fogo;
4. Colocar a frigideira no fogo;
5. Esperar o óleo esquentar;
6. Colocar o ovo na frigideira;
7. Retirar o ovo quando pronto.



# Algoritmos - Trocar Lâmpadas

---

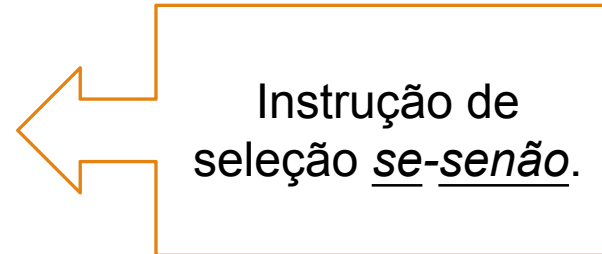
1. **Se** (lâmpada estiver alta)  
pegar escada;
2. Pegar lâmpada nova;
3. **Se** (lâmpada estiver quente)  
pegar pano;
4. Tirar lâmpada queimada;
5. Colocar lâmpada nova.



# Algoritmos - Fazer um Bolo

---

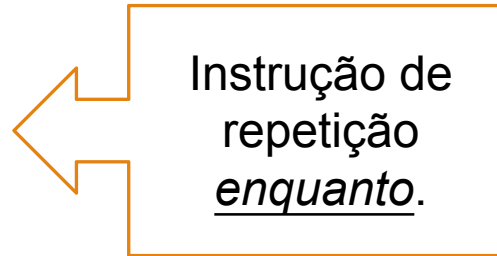
1. Pegar ingredientes;
2. Se (roupa for branca)  
colocar avental;
3. **Se** (tiver batedeira)  
bater ingredientes na batedeira;  
**Senão**  
bater ingredientes à mão;
4. Colocar a massa na forma;
5. Colocar forma no forno;
6. Esperar o bolo assar;
7. Retirar o bolo do forno.



# Algoritmos - Descascar Batatas

---

1. Pegar faca, bacia e batatas;
2. Se (roupa for branca)  
colocar avental;
3. **Enquanto** (houver batatas)  
descascar uma batata;
4. Limpar e guardar faca e bacia.



# Algoritmos - Os Canibais e os Missionários

---

Ajude os três missionários a atravessarem o rio. Não deixe os canibais em maior número junto dos missionários em nenhum dos lados, senão eles comem os missionários.





# Algoritmos - Torre de Hanói

---

Passe os seis discos do pino 01 para o pino 03 de maneira que um disco maior nunca fique em cima de outro menor em nenhuma situação. O pino 02 pode ser utilizado como pino auxiliar.



# Algoritmos - Algoritmos Computacionais

---

Construir [algoritmos para computadores](#) requer, em primeiro lugar, a transformação do nosso "passo a passo" em uma estrutura que possa ser entendida e executada pelo computador.

Utilizaremos inicialmente uma linguagem conhecida como [Portugol](#) (mais parecida com o português) e mais tarde, assim que os principais conceitos tenham sido assimilados, a [Linguagem C](#).

Para iniciar os estudos de algoritmos computacionais devemos aprender antes alguns conceitos básicos:

- variável;
- constante;
- operadores e;
- comandos de atribuição, entrada e saída.

# Variável - Memória

---

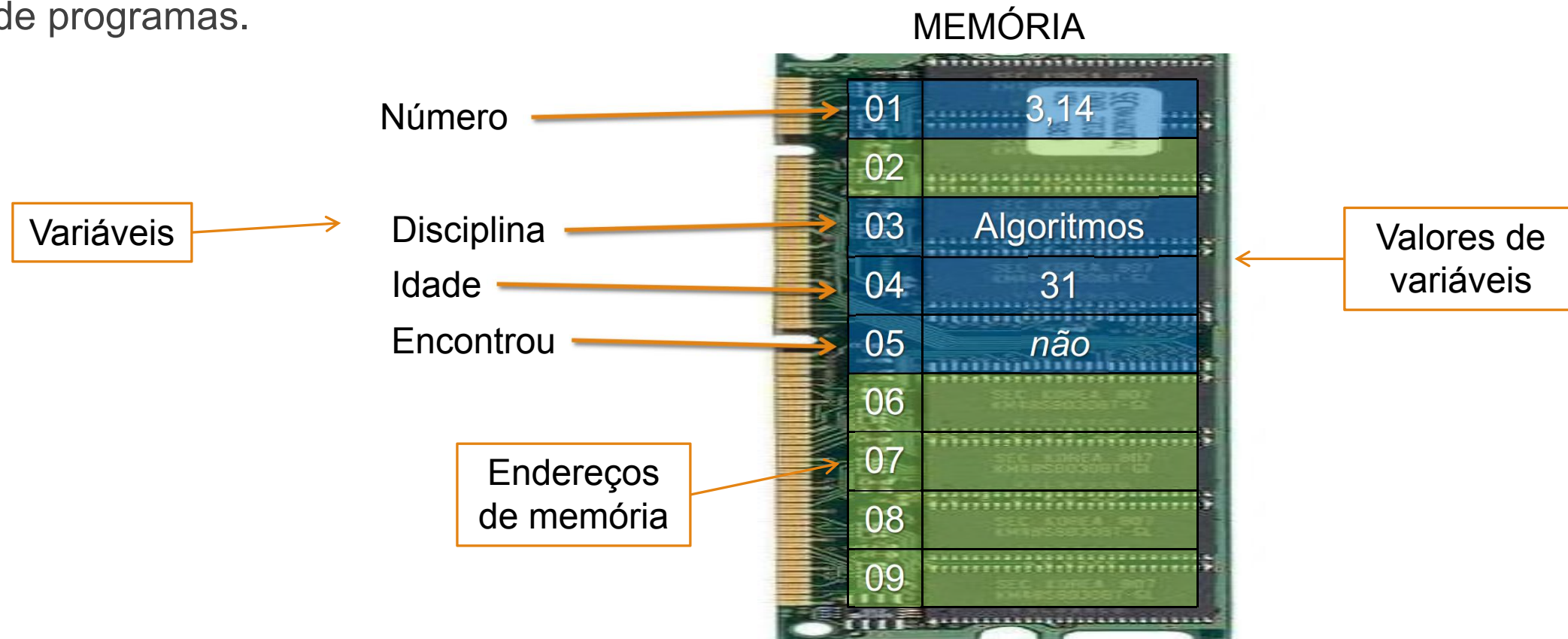
**Memória:** Meio físico para armazenar dados temporariamente ou permanentemente (TANENBAUM, 1997).

É na **memória** do computador que o programa será executado.

É necessário informar ao computador, no algoritmo, que precisaremos utilizar memória para armazenar dados inerentes à solução de um problema, como o resultado de uma operação, por exemplo.

# Variável - Memória

Variável é um **local na memória principal** onde serão armazenados valores utilizados durante a execução de programas.



# Variável - Tipos

---

Para cada tipo de dado (valor), existe um **tipo de variável** específica:

- **Inteiro**: valores inteiros (positivos ou negativos).
  - Exemplo: 1; 2000; -3; 0
- **Real**: valores reais (números fracionários).
  - Exemplo: 1,0; 2,001; -3,14; -3
- **Caractere**: um único caractere (uma letra ou um símbolo).
  - Exemplo: 'M'; 'F'; '@'; '1'
- **Cadeia (ou String)**: sequência de caracteres.
  - Exemplo: "Masculino"; "Feminino", "Algoritmos e Programação I"
- **Lógico**: valores lógicos.
  - Apenas: *VERDADEIRO* ou *FALSO*

Valores de **variáveis** **caracteres** devem ser escritos sempre entre **aspas**.

# Variável - Declaração de Variáveis

---

A **declaração** (escrita) do nome de uma variável (ou, simplesmente, declaração de uma variável) permite a identificação do tipo do valor armazenado nessa variável.

Regras básicas:

1. O nome de uma variável deve ser iniciado com uma letra e seguido por outras letras ou números.
2. Nunca devem ser utilizados caracteres que não sejam alfanuméricos, com exceção do *underline* (\_).

Sintaxe: **<tipo de variável>** *<lista de variáveis>*;

Exemplos:

- **inteiro** idade;
- **inteiro** idade, número, num, n, a, b;
- **real** altura, salário, x, fx, média\_das\_notas;
- **caractere** sexo, letra;
- **cadeia** nome, cidade, disciplina, frase;
- **lógico** achou, positivo, errado;

Exemplos de declarações erradas:

- **inteiro** 1número;
- **real** s@l@rio, média-das-notas, f(x);
- **cadeia** nome do aluno;

# Constante

---

As **constantes** são criadas com base nas mesmas regras e tipos já vistos em variável. Diferem apenas no fato de armazenar um valor constante, ou seja, que não se modifica durante a execução de um programa.

Sintaxe: **const** *<nome da constante>* = *<valor da constante>*;

Exemplo:

- **const** pi = 3,14;
- **const** juros = 0,99;
- **const** primeira\_letra = 'A';
- **const** nome\_empresa = "UFPI";

# Operadores - Aritméticos

---

Os **operadores aritméticos** são empregados em expressão aritmética em que são utilizados constantes e/ou variáveis do tipo real ou inteiro como operandos.

Operador	Símbolo
adição	+
subtração	-
multiplicação	*
divisão	/
divisão inteira	div
resto da divisão inteira	mod



# Operadores – Aritméticos (mod vs. div)

---

Diagram illustrating the relationship between division, modulus, and integer division for 7 and 2:

Division: 
$$\begin{array}{r} 7 \overline{) 2} \\ (1) \quad 3 \end{array}$$

Modulus:  $7 \bmod 2 = 1$

Division (Real):  $7 / 2 = 3,5$

Integer Division:  $7 \text{ div } 2 = 3$

Arrows indicate that the modulus result (1) and the integer division result (3) are derived from the division operation.

Diagram illustrating the relationship between division, modulus, and integer division for 3 and 5:

Division: 
$$\begin{array}{r} 3 \overline{) 5} \\ (3) \quad 0 \end{array}$$

Modulus:  $3 \bmod 5 = 3$

Integer Division:  $3 \text{ div } 5 = 0$

Arrows indicate that the modulus result (3) and the integer division result (0) are derived from the division operation.

# Operadores - Relacionais

---

Os **operadores relacionais** realizam comparações entre variáveis , constantes e/ou expressões.

Operador	Símbolo
igual	=
maior	>
menor	<
menor ou igual	≥
maior ou igual	≤
diferente	≠

# Operadores - Lógicos

---

Os **operadores lógicos** retornam *Falso* (F) ou *Verdadeiro* (V) , de acordo com seus operandos.

Operador	Símbolo
conjunção	E
disjunção	OU
negação	NÃO

# Comando de Atribuição

---

Na construção de algoritmos, é necessário indicar que uma variável deve armazenar um determinado valor durante a execução do programa.

Para fazer essa indicação é necessário a utilização do **comando de atribuição**, representado por uma **seta ( $\leftarrow$ )** apontando para a esquerda.

Exemplos:

- `num  $\leftarrow$  10;`
- `sexo  $\leftarrow$  'F';`
- `salário  $\leftarrow$  6.000,00;`
- `achou  $\leftarrow$  falso;`
- `disciplina  $\leftarrow$  "Algoritmos e Programação I";`
- `delta  $\leftarrow$  b*b + 2*a*c;`

Dizemos que a variável **num** recebeu o valor **10 (dez)**, ou seja, o local de memória reservado a variável num armazenou o valor inteiro 10, supondo que a variável é do tipo inteiro.

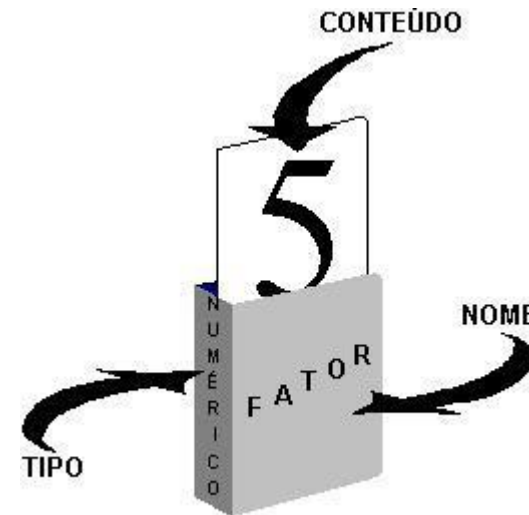
# Comando de Atribuição

Resumindo ...

```
inteiro fator;  
fator ← 5;
```

A variável chamada de **fator**, do tipo **inteiro**, armazena o valor **5**.

Analogicamente



# Comando de Entrada de Dados

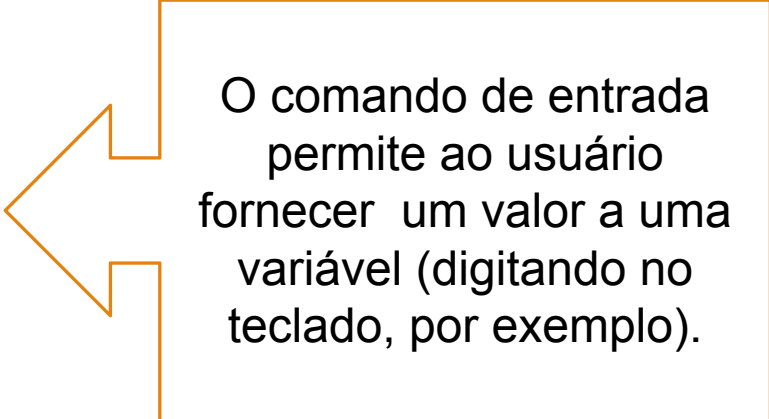
---

O **comando de entrada** é responsável pela leitura e armazenamento de dados em uma variável.

Sintaxe: **leia** (<variável>);

Exemplos:

- **leia** (num);
- **leia** (sexo);
- **leia** (salário);
- **leia** (disciplina);
- **leia** (a, b, c);



O comando de entrada permite ao usuário fornecer um valor a uma variável (digitando no teclado, por exemplo).

# Comando de Saída de Dados

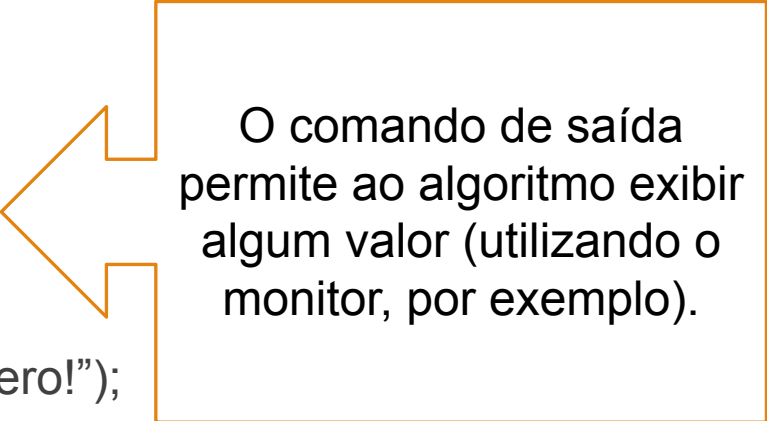
---

O **comando de saída** é responsável pela exibição de variáveis, constantes e/ou expressões.

Sintaxe: **escreva** (<variável, constante e/ou expressão>);

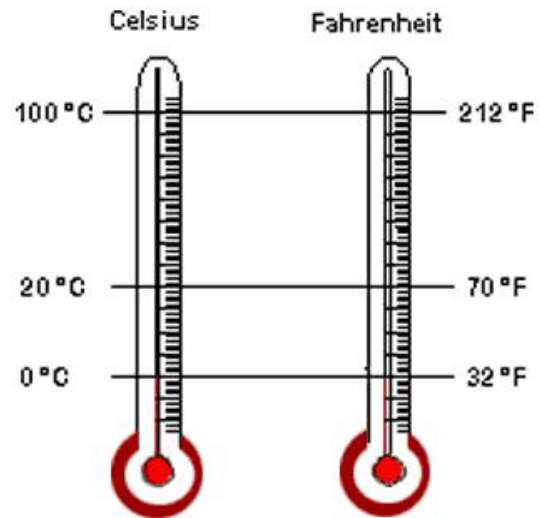
Exemplos:

- **escreva** (num);
- **escreva** (salário);
- **escreva** (disciplina);
- **escreva** (a);
- **escreva** ( $b*b + 2*a*c$ );
- **escreva** ("Digite o valor do número!");
- **escreva** ("O nome é: ", nome);
- **escreva** ("O resultado é ", rendimento, " reais de comissão!");



O comando de saída  
permite ao algoritmo exibir  
algum valor (utilizando o  
monitor, por exemplo).

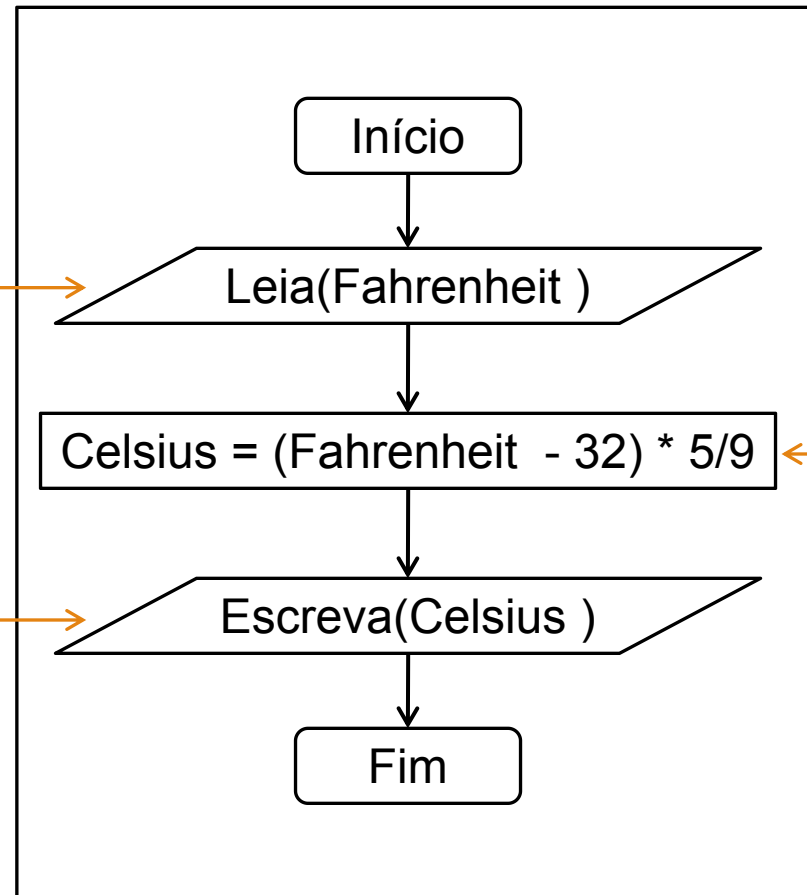
# Fluxograma – Conv. de Fahrenheit para Celsius



$$\frac{^{\circ}\text{C}}{5} = \frac{^{\circ}\text{F} - 32}{9}$$

Comando de entrada.

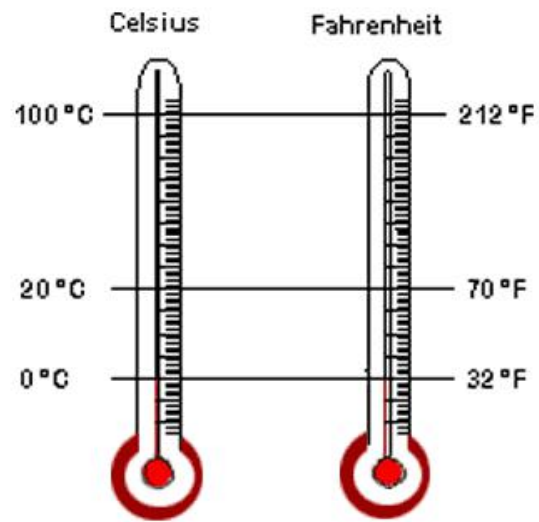
Comando de saída.



Comando de atribuição.



# Peseudo-Código – Conv. de Fahrenheit para Celsius



$$\frac{^{\circ}C}{5} = \frac{^{\circ}F - 32}{9}$$

**Algoritmo** Converte\_ Graus;

{

**real** fahrenheit, celsius;

**escreva** ("Digite a temperatura:");

**leia** (fahrenheit);

celsius ← (fahrenheit - 32) \* 5/9;

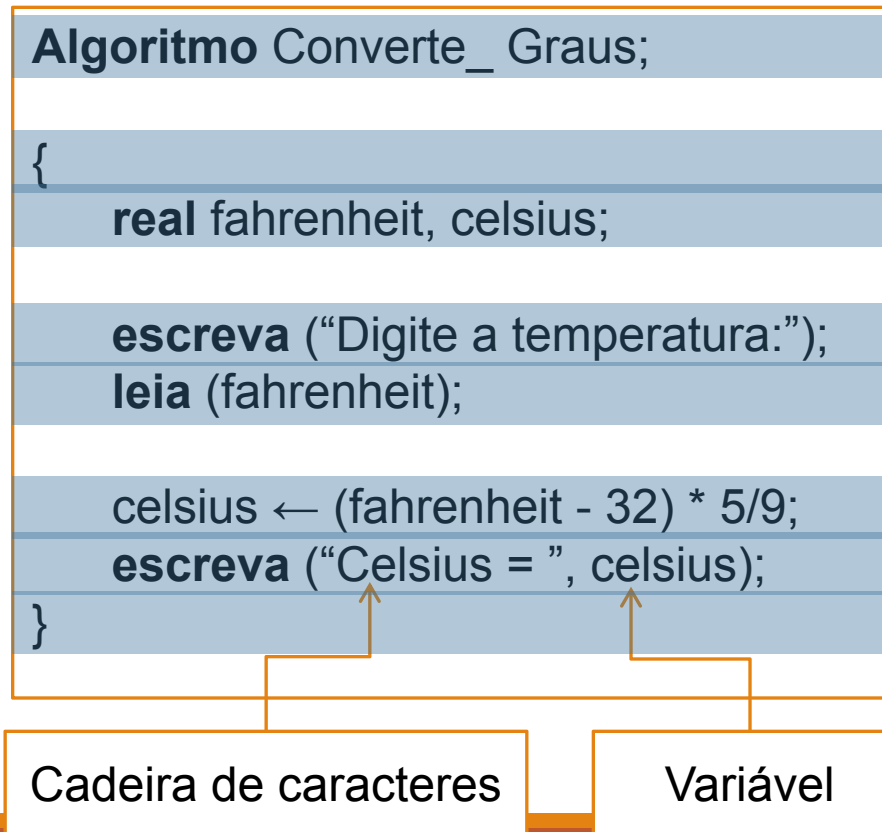
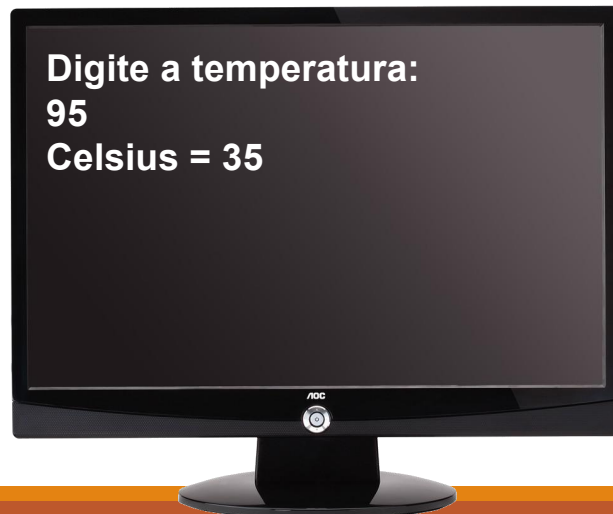
**escreva** ("Celsius = ", celsius);

}

O ; (ponto-e-vírgula)  
indica o fim de um  
comando.

# Peseudo-Código – Conv. de Fahrenheit para Celsius

MEMÓRIA	
fahrenheit →	01 95
	02
celsius →	03 35
	04
	05



# Exemplo - Velocidade Média



$$\text{Velocidade Média} = \frac{\text{Distância}}{\text{Tempo}}$$

**Algoritmo** Velocidade\_Média;

{

**real** velocidade, tempo, distância;

**escreva** ("Digite a distância:");

**leia** (distância);

**escreva** ("Digite o tempo:");

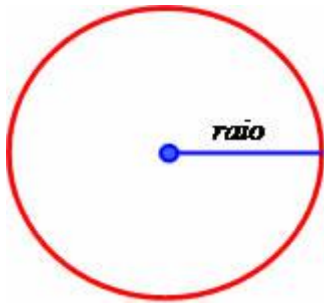
**leia** (tempo);

velocidade ← distância / tempo;

**escreva** ("Velocidade Média= ", velocidade);

}

# Exemplo - Operações sobre Circunferência



$$\text{Perímetro} = 2\pi r$$

$$\text{Área} = \pi r^2$$

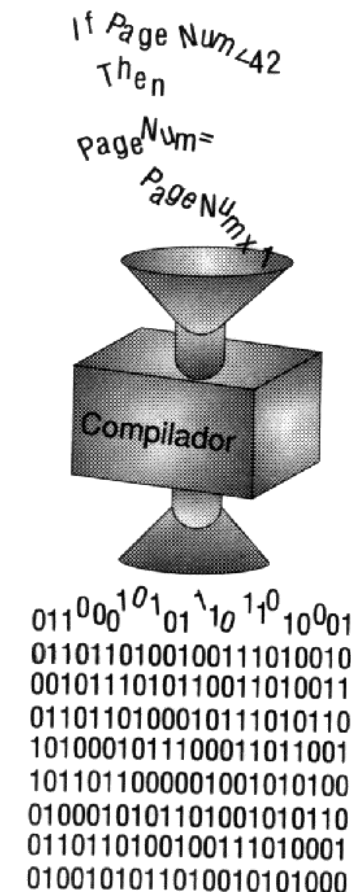
$$\pi = 3,14$$

```
Algoritmo Circunferência;  
{  
    const pi = 3,14;  
    real perímetro, área, raio;  
  
    escreva ("Digite a valor do raio:");  
    leia (raio);  
  
    perímetro ← 2*pi*raio;  
    área ← pi*raio*raio;  
  
    escreva ("Perímetro = ", perímetro);  
    escreva ("Área = ", área);  
}
```

# Compilador

Um compilador é um programa que traduz um programa descrito em uma **linguagem de programação (código-fonte)** para um programa equivalente em **linguagem de máquina (código de máquina)**.

O compilador nos permite escrever algoritmos em uma linguagem de programação que é mais facilmente entendida pelos seres humanos para , em seguida, traduzi-los para a linguagem de máquina que é entendida pelo computador.



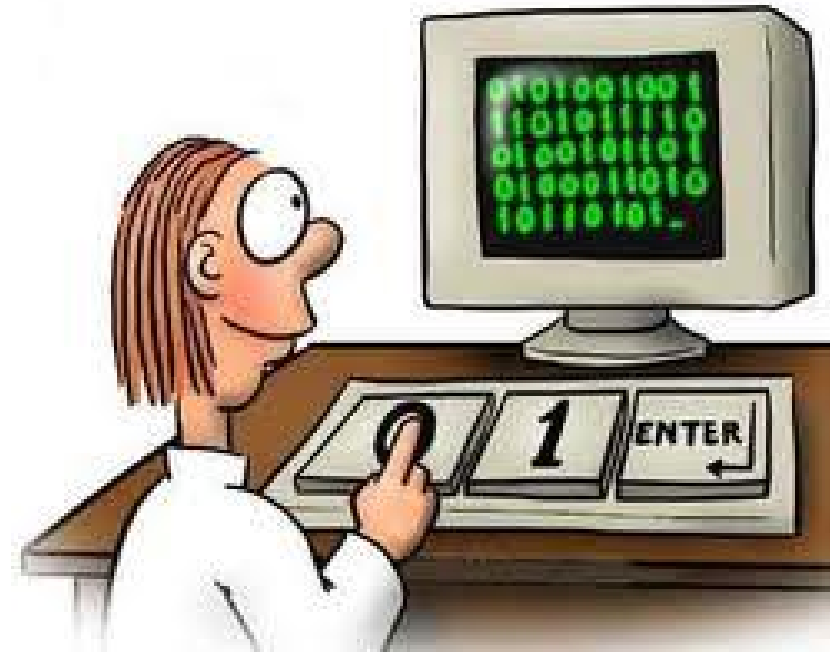
# Compilador - Linguagem de Máquina

---

A **linguagem de máquina** é a linguagem que o computador compreende e armazena as instruções das ações que deve realizar. É muito difícil de ser compreendida e utilizada pelos seres humanos.

A linguagem de máquina é formada por um conjunto de **bits** (*Binary Digit* ou Dígito Binário), representado graficamente por **0 (zero)** ou **1 (um)**.

Na realidade a linguagem de máquina compreende apenas a **ausência (0)** e a **presença (1)** de energia.



# Compilador - Linguagem de Programação

A **linguagem de programação** é uma linguagem mais facilmente compreendida pelos seres humanos, onde algoritmos são escritos com o objetivo de prover programas de computador.



# Compilador - Compilação

---

## Código-fonte

```
if (fimDeSemana)  
    printf ("sair");  
else  
    printf ("estudar");
```

Compilação

## Código-objeto

```
0111 0100 0101 0010  
1101 0000 0111 1110  
0111 0100 0101 0010  
1101 0000 0111 1110  
0111 0100 0101 0010  
1101 0000 0111 1110
```



# Exercício 01

---

Faça um algoritmo em que sejam digitadas as três notas de um aluno e, em seguida, calcule e exiba a média dessas notas.

**Entrada de dados:**

- Primeira nota: **n1**
- Segunda nota: **n2**
- Terceira nota: **n3**

**Saída de dados:**

- Média das notas: **média**

$$\text{Média das notas} = \frac{(n1 + n2 + n3)}{3}$$

```
Algoritmo Média_notas;  
{  
    real n1, n2, n3, média;  
  
    escreva ("Digite a primeira nota:");  
    leia (n1);  
    escreva ("Digite a segunda nota:");  
    leia (n2);  
    escreva ("Digite a terceira nota:");  
    leia (n3);  
  
    média ← (n1 + n2 + n3) / 3;  
    escreva ("Média das notas= ", média);  
}
```

# Exercício 02

Dados os coeficientes **a**, **b** e **c** de uma equação de segundo grau ( **$ax^2 + bx + c = 0$** ), exibir as raízes da equação.

**Entrada de dados:**

- Coeficiente A: **a**
- Coeficiente B: **b**
- Coeficiente C: **c**

**Saída de dados:**

- Raiz X': **x1**
- Raiz X'': **x2**

$$\Delta = b^2 - 4ac$$

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

**Algoritmo** Equação\_2grau;

```
{  
    real a, b, c, x1, x2, delta;  
    escreva ("Digite o coeficiente A:");  
    leia (a);  
    escreva ("Digite o coeficiente B:");  
    leia (b);  
    escreva ("Digite o coeficientes C");  
    leia (c);  
    delta ← b*b – 4*a*c;  
    x1 = ( -b + raiz (delta) )/2;  
    x2 = ( -b - raiz (delta) )/2;  
    escreva ("X' = ", x1, " X'' = ", x2);  
}
```