

**Algoritmos: Operadores, Expressão,
Sequência, comandos de entrada e
saída.**

Rômulo de Oliveira Nunes



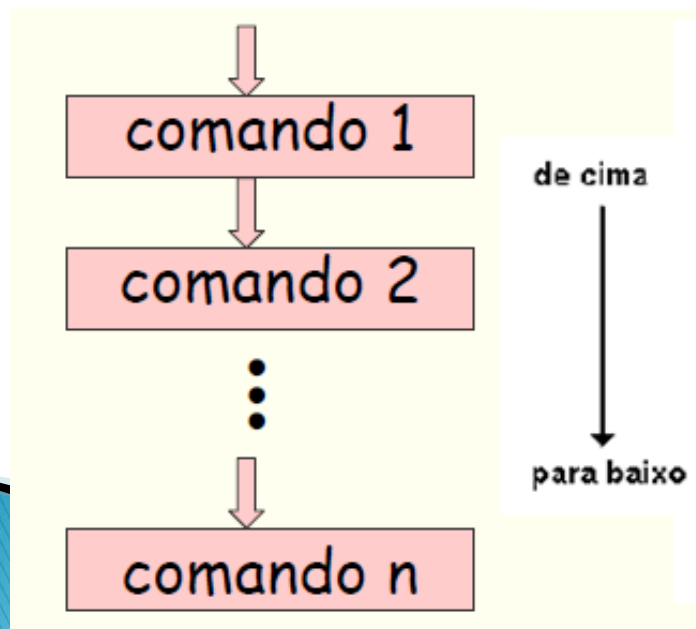
Definições

➤ Algoritmo

- Sequência finita e não ambígua de instruções computáveis para solucionar um determinado problema.
- Dentro do algoritmo haverá uma necessidade de tomada de decisões.
- **Estruturas básicas de controle do fluxo de execução**
 - Sequencial
 - Condicional ou de Seleção
 - Repetição

Estrutura sequencial

- Conjunto de comandos executados numa sequência linear;
- De cima para baixo;
- Cada comando é executado apenas após o término do comando anterior.
- Composta por atribuição, comandos de entrada e comandos de saída.



Sintaxe Geral:

```
inicio
    Comando 1;
    Comando 2;
    Comando 3;
    ....
    Comando n
fim
```

Estrutura geral

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    declaração de variáveis
```

```
    ....
```

```
    sentenças
```

```
    ....
```

```
}
```



O que são variáveis?

- ▶ A memória do computador é armazenada em blocos de dados



- ▶ Cada bloco contém uma informação nele

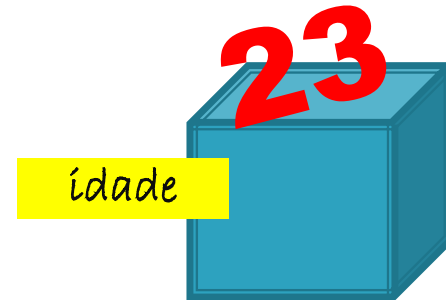


- ▶ ... e ao menos uma “etiqueta” para sabermos que informação é aquela
- ▶ Chamamos de “variáveis” porque o conteúdo das caixas podem mudar ao longo do tempo

Variáveis

Declaração da variável **idade**...
...atribuição do valor 23 a ela

```
int Idade = 23;  
float Peso = 74.5;  
char sexo = 'M';
```



Variáveis

Declaração da variável **peso** e...
...atribuição do valor 74.5 a ela

```
int Idade = 23  
float Peso = 74.5  
char sexo = 'M'
```

idade

23

peso

74.5

Variáveis

Declaração da variável **sexo** e...
...atribuição do valor 'M' a ela

```
int Idade = 23  
float Peso = 74.5  
char sexo = 'M'
```

idade

23

peso

74.5

Sexo

'M'

Os valores podem ser de
tipos diferentes

Variáveis

- Toda variável precisa ser declarada antes de ser usada.
- Somente valores do tipo especificado podem ser armazenados na variável.

DECLARAÇÃO:

```
tipo lista_de_variáveis;
```

EXEMPLO:

```
double lucro;  
int i, j;  
char a, b, c;
```

Tipos básicos

Tipo de Dado	Bits	Faixa de Valores
char	8	-128 a 127
int	16	-32768 a 32767
float	32	7 dígitos significativos
double	64	15 dígitos significativos

Identificadores

Nomes de variáveis, funções e outros objetos definidos pelo usuário.

- Pode ser composto por letras, números ou sublinhados.
- Deve começar com uma letra ou um sublinhado.
- Não pode ser uma palavra reservada da linguagem C

Palavras reservadas

auto	break	case	char	const	continue
default	do	double	else	enum	extern
float	for	goto	if	int	long
main	register	return	short	signed	sizeof
static	struct	switch	typedef	union	unsigned
void	volatile	while			

Declarando constantes

- Modificador **const**
- Garantir que o valor salvo em um endereço de memória não seja alterado durante a execução do programa.

```
const <tipo> nome = <valor>;
```

```
const int A = 10;  
const float GRAVIDADE = 9.8;
```

O operador de atribuição

Uma atribuição é uma expressão cujo valor resultante corresponde ao valor atribuído.

SINTAXE

nome_da_variável = expressão;

x = 2 + 4;

variável

operador de atribuição

Atribuições múltiplas

- O mesmo valor pode ser atribuído a muitas variáveis.

EXEMPLO

```
x = y = z = 0;
```

- A ordem de avaliação é da direita para a esquerda.

1º) $z = 0$

2º) $y = 0$

3º) $x = 0$

Operadores aritméticos

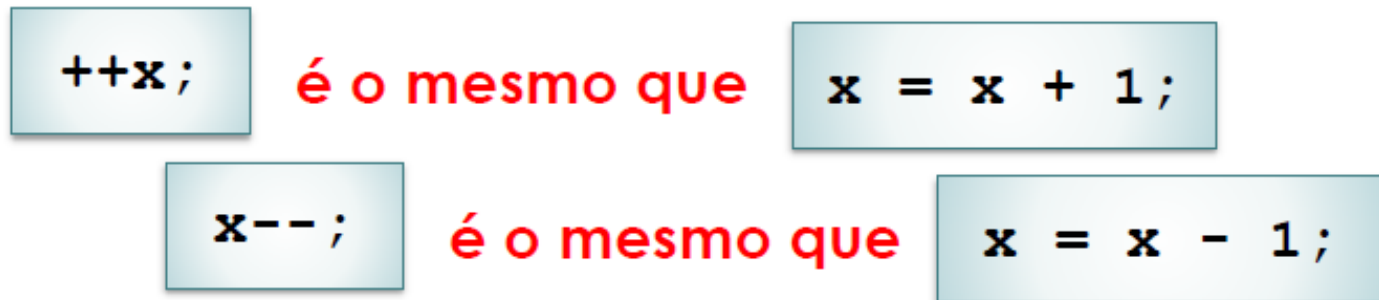
Operador	Ação
-	Subtração, menos unário
+	Adição
*	Multiplicação
/	Divisão
%	Módulo da divisão

ATENÇÃO!

As operações são feitas na precisão dos operandos.

Incremento e decremento

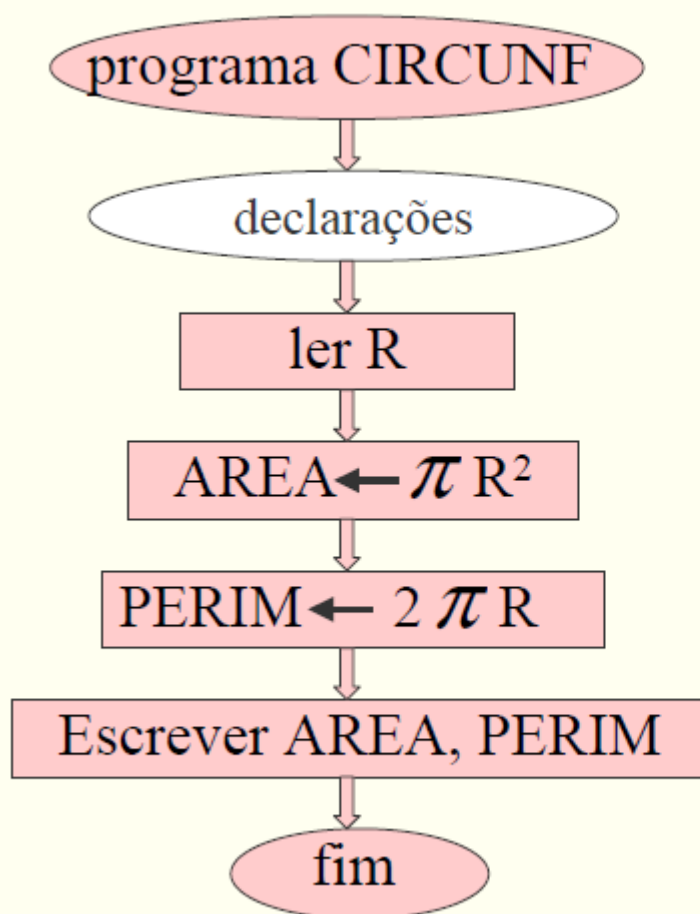
Operador	Ação
++	Soma 1 ao seu operando
--	Subtrai 1 ao seu operando



ATENÇÃO!

Há uma diferença entre `x++` e `++x` quando usados em uma expressão.

Exemplo Sequenciamento



programa CIRCUNF

declarações

início

leia (R)


AREA ← πR^2

PERIM ← $2 \pi R$

escreva (AREA, PERIM)

fim

Precedência dos operadores aritméticos

Mais alta	++ --
	- (menos unário)
	* / %
Mais baixa	+ -

- Operadores do mesmo nível de precedência são avaliados da esquerda para a direita.
- Parênteses podem ser usados para forçar uma (ou mais) operação a ter precedência maior.

Conversão de tipos em expressões

Quando operandos de tipos diferentes são misturados em uma expressão, os valores são convertidos no tipo do maior operando.

**prioridade de
conversão**



double

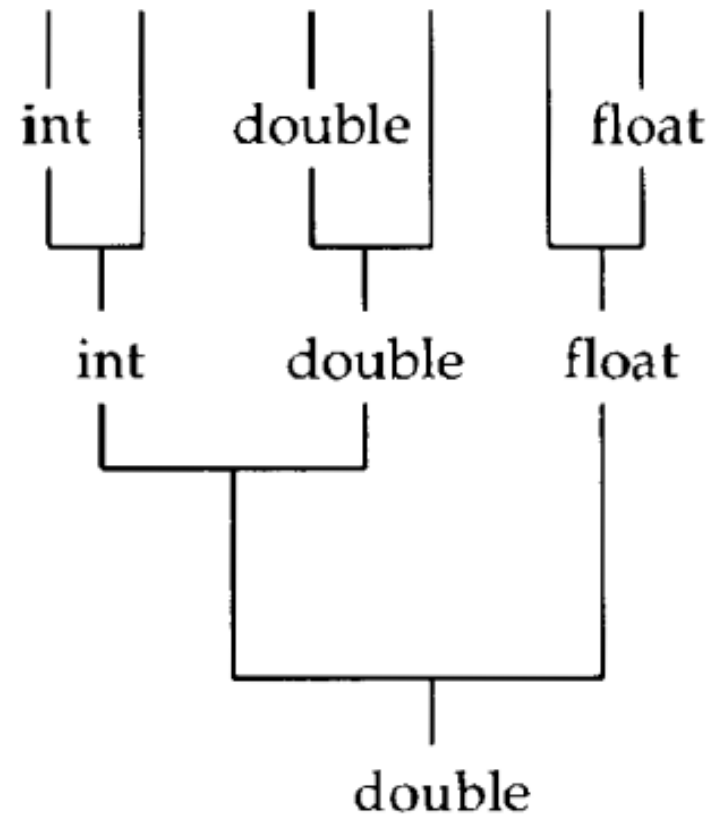
float

int

char

Exemplo

```
char ch;  
int i;  
float f;  
double d;  
result = (ch/i) + (f*d) - (f+i);
```



Operadores aritméticos de atribuição

variável = variável operador expressão;

é o mesmo que

variável operador = expressão;

EXEMPLOS

`i = i + 10;`

`i += 10;`

`x = x * (y + 1);`

`x *= y + 1;`

`a = a - 120;`

`a -= 120;`

Entrada e saída de dados

Comando de entrada

- Usado para transmitir informações (dados) do usuário para o computador;
- As informações serão armazenadas em variáveis declaradas no algoritmo;
- Uma das formas de se atribuir valores as variáveis;

Sintaxe: **scanf ("formatos", &var1, &var2,...);**

Comando de entrada

Sintaxe: `scanf ("formatos", &var1, &var2,...);`

Exemplos:

```
int i, j;
```

```
float x;
```

```
char c;
```

```
scanf("%d", &i);
```

```
scanf("%d %f", &j, &x);
```

```
scanf("%c", &c);
```

<code>%d</code>	para inteiros
<code>%i</code>	para inteiros
<code>%f</code>	para float
<code>%lf</code>	para double
<code>%c</code>	para char

Comando de entrada

Sintaxe: `scanf ("formatos", &var1, &var2,...);`

Exemplos:

```
int i, j;
```

```
float x;
```

```
char c;
```

```
scanf("%d", &i);
```

```
scanf("%d %f", &j, &x);
```

```
scanf("%c", &c);
```

**Não esquecer
do & antes do
nome da
variável!**

<code>%d</code>	para inteiros
<code>%i</code>	para inteiros
<code>%f</code>	para float
<code>%lf</code>	para double
<code>%c</code>	para char

Comando de saída

- Usado para transmitir informações do computador para o meio externo;
- Permite que o algoritmo possa mostrar os dados que calculou, como resposta ao problema que resolveu;
- Finalidade de exibir o conteúdo da variável identificada;

```
#include <stdio.h>

int main()
{
    printf("Hello World");

    return 0;
}
```

Comando de saída

- Usado para transmitir informações do computador para o meio externo;
- Permite que o algoritmo possa mostrar os dados que calculou, como resposta ao problema que resolveu;
- Finalidade de exibir o conteúdo da variável identificada;

```
printf ("formatos", var1, var2,...);
```

Comando de saída

```
printf ("formatos", var1, var2,...);
```

Exemplos:

```
int i, j;
```

```
float x;
```

```
char c;
```

```
printf("%d", i);
```

```
printf("%d, %f", j, x);
```

```
printf("%c", c);
```

Comando de saída

Exemplo:

```
#include /* Biblioteca de entrada e saída */
main() /* Aqui começa o programa principal */
{
    int ano = 2004; /* Declarei ano como inteiro e já defini seu valor */
    printf("Estamos no ano de %d", ano); /* Imprime o valor do ano */
}
```

Exemplo

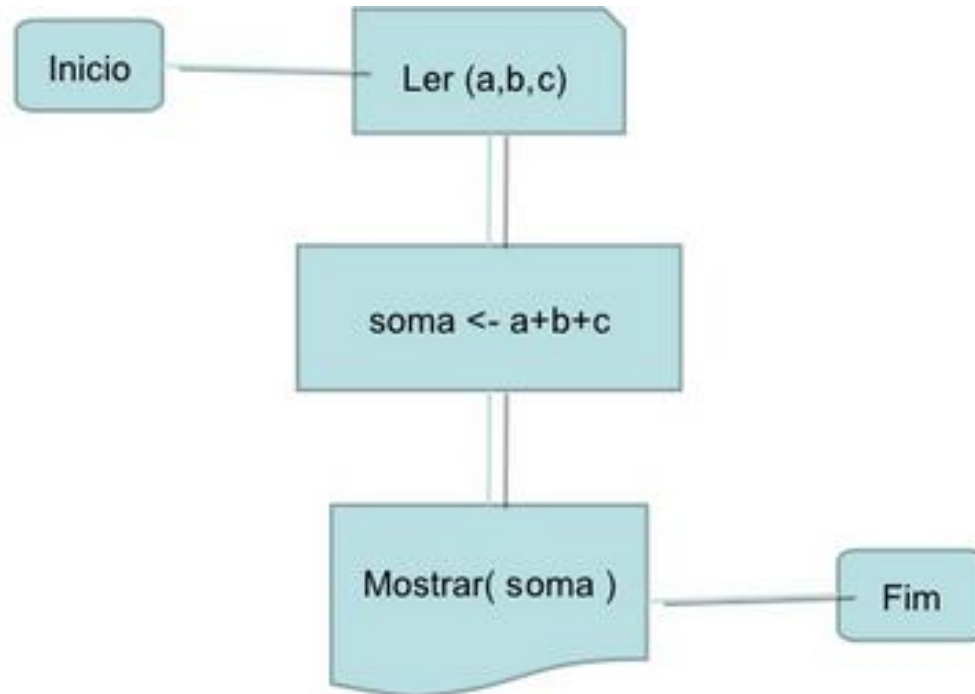
Faça um programa que recebe do usuário o valor do LADO e calcula a área de um quadrado.

Exemplo

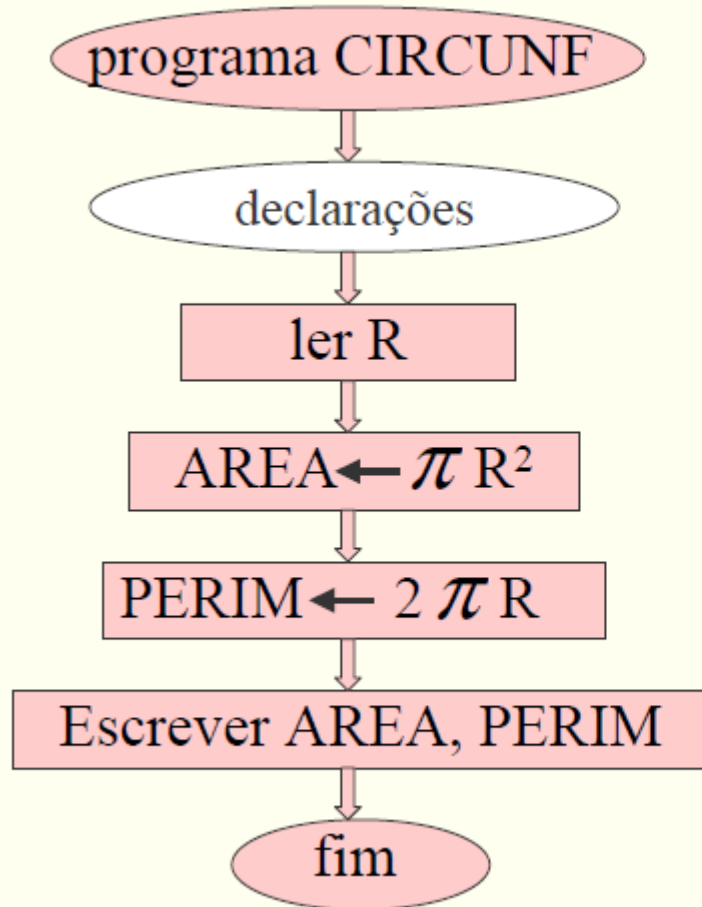
```
#include <stdlib.h>
#include <stdio.h>
int main( )
{
    int    lado, area_quadr;
    printf("Lado do quadrado: ");
    scanf( "%d", &lado);
    area_quadr = lado * lado;
    printf ("Area do Quadrado = %d ", area_quadr);
    system("PAUSE");
    return 0;
}
```


Exemplo

➤ Somar TRÊS números;



Exercício (RETOMANDO)



programa CIRCUNF

declarações

início

leia (R)

AREA ← πR^2

PERIM ← $2 \pi R$

escreva (AREA, PERIM)

fim