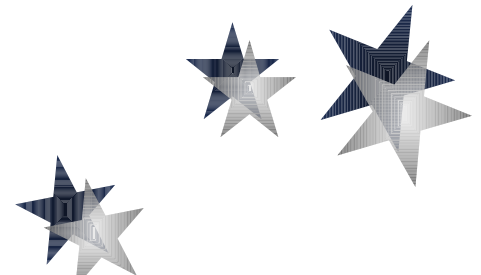


Algoritmos: Conceitos Fundamentais

Instituto de Ciências Matemáticas e de
Computação

Slides elaborados pela
Prof(a). Simone do Rocio Senger de Souza
1o semestre/2012



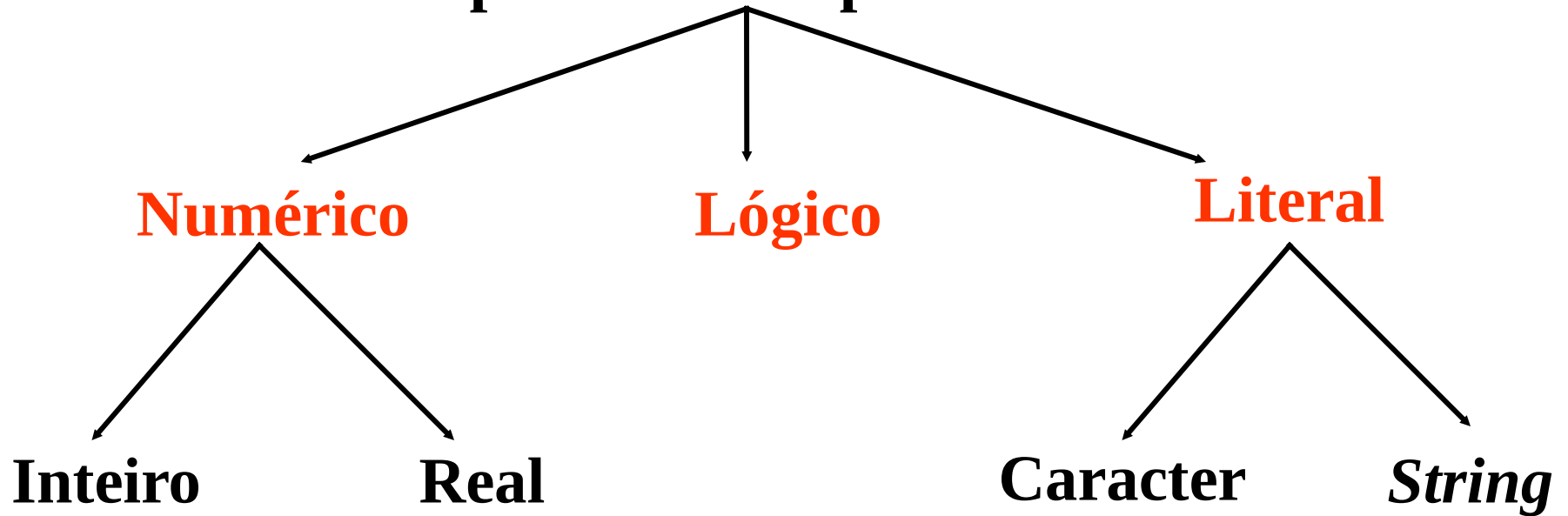


DADOS, INSTRUÇÕES E EXPRESSÕES



TIPO DE DADOS

Tipos de dados primitivos



TIPO DE DADOS - INTEIRO

- Inteiro: Toda e qualquer informação numérica que pertença ao conjunto dos **números inteiros** relativos (negativa, nula ou positiva)
- Exemplos:
 - Ele tem **15** irmãos
 - A escada possui **8** degraus
 - Meu vizinho comprou **2** carros novos
- Linguagem C: tipo **int, long int, unsigned int, ...**

TIPO DE DADOS - REAL

- Números reais muito grandes ou muito pequenos são escritos em forma de ponto flutuante (ou notação científica)

- Exemplo:

+3.14E+07 (3.14×10^7)

5.1E-18 (5.1×10^{-18})

- Linguagem C: tipo **float, double, long double**

TIPO DE DADOS - Literal

- **Literal**: Toda e qualquer informação composta por um conjunto de caracteres alfanuméricos: **numéricos** (0...9), **alfabéticos** (A...Z, a... Z) e **especiais** (por exemplo, #, ?, !, @).
- Exemplos:
 - Constava na prova: *‘Use somente caneta!’*.
 - O parque municipal estava repleto de placas: *‘Não pise na grama’*.
 - O nome do vencedor é *‘Felisberto Laranjeira.’*
- **Linguagem C**: tipo **char, unsigned char**

TIPO DE DADOS - LÓGICO

Lógico: Toda e qualquer informação que pode assumir apenas duas situações (**biestável**)

- Exemplos:
 - A porta pode estar ***aberta*** ou ***fechada***.
 - A lâmpada pode estar ***apagada*** ou ***acesa***.
- Linguagem C: **não tem tipo lógico!!!**

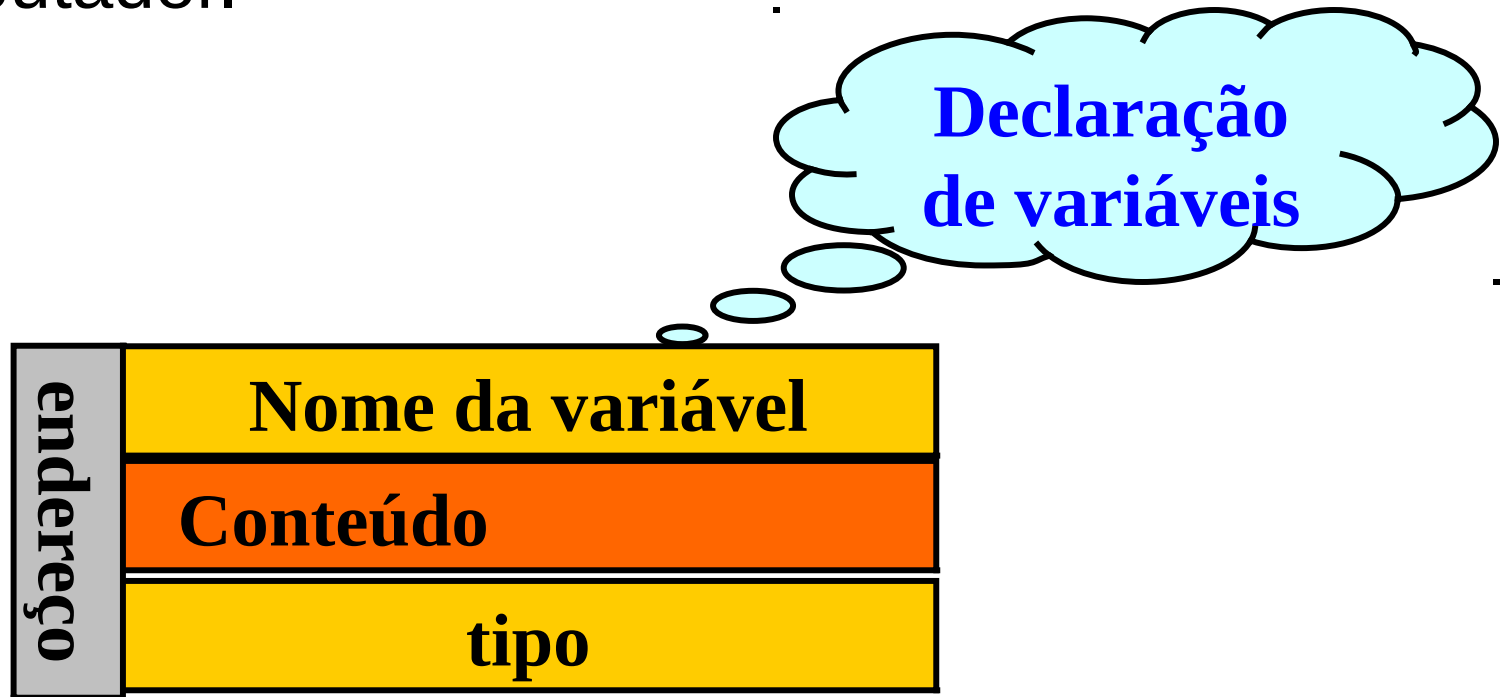
2- TIPO DE DADOS

Valores típicos para a linguagem C (padrão ANSI)

Tipo	Tamanho (bytes)	Intervalo
Char	1	-127 a 127
Unsigned char	1	0 a 255
Int	2	-32.768 a 32.767
Unsigned int	2	0 a 65.535
Long int	4	-2.147.483.647 a 2.147.483.647
Unsigned long int	4	0 a 4.294.967.295
Float	4	Seis dígitos de precisão
Double	8	Dez dígitos de precisão
Long double	10	Dez dígitos de precisão

Variáveis

- Armazenamento das variáveis na memória do computador.



3- Declaração de Variáveis

- Forma de declaração:

Linguagem C:

```
int idade;
```

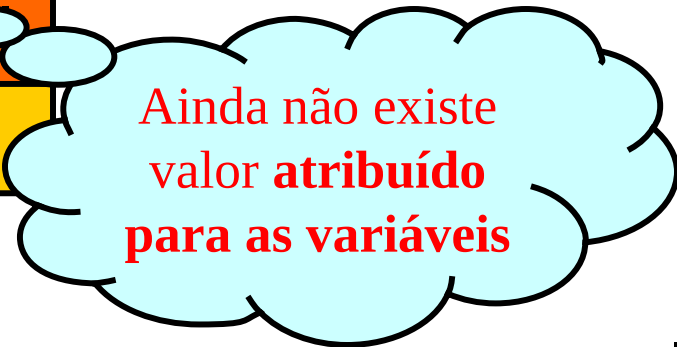
```
float peso;
```

```
char sexo, nome[30];
```

```
char tem-filhos;
```

3- Declaração de Variáveis

- Após a declaração:

44F8	idade	43330	sexo	44322	tem-filhos
	???		???		???
	int		char		char
44F10	peso	44F15	nome		
	???		???		
	float		char[30]		

Ainda não existe
valor **atribuído**
para as variáveis

Declaração de Variáveis

- Nome das variáveis:
 - Devem começar por um caracter **alfabético**;
 - Podem ser seguidos por caracteres **alfabético** e **alfanuméricos**;
 - Não é permitido o uso de **caracteres especiais** (menos o **sublinha**);

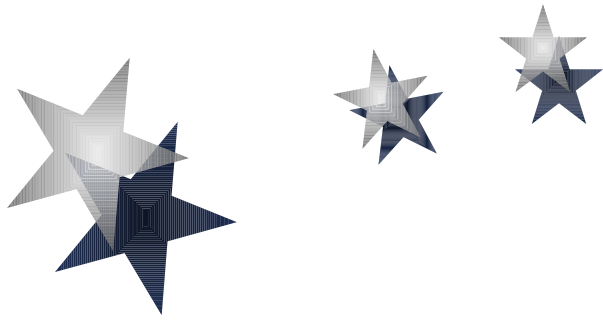
Declaração de Variáveis

- Nome das variáveis:

- Por convenção, os nomes das variáveis utilizam letras **minúsculas** (maiúsculas para nome de **constantes**);
 - Em C existe diferença entre maiúsculo e minúsculo:
 - nome
 - NOME
 - Nome
- Podem ter qualquer tamanho, entretanto, apenas os **31 primeiros caracteres** são utilizados pelo compilador;
- Não é permitido o uso de **palavras reservadas** da linguagem C
 - Exemplo: *for, while, do, if, else, nome de funções existentes...*

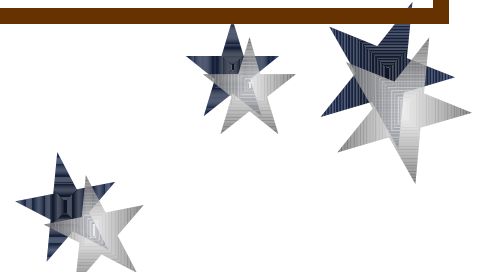


variáveis diferentes!



INSTRUÇÕES

**Comandos que determinam ações que
devem ser realizadas**



1- Instruções Primitivas

- Comando de atribuição
- Comando de entrada
- Comando de saída

1- Comando de Atribuição

Linguagem Algorítmica	C
k = 1; TOTAL = $\sqrt{N} + X^2$ cor = “VERDE” ligado = ‘s’;	k = 1; TOTAL = sqrt (N) + pow(X,2); strcpy(cor, “VERDE”) ; ligado = ‘s’;

1- Comando de Atribuição

Linguagem Algorítmica	
k = 1; TOTAL = $\sqrt{N} + X^2$ cor = "VERDE" ligado = 's';	k = 1; TOTAL = sqrt (N) + pow(X,2); strcpy(cor, "VERDE"); ligado = 's';

constante numérica

constante literal

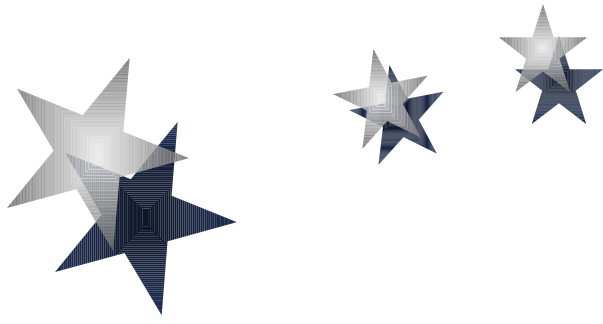
expressão aritmética

2- Comando de Entrada

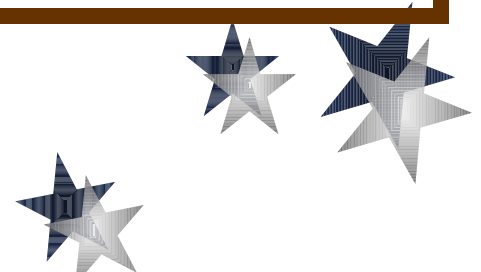
Algoritmo	C
leia(nome); leia(num); leia(salario); leia(sexo); leia(num1, num2);	gets(nome); scanf("%d", &num); scanf("%f", &salario); sexo = getch(); scanf("%d %d", &num1, &num2);

3- Comandos de Saída

Algoritmo	C
escreva(nome); escreva(nome, num); escreva("valor inválido!");	printf("%s", nome); printf("%s %d", nome, num); printf("valor inválido!");



EXPRESSÃO ARITMÉTICA



Expressão Aritmética

OPERADORES NUMÉRICOS

+	adição
-	subtração
*	multiplicação
/	divisão
/	quociente inteiro (para int)
%	resto da divisão

Expressão Aritmética

- Operador **/**:
 - Para integer: parte inteira da divisão
 - Para float: considera a parte decimal
- Operador **%**: resto da divisão
 - Só se aplica para operadores integer
- Exemplo:

30 / 7 = resulta 4

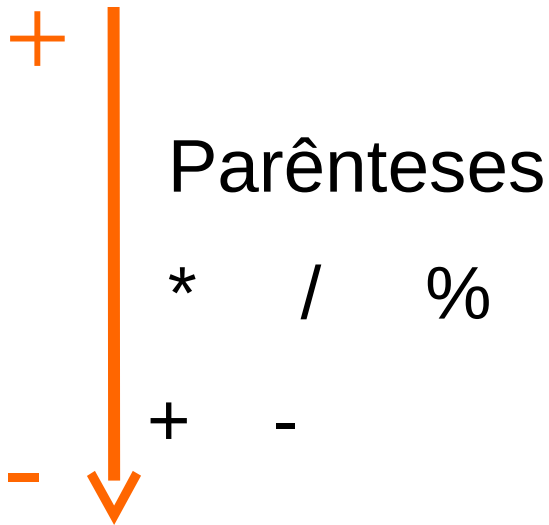
30 % 7 = resulta 2

5 / 7 = resulta 0

5 % 7 = resulta 5

Expressão Aritmética

- Na resolução das expressões aritméticas, as operações guardam uma hierarquia entre si, sendo resolvidos da esquerda para direita se são de igual prioridade:





EXPRESSÃO LÓGICA



Expressão Lógica

- Denomina-se **expressão lógica** aquela cujos operadores são lógicos ou relacionais e cujos operandos são relações ou variáveis ou constantes do tipo lógico.
- Exemplo:

$(A+B == 0) \text{ and } (C \neq 1)$

OPERADORES RELACIONAIS

OPERADORES	==	igual
	!=	diferente
	>	maior
	<	menor
	>=	maior ou igual
	<=	menor ou igual

Expressão Lógica

OPERADORES RELACIONAIS

- Exemplos:

$$\begin{array}{ccc} 2 * 4 & == & 24 / 3 \\ 8 & == & 8 \\ & \text{V} & \end{array}$$

$$\begin{array}{ccc} 2 + 8 \% 7 & >= & 3 * 6 - 15 \\ 2 + 1 & >= & 18 - 15 \\ 3 & >= & 3 \\ & \text{V} & \end{array}$$

Expressão Lógica

OPERADORES LÓGICOS

- Utilizam-se três **conectivos** básicos para a formação de novas proposições lógicas compostas a partir de outras proposições lógicas simples.

OPERADORES LÓGICOS

&&

e

||

ou

!

não

Expressão Lógica

TABELAS VERDADE

Operação de Negação

A	! A
true	false
false	true

Expressão Lógica

TABELAS VERDADE

Operação de Conjunção (e)

A	B	A && B
true	true	true
true	false	false
false	true	false
false	false	false

Exemplo:

A = Tenho dinheiro?

B = Tenho onde ir?

Expressão Lógica

TABELAS VERDADE

Operação de Disjunção Não-Exclusiva (ou)

A	B	$A \parallel B$
true	true	true
true	false	true
false	true	true
false	false	false

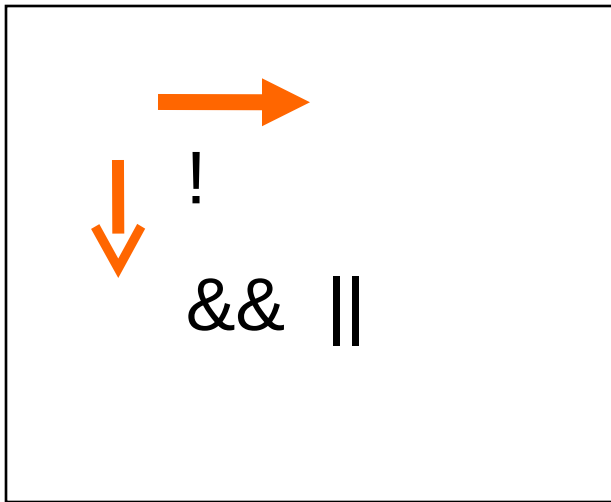
Exemplo:

A = Tenho dinheiro?

B = Tenho onde ir?

OPERADORES LÓGICOS


- Na resolução das expressões lógicas, os operadores lógicos guardam uma hierarquia entre si:



Expressão Lógica

OPERADORES LÓGICOS

- Na resolução das expressões lógicas, os diversos operadores guardam uma hierarquia entre si:



parenteses mais internos
operadores aritméticos
operadores relacionais
operadores lógicos

Expressão Lógica

- Exemplos:

$2^4 \neq 4 + 2 \parallel 2 + 3 * 5 / 3 \% 5 > 0$

$16 \neq 6 \parallel 2 + 15 / 3 \% 5 > 0$

$16 \neq 6 \parallel 2 + 5 \% 5 > 0$

$16 \neq 6 \parallel 2 + 0 > 0$

$16 \neq 6 \parallel 2 > 0$

$V \parallel V$

V

Estrutura Condicional Simples C

if (condição)

<comando> **;**

Estrutura C

a condição deve ser
uma expressão lógica

if (condição)

<comando> ;

O comando só será
executado se a condição
for verdadeira

Estrutura Condicional Simples C

if (condição)

<comando>

se **mais de um comando** deve ser executado quando a **condição** for verdadeira, esses comandos devem ser transformados em um **comando composto**.

Estrutura Condicional Simples C

```
if (condição)  
{
```

```
<comando> ;
```

```
    <comando> ;
```

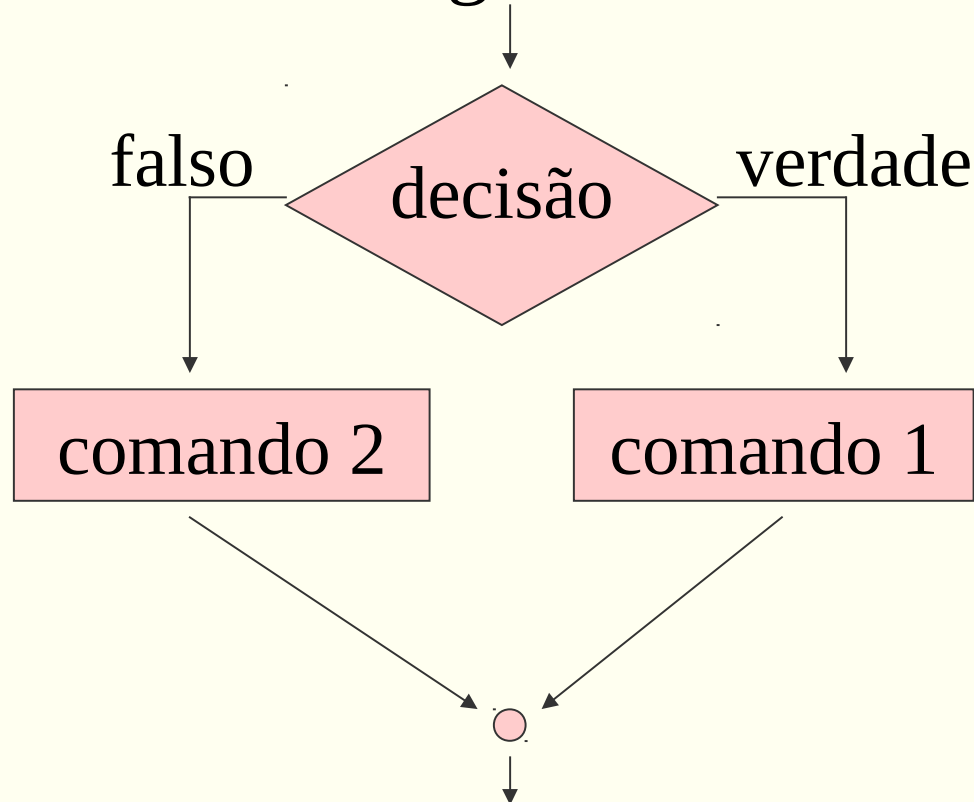
```
    <comando> ;
```

```
}
```

Estrutura Condicional Composta

Formas de Representação no Algoritmo

Fluxograma



Estrutura Condicional Composta C

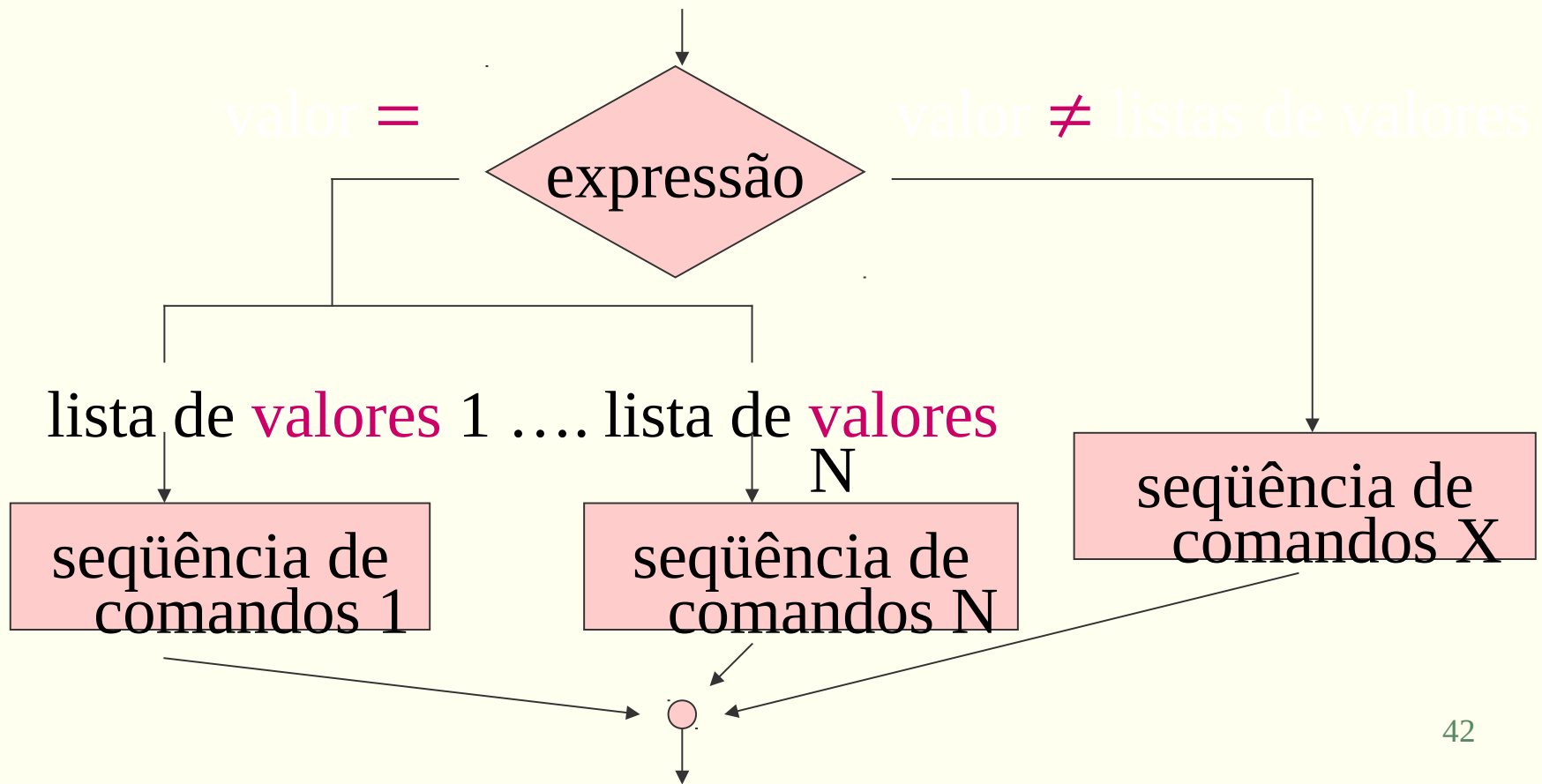
```
if (condição)
    <comando 1> ;
else <comando 2>;
```


Estrutura Condicional Composta C

```
if (condição)
{
    <comando> ;
    <comando> ;
    <comando> ;
}
else
{
    <comando> ;
    <comando> ;
}
```

SELEÇÃO ENTRE DUAS OU MAIS SEQÜÊNCIAS DE COMANDOS

Formas de Representação no Algoritmo
Fluxograma



SELEÇÃO ENTRE DUAS OU MAIS SEQÜÊNCIAS DE COMANDOS C

```
        switch (variável) {  
case <constante1> : <seqüência de comandos 1> ; break;  
....  
case <constante N> : <seqüência de comandos N> ; break;  
default seqüência de comandos default;  
}
```

SELEÇÃO ENTRE DIAS OU

A expressão é **avaliada** e então a seqüência associada ao seu **valor** é executada.

```
switch (variável) {  
case <constante1> : <seqüência de comandos 1> ; break;  
....  
case <constante N> : <seqüência de comandos N> ; break;  
default seqüência de comandos default;  
}
```

Se o comando **break** não for utilizado os demais **cases** continuam sendo avaliados.

SELEÇÃO ENTRE DUAS OU MAIS SEQUÊNCIAS DE

COMANDOS C

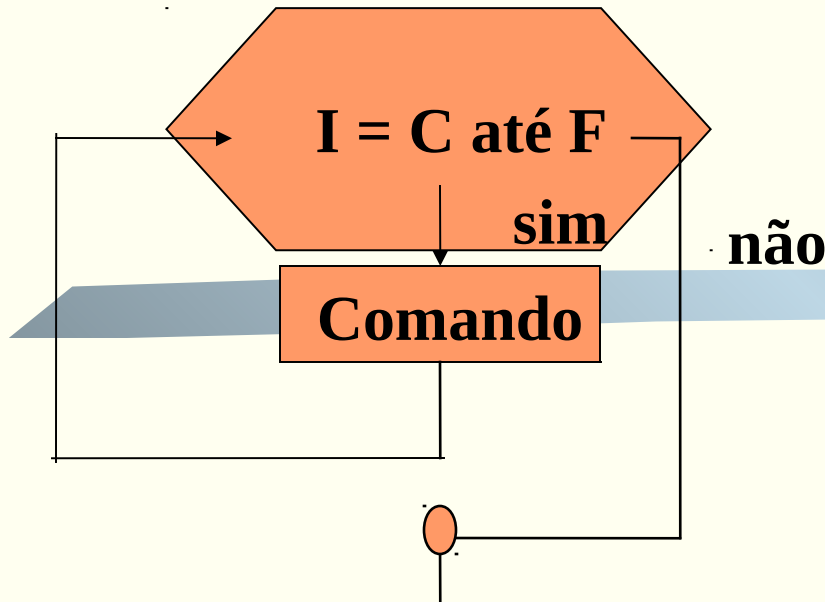
Se o valor da expressão **não** se encontra em nenhuma das listas de valores, a **seqüência default** será executada.

```
switch (variável) {  
case <constante 1> : <seqüência de comandos 1> ; break;  
....  
case <constante N> : <seqüência de comandos N> ; break;  
default seqüência de comandos default;  
}
```

A opção **default** é facultativa

Repetição Contada

Fluxograma




Linguagem C

```
For (i=C; i<=F; i++){  
    comando;  
}
```

Comando for

```
for (inicialização; teste; incremento)  
    instrução;
```

```
for (inicialização; teste; incremento)  
{  
    instrução_1;  
    instrução_2;  
    ...  
    instrução_n;  
}
```

A blue brushstroke underline is positioned beneath the code block, starting from the left margin and extending to the right, with a slightly irregular, hand-drawn appearance.

Comando for

Exemplos:

```
//Usando contador float  
float k;  
for(k = 0; k<=1; k += 0.1)  
    printf("%3.1f ", k);
```

⇒ 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0

```
//Usando contador char  
char letra;
```

```
for(letra = 'A'; letra<='Z'; letra++)  
    printf("%c ", letra);    ⇒ A B C ....X Y Z  
for(letra = 'z'; letra>='a'; letra--)  
    putchar(letra);        ⇒ z y x ....c b a
```


Repetição Contada

Exemplo

Desenvolver algoritmo para o problema:

Dado um conjunto de N números,
calcular a média aritmética dos
mesmos

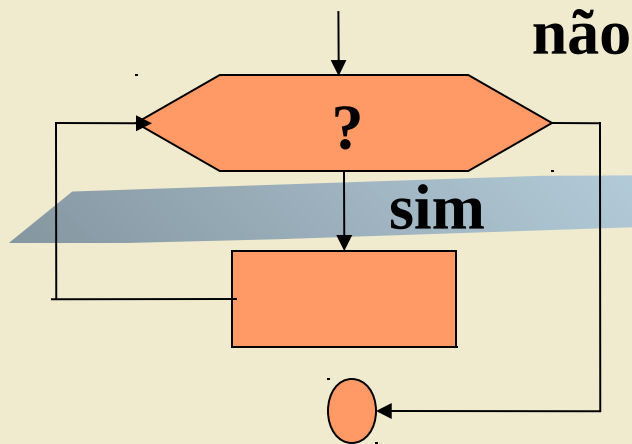
```
int main(){
    int i, n;
    float s=0, x;
    printf("n="); scanf("%d",&n);
    for(i=1; i <= n; i++){
        printf("X="); scanf("%f",&x);
        s = s + x;
    }
    printf("média=%.2f",s/n);
    return 0;
}
```

quantidade de notas

nota

Repetição com Teste no Início

Fluxograma



Linguagem C

```
while (condição){  
    comando;  
}
```

Comando while

```
while (expressão de teste)  
instrução;
```

```
while (expressão de teste)  
{  
    instrução_1;  
    instrução_2;  
    ...  
    instrução_n;  
}
```

Comando while

- Há equivalência entre comandos for e while

```
for (expr1; expr2; expr3)  
    instrução1;  
Instrução2;
```

```
expr1;  
while(expr2){  
    instrução1;  
    expr3;  
}  
Instrução2;
```

Repetição com Teste no Início

Exemplo

```
int main(){
    char sexo;
    printf("sexo:"); scanf("%c",&sexo);
    while((sexo!='F')&&(sexo!='f')&&(sexo!='M')&&(sexo!='m')){
        printf("erro");
        printf("sexo:"); scanf("%c",&sexo);
    }
    printf("sexo:%c",sexo);
    return 0;
}
```

Repetição com Teste no Início

Esta variável de controle deve ter um valor conhecido
Neste caso ela foi lida antes

```
int main(){  
    char sexo;  
    printf("sexo:"); scanf("%c",&sexo);
```

laço {

```
while((sexo!='F')&&(sexo!='f')&&(sexo!='M')&&(sexo!='m')){  
    printf("erro");  
    printf("sexo:"); scanf("%c",&sexo);  
}
```

```
printf("sexo:%c",sexo);  
return 0;  
}
```

A variável de controle deve ter seu valor modificado dentro do “laço”.
Neste caso a variável foi lida novamente

Repetição com Teste no Início

Exemplo

Desenvolver programa para resolver o problema:

Ler N números reais (o valor de N também deve ser lido), sendo impressa a média desses números.


```
int main(){
    int i,j,n;
    float num, media;
    printf("n:");scanf("%d",&n);
    media = 0;
    i = 1;
    while(i ≤ n){
        printf("num:");
        scanf("%d",&num);
        media = media + num;
        i = i + 1;
    }
    media = media / n;
    printf("media:%.2f",media);
    return 0;
}
```

```
int main(){
    int i,j,n;
    float num, media;
    printf("n:");
    media = 0;
    i = 1;
    while(i ≤ n){
        printf("num:");
        scanf("%d",&num);
        media = media + num;
        i = i + 1;
    }
    media = media / n;
    printf("media:%.2f",media);
    return 0;
}
```

i é um contador que determina a parada

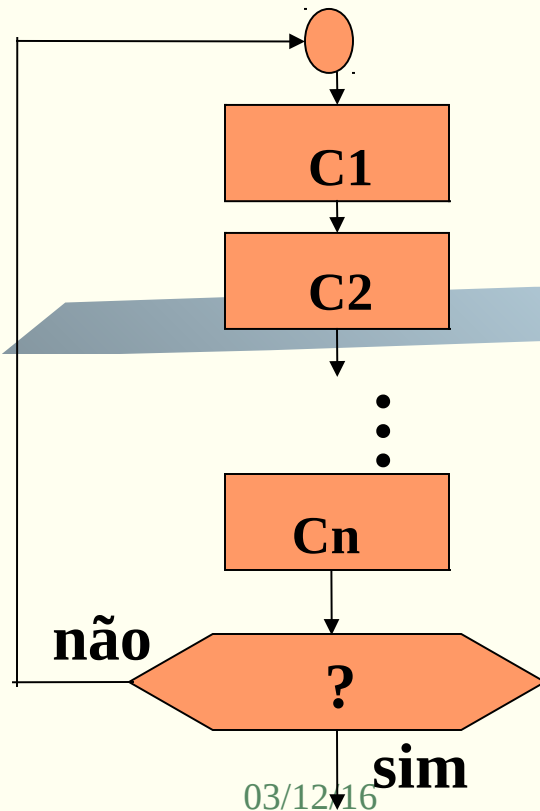
i é declarado como inteiro

i é inicializado

i é incrementado de 1

Repetição com Teste no Final

Fluxograma

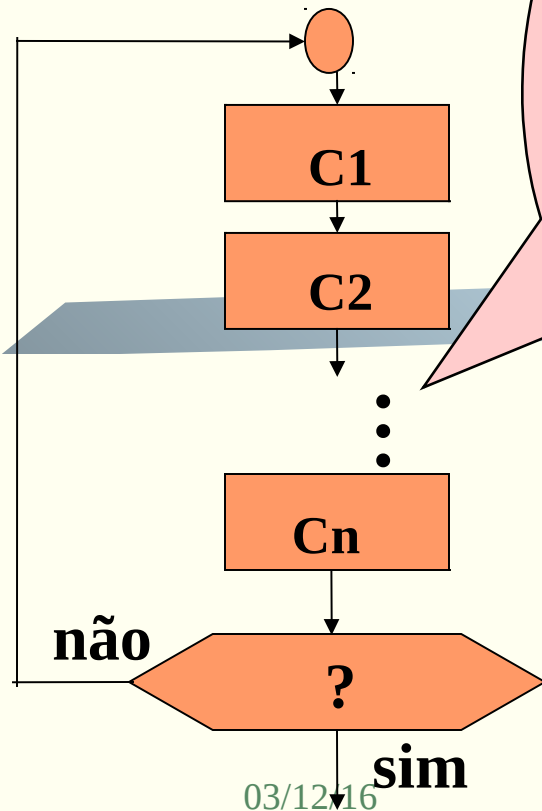


Linguagem C

```
do{  
    comando 1;  
    comando 2;  
  
    comando n;  
while(<condição>);
```

Repetição com Teste no Final

Fluxograma



Os comandos
são executados
pelo menos
uma vez

Linguagem C

```
do{
```

```
    comando 1;
```

```
    comando 2;
```

```
    comando n;
```

```
while(<condição>);
```

Comando do-while

```
do  
{  
    instrução;  
} while(expressão de teste);
```

- Permite executar o bloco “do-while” mesmo se o teste for falso no início.
- Normalmente pouco utilizado.

Comando do-while

Exemplo 1:

//Soma os valores inteiros até receber valor 0

```
int i=0;
```

```
int sum=0;
```

```
do{
```

```
    sum += i;
```

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

```
}while (i>0);
```

```
printf("Somatorio eh: %d", sum);
```

Comando do-while

Exemplo : Código realizado com o comando WHILE.

```
char sair;
```

```
//Pedindo pra sair em um while
```

```
printf("\nPede pra sair!!! (digite S):");
```

```
sair = toupper(getchar());
```

```
while(sair!='S'){
```

```
    printf("\nPede pra sair!!! (digite S):");
```

```
    sair = toupper(getchar());
```

```
}
```

```
printf("\nSaiu!!!");
```

Comando break

- Utilizado no corpo de qualquer estrutura de laço.
- Causa a saída imediata do laço, desviando o programa para a próxima instrução após o laço atual.
- Se estiver em laços aninhados, o **break** afetará somente o laço que o contém e seus laços internos.
- No caso de laços como **for**, **while** e **do-while**, o laço é interrompido e os comandos do programa são retomados a partir da primeira linha fora do laço.
- No caso do **switch**, a execução da sequência de comandos é interrompida.

Comando break

Exemplo:

```
int x, sum;  
sum=0;  
while(1){  
    scanf("%f",&x);  
    if(x<0.0)  
        break;  
    sum += sqrt(x);  
}  
printf("sum= %f\n", sum);
```

Comando continue

- Força a execução da próxima iteração do laço, não executando o código que vem a seguir.
- Esse comando ocorre apenas nos comandos “for”, “while” e “do-while”.
- Esse comando deve ser evitado, pois dificulta a leitura e manutenção do código.

Comando continue

No caso do “while” e “do-while” a execução é desviada para o teste condicional e depois segue para o corpo do laço.

Interrompe a sequência de execuções dentro do laço,
Verifica a condição.

No caso do “for”, o desvio é feito para o incremento, seguido pelo teste condicional e corpo do laço.

Interrompe a sequência de execuções dentro do laço,
Incrementa ou decrementa o contador,
Verifica a condição.

Comando continue

Exemplo:

```
int main(){
    int valor;
    for(valor=1; valor<=10; valor++){
        if(valor%2)
            continue;
        printf("%d ", valor);
    }
    return 0;
}
```