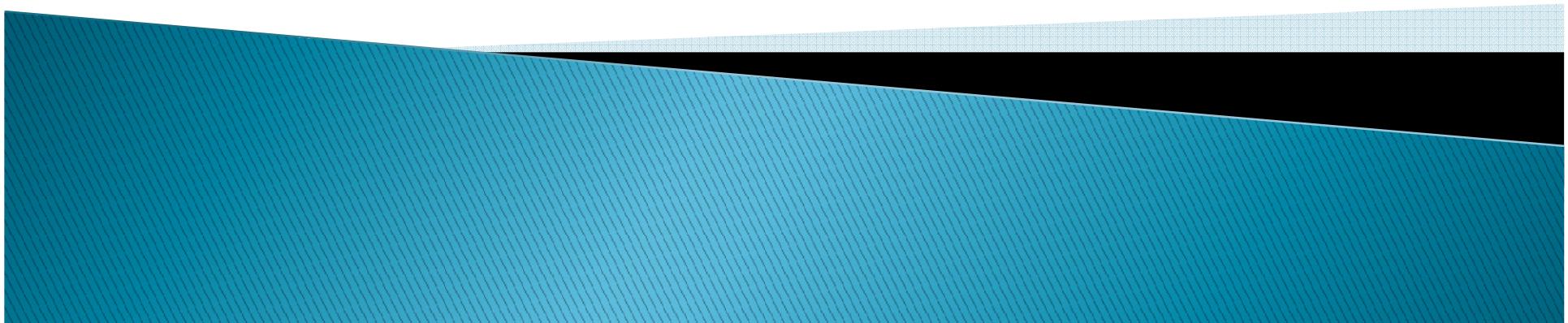


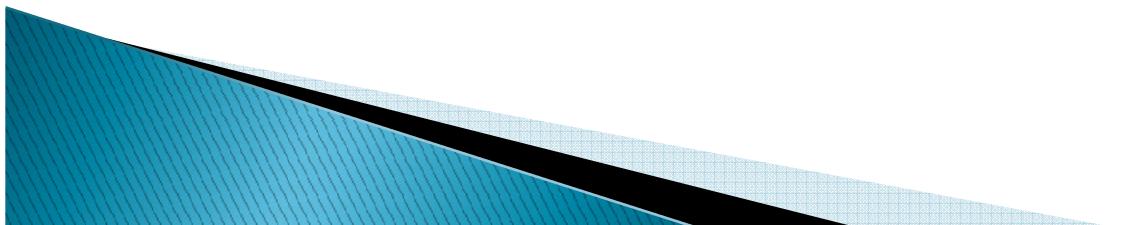
Instituto Federal da Bahia-IFBa
Curso: Engenharia Elétrica
Curso da Linguagem C
Prof. Luiz Cláudio Machado



Linguagem C

- ▶ Função chamada main, que significa principal.
- ▶ main () ← função principal programa C
- ▶ { ← inicia o corpo da função (BEGIN)

- ▶ } ← termina a função (END)



Comentários em C

Os comentários podem ocupar uma ou várias linhas, devendo ser inseridos nos programas utilizando os símbolos /* */ ou //.

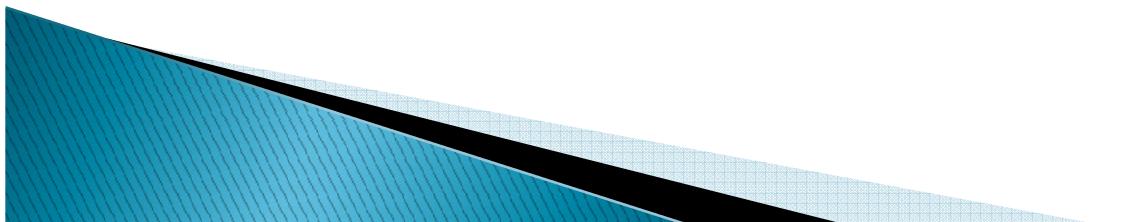
Exemplo:

```
/*
linhas de comentário
linhas de comentário
*/
```

A região de comentários é aberta com os símbolos /* e é encerrada com os símbolos */.

```
// comentário
```

A região de comentários é aberta pelos símbolos // e é encerrada automaticamente ao final da linha.

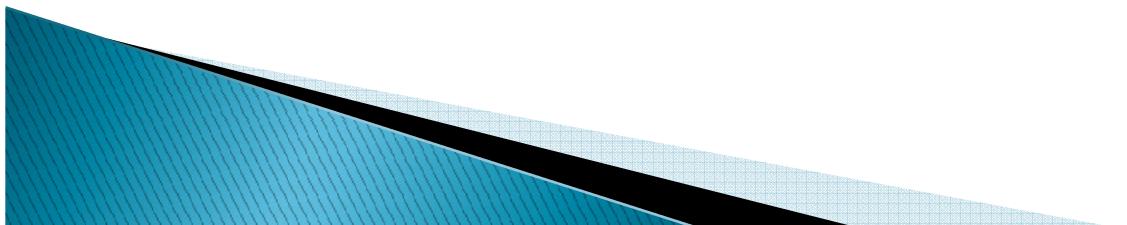


Comando de Atribuição

O *comando de atribuição* é utilizado para atribuir valores ou operações a variáveis, sendo representado por = (sinal de igualdade).

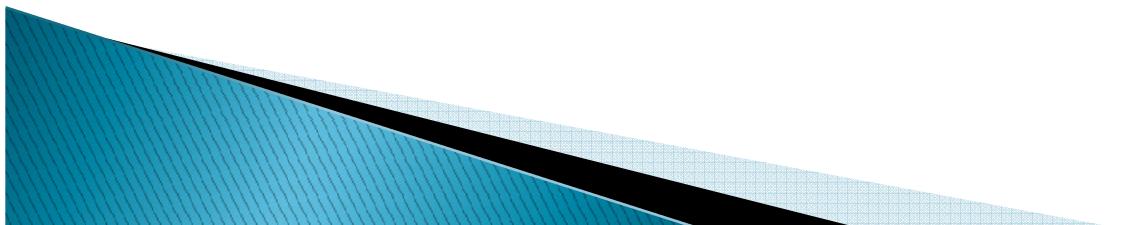
Exemplo:

```
x = 4;  
x = x + 2;  
y = 2.5;  
sexo = 'F';
```



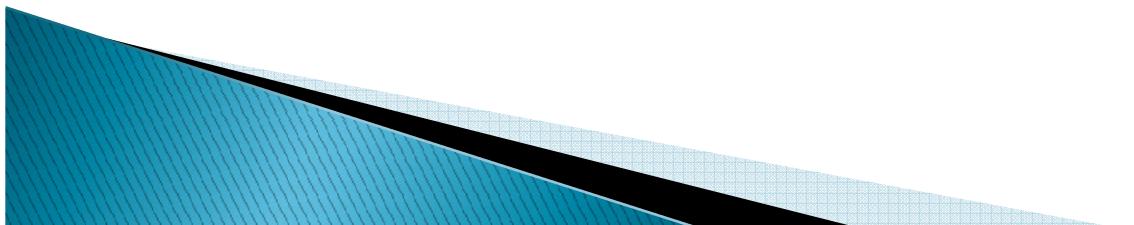
Operadores Matemáticos

OPERADOR	EXEMPLO	COMENTÁRIO
+	$x + y$	Soma o conteúdo de x e de y .
-	$x - y$	Subtrai o conteúdo de y do conteúdo de x .
*	$x * y$	Multiplica o conteúdo de x pelo conteúdo de y .
/	x / y	Obtém o quociente da divisão de x por y .
%	$x \% y$	Obtém o resto da divisão de x por y .
++	$x++$	Aumenta o conteúdo de x em uma unidade.
--	$x--$	Diminui o conteúdo de x em uma unidade.



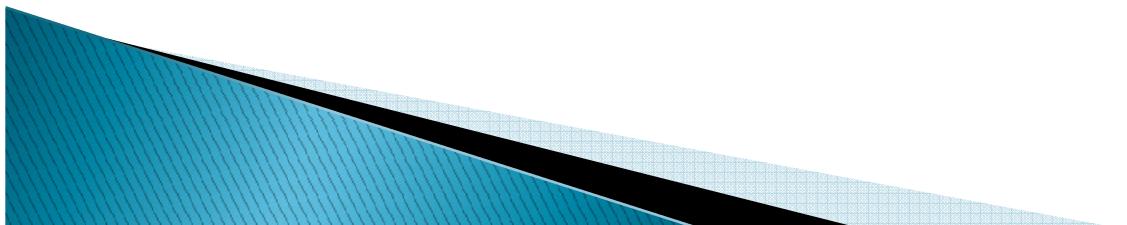
Operadores Matemáticos de Atribuição

OPERADOR	EXEMPLO	COMENTÁRIO
<code>+=</code>	<code>x += y</code>	Equivale a <code>x = x + y</code>
<code>-=</code>	<code>x -= y</code>	Equivale a <code>x = x - y</code>
<code>*=</code>	<code>x *= y</code>	Equivale a <code>x = x * y</code>
<code>/=</code>	<code>x /= y</code>	Equivale a <code>x = x / y</code>
<code>%=</code>	<code>x %= y</code>	Equivale a <code>x = x % y</code>



Operadores Relacionais

OPERADORES RELACIONAIS		
OPERADOR	EXEMPLO	COMENTÁRIO
<code>==</code>	<code>x == y</code>	O conteúdo de x é igual ao conteúdo de y.
<code>!=</code>	<code>x != y</code>	O conteúdo de x é diferente do conteúdo y.
<code><=</code>	<code>x <= y</code>	O conteúdo de x é menor ou igual ao conteúdo de y.
<code>>=</code>	<code>x >= y</code>	O conteúdo de x é maior ou igual ao conteúdo de y.
<code><</code>	<code>x < y</code>	O conteúdo de x é menor que o conteúdo de y.
<code>></code>	<code>x > y</code>	O conteúdo de x é maior que o conteúdo de y.



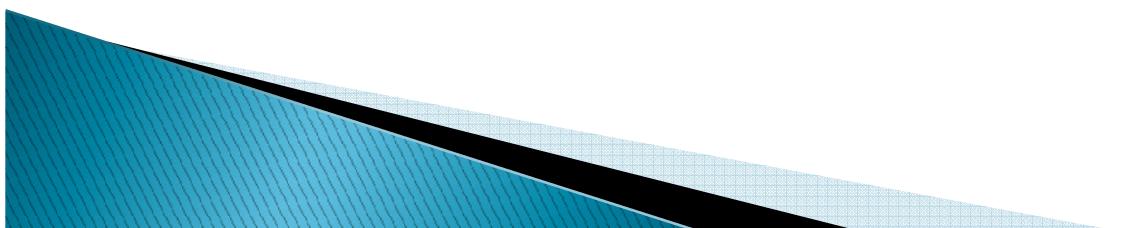
Operadores (Incremento e Decremento)

Operadores (Incremento e Decremento)

PÓS-INCREMENTO E PRÉ-INCREMENTO

```
#include <stdio.h>
main ( )
{
    int a, b;
    a = 2;
    b = a++;
    printf ("%d %d", a, b);
}
```

```
#include <stdio.h>
main ( )
{
    int a, b;
    a = 2;
    b = ++ a;
    printf ("%d %d", a, b);
}
```



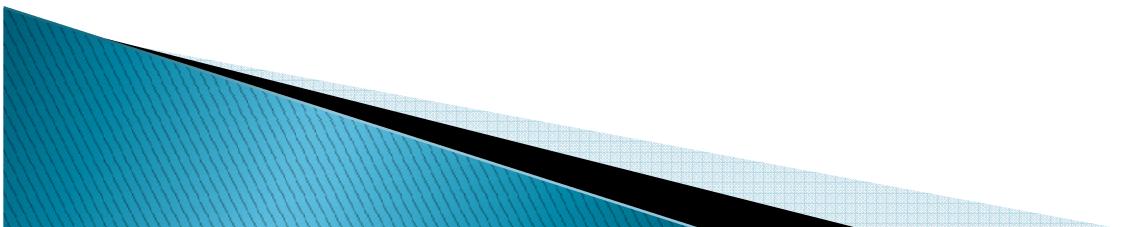
Operadores (Incremento e Decremento)

main()

{

```
int a=1,  
b=3,  
c=4,  
d, e;  
a++;  
d = --b;  
c += a;  
e = a + b * c;  
--e;  
printf("%d %d %d %d %d", a, b, c, d, e);
```

}



Operadores de Bits

- ▶ Não podem ser aplicados às variáveis do tipo float e double. São eles:

&	E bit-a-bit
	OU inclusivo bit-a-bit
^	OU exclusivo bit-a-bit
<<	Deslocamento a esquerda
>>	Deslocamento a direita
~	Complemento unário

Operadores Lógicos

- ▶ Conhecidos como conectivos lógicos de operação, pois objetivam conectar expressões lógicas que, geralmente, são apresentadas através de comandos de decisão. São eles:

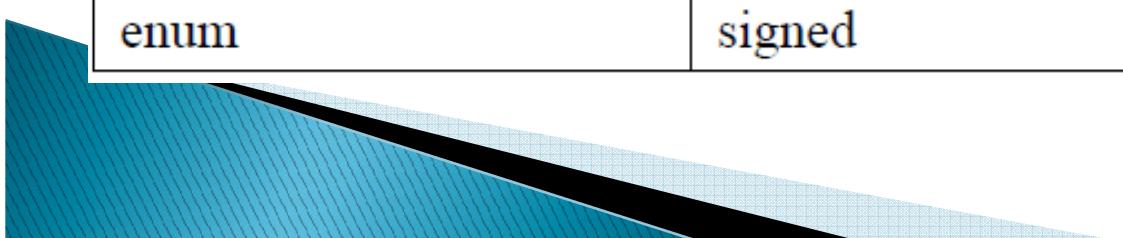
OPERADOR	FUNÇÃO
<code>&&</code>	E lógico
<code> </code>	Ou lógico
<code>!</code>	Não lógico

Funções Matemáticas

FUNÇÃO	EXEMPLO	COMENTÁRIO
ceil	ceil(x)	Arredonda um número real para cima, por exemplo, ceil(3.2) é 4.
cos	cos(x)	Calcula o cosseno de x (x deve estar representado em radianos).
exp	exp(x)	Obtém o logaritmo natural e elevado à potência x.
fabs	fabs(x)	Obtém o valor absoluto de x.
floor	floor(x)	Arredonda um número real para baixo, por exemplo, floor(3.2) é 3.
log	log(x)	Obtém o logaritmo natural de x.
log10	log10(x)	Obtém o logaritmo de base 10 de x.
modf	modf(x, y)	Decompõe um determinado número real em duas partes: x recebe a parte fracionária e y recebe a parte inteira do número.
pow	pow(x, y)	Calcula a potência de x elevado a y.
sin	sen(x)	Calcula o seno de x (x deve estar representado em radianos).
sqrt	sqrt(x)	Calcula a raiz quadrada de x.
tan	tan(x)	Calcula a tangente de x (x deve estar representado em radianos).

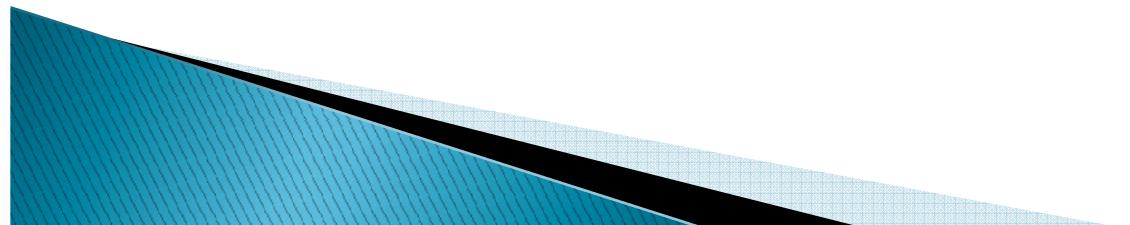
Palavras Reservadas

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



Tipos de variáveis em C

TIPO	BIT	BYTES	ESCALA
char	8	1	-128 a 127
int	16	2	-32768 a 32767
float	32	4	3.4e-38 a 3.4e+38
double	64	8	1.7e-308 a 1.7e+308
void	0	0	sem valor

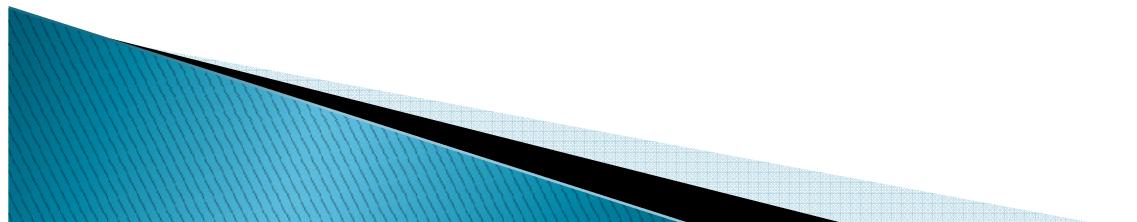


Tipos de variáveis (Válidas e não válidas)

Exemplos de identificadores válidos: Exemplos de identificadores inválidos:

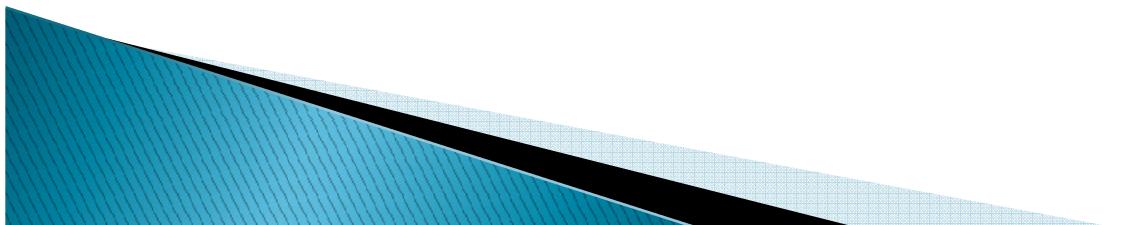
A
a
nota
NOTA
X5
A32
NOTA1
MATRICULA
nota_1
dia
IDADE

5b - por começar por número
e 12 - por conter espaço em branco
x-y - por conter o caractere especial -
prova 2n - por conter espaço em branco
nota(2) - por conter os caracteres especiais ()
case - por ser palavra reservada
SET - por ser palavra reservada



Tipo Inteiro

```
#include <stdio.h>
main ()
{
    int num;
    num = 2;
    printf ("Este é o numero dois: %d", num);
}
```



Tipo Inteiro

```
▶ #include "stdio.h"  
▶ main( )  
{  
    int ano=2013;  
  
    printf("Estamos em %4d.", ano);  
    * A saída será Estamos em 2013. */  
  
    printf("\nEstamos em %6d.", ano);  
    * A saída será Estamos em 2013. */  
  
    printf("\nFeliz %8d.", ano);  
    * A saída será Feliz 2002. */
```



Tipo Inteiro

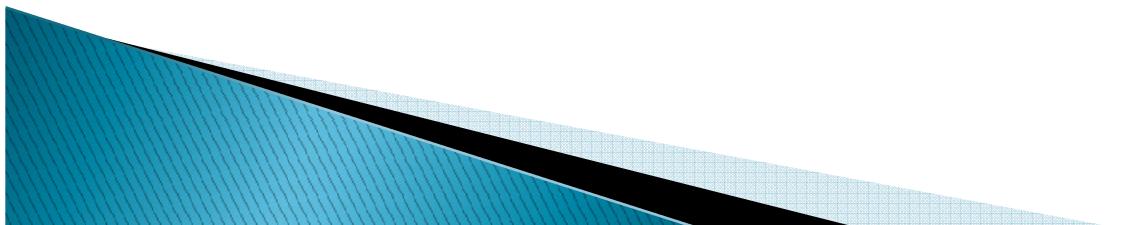
- ▶ main()

- ▶ {

```
    int a=89;  
    printf("%02d", a); /* A saída será 89 */  
    printf("%04u", b); /* A saída será 0095 */  
    printf("%06d", a); /* A saída será 000089 */  
    printf("%08u", b); /* A saída será 00000095
```

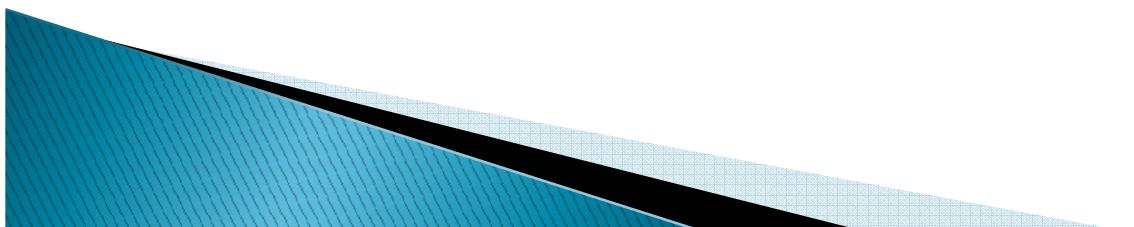
- ▶ */

- ▶ }



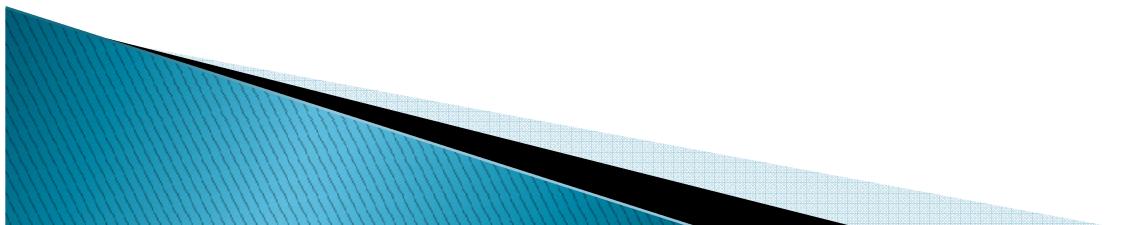
Tipo Real

```
▶ #include <stdio.h>
main ()
{
    float n1, n2;
    n1=6;
    n2=5.5;
}
```



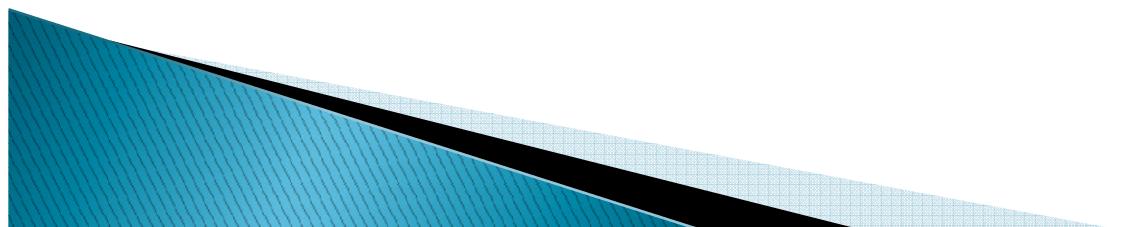
Tipo Real

```
▶ #include "stdio.h"
main( )
{
    double result;
    int num;
    num = 59;
    printf ("O resultado é %f", 3.1415 * num );
}
```



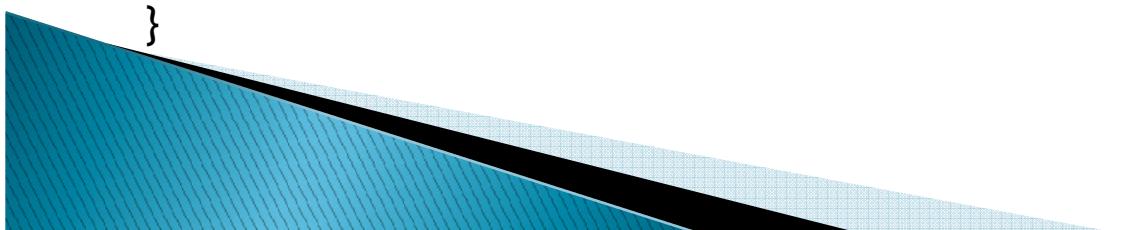
Tipo Caracter

```
▶ void main ()  
{  
    char letra;  
    letra='a';  
    printf ("%c e a primeira letra do  
alfabeto", letra);  
}
```



Tipo Caracter

```
▶ main( )
{
    char nome[30],
        endereço[20],
        cidade[15],
        uf[2],
        tel[13];
    clrscr( );
    puts("Entre com seu nome:");
    puts("Seu endereço:");
    puts("Cidade:");
    puts("UF:");
    puts("Telefone:");
    clrscr( );
    gotoxy(10,5);
    printf("%s %s",nome,tel);
}
```



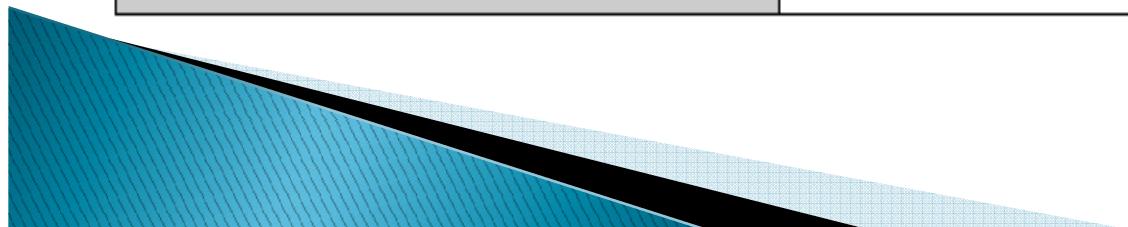
Formatadores da função

CÓDIGO printf ()	FORMATO
%c	APENAS UM CARACTER
%d	DECIMAL INTEIRO
%e	NOTAÇÃO CIENTÍFICA
%f	PONTO FLUTUANTE – float ou double
%g	%e OU %f (O MAIS CURTO)
%o	OCTAL
%s	CADEIA DE CARACTERES – STRING
%u	DECIMAL SEM SINAL
%x	HEXADECIMAL

Tipo	Num de bits	Formato para leitura com scanf	Intervalo	
			Inicio	Fim
char	8	%c	-128	127
unsigned char	8	%c	0	255
signed char	8	%c	-128	127
int	16	%i	-32.768	32.767
unsigned int	16	%u	0	65.535
signed int	16	%i	-32.768	32.767
short int	16	%hi	-32.768	32.767
unsigned short int	16	%hu	0	65.535
signed short int	16	%hi	-32.768	32.767
long int	32	%li	-2.147.483.648	2.147.483.647
signed long int	32	%li	-2.147.483.648	2.147.483.647
unsigned long int	32	%lu	0	4.294.967.295
float	32	%f	3,4E-38	3.4E+38
double	64	%lf	1,7E-308	1,7E+308
long double	80	%Lf	3,4E-4932	3,4E+4932

Caracteres Especiais

CÓDIGOS ESPECIAIS	SIGNIFICADO
\n	NOVA LINHA
\t	TAB
\b	RETROCESSO
\”	ASPAS
\	BARRA INVERSA
\f	SALTA PÁGINA DE FORMULÁRIO
\0	NULO
\x	MOSTRA CARACTER HEXADECIMAL



Comando Entrada de Dados

Exemplo:

```
cin >> X;
```

Um valor digitado pelo usuário será armazenado na variável X.

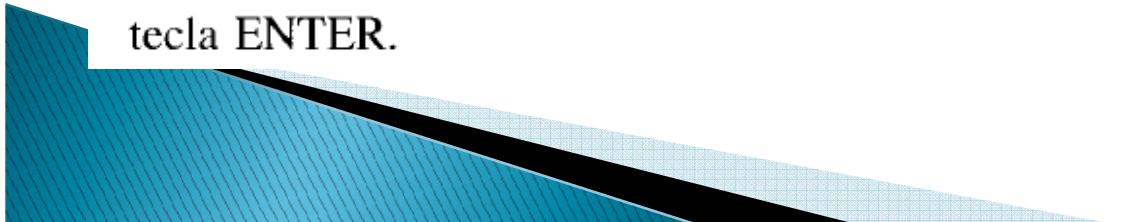
```
gets(NOME);
```

Um ou vários caracteres digitados pelo usuário serão armazenados na variável NOME.

```
scanf(&X);
```

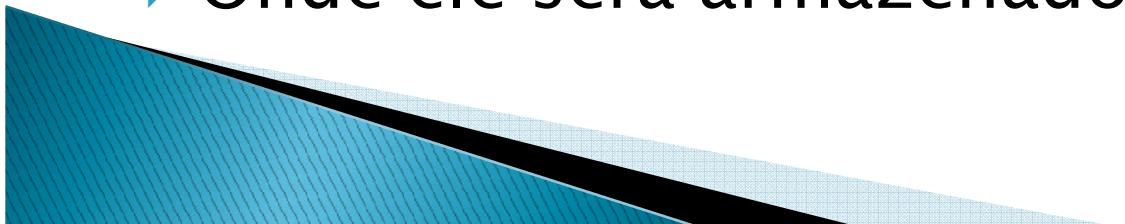
Um valor digitado pelo usuário será armazenado na variável X.

O comando gets deve ser utilizado quando se deseja digitar uma cadeia contendo espaços em branco, por exemplo, um nome completo como João da Silva. O comando cin consegue armazenar os caracteres até que seja encontrado o primeiro espaço em branco e os caracteres posteriores serão desprezados (sendo assim, seria armazenado apenas João). O comando gets e o comando scanf armazenam toda a cadeia até que seja pressionada a tecla ENTER.



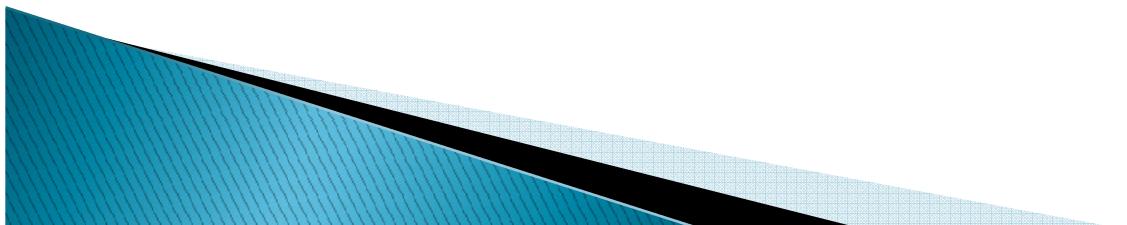
Entrada de Dados

- ▶ **SCANF**, serve como entrada de dados em seu programa.
- ▶ Tudo o que for digitado no teclado será lido por essa função.
- ▶ Para o **scanf** deverá ser informado duas coisas:
 - ▶ Em qual formato o dado será lido?
 - ▶ Onde ele será armazenado?



Entrada de Dados

```
▶ #include <stdio.h>
▶ main()
{
    float a, b, c;
    printf("Digite tres numeros");
    scanf("%f %f %f", &a, &b, &c);
    printf("A media dos numeros %f , %f e %f e igual a %f",
a, b, c, (a + b + c)/3);
}
```



Comando Saída de Dados

O *comando de saída* é utilizado para mostrar dados na tela ou na impressora. Os comandos de saída mais utilizados na linguagem C/C++ são cout e printf.

Exemplo:

```
cout << X;
```

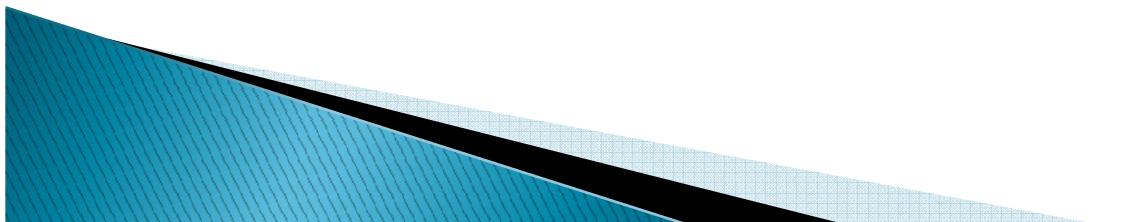
Mostra o valor armazenado na variável X.

```
cout << "Conteúdo de X = " << X;
```

Mostra a mensagem "Conteúdo de X = " e em seguida o valor armazenado na variável X.

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

Mostra o número inteiro armazenado na variável Y.



Comando Saída de Dados

```
printf("Conteúdo de Y = %d", Y);
```

Mostra a mensagem "Conteúdo de Y = " e em seguida o número inteiro armazenado na variável Y.

```
printf("%f", X);
```

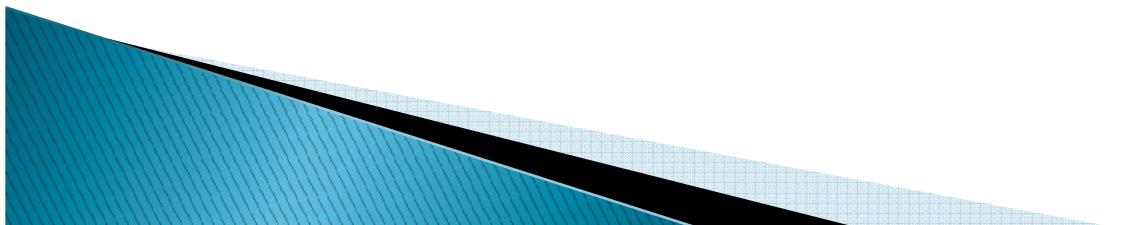
Mostra o número real armazenado na variável x.

```
printf("%5.2f", X);
```

Mostra o número real armazenado na variável x, utilizando cinco casas para a parte inteira e duas casas decimais.

```
printf("Conteúdo de X = %5.2f", X);
```

Mostra a mensagem "Conteúdo de X =" e em seguida o número real armazenado na variável X, utilizando cinco casas para a parte inteira e duas casas decimais.



Saída de Dados

```
/* Representa uma linha de comentários */  
#include <stdio.h>
```

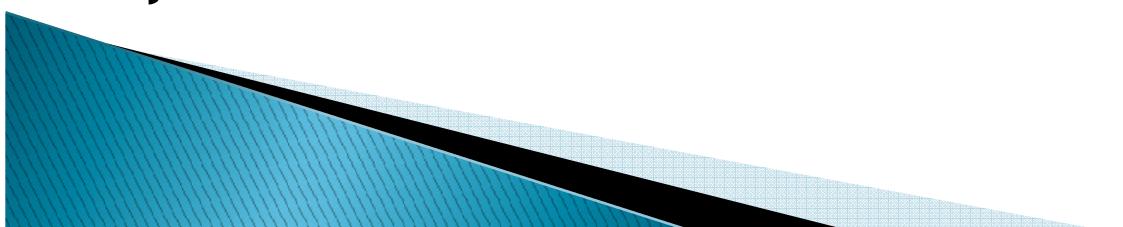
```
/* Início do corpo principal do programa */
```

```
main ()  
{
```

```
/* Função utilizada para saída de dados */
```

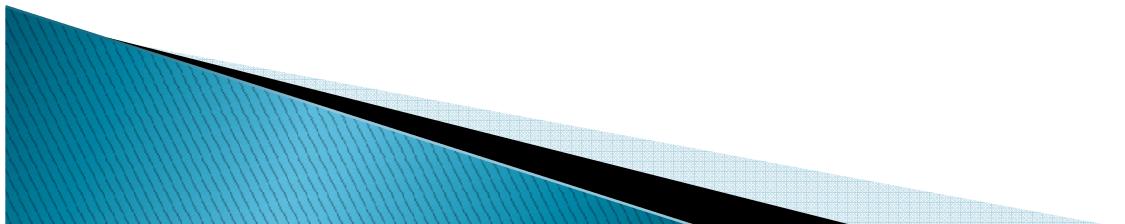
```
printf ("Olá Alunos, estou testando....!");
```

```
}
```



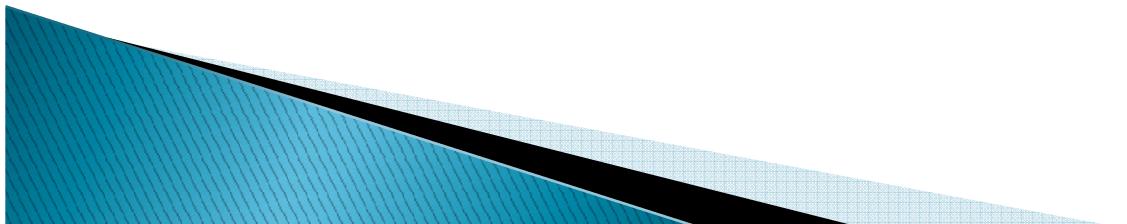
Saída de Dados (Printf)

- ▶ O printf () é uma função utilizada para saída de informações.
- ▶ Sua sintaxe é:
- ▶ printf(Expressao de controle, Lista de argumentos);
- ▶ #include “stdio.h”
- ▶ main ()
- ▶ {
 printf (“Tenho %d anos de idade”, 25);
- ▶ }



Saída de Dados (Printf)

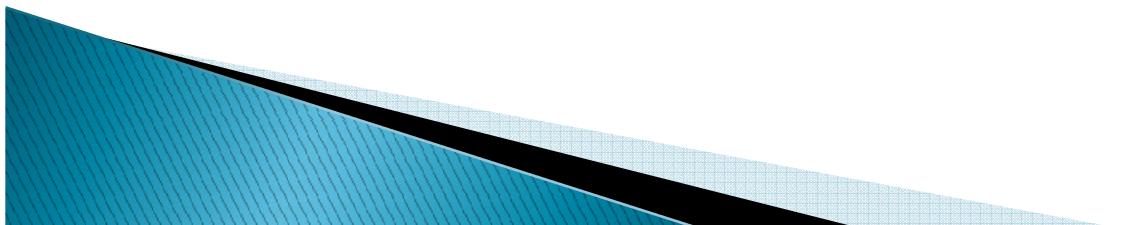
- ▶ A exibição dos resultados do processamento e de mensagens é feita através da função predefinida
- ▶ *printf(), cujo protótipo está contido também no arquivo stdio.h. Sua sintaxe é a seguinte:*



Saída de Dados (Printf)

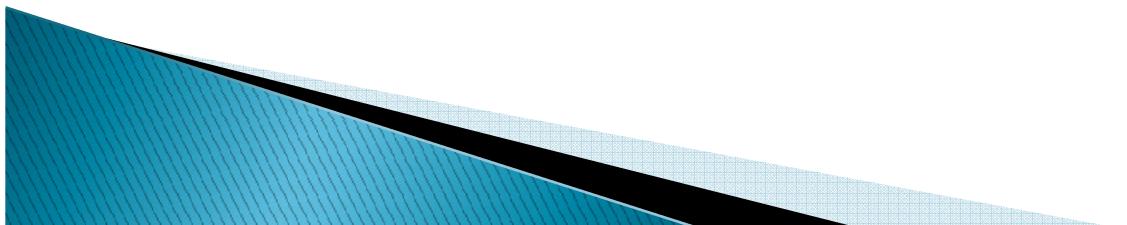
```
▶ #include <stdio.h>

▶ main()
▶ {
    printf("%d", 5 > 3);
▶ }
```



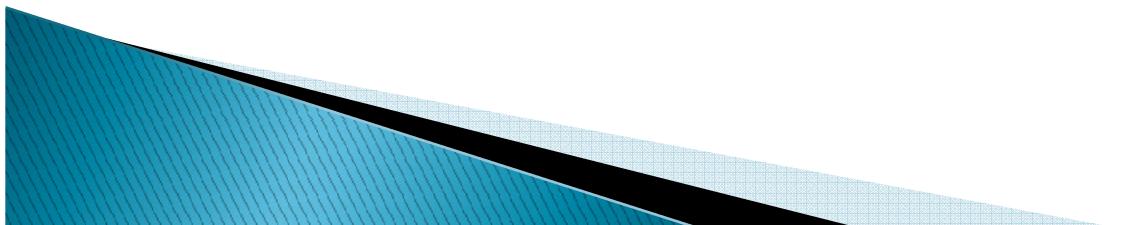
Saída de Dados (Printf)

```
► int main(int argc, char *argv[])
{
    printf ("\n %s tem %d anos de idade \n ", "Luiz",
18);
    system("PAUSE");
    return 0;
}
```



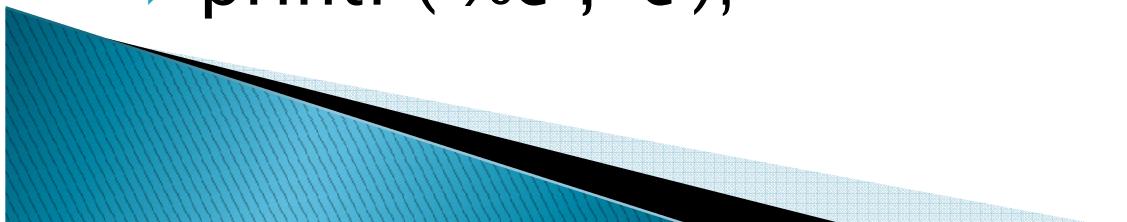
Função puts

- ▶ FUNÇÃO **puts()**
- ▶ Representa a saída de uma única STRING por vez, seguida do caracter de nova linha, ou seja, a função muda de linha automaticamente sem que haja a necessidade do pressionamento da tecla de entrada de dados, ENTER.
- ▶ *Exemplos:*
- ▶ puts (“IFBA”);
- ▶ puts (“BAHIA”);
- ▶ puts (“VITORIA”);



Função putchar

- ▶ FUNÇÃO **putchar ()**
- ▶ Representa a saída de apenas um caracter na tela do computador e não acrescenta uma nova linha automaticamente como a função puts (). As duas instruções seguintes são equivalentes:
 - ▶ putchar ('c');
 - ▶ printf ("%c", 'c');



Referências Bibliográficas

- ▶ **Básica**

SEBESTA, R. W. Conceitos de Linguagens de Programação. 5^a ed. Bookman, 2006.

- ▶ HERBERT SCHILDT. "C Completo e Total". Editora Makron Books, 1997.

- ▶ **Complementar**

- ▶ KERNIGHAN Brian W; RITCHIE, Dennis N. "C: A Linguagem de Programação". Edisa, Editora Campus.
- ▶ KERNIGHAN, B. W.; RITCHIE, D. M. A Linguagem de Programação C ANSI. Elsevier, 1989.
- ▶ VAREJÃO, F. Linguagens de Programação Java, C, C++ e outras. Elsevier, 2004.

