



INSTITUTO FEDERAL DE  
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA  
**PARAÍBA**  
Campus João Pessoa



Tecnologia em Sistemas para Internet

# Linguagem “C”

String

valeriacavalcanti.com.br

**Profa. Valéria Cavalcanti**

valeria.cavalcanti@ifpb.edu.br



# Caracteres

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

int main(void) {
    char c;

    c = 'A';

    printf("%c - %d - %#o - %#X", c, c, c, c);

    // Será impresso: A - 65 - 0101 - 0X41

    getch();
}
```

Diagram illustrating the conversion of the character 'A' to its decimal, octal, and hexadecimal representations using the `printf` function.

The code defines a `char` variable `c` and assigns it the value 'A'. The `printf` function is used to print the character and its representations in decimal, octal, and hexadecimal formats.

The output of the program is: `A - 65 - 0101 - 0X41`.

C realiza a conversão “automática” de **char** para **int** e vice-versa;

Respeitando a tabela ASCII

# Tabela ASCII

Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex	Char	Dec	Oct	Hex
(nul)	0	0000	0x00	(sp)	32	0040	0x20	@	64	0100	0x40	`	96	0140	0x60
(soh)	1	0001	0x01	!	33	0041	0x21	A	65	0101	0x41	a	97	0141	0x61
(stx)	2	0002	0x02	"	34	0042	0x22	B	66	0102	0x42	b	98	0142	0x62
(etx)	3	0003	0x03	#	35	0043	0x23	C	67	0103	0x43	c	99	0143	0x63
(eot)	4	0004	0x04	\$	36	0044	0x24	D	68	0104	0x44	d	100	0144	0x64
(enq)	5	0005	0x05	%	37	0045	0x25	E	69	0105	0x45	e	101	0145	0x65
(ack)	6	0006	0x06	&	38	0046	0x26	F	70	0106	0x46	f	102	0146	0x66
(bel)	7	0007	0x07	'	39	0047	0x27	G	71	0107	0x47	g	103	0147	0x67
(bs)	8	0010	0x08	(	40	0050	0x28	H	72	0110	0x48	h	104	0150	0x68
(ht)	9	0011	0x09	)	41	0051	0x29	I	73	0111	0x49	i	105	0151	0x69
(nl)	10	0012	0x0a	*	42	0052	0x2a	J	74	0112	0x4a	j	106	0152	0x6a
(vt)	11	0013	0x0b	+	43	0053	0x2b	K	75	0113	0x4b	k	107	0153	0x6b
(np)	12	0014	0x0c	,	44	0054	0x2c	L	76	0114	0x4c	l	108	0154	0x6c
(cr)	13	0015	0x0d	-	45	0055	0x2d	M	77	0115	0x4d	m	109	0155	0x6d
(so)	14	0016	0x0e	.	46	0056	0x2e	N	78	0116	0x4e	n	110	0156	0x6e
(si)	15	0017	0x0f	/	47	0057	0x2f	O	79	0117	0x4f	o	111	0157	0x6f
(dle)	16	0020	0x10	0	48	0060	0x30	P	80	0120	0x50	p	112	0160	0x70
(dc1)	17	0021	0x11	1	49	0061	0x31	Q	81	0121	0x51	q	113	0161	0x71
(dc2)	18	0022	0x12	2	50	0062	0x32	R	82	0122	0x52	r	114	0162	0x72
(dc3)	19	0023	0x13	3	51	0063	0x33	S	83	0123	0x53	s	115	0163	0x73
(dc4)	20	0024	0x14	4	52	0064	0x34	T	84	0124	0x54	t	116	0164	0x74
(nak)	21	0025	0x15	5	53	0065	0x35	U	85	0125	0x55	u	117	0165	0x75
(syn)	22	0026	0x16	6	54	0066	0x36	V	86	0126	0x56	v	118	0166	0x76
(etb)	23	0027	0x17	7	55	0067	0x37	W	87	0127	0x57	w	119	0167	0x77
(can)	24	0030	0x18	8	56	0070	0x38	X	88	0130	0x58	x	120	0170	0x78
(em)	25	0031	0x19	9	57	0071	0x39	Y	89	0131	0x59	y	121	0171	0x79
(sub)	26	0032	0x1a	:	58	0072	0x3a	Z	90	0132	0x5a	z	122	0172	0x7a
(esc)	27	0033	0x1b	;	59	0073	0x3b	[	91	0133	0x5b	{	123	0173	0x7b
(fs)	28	0034	0x1c	<	60	0074	0x3c	\	92	0134	0x5c		124	0174	0x7c
(gs)	29	0035	0x1d	=	61	0075	0x3d	]	93	0135	0x5d	}	125	0175	0x7d
(rs)	30	0036	0x1e	>	62	0076	0x3e	^	94	0136	0x5e	~	126	0176	0x7e
(us)	31	0037	0x1f	?	63	0077	0x3f	_	95	0137	0x5f	(del)	127	0177	0x7f

# Funções para Manipular “char”

Função	Descrição
int <b>isdigit</b> (int c)	Verifica se ‘c’ é um número.
int <b>isalpha</b> (int c)	Verifica se ‘c’ é uma letra.
int <b>isalnum</b> (int c)	Verifica se ‘c’ é dígito ou letra.
int <b>isxdigit</b> (int c)	Verifica se ‘c’ é um caractere hexadecimal.
int <b>islower</b> (int c)	Verifica se ‘c’ é uma letra minúscula.
int <b>isupper</b> (int c)	Verifica se ‘c’ é uma letra maiúscula.
int <b>tolower</b> (int c)	Converte ‘c’ para minúscula.
int <b>toupper</b> (int c)	Converte ‘c’ para maiúscula.

# Exemplo: Manipulação de “char”

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

int main() {
    char c;

    printf("Informe um caractere: ");
    c = getchar();

    if (isdigit(c)) puts("Numero");
    if (isalpha(c)) puts("Letra");
    if (isalnum(c)) puts("Numero ou letra");
    if (isxdigit(c)) puts("Caractere hexadecimal");
    if (islower(c)) puts("Letra minuscula");
    if (isupper(c)) puts("Letra maiuscula");

    return 0;
}
```

# Strings em C

- É uma sequência de caracteres, seu término é representado pelo caractere nulo **'\0'**;

<i>l</i>	<i>i</i>	<i>n</i>	<i>d</i>	<i>o</i>	<b>\0</b>
0	1	2	3	4	5

- Vetor de caracteres !!

# Strings em C

## Declarando:

- `char nome[31];`
- `char instituicao[] = "LINDO";`
- `char instituicao[] = {'L', 'I', 'N', 'D', 'O', '\0'};`
- `char cidade[20] = "João Pessoa";`
- `char estado[3] = "Paraíba";` << erro !! >>
- `char teste[] = {"o", "i"};` << erro !! >>
- `char comandos[] = {'\n', 'a', 't', 'i', 'r', 'o'};`

## Atenção:

- Qual a saída de `puts("teste")` e `printf("teste")`?

# Strings em C

## Inicializando:

- nome = "Fulano de Tal"; << erro !! >>
- cidade[0] = 'j';
- estado[0] = "P"; << erro !! >>
- nome[0] = '\0'; << string vazia >>

## Atenção:

- Cuidado com strings maiores do que o espaço reservado, o excedente irá sobrescrever a memória após o vetor.



# Strings: Exercício 1

- Escreva um programa, em C, para ler uma frase do usuário, armazene num vetor de 50 posições, no final exiba a quantidade de dígitos e letras.

# Strings: Exercício 1 (Solução)

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

int main() {
    char frase[51];
    int digitos = 0, letras = 0, i;

    printf("Informe uma frase: ");
    gets(frase);

    for (i = 0; frase[i] != '\0'; ++i) {
        if (isdigit(frase[i])) ++digitos;
        else if (isalpha(frase[i])) ++letras;
    }

    printf("Digitos: %d\nLetras: %d\n", digitos, letras);

    return 0;
}
```

## Strings: Exercício 2

- Escreva um programa, em C, para ler uma frase do usuário depois converta todos os caracteres para maiúsculo.

## Strings: Exercício 2 (Solução)

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

int main() {
    char frase[51];
    int i;

    printf("Informe uma frase: ");
    gets(frase);

    for (i = 0; frase[i] != '\0'; ++i) {
        frase[i] = toupper(frase[i]);
    }

    printf("Frase: %s\n", frase);

    return 0;
}
```

# Biblioteca stdlib.h

## CONVERSÃO

Comando	Descrição	Exemplo
atoi	Converte para int	int i = atoi("15");
atol	Converte para long	long int i = atol("15");
atof	Converte para float	float f = atof("15.2");
strtod	Converte para double	double d = strtod("15.2", NULL);
strtol	Converte para long	long int i = atol("15");

# (stdlib.h) Exercício 1

- Escreva um programa, em C, para ler uma string e a converter em um número inteiro.

# (stdlib.h) Exercício 1: Solução

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

int main() {
    char texto[10];
    int numero;

    printf("Informe um numero: ");
    gets(texto);

    numero = atoi(texto);

    printf("Texto: %s\nNumero: %d\n", texto, numero);

    return 0;
}
```

**Ao contrário ?!**



# string.h

# Biblioteca string.h

Função	Sintaxe	Descrição
strlen	strlen(st)	Tamanho da string

## Exemplo:

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

int main() {
    char frase[31];

    printf("Informe uma frase: ");
    gets(frase);

    printf("Frase '%s', possui %d caracteres.\n", frase, strlen(frase));

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strcpy	strcpy(dest, orig)	Copiar string.

## Exemplo:

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

int main() {
    char frase[31];

    strcpy(frase, "Que bom!");

    printf("Frase: '%s'.\n", frase);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strncpy	strncpy(dest, orig,n)	Copiar 'n' caracteres da string.

## Exemplo:

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

int main(){
    char frase[31];

    strncpy(frase, "Que bom!", 3);
    frase[3] = '\0';

    printf("Frase: '%s'.\n", frase);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strcat	strcat(dest, orig)	Concatenar string.

## Exemplo:

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

int main() {
    char frase[31] = "Funciona!";

    strcat(frase, " Que bom!");

    printf("Frase: '%s'\n", frase);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strncat	strncat(dest, orig, n)	Concatenar caracteres de string.

## Exemplo:

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

int main() {
    char frase[31] = "Funciona!";

    strncat(frase, " Que bom!", 4);

    printf("Frase: '%s'\n", frase);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strupr	strupr(st)	Converter para maiúsculo

## Exemplo:

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

int main() {
    char frase[31] = "Funciona!";

    printf("Frase: '%s'\n", frase);
    printf("Frase: '%s'\n", strupr(frase));
    printf("Frase: '%s'\n", frase);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strlwr	strlwr(st)	Converter para minúsculo

## Exemplo:

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

int main() {
    char frase[31] = "FUNCIONA!";

    printf("Frase: '%s'\n", frase);
    printf("Frase: '%s'\n", strlwr(frase));
    printf("Frase: '%s'\n", frase);

    return 0;
}
```



# Biblioteca string.h

Função	Sintaxe	Descrição
strchr	strchr(st, char)	Localizar primeiro char na string

## Exemplo:

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

int main(){
    char frase[31] = "Funciona!";
    int indice;

    indice = strchr(frase, 'n') - frase;

    printf("Caractere 'n' estah no indice %d.", indice);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strchr	strchr(st, char)	Localizar último char na string

## Exemplo:

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

int main() {
    char frase[31] = "Funciona!";
    int indice;

    indice = strchr(frase, 'n') - frase;

    printf("Caractere 'n' estah no indice %d.", indice);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strstr	strstr(st1, st2)	Localizar string em outra

## Exemplo:

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

int main(){
    char frase[31] = "Funciona!";
    int indice;

    indice = strstr(frase, "io") - frase;

    printf("String 'io' estah no indice %d.", indice);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strrev	strrev(st)	Inverter uma string

## Exemplo:

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

int main() {
    char frase[31] = "Funciona!";

    printf("Frase: %s.\n", frase);
    printf("Frase invertida: %s.\n", strrev(frase));
    printf("Frase: %s.\n", frase);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strset	strset(st, char)	Substituir todos os caracteres

## Exemplo:

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

int main() {
    char frase[31] = "Funciona!";

    strset(frase, 'v');
    printf("Frase: %s.\n", frase);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strcmp	strcmp(st1, st2)	Comparar strings

## Exemplo:

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

int main(){
    char frase1[] = "Funciona!", frase2[] = "funciona!";
    int resultado = strcmp(frase1, frase2);

    if (resultado > 0) printf("%s > %s\n", frase1, frase2);
    else if (resultado < 0) printf("%s < %s\n", frase1, frase2);
    else printf("%s = %s\n", frase1, frase2);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
strncmp	strncmp(st1, st2)	Comparar 'n' caracteres - strings

## Exemplo:

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

int main(){
    char frase1[] = "FuncionO!", frase2[] = "FuncionA!";
    int resultado = strncmp(frase1, frase2, 7);

    if (resultado > 0) printf("%s > %s\n", frase1, frase2);
    else if (resultado < 0) printf("%s < %s\n", frase1, frase2);
    else printf("%s = %s\n", frase1, frase2);

    return 0;
}
```

# Biblioteca string.h

Função	Sintaxe	Descrição
stricmp	stricmp(st1, st2)	Comparar strings, ignorando ↑↓

## Exemplo:

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

int main() {
    char frase1[] = "IGUAIS!", frase2[] = "iguais!";
    int resultado = stricmp(frase1, frase2);

    if (resultado > 0) printf("%s > %s\n", frase1, frase2);
    else if (resultado < 0) printf("%s < %s\n", frase1, frase2);
    else printf("%s = %s\n", frase1, frase2);

    return 0;
}
```