

Comandos de Seleção + Caracteres

Aula 05

Marcos Silvano Almeida
marcossilvano@utfpr.edu.br
Departamento de Computação
UTFPR Campo Mourão

Roteiro

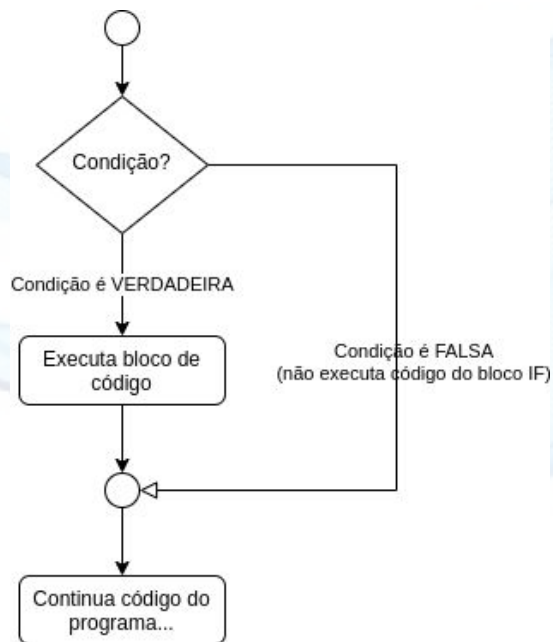
- Comandos de seleção e exemplos
 - IF
 - IF ELSE
 - IF ELSE aninhados
 - Verificando intervalos
 - IF ELSE encadeados
 - SWITCH
- Trabalhando com caracteres
 - Tabela ASCII
 - `int` \Leftrightarrow `char`

Comandos de Seleção

IF

Comando IF

- A estrutura de seleção ou decisão é utilizada para decidir se um bloco de comandos será executado (ou não).
 - O bloco interno é executado somente se a **condição for verdadeira**
 - O código que encontra-se após o condicional, **continuará a ser executado**



Comando IF

```
if (condição) {  
    linha1;  
    linha2;  
    linha3;  
    ...  
}
```

- Os comandos são executados se a condição for verdadeira. Uma condição é uma expressão booleana que resulta em dois valores possíveis: verdadeiro (true/1) ou falso (false/0).
 - Em C/C++ o valor 0 é false e qualquer outro, é considerado true
- Em C/C++, torna-se obrigatória a utilização de bloco (chaves) quando existe mais de um comando a executar.

Comando IF: exemplo

```
#include <stdio.h>

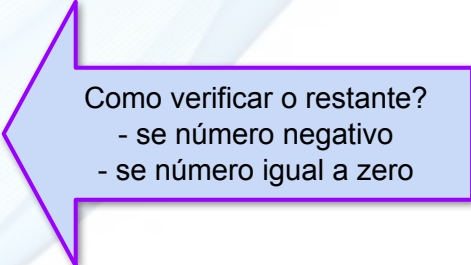
int main () {

    int a;

    printf("Informe um inteiro: ");
    scanf(" %d", &a);

    // verifica condição
    if( a > 0 ) {
        // se condição for verdadeira,
        // executa este bloco de código
        printf("O número informado é positivo\n" );
    }

    // após o bloco do comando IF, continua execução do programa
    printf("Valor: %d\n", a);
    return 0;
}
```



Como verificar o restante?
- se número negativo
- se número igual a zero

Comandos de Seleção IF-ELSE

Comando IF-ELSE

```
if (condição) {  
    // bloco 1/verdadeiro  
    linha1;  
    linha2;  
}  
else {  
    // bloco 2/falso  
    linha3;  
    linha4;  
}
```

```
if (condição) {  
    // bloco 1/verdadeiro  
    linha1;  
    linha2;  
}  
else {  
    // bloco 2/falso  
    linha3;  
    linha4;  
}
```

Um padrão de escrita é colocar o ELSE ao lado do fecho { do IF. Outros preferem colocá-lo na próxima linha, alinhado ao IF a que pertence.

- A estrutura de seleção IF-ELSE provê duas possibilidades de caminhos.
- Ao contrário do IF sozinho, em IF-ELSE um dos caminhos será executado
 - Se a condição for verdadeira, **bloco 1** será executado.
 - Caso contrário (condição é falsa), **bloco 2** será executado.
 - O caminho do ELSE é opcional e o utilizamos quando necessitamos de um caminho de execução alternativo (e contrário) ao caminho do IF.

Exemplo IF-ELSE

```
#include <stdio.h>

int main () {
    int a;

    printf("Informe o valor: ");
    scanf(" %d", &a);

    if( a > 0 ) {
        // se condição for verdadeira, executa este bloco de código
        printf("O número é positivo.\n" );
    } else {
        // se condição for falsa, executa este bloco de código
        printf("O número não é positivo.\n" );
    }

    printf("Valor: %d\n", a); // após o bloco IF-ELSE, execução continua
    return 0;
}
```

Comandos IF-ELSE aninhados

- Podemos adicionar qualquer tipo de código dentro de um dos blocos do comando IF-ELSE, incluindo outro(s) comando(s) IF-ELSE:

```
if (condição1) {  
    // executa se condição1 é TRUE  
    if (condição2) {  
        // executa se condição2 é TRUE  
    }  
} else {  
    // executa se condição1 é FALSE  
    if (condição3) {  
        // executa se condição3 é TRUE  
    } else {  
        // executa se condição3 é FALSE  
    }  
}
```

Exemplo IF-ELSE aninhado: sinal

```
int main () {  
    int a;  
    scanf(" %d", &a);  
    if( a > 0 ) {  
        printf("O número é positivo.\n" ); // primeira condição verdadeira  
    } else {  
        // primeira condição é falsa, aninhamos outro IF-ELSE  
        if (a < 0) {  
            printf("O número é negativo.\n" ); // segunda condição é verdadeira  
        } else {  
            printf("O número é igual a zero.\n" ); // segunda condição é falsa  
        }  
    }  
    printf("Valor: %d\n", a); // após o bloco IF-ELSE, execução continua  
    return 0;  
}
```

Exemplo IF-ELSE aninhado: maior entre dois

```
int main () {  
    int a, b;  
    printf("Informe dois inteiros:\n");  
    scanf(" %d %d", &a, &b);  
  
    if (a == b) {  
        printf("Os numeros são iguais.\n");  
    } else {  
        if (a > b) {  
            printf("Maior: %d\n", a);  
        } else {  
            printf("Maior: %d\n", b);  
        }  
    }  
    return 0;  
}
```

Maior entre 3 valores

Exemplo IF-ELSE aninhado: primeira solução

```
int main() {  
    int a, b, c;  
    printf("Informe tres inteiros: ");  
    scanf(" %d %d %d", &a, &b, &c);  
  
    if (a > b && a > c) {  
        printf("Maior: %d\n", a);  
    }  
    else if (b > c) {  
        printf("Maior: %d\n", b);  
    }  
    else {  
        printf("Maior: %d\n", c);  
    }  
    return 0;  
}
```

Exemplo IF-ELSE aninhado: solução aninhada

```
int main() {  
    int a, b, c;  
    printf("Informe tres inteiros: ");  
    scanf(" %d %d %d", &a, &b, &c);  
    if (a > b) {  
        if (a > c)  
            printf("Maior: %d\n", a);    // a > b > c  
        else  
            printf("Maior: %d\n", b);    // c > a > b  
    } else {  
        if (b > c)  
            printf("Maior: %d\n", b);    // b > a > c  
        else  
            printf("Maior: %d\n", c);    // c > b > a  
    }  
    return 0;  
}
```

Exemplo IF-ELSE aninhado: terceira solução (+simples)

```
int main() {  
    int a, b, c;  
    printf("Informe tres inteiros: ");  
    scanf(" %d %d %d", &a, &b, &c);  
  
    int max = a;  
    if (b > max) {  
        max = b;  
    }  
    if (c > max) {  
        max = c;  
    }  
    printf("Maior: %d\n", max);  
  
    return 0;  
}
```

Utiliza uma variável para guardar o maior valor encontrado até o momento.

Verificando intervalos

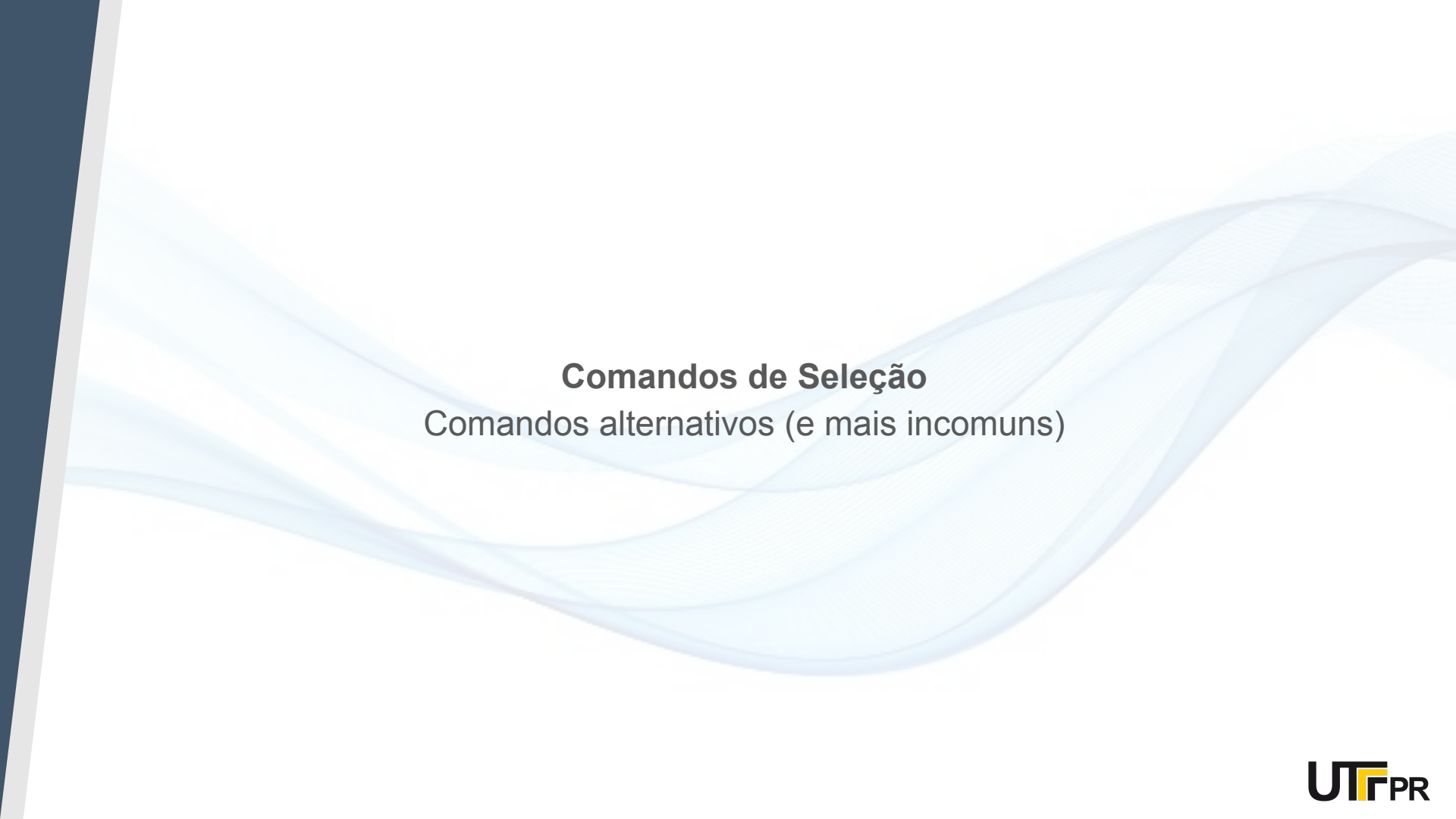
- Suponha que precisemos verificar se um número pertence à alguns intervalos.
 - Para tanto, podemos utilizar expressões compostas.

```
int main() {  
    int num;  
    printf("Informe um número: ");  
    scanf(" %d", &num);  
  
    if ( (num >= 55 && num <= 160) || (num >= 750 && num <= 980) ) {  
        printf("Dentro do intervalo.\n");  
    } else {  
        printf("Fora do intervalo.\n");  
    }  
    return 0;  
}
```

Comandos IF-ELSE “encadeados”

- Quando desejados verificar condições mutuamente exclusivas, isto é, só haverá um caminho para cada condição, podemos encadear vários IF-ELSE.

```
float score;
printf("Informe uma nota (0-10): ");
scanf("%f", &score);
if (score >= 9) {
    printf("Conceito: A\n");
} else if (score >= 8) {
    printf("Conceito: B\n");
} else if (score >= 7) {
    printf("Conceito: C\n");
} else if (score >= 6) {
    printf("Conceito: D\n");
} else {
    printf("Conceito: F\n");
}
```



Comandos de Seleção

Comandos alternativos (e mais incomuns)

Operador Ternário (opcional)

- C possui um operador que permite realizar uma atribuição condicional.
- Bastante limitado e incomum.

```
int value = condição ? retorno_se_verdadeiro : retorno_se_falso;
```

- Exemplo:

```
int value = num > 0 ? 500 : 150;
```

- É o mesmo que:

```
if (num > 0) {  
    value = 500;  
} else {  
    value = 150;  
}
```

Comando SWITCH (opcional)

- Caso queiramos um encadeamento de seletores para **comparar a igualdade um valor**, podemos utilizar o comando SWITCH
 - Bastante limitado... sempre podemos usar IF ELSE-IF no lugar do SWITCH
 - Para verificar expressões, somente IF ELSE-IF

```
int num;
printf("Informe um número (1-5): ");
scanf(" %d", &num);
switch (num) {
    case 1: printf("Um\n"); break; // << não pode faltar o break
    case 2: printf("Dois\n"); break; // para cada caso!
    case 3: printf("Tres\n"); break;
    case 4: printf("Quatro\n"); break;
    case 5: printf("Cinco\n"); break;
    default: printf("ERRO: fora do intervalo (1-5)\n");
}
```



Caracteres

O tipo char e a tabela ASCII

- ASCII é uma tabela de caracteres padronizada
 - Antiga, início dos anos 1960
- Tipo char C \Rightarrow 1 Byte para cada caractere (8 bits)
 - Parte padrão da ASCII \Rightarrow índices 0 à 127
 - Parte estendida \Rightarrow índices 128 à 255 (pode ser trocada)
 - Caracteres imprimíveis \Rightarrow 32 ao 126

- Sugestão para facilitar a vida:
 - Não use acentos!

- Caso queira tentar a sorte:
 - <http://linguagemc.com.br/localizacao-de-programas-com-locale-h/>

Dec	Hex	Oct	Char	Dec	Hex	Oct	Html	Chr	Dec	Hex	Oct	Html	Chr	Dec	Hex	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	&	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	:	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	>	93	5D	135]	}	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

ASCII TABLE

<https://pt.m.wikipedia.org/wiki/Ficheiro:ASCII-Table-wide.svg>

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Caracteres
de controle

imprimível

imprimível

PR

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	 	Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	\$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	((72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051))	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[[123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174	|	
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135]]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177		DEL

Caracteres de controle

imprimível

imprimível

Exemplo char

```
int main() {  
    // Em C, char é um tipo numérico tratado como caractere (símbolo)  
    printf(" %d = %c\n", 65, 65);  
    printf(" %d = %c\n", 'A', 'A');  
    // 32 é a diferença de posições entre maiúscula e minúscula na ASCII  
    int c = 'A' + 32;  
    printf(" %d = %c\n", c, c);  
  
    // Podemos usar operadores relacionais com char (é um número)  
    c = '#';  
    if ( (c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z') ) {  
        printf("Caractere %c(%d) => LETRA\n", c, c);  
    } else {  
        printf("Caractere %c(%d) => NAO LETRA\n", c, c);  
    }  
    return 0;  
}
```

Referências

- Algoritmos e Programação
 - Marcela Gonçalves dos Santos
 - Disponível pelo Moodle
- Estruturas de Dados, Waldemar Celes e José Lucas Rangel
 - PUC-RIO - Curso de Engenharia
 - Disponível pelo Moodle
- Linguagem C, Silvio do Lago Pereira
 - USP - Instituto de Matemática e Estatística
 - Disponível pelo Moodle
- Curso Interativo da Linguagem C
 - <https://www.tutorialspoint.com/cprogramming>