

# MC202 - Estruturas de Dados

Alexandre Xavier Falcão

Instituto de Computação - UNICAMP

[afalcao@ic.unicamp.br](mailto:afalcao@ic.unicamp.br)

# Cadeias de caracteres

- Cadeias de caracteres são variáveis compostas homogêneas do tipo **char** e terminadas pelo caracter `'\0'`.

# Cadeias de caracteres

- Cadeias de caracteres são variáveis compostas homogêneas do tipo **char** e terminadas pelo caracter `'\0'`.
- Alguns comandos já colocam o `'\0'` automaticamente no final da cadeia (e.g., `printf`). Portanto, uma cadeia com  $n$  posições possui  $n - 1$  posições úteis.

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

int main()
{
    char *s1 = "MC202"; // cadeia no heap para leitura apenas
    char s2[6] = {'M','C','2','0','2','\0'}; // na pilha
    char *s3 = (char *)calloc(6,sizeof(char)); // no heap
    char s4[] = "MC202"; // na pilha

    // s3 = "MC202"; /* O mesmo que s3 = s1, não deve ser feito. */
    s3[0]='M'; s3[1]='C'; s3[2]='2'; s3[3]='\0'; s3[4]='2'; s3[5]='\0';

    printf("%s %s %s %s\n",s1,s2,s3,s4);

    return(0);
}
```

- Tabela ASCII.
- Operações úteis com cadeias de caracteres.

# Tabela ASCII

Caracteres e comandos do teclado são codificados em [0,127].

Dec	Char	Dec	Char	Dec	Char	Dec	Char
0	NUL (null)	32	SPACE	64	@	96	`
1	SOH (start of heading)	33	!	65	A	97	a
2	STX (start of text)	34	"	66	B	98	b
3	ETX (end of text)	35	#	67	C	99	c
4	EOT (end of transmission)	36	\$	68	D	100	d
5	ENQ (enquiry)	37	%	69	E	101	e
6	ACK (acknowledge)	38	&	70	F	102	f
7	BEL (bell)	39	'	71	G	103	g
8	BS (backspace)	40	(	72	H	104	h
9	TAB (horizontal tab)	41	)	73	I	105	i
10	LF (NL line feed, new line)	42	*	74	J	106	j
11	VT (vertical tab)	43	+	75	K	107	k
12	FF (NP form feed, new page)	44	,	76	L	108	l
13	CR (carriage return)	45	-	77	M	109	m
14	SO (shift out)	46	.	78	N	110	n
15	SI (shift in)	47	/	79	O	111	o
16	DLE (data link escape)	48	0	80	P	112	p
17	DC1 (device control 1)	49	1	81	Q	113	q
18	DC2 (device control 2)	50	2	82	R	114	r
19	DC3 (device control 3)	51	3	83	S	115	s
20	DC4 (device control 4)	52	4	84	T	116	t
21	NAK (negative acknowledge)	53	5	85	U	117	u
22	SYN (synchronous idle)	54	6	86	V	118	v
23	ETB (end of trans. block)	55	7	87	W	119	w
24	CAN (cancel)	56	8	88	X	120	x
25	EM (end of medium)	57	9	89	Y	121	y
26	SUB (substitute)	58	:	90	Z	122	z
27	ESC (escape)	59	;	91	[	123	{
28	FS (file separator)	60	<	92	\	124	
29	GS (group separator)	61	=	93	]	125	}
30	RS (record separator)	62	>	94	^	126	~
31	US (unit separator)	63	?	95	_	127	DEL

# Tabela ASCII

Isso permite, por exemplo, a identificação de letras minúsculas e maiúsculas nos intervalos [97, 122] e [65, 90], respectivamente, e a conversão para um mesmo padrão de busca usando uma dessas duas opções.

```
char *MinusculasParaMaiusculas(char *cadeia)
{
    long comp_cadeia = ComprimentoCadeia(cadeia);
    char *maiusculas = CriaCadeia(comp_cadeia);

    for (long i=0; i < comp_cadeia; i++){
        if ((cadeia[i]>=97)&&(cadeia[i]<=122)) /* se for letra minúscula */
            maiusculas[i] = cadeia[i]-32; /* converte para maiúscula */
        else
            maiusculas[i] = cadeia[i]; /* apenas copia o caracter */
    }

    return(maiusculas);
}
```

Exemplos das várias operações envolvendo cadeias de caracteres são

- comparação,
- concatenação,
- inversão da ordem dos caracteres,
- separação das partes entre delimitadores,
- busca por um padrão,
- inserção e remoção de um padrão,
- conversões entre números e cadeias numéricas,
- conversões de padrões de caracteres entre somente com letras maiúsculas ou minúsculas, etc.

# Operações com cadeias de caracteres

- A inclusão de `string.h` no programa permite o uso de diversas funções, como `strcmp`, `strcpy`, `strlen`, mas nem sempre essas funções são fáceis de usar e apresentam um comportamento intuitivo.



# Operações com cadeias de caracteres

- A inclusão de `string.h` no programa permite o uso de diversas funções, como `strcmp`, `strcpy`, `strlen`, mas nem sempre essas funções são fáceis de usar e apresentam um comportamento intuitivo.
- Por exemplo, quando duas cadeias são iguais, `strcmp` retorna 0 (falso), quando seria mais intuitivo retornar 1 (verdadeiro).

# Operações com cadeias de caracteres

- A inclusão de `string.h` no programa permite o uso de diversas funções, como `strcmp`, `strcpy`, `strlen`, mas nem sempre essas funções são fáceis de usar e apresentam um comportamento intuitivo.
- Por exemplo, quando duas cadeias são iguais, `strcmp` retorna 0 (falso), quando seria mais intuitivo retornar 1 (verdadeiro).
- A função `strtok` extrai partes (*tokens*) de uma cadeia separadas por um delimitador (e.g., '-'), mas quebra a cadeia de entrada nessas partes, alterando o seu conteúdo dentro da função – **devemos evitar que argumentos de entrada sejam alterados**, a menos que este seja o intuito principal da função.

# Operações com cadeias de caracteres

- A inclusão de `string.h` no programa permite o uso de diversas funções, como `strcmp`, `strcpy`, `strlen`, mas nem sempre essas funções são fáceis de usar e apresentam um comportamento intuitivo.
- Por exemplo, quando duas cadeias são iguais, `strcmp` retorna 0 (falso), quando seria mais intuitivo retornar 1 (verdadeiro).
- A função `strtok` extrai partes (*tokens*) de uma cadeia separadas por um delimitador (e.g., '-'), mas quebra a cadeia de entrada nessas partes, alterando o seu conteúdo dentro da função – **devemos evitar que argumentos de entrada sejam alterados**, a menos que este seja o intuito principal da função.

Vamos continuar a aula preenchendo o código do programa `operacoes-cadeia.c`.