Programação de Computadores

Linguagem C

Cadeia de Caracteres

- Em C uma cadeia de caracteres (string) é implementada como um vetor do tipo char.
- Variáveis do tipo char são usadas para armazenar um caracter (tamanho = 1 byte).
- Caracteres literais são representados por aspas simples:

```
char c1 = ‘a’;
char c2 = ‘A’;
```

Programação de Computadores

Linguagem C

Cadeia de Caracteres

- Variáveis do tipo char podem receber valores literais do tipo caractere ou também valores inteiros
- Que representam o caractere correspondente, conforme o sistema de codificação adotado. **Lembram da tabela ASCII?**
- Assim, variáveis do tipo char podem também ter seu valor comparado com inteiros.

Tabela ASCII

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	[ENG 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]

Programação de Computadores

Linguagem C

Cadeia de Caracteres

- Exemplo:

Crie um programa em C que imprime o código (em decimal) relativo a um caractere digitado pelo usuário.

```
for (int ch = 48; ch <= 57; ch++) {
   printf("%c \t %d \t %x \n", ch, ch, ch);
}
for (int ch = 65; ch <= 122; ++ch) {
   printf("%c \t %d \t %x \n", ch, ch, ch);
}
                                    }
```

Linguagem C

Cadeia de Caracteres

- Cadeias de caracteres são simplesmente arrays/vetores de caracteres que terminam com o caractere ‘\0’:
- O caractere especial ‘\0’ indica o final da cadeia de caracteres
- Note que para armazenar 10 caracteres precisamos de 11 posições. Uma posição adicional para o caractere ‘\0’
- Estas cadeias são também chamadas de strings

```
int main()
{
    char chs[30];
    int i = 0;
    char ch = getchar();
    while(ch != '\n'){
        chs[i++] = ch;
        ch = getchar();
    }
    chs[i]='\0';
    printf("%s\n",chs);
    return 0;
}
```

Exemplos

Suponha um array de 15 caracteres

- `char nome[15];`

?	?	?	?	?	?	?	?	?	?	?	?	?	?	?

- `scanf("%s", nome);`
  (suponha que o usuário digitou Puca)

P	u	c	a	\0	?	?	?	?	?	?	?	?	?	?

Programação de Computadores

Linguagem C

Operações em cadeia de caracteres

A função **strlen()** (abreviação de string length) é utilizada para calcular o tamanho de uma string.

- Sim, calcular o tamanho: a função percorrerá o array de caracteres em busca do caractere ‘\0’.

- Lembre-se, portanto, que há um custo elevado em chamar essa função várias vezes.

Programação de Computadores

Linguagem C

Operações em cadeia de caracteres

A função **strcpy(destino, origem)**

```
#include < stdio.h>                /* diretiva */
#include < string.h>
#define LENGHT 17

main()                    /* função principal */
{

   char origem[LENGHT} =  "Aqui vamos de novo!",
           destino[LENGHT];

    strcpy (destino, "constante string");
    printf ("%s\n", destino);
    strcpy (destino, origem);

    printf ("%s\n", destino);
   }
```

Programação de Computadores

Linguagem C

Operações em cadeia de caracteres

A função **strcmp(string1,string2)**

- A função strcmp devolve um valor inteiro que indica o relacionamento entre string1 e string2:

✓ Um Um valor menor que zero significa que string1 é menor que string2.

✓ valor zero significa que ambas as strings são iguais.

✓ Um valor maior que zero significa que string1 é maior que string2.

Programação de Computadores

Linguagem C

Operações em cadeia de caracteres

A função **strcat(string1,string2)**

- Como resultado desse comando, o valor de string2 é anexado ao final de string1. Uma vez que o valor concatenado é realmente armazenado em string1, essa variável deve estar definida com o tamanho adequado para armazenar a string combinada.

A função **strcasecmp(string1,string2)**

- Ignora o “case sensitive” (maiúsculas e minúsculas)

Programação de Computadores

Linguagem C - strcasecmp(string1,string2)

```
#include<stdio.h>
#include<string.h>
int main()
{
    char str1[] = {'S','T','R','I','N','G'};
    char *str2 = "string";
    int result;
    result = strcasecmp(str1, str2);
    if (result == 0)
        printf("Strings são iguais.\n");
    else if (result < 0)
        printf("\"%s\" é menor que \"%s\".\n", str1, str2);
    else
        printf("\"%s\" é maior que \"%s\".\n", str1, str2);
    return 0;
}
```

Programação de Computadores

Linguagem C

Comandos de Entrada/Saída

- **putchar( )** - Função é usada para exibir um único caractere na saída padrão.

- **puts( )** - Função é usada para imprimir uma sequência de caracteres (string) na saída padrão. Ela adiciona automaticamente uma nova linha após a string exibida.

```c
#include <stdio.h>
int main() {
    char ch = 'A';
    putchar(ch);
    putchar('\n');
    char mensagem[] = "Exemplo de string";
    puts(mensagem);

    return 0;
}
```

Programação de Computadores

Linguagem C

Comandos de Entrada/Saída

- **getchar( )** - Essa função é usada para receber um único caractere de entrada do usuário. Ela retorna o caractere lido como um valor int.

```
char caractere;
printf("Digite um caractere: ");
caractere = getchar();
printf("O caractere digitado foi: %c\n", caractere);
```

Programação de Computadores

Linguagem C

Comandos de Entrada/Saída

- **gets( ) e fgets( )** - Essas funções são usadas para receber uma linha de texto (uma sequência de caracteres) do usuário. A função gets( ) está obsoleta e não é recomendada devido a possíveis problemas de segurança. **É preferível usar a função fgets( ), que é mais segura porque permite especificar o tamanho máximo da entrada a ser lida.**
- **sizeof( )** - Função usada para especificar o tamanho máximo da entrada a ser lida.

```
char palavra[100];
fflush(stdin);//Limpeza do buffer
printf("Digite uma nome:");
scanf("%100[^\n]",palavra);
printf("Nome: %s\n",palavra);
```

```
char nome[50];
printf("Digite o seu nome: ");
fgets(nome, sizeof(nome), stdin);
printf("O seu nome é: %s", nome);
```