LP – Linguagem de Programação I

Manipulação de String

Strings são vetores de **chars**. As strings são o uso mais comum para os vetores. Devemos apenas ficar atentos para o fato de que as strings têm o seu último elemento com um '\0'. A declaração geral para uma string é:

char nome_da_string [tamanho];

Ex.:

char nome[10];

C	R	I	S	T	I	A	N	E	\0

Devemos lembrar que o tamanho da string deve incluir o '\0' final. A biblioteca padrão do C possui diversas funções que manipulam strings.

gets

A função **gets()** lê uma string do teclado. Sua forma geral é:

gets (nome_da_string);

O programa abaixo demonstra o funcionamento da função **gets()**:

```
 #include <stdio.h>

int main ()
{
        char string[100];
        printf ("Digite o seu nome: ");
        gets (string);
        printf ("\n\n Ola %s",string);
        return(0);
}

Obs.: O gets não limita o número de caracteres digitados.
```

Strcmp (comparação de String)

Muitas vezes é necessário realizar a comparação entre duas strings; mas elas não podem ser comparadas diretamente através dos operadores de comparação (símbolo = =).
A função do C, chamada *strcmp(string1, string2)* compara duas strings, e retorna o resultado da comparação, através de um valor. Se este valor for 0 (zero), as duas strings são iguais, caso contrário são diferentes.
Para comparar elementos individuais da string, basta acessá-lo através de seu índice. Lembre-se que cada posição do vetor contém um único valor. Isto é ilustrado pelo exemplo a seguir:

```
... // se o primeiro caracter da string cep for igual a 8 ...
if (cep[0] == '8')  printf("Curitiba");
```

Entende-se por comparação entre strings, sua posição em ordem alfabética. A ordem alfabética é baseada na tabela ASCII. Portanto, cuidado ao comparar maiúsculas com minúsculas, pois na tabela ASCII as letras maiúsculas possuem um valor menor que as letras minúsculas, ou seja, o caractere 'Z' vem antes do caractere 'a'.

```
#include <stdio.h>
#include <string.h>
int main ()
{
        char str1[100],str2[100];
        printf ("Entre com uma string: ");
        gets (str1);
        printf ("\n\nEntre com outra string: ");
        gets (str2);
        if (strcmp(str1,str2)==0)
                printf ("\n\nAs duas strings são iguais.");
       else
                printf ("\n\nAs duas strings são diferentes.");
        return(0);
}
```

strcpy

A função **strcpy()** copia a string-origem para a string- destino.

```
 #include <stdio.h>

#include <string.h>
int main ()
{
        char str1[100],str2[100],str3[100];
        printf ("Entre com uma string: ");
        gets (str1);
        strcpy (str2,str1);    /* Copia str1 em str2 */
        strcpy (str3,"Voce digitou a string "); /* Copia "Voce digitou
a string" em str3 */
        printf ("\n\n%s%s",str3,str2);
        return(0);
}
```

putchar: escreve um caracter na tela.

strncpy

Conforme exemplo abaixo, copia até qtde caracteres da cadeia apontada por orig para a cadeia apontada por dest. Se a cadeia de origem é menor que quantidade de caracteres, caracteres nulos são inseridos em dest até completar a quantidade especificada; se é maior ou igual, apenas as quantidades de caracteres iniciais da origem são copiados. As cadeias não podem estar sobrepostas e a cadeia de origem deve terminar com o caractere nulo, se for menor que a quantidade de caracteres especificada.

```
# include <stdio.h>
# include <string.h>
int main()
{
```

```
   char linha [16];
   char copia [20] = " @@@@@@@@@@@@@@@@@@@@";
   int qtd,i ;
   printf("Informe um texto");
   gets(linha);

   printf (" Qtd caracteres p/ copia : ");
   scanf ("%i", &qtd );
   strncpy (copia , linha , qtd );

   printf (" Origem : |%s|\n", linha );
   printf (" Destino : |");

   for (i = 0; i < 20; i++)
   {
      if ( copia[i] == '\0')
          {
        putchar ( '^');
      }
          else
          {
         putchar ( copia[i]);
      }
    }
    putchar ( '|');
    return 0;

}
```

strcat

A string de origem permanecerá inalterada e será anexada ao fim da string de destino.

```
 #include <stdio.h>

#include <string.h>
int main ()
{
        char str1[100],str2[100];
        printf ("Entre com uma string: ");
        gets (str1);
        strcpy (str2,"Voce digitou a string ");
        strcat (str2,str1);    /* str2 armazenara' Voce digitou a
string + o conteudo de str1 */
        printf ("\n\n%s",str2);
        return(0);
}
```

strlen

A função **strlen()** retorna o comprimento da string fornecida. O terminador nulo não é contado. Isto quer dizer que, de fato, o comprimento do vetor da string deve ser um a mais que o inteiro retornado por **strlen()**. Um exemplo do seu uso:

```
 #include <stdio.h>

#include <string.h>
int main ()
{
        int size;
```

```
        char str[100];
        printf ("Entre com uma string: ");
        gets (str);
        size=strlen (str);
        printf ("\n\nA string que voce digitou tem tamanho %d",size);
        return(0);
}
```

toupper e tolower – Transforma cada caracter da string passada como argumento para maiúsculo e minúsculo respectivamente.

```
#include <stdio.h>
#include <string.h>
int main ()
{
        int size,i;
        char str[21], str1[21], maiusculo[21], minusculo[21];
        printf ("Entre com uma string: ");
        gets (str);

        for (i=0;i<21;i++)
        {
           maiusculo[i]=toupper(str[i]);
           minusculo[i]=tolower(str[i]);
        }

        printf ("\n\nMaiusculo: %s e Minusculo: %s\n",maiusculo,
minusculo);

        return(0);
}
```

Matriz de Caracter

Caso as matrizes sejam de caracteres, isto é equivalente a termos um vetor de strings. Sua inicialização pode se dar da forma:

```
char Nome[5][10] = {"Joao",
                    "Jose",
                    "Maria",
                    "Geraldo",
                    "Lucia"};
```

A matriz será inicializada como

J	o	a	o	\0	\0	\0	\0	\0	\0
J	o	s	e	\0	\0	\0	\0	\0	\0
M	a	r	i	a	\0	\0	\0	\0	\0
G	e	r	a	l	d	o	\0	\0	\0
L	u	c	i	a	\0	\0	\0	\0	\0

Para imprimir um nome, utilizar apenas o índice da linha.
Ex.: nomes[l];