Disciplina: Linguagem de Programação II
Professora: Caroline Mazetto Mendes

Aula 11 - 18/04/17

1) Vetor de Caracteres

Vetores de caracteres são utilizados para armazenar palavras ou textos. Para declarar um vetor de caracteres, deve-se informar o tipo de dados como sendo do tipo char:

char palavra[10]; / Declarando um vetor de char */*

Assim como vetores de números, um vetor de caracter pode ser inicializado durante a sua declaração:

char nome[] = {'M', 'a', 'r', 'i', 'a'};

Para facilitar a inicialização, ao invés de definir cada caracter do texto usando apóstrofo, pode-se inicializar o vetor usando aspas:

char nome[] = "Maria";

Strings

Strings são vetores de caracteres que termiman com '\0' (caracter de terminação). Considere a seguinte declaração:

char palavra[10];

O vetor acima pode armazenar até 9 caracteres, pois a última posição é usada para armazenar o caracter de terminação.

Assim, é necessário declarar o vetor de caracteres adicionando 1 ao tamanho máximo de caracteres esperado. Por exemplo, para armazenar o texto "Linguagem de Programacao", é necessário criar um vetor de tamanho 25 (24 caracteres + 1 para o caracter de terminação).

Ao inicializar uma string usando aspas, não é necessário definir o caracter de terminação, pois o compilador faz isso automaticamente. Por exemplo:

char nome[] = "Maria";

O vetor acima possui tamanho igual a 6, e a última posição do vetor armazena o caracter de terminação.

Imprimindo Strings

Strings podem ser impressas com a função *printf()*, usando o especificador de formato %s. Por exemplo:

```
char nome[] = "Maria";
printf("Seja bem vindo(a) %s\n", nome);
```

O trecho de código acima exibe a mensagem:

Seja bem vinda Maria

A função *puts()* também pode ser utilizada para imprimir uma string. No entanto, essa função apenas imprime uma string e pula linha. Nenhuma opção de formatação pode ser definida. O trecho de código abaixo exibe a mesma mensagem de boas vindas apresentada anteriormente:

```
char nome[] = “Maria”;
printf(“Seja bem vindo(a) ”);
puts(nome);
```

Lendo Strings do teclado

A função *scanf()* pode ser usada para ler uma string do teclado. No entanto, a função obtém apenas a primeira palavra digitada. Por exemplo:

```
char nome[100];

printf(“Entre com o nome: ”);
scanf(“%s”, nome);   /* Nao é necessário usar o & antecedendo o nome de uma string */
printf(“Seja bem vindo(a) %s\n”, nome);
```

Considere que o usuário digitou o nome Maria da Silva. O trecho de código acima exibe a mensagem:

Seja bem vinda Maria

Observe que a função *scanf()* obtém apenas o primeiro nome. Para obter todo o texto informado pode-se utilizar a função *gets()*. Por exemplo:

```
char nome[100];

printf(“Entre com o nome: ”);
gets(nome);
printf(“Seja bem vindo(a) %s\n”, nome);
```

A função *gets()* obtém todo o texto digitado pelo usuário (toda linha até o usuário digitar ENTER). Assim, o trecho de código acima pode exibir a mensagem:

Seja bem vinda Maria da Silva

Obs.: Se o usuário digitar mais caracteres que o tamanho declarado, isso causará um erro de acesso fora dos limites do vetor.

Funções de String

A biblioteca padrão do C disponibiliza algumas funções para manipulação de Strings. Para ter acesso a essas funções, deve-se adicionar a diretiva #include <string.h>. Exemplo de funções:

strlen() - Retorna um número inteiro que é o tamanho da string (o número de caracteres da string, sem contar o caractere de terminação).

Exemplo: Imprimir o comprimento de uma string digitada pelo usuário.

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

int main(void) {
	char nome[100];
	int comprimento;

	printf(“Entre com seu nome:\n”);
	gets(nome);
	comprimento = strlen(nome);
	printf(“Seu nome tem  %d caracteres.\n”, comprimento);
}
```

Código 1. Exemplo de utilização da função *strlen().*

strcmp() - É usada para comparar duas strings. Recebe como parâmetros de entrada duas strings e retorna um número inteiro:

- 0 (zero) se as duas strings são iguais
- Um número negativo se a primeira string é menor que a segunda string
- Um número positivo se a primeira string é maior que a segunda string

Obs.: Lembre-se que strings são vetores, por isso não se pode comparar duas strings usando o operador de igualdade (==). Strings devem ser comparadas caracter por caracter. Nesse caso, use a função *strcmp()*.

Exemplo: Imprimir uma mensagem informando se duas strings são iguais.

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

int main(void) {
	char palavra1[100];
	char palavra2[100];
	int resultado;

	printf(“Entre com a primeira palavra:\n”);
	gets(palavra1);
	printf(“Entre com a segunda palavra:\n”);
	gets(palavra2);

	resultado = strcmp(palavra1, palavra2);

	if (resultado == 0) {
		printf(“As palavras são iguais\n”);
	}
}
```

Código 2. Exemplo de utilização da função *strcmp()*.

strcpy() - É usada para copiar o conteúdo de uma string para outra. Recebe como parâmetros de entrada duas strings. A função copia o conteúdo da segunda string para a primeira string.

Exemplo: Copiar duas strings e imprimir os seus valores.

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

int main(void) {
	char palavra[100],
	char copia[100];

	printf(“Entre com uma palavra:\n”);
	gets(palavra);
	strcpy(copia, palavra);

	printf("Entre com outra palavra:\n”);
	gets(palavra);

	printf("A primeira palavra digitada foi: %s\n", copia);
	printf("A segunda palavra digitada foi: %s\n", palavra);
}
```

Código 3. Exemplo de utilização da função *strcpy()*.

Obs.: Lembre-se que strings são vetores, por isso não se pode copiar duas strings usando o operador de atribuição (=). Strings devem ser copiadas caracter por caracter. Nesse caso, use a função *strcpy()*.

2) Exercícios

- Implementar e testar os Códigos 1, 2 e 3.