Strings

Módulo 7 Aula 4

Linguagem C, o Curso Definitivo WR Kits

Autor: Dr. Eng. Wagner Rambo

Strings são conjuntos de caracteres. Na Linguagem C, uma *string* é um conjunto de caracteres armazenados em um vetor contendo um caractere nulo no final. Não existe um tipo de dado *string* na Linguagem C, logo não podemos atribuir *strings* a variáveis e concatena-las diretamente através de expressões, como soma por exemplo. As *strings* são representadas utilizando aspas entre os caracteres, confira alguns exemplos no Box 1.

```
“Van Halen”
“Beagle”
“40 anos”
“W”
```

Box 1 - Exemplos de Strings.

Lembrando que os caracteres são sempre representados entre apóstrofos (ou aspas simples se preferir). Tomando como exemplo o “W” do Box 1, o mesmo não é um simples caractere. Consiste em uma *string* que ocupa 2 bytes (o próprio W, mais o caractere nulo), afinal está entre aspas. Se considerarmos ‘W’ (entre apóstrofos), este sim é um caractere e ocupa apenas 1 byte na memória.

Como *string* não consiste em um tipo de dado básico no C, precisamos recorrer a vetores para armazenar o conjunto de caracteres. Caso tenhamos um vetor com tamanho 10 e armazenarmos nele apenas “BR” (Box 2), como saber quando a *string* encerrou?

B	R								
v[0]	v[1]	v[2]	v[3]	v[4]	v[5]	v[6]	v[7]	v[8]	v[9]

Box 2 - String BR em um vetor de tamanho 10.

A solução para isso é inserir um caractere nulo representado por \0 ao final da *string*, conforme Box 3.

B	R	\0							
v[0]	v[1]	v[2]	v[3]	v[4]	v[5]	v[6]	v[7]	v[8]	v[9]

Box 3 - String com caractere nulo.

Dessa forma, quando você for declar uma *string* em seu código, necessariamente deverá estar previsto um espaço adicional para o caractere nulo '\0', conhecido também como delimitador. No Box 4 demonstramos a forma de declarar uma *string* para 10 caracteres.

```
char str[11];            /* string para 10 caracteres */
```

Box 4 - Declaração de uma string.

Como podemos ver, se desejamos reservar um espaço para 10 caracteres em nossa *string*, devemos necessariamente prevenir um espaço adicional para o caractere delimitador, por este motivo que a *string* do Box 4 tem tamanho 11 e poderá ser usada para 10 caracteres.

No Box 5, veja 3 formas de realizar a carga inicial de uma *string*.

```
char str1[12] = "WR Kits";
char str2[12] = {'W','R',' ','K','i','t','s'};
char str3[]   = "WR Kits";
```

Box 5 - Três formas de se realizar a carga inicial de strings.

Em str1 e str2 o compilador adiciona o caractere nulo automaticamente ao final da *string* e elas representam a carga inicial de uma *string* em um vetor de tamanho 12. Já str3 consiste em uma *string* onde o compilador calcula a dimensão necessária para inserir WR Kits mais o caractere nulo no final.

No Box 6, confira a diferença entre *string* e um simples vetor de caracteres.

```
char teste1[]   = "abcde";                /* string */
char teste2[10] = {'a','b','c','d','e'};  /* string */
char teste3[]   = {'a','b','c','d','e'};  /* vetor de caracteres */
```

Box 6 - String vs. Vetor de Caracteres.

Não há necessidade de um caractere limitador no teste3, por isso consiste em um vetor de caracteres, onde iremos utilizar os caracteres contidos nele individualmente.

Para imprimir uma *string,* você pode utilizar o formato de leitura %s, conforme Box 7.

```
char hello[25] = "Hello Universe!";

main()
{
  printf("%s", hello);

}
```

Box 7 - Imprimindo uma string completa.

Utilizando a função *scanf*, você pode ler *strings* que contenham uma única palavra, como demonstrado no Box 8.

```
main()
{
  char nome[50],
       sobrenome[50];

  printf("Digite o seu nome: ");
  scanf("%s", nome);
  printf("Digite o sobrenome: ");
  scanf("%s", sobrenome);

}
```

Box 8 - Lendo strings com scanf.

As *strings* lidas por *scanf* não são precedidas do operador &, como acontece com as variáveis. A função *scanf* faz a leitura da palavra e ao encontrar um espaço, tabulação ou enter, insere o caractere limitador de forma automática.

Uma função que permite a leitura de *strings* de mais palavras é a *gets* e o seu uso é demonstrado no Box 9.

```
main()
{
  char rua[50];

  printf("Digite o nome da rua: ");
  gets(rua);

}
```

Box 9 - Lendo string com gets.

Exercício resolvido: desenvolva um projeto que solicite ao usuário a entrada de nomes de países diversos e o programa encerre quando a entrada for uma *string* vazia.

Exercício proposto: desenvolva um programa que solicita ao usuário a entrada do Nome, Sobrenome, Endereço, Bairro, Cidade, CEP e Telefone. Após todas as entradas, os dados completos devem ser impressos na tela.

Bibliografia: DAMAS, Luís; Linguagem C, décima edição.

Disponível em: https://amzn.to/3nGdIbN