VOLTAR

Strings

Apresentar o conceito de strings e aplicar tal conceito utilizando a linguagem C.

NESTE TÓPICO

- Introdução
- Definição
- Declaração da String

Introdução

No cotidiano sempre nos deparamos com frases, textos e palavras, os quais são representados por seqüências (ou cadeias) de caracteres. Como exemplos, podemos citar o envio de mensagens por correios eletrônicos, os editores de textos e os programas de cadastro, onde os dados são representados textualmente.

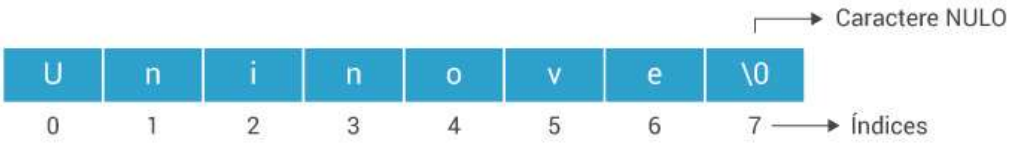
Dentro do contexto de linguagens de programação, seqüências de caracteres são denominadas strings.

A maioria das linguagens de programação permite a representação de cadeias de caracteres. Dentre elas, podemos citar a linguagem Pascal, que possui um tipo de dados específico chamado string para tal finalidade. A Linguagem C não possui este tipo de dados e, por este motivo, é utilizado o conceito de vetores, os quais permitem agupar, em uma só variável, vários caracteres.

Definição

Na Linguagem C, strings são representadas por vetores de caracteres e devem ser terminadas, obrigatoriamente, pelo caractere nulo (\0). Nesse caso, devemos reservar uma adicional para o caractere, que indica o fim de

cadeia. Na Figura abaixo é mostrada a representação gráfica de uma string. Como queremos representar a palavra “Uninove”, composta por 7 caracteres, o vetor deve ter tamanho 8 para considerar o caractere nulo.



Representação gráfica de uma string em C

Declaração da String

Como uma variável simples, uma string deve ser declarada antes de sua utilização. A declaração de uma string é feita indicando o tipo de dado como char, o seu nome e o seu tamanho.

A forma geral de declaração de uma string em linguagem C é:

char nome [tamanho];

Onde, nome é o nome pelo qual a string será referenciada e tamanho é a quantidade de caracteres que a string pode conter.

Exemplos:

- Declaração de uma string denominada palavra de 8 posições de caracteres.

char palavra [8];

Assim como uma variável simples, a string pode ser declarada e inicializada no início de um programa. O trecho de código a seguir mostra a inicialização da string sob 2 formas.

char palavra [8] = {‘U’ , ‘n’, ‘i’, ‘n’, ‘o’, ‘v’, ‘e’, ‘\0’};

ou

char palavra [8] = “Uninove”;

DICA IMPORTANTE

O conteúdo das variáveis do tipo char deve ser referenciado entre apóstrofes (‘ ‘).

O conteúdo das strings deve ser referenciado entre aspas (“ ”).

Referenciando um Caractere da String

Os caracteres de uma string podem ser referenciados de forma individualizada. Não é possível referenciar todos ao mesmo tempo, já que sabemos que a string é um vetor. Um caractere é referenciado pelo nome da string seguido do índice onde ele está armazenado entre colchetes.

O exemplo abaixo referencia o terceiro caractere da string denominada palavra. O terceiro caractere, que no exemplo anterior, é a letra “i”, é referenciado pelo índice 2, já que o primeiro elemento tem índice 0.

palavra [2] = ‘i’;

Lendo e Exibindo Dados para uma String

Imagine que temos uma string denominada frase, que pode armazenar, no máximo, 50 caracteres. Queremos solicitar ao usuário que informe os dados para preencher esse vetor de caracteres. O trecho de código a seguir lê os dados utilizando-se a função scanf.

```
1.  printf ("Digite a String=");
2.  scanf ("%s", frase);
```

Devemos notar que não usamos o caractere & na passagem da variável palavra para a função, pois a cadeia de caracteres é um vetor. O uso do especificador de formato % na leitura é limitado, pois o fragmento de código acima funciona apenas para capturar nomes simples. Para evitarmos este tipo de limitação, é necessário especificar o número máximo de caracteres e a leitura até que seja encontrado o caractere de mudança de linha (‘\n’), o qual está representado no trecho de código a seguir.

```
1.  printf ("Digite a String=");
2.  scanf (" %49[^\n]", frase);
```

Entretanto, a leitura feita com esta função não é muito utilizada, pois seu uso é muito limitado. Para tanto, existe uma função específica para a leitura de strings, a qual trataremos no próximo tópico.

Embora também não seja muito usual, para exibir o conteúdo de uma string, podemos utilizar a função printf, juntamente com o especificador de formato %s. O trecho de código a seguir mostra a exibição da String utilizando a função printf.

```
1.  printf ("%s \n", frase);
```

Funções para manipulação de Strings

Um dos recursos adicionais da linguagem C é a possibilidade de declararmos bibliotecas, cuja principal vantagem é a utilização de funções pré-definidas. Isso deixa o código mais claro e de fácil manutenção. Para utilizarmos as funções que manipulam strings, temos que declarar, no início do programa, a biblioteca “string.h”. Dentre as funções mais usadas, destacam-se: gets, puts, strcmp, strcpy e strlen.

1. gets: utilizada para a leitura de uma string via teclado. Esta é a função mais indicada e utilizada para a leitura, pois não tem as limitações da função scanf. O trecho de código abaixo mostra a utilização da função gets. A sintaxe da função é a seguinte:

gets (string)

```
1.  char str [40];
2.  printf ("\n Digite a String desejada= ");
3.  gets (str);
```

2. puts: utilizada para a exibição de uma string. O trecho de código abaixo mostra a utilização da função puts. A sintaxe da função é a seguinte:

puts (string)

```
1.  char str [40];
2.  printf ("\n Digite a String desejada= ");
3.  gets (str);
4.  puts (str);
```

3. strcpy: utilizada para copiar o conteúdo de uma string origem para uma string destino. A primeira string terá o mesmo valor da segunda string. Podemos, também, colocar uma string qualquer entre aspas ao invés de uma variável no lugar da segunda string (origem). Importante lembrar que as duas strings devem ter o mesmo tamanho definido, sendo que a segunda string (origem) pode ser menor, nunca maior que a primeira (destino). O trecho de código abaixo mostra a utilização da função strcpy. A sintaxe da função é a seguinte:

strcpy (string destino, string origem)

```
1.  char strdest [40], strorig [40];
2.  printf ("\n Digite a String Origem= ");
3.  gets (strorig);
4.  strcpy(strdest, strorig);
```

4. strcmp: utilizada para comparar se o conteúdo de uma String é igual ao conteúdo de outra String. Nesse caso, a função retorna o valor 0 (zero) se as duas cadeias forem iguais, um valor menor que zero se a primeira string for alfabeticamente menor que segunda string ou um valor maior que zero se a

primeira string for alfabeticamente maior que a segunda string. Esta função diferencia maiúsculas de minúsculas. O trecho de código abaixo mostra a utilização da função strcmp. A sintaxe da função é a seguinte:

strcmp (string1 , string2)

```
1.  char str1[50], str2[50];
2.  strcpy(str1, "Linguagem C");
3.  printf ("\n Digite a Segunda String=");
4.  gets (str2);
5.  if ( strcmp (str1, str2) == 0 ) {
6.       printf ("\n As Strings sao Iguais");}else { printf ("\n As Strings sao Diferent
    es");
7.  }
```

5. strlen: utilizada para retornar o tamanho (quantidade de letras) de uma string, desprezando o caractere nulo final (\0). Ela retorna o valor exato de caracteres. A sintaxe da função é a seguinte:

strlen (string)

```
1.  char str [50];
2.  int tam;
3.  strcpy(str, "UNINOVE");
4.  tam = strlen(str);
5.  printf("\n A quantidade de caracteres da palavra e= %d", tam);
```

Exemplo 1

A seguir, apresentamos um exemplo de um programa em linguagem C que trabalha com strings. O programa solicita que o usuário digite uma string e, em seguida, percorre a string inteira, exibindo caractere por caractere.

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.  #include <string.h>
4.  main () {
5.      // declaracao da string
6.      char palavra [20];
7.      int i;
8.      // leitura da string
9.      printf ("*** Digite a string ***\n");
10.     gets (palavra);
11.     //laco para percorrer a string
12.     while (palavra [i] != '\0'){
13.             printf ("\n %c", palavra [i]);
14.             i++;
15.     }
16.     system ("PAUSE");
17. }
```

Exemplo 2

Vamos estudar outro exemplo de utilização de strings em C. Um palíndromo é uma palavra, frase ou qualquer outra sequência de caracteres que tenha a propriedade de poder ser lida tanto da direita para a esquerda como da

esquerda para a direita. O programa solicita que o usuário digite uma string, faz a inversão dos caracteres e verifica se a palavra é um palíndromo.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main () {
    // declaração das strings
    char palavra [20], palavra_invertida [20];
    int i, j=0, tam;
    // leitura da string
    printf ("*** Digite a string ***\n");
    gets (palavra);
    tam = strlen (palavra); // tam armazenará o tamanho da string
    for (i = tam-1; i>=0; i--){
          palavra_invertida [j] = palavra [i];
          j++;
    }
    palavra_invertida [j] = '\0';
    if  ( strcmp (palavra, palavra_invertida) == 0)
          printf ("\n A palavra e um palindromo!");
    else printf ("\n A palavra e nao palindromo!");
    system ("PAUSE");
}
```

Exemplo 3

Neste programa é feita a contagem de todas as vogais existentes em uma string de tamanho 30. Por fim, é mostrada a quantidade de vogais.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main() {
    // declaração da string
    char palavra [30];
    int i= 0, qtde=0;
    // leitura da string
    printf ("*** Digite a string ***\n");
    gets (palavra);
    //laço para percorrer a string
    while (palavra [i] != '\0'){
           if (palavra [i] == 'a' || palavra [i] == 'e' || palavra [i] == 'i' || pal
avra [i] == 'o' || palavra [i] == 'u')
          {
                qtde++;
          }
          i++;
    }
    printf ("\n A quantidade de vogais da palavra e = %d\n", qtde);
    system ("PAUSE");
}
```

Exemplo 4

Neste programa, a letra “a” é substituída pela letra “i” em uma string de tamanho 15. Por fim, a string alterada é exibida.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main () {
    // declaração da string
    char palavra [15];
    int i, tam;
    // leitura da string
    printf ("*** Digite a string ***\n");
    gets (palavra);
    tam = strlen (palavra); // tam armazenará o tamanho da string
    for (i = 0; i<tam; i++){
          if (palavra [i] == 'a')
                palavra [i] = 'i';
    }
    puts (palavra);
    system ("PAUSE");
}
```

Agora que você já estudou essa aula acesse a plataforma AVA, resolva os exercícios e verifique o seu conhecimento. Caso fique alguma dúvida, leve a questão ao Fórum e divida com seus colegas e professor.

Quiz

Exercício

Strings

INICIAR

Quiz

Exercício Final

Strings

INICIAR

Referências

MIZRAHI, V. V. *Treinamento em linguagem C*, São Paulo: Pearson, 2008.

SCHILDT, H. C — *Completo e Total*. São Paulo: Pearson, 2006.

Avalie este tópico

ANTERIOR
Matrizes

Índice

Biblioteca (https://www.uninove.br/conheca-a-uninove/biblioteca/sobre-a-biblioteca/apresentacao/)
Portal Uninove (http://www.uninove.br)
Mapa do Site

Ajuda?
PRÓXIMO
Portfólio
(https://ava.uninove.br/seu/AVA/topico/topico.php?idCurso=)