Linguagem de Programação

14/10/2013

Strings em Linguagem C

mauricioow@gmail.com

String

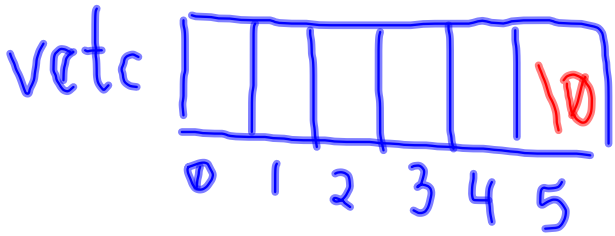
String é uma seqüência de caracteres utilizada para o armazenamento de texto. Na linguagem C strings são vetores de caracteres que possuem um caracter que indica o término de seu conteúdo, o caracter nulo ‘\0’ (contrabarra zero).

Declaração de strings

Como a string possui o caracter nulo para delimitar o final do seu conteúdo, o tamanho da string deve ser definido com um caracter a mais do que será efetivamente necessário.

Sintaxe: char identificador-da-string [tamanho+1];

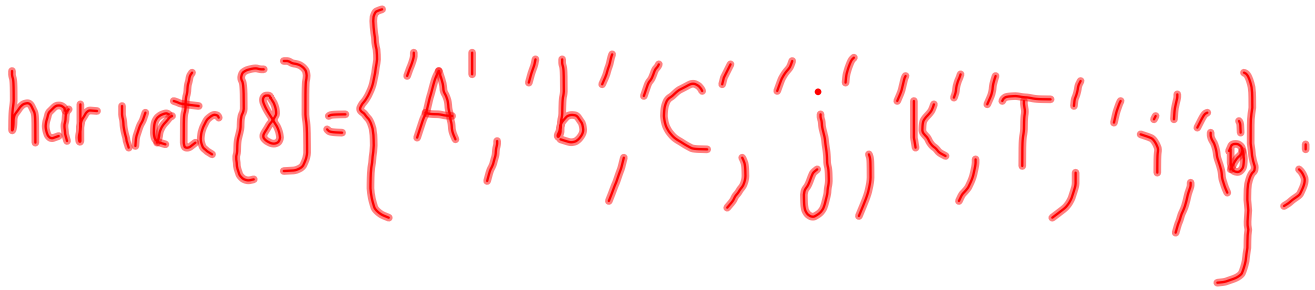
char vetc [6]; vetc é um vetor de caracteres (string) de tamanho 6. Pode receber uma palavra de no máximo 5 letras



Inicialização de strings

Uma string pode ser inicializada na sua declaração com uma sequência de caracteres entre chaves e separadas por virgula.

```
char vetc[6]= {'T', 'e', 'x', 't', 'o', '\0'};
```



Lembre-se que o compilador só reconhecerá um caractere se este estiver entre aspas simples, logo, usar uma atribuição do tipo {t,e,x,t,o,\0} ou {texto\0} irá gerar um erro de compilação.

Uma string pode também ser inicializada por uma seqüência de caracteres entre aspas duplas. Neste caso, não é necessário o uso de aspas simples e virgulas, o compilador C coloca automaticamente o '\0' no final.

```
char vetc[6] = “Texto”;
```

Assim como vetores e matrizes, na inicialização de uma string o compilador vai verificar e considerar o tamanho declarado na inicialização.

```
char vetc[ ] = "Texto"; /* vetor não-dimensionado, o
compilador coloca automaticamente o '\0' no final */
```

Funções de entrada e saída (<stdio.h>)

gets(s) - Lê uma string do dispositivo de entrada padrão e armazena esta string em s. Não é uma função segura, pois o tamanho da string não é especificado.

fgets(s, TAM, stdin) - Lê uma string de tamanho TAM do dispositivo de entrada padrão e armazena esta string em s.

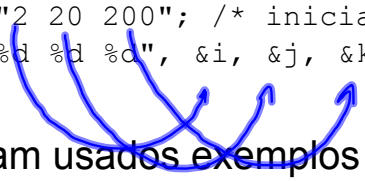
puts(s) - Imprime a string s no dispositivo de saída padrão.

sscanf(s, "expressão-controle", end1, end2, ...) - Faz a leitura formatada em uma string s

sprintf(s, "expressão-controle", arg1, arg2, ...) - Faz a escrita formatada em uma string s

Trecho de código

```
char string[ ]= "2 20 200"; /* inicializa a string str */
sscanf(string, "%d %d %d", &i, &j, &k); /* armazena 2 em i, 20 em
j e 200 em k */
```



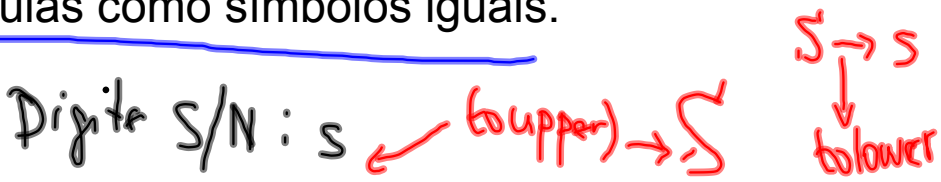
Observe que foram usados exemplos numéricos como valores, neste caso o compilador interpreta os numeros como sendo uma string. Observe as notações abaixo:
1 para o compilador pode ser um número inteiro ou real.
'1' para o compilador é o alfanumerico (caracter) char.
"11..." para o compilador é uma string de conteudo 11...
logo deve-ser ter muita atenção ao uso de aspas para manipular strings.

Trecho de código

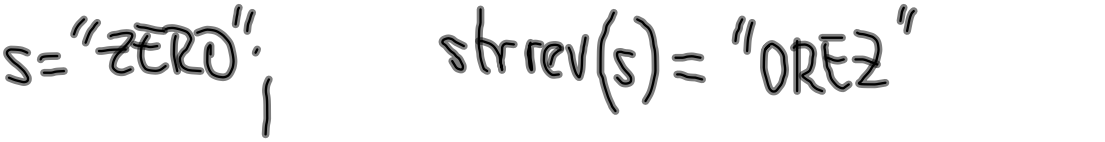
```
char string[10];
int i=2;
sprintf(string,"%d", i); /*armazena 2 em str
*/
```

Funções de manipulação de strings <string.h>

Strings não podem ser comparadas com o operador de comparação padrão (==), neste caso deve-se usar função strcmp() ou a função stricmp().
strcmp(s1,s2) – Retorna 0 se s1 e s2 são iguais; menor que 0 se s1 < s2; maior que 0 se s1 > s2 (comparação alfabética).
stricmp(s1,s2) – Retorna 0 se s1 e s2 são iguais; menor que 0 se s1 < s2; maior que 0 se s1 > s2 (comparação alfabética). Essa função considera letras maiúsculas ou minúsculas como símbolos iguais.



Strings não podem ser atribuídas com o operador de atribuição (=), para uma atribuição usa-se a função strcpy( ).
strcpy(s1,s2) – Copia s2 em s1.
Strings não podem ser concatenadas com o operador (+), para tal usa-se a função strcat().
strcat(s1,s2) – Concatena s2 ao final de s1.
strlen(s) – Retorna o número de caracteres em s (sem contar o caracter nulo (\0)).
strchr(s,c) – Retorna um ponteiro na primeira ocorrência do caracter c na string s.
strstr(s1,s2) - Retorna um ponteiro na primeira ocorrência se s2 em s1.
strrev(s) – Inverte a string s sobre ela mesma.



```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main()
{
   char nome[] = "fulano";
   char sobrenome[] = " de tal";
   char nomeCompleto[15];
   strcat(nome, sobrenome);
   strcpy(nomeCompleto, nome);
   puts(nomeCompleto);
   system("pause");
   return(0);
}
```

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main()
{
  char f1[50], f2[50],f3[100],c;
  int tam1, tam2, i,cont1,cont2;

  printf("Digite uma frase: ");
  gets(f1);
  printf("Digite uma frase: ");
  gets(f2);
  tam1=strlen(f1);
  tam2=strlen(f2);
  c='a';
  cont1=cont2=0;
  for(i=0;i<=tam1;i++)
     if(f1[i]==c)
       cont1++;
  for(i=0; i<=tam2;i++)
     if(f2[i]==c)
       cont2++;

 printf("\nA letra %c aparece %d vez(es) na primeira e
d vez(es) na segunda frase.\n", c, cont1, cont2);
  if(!strcmp(f1,f2))
    printf("\nAs frases sao iguais\n");
  else{
        strcat(f3,f1);
        strcat(f3," ");
        strcat(f3,f2);
        printf("Nova frase: ");
        puts(f3);
        if(strchr(f3,'i')) printf("A letra 'i' aparece
nesta frase\n\n");
        if(strstr(f3,"string"))printf("A sequencia
\"string\" tambem aparece nesta frase\n\n");
        printf("Frase invertida: ");
        strrev(f3);
        puts(f3);
        system("PAUSE");
    }
  }
```

frase 1 = testando o uso
frase 2 = de string.

saida

Digite uma frase: testando o uso
Digite uma frase: de string.

A letra a aparece 1 vez(es) na primeira e 0 vez(es) na segunda frase.
Nova frase: testando o uso de string.
A letra 'i' aparece nesta frase

A sequencia "string" tambem aparece nesta frase

Frase invertida: .gnirts ed osu o odnatset

Exercícios de Strings

1. Faça um programa que carregue e imprima um vetor com o seu nome (nome completo) e o total de letras que ele possui.

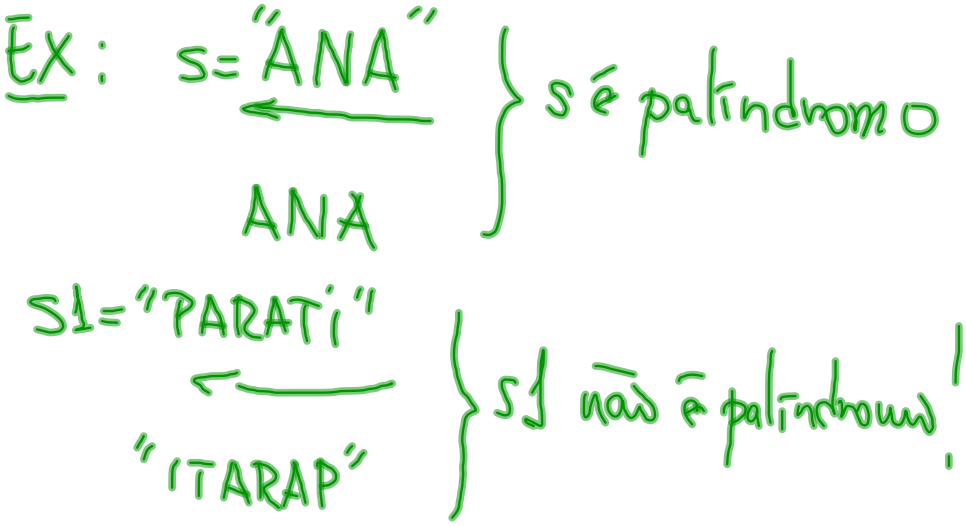
```
#include <stdio.h>
#include <string.h>
main()
{
  char nome[50];
  int i,tam,cont;
  printf("\nDigite seu nome completo: \n");
  fgets(nome,50,stdin);
  tam = strlen(nome);
  cont = tam;
  for(i=0;i<=tam;i++)
    if (nome[i]==' ')
      cont--;
  printf ("\nO nome possui %d letra(s)",cont-1);
  getch();
}
```

2. Faça um programa que carregue o vetor nome com o seu nome (nome completo), fornecido via teclado.Utilize a função gets para a leitura e em seguida use a função puts para imprimir o nome.

Substitua a função gets pela função scanf e avalie os resultados.

```
#include <stdio.h>
#include <string.h>
main()
{
  char nome[50];
  printf("\nDigite seu nome completo: \n");
  gets(nome);
  printf("O seu nome completo eh: \n\n");
  puts(nome);
  printf("\nReDigite seu nome completo: \n");
  scanf("%s",&nome);
  printf("Seu nome completo, usando scanf:\n\n");
  puts(nome);
  getch();
}
```

3. Faça um programa em Linguagem C, que verifique se uma palavra armazenada em um vetor de caracteres é um palíndromo.



```
#include <stdio.h>
#include <string.h>
main()
{
  char palavra[15], inversa[15];
  printf("\nDigite uma palavra:");
  gets(palavra);
  strcpy(inversa,palavra);
  strrev(inversa);
  if (!strcmp(palavra,inversa))
    printf("\nA PALAVRA %s E UM PALINDROMO.\n\n",palavra);
  else{
    printf("\nA PALAVRA %s NAO E UM PALINDROMO.\n\n",palavra);
    puts(inversa);
   }
  getch();
}
```

4. Faça um programa que carregue um vetor de caracteres via teclado e imprima o mesmo vetor com as letras minúsculas substituídas por letras maiúsculas.

Use `toupper(s)`

5. Faça um programa que carregue um vetor de caracteres, via teclado, gere e imprima um outro vetor onde as vogais, do primeiro vetor, sejam substituídas pelo caracter *.