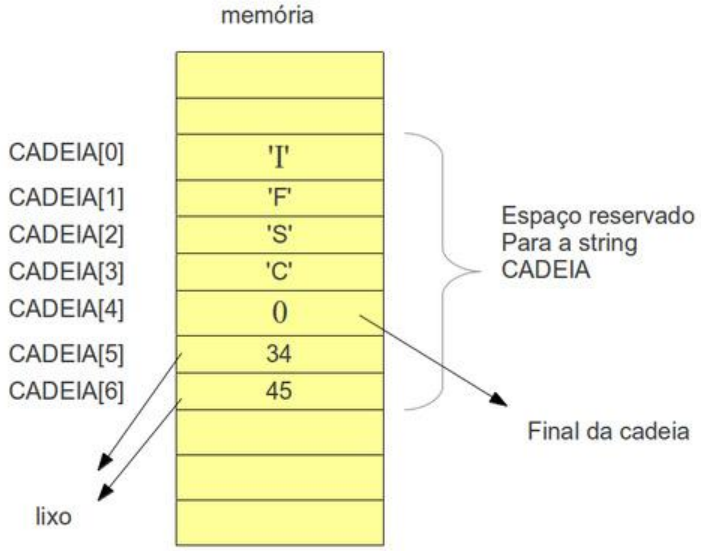
Vetor de caracter



Introdução

- O termo string serve para identificar uma sequência de caracteres. Esta sequência é utilizada para representar textos.
- Em linguagem C, ao contrário de outras linguagens, não existe um tipo de dados string nativo, apenas o tipo char (um byte - um caracter).

Introdução

- Para representar uma string em C, devemos criar um vetor de caracteres, ou seja um vetor do tipo char.
- Porém, obrigatoriamente o último caracter do vetor deve ser o nulo, ou seja, o '\0‘ (zero). Ele é usado para marcar o final da string.
- Funções que manipulam string adicionam o '\0‘ ao final dela.

Declarando uma string

Exemplo:

char n [7];

Se inicializarmos a string de 7 posições declarada acima, atribuindo a ela a palavra Joao da seguinte forma:

char n [7]="Joao";

Teremos:

J	o	a	o	\0		
0	1	2	3	4	5	6

Inicializando uma string

Formas de inicialização:

char n [7]="Joao"; ou char n []="Joao";

ou

char n []={'J','o','a','o','\0'};

ou

```
char n [7];
n [0]='J';
n [1]='o';
n [2]='a';
n [3]='o';
n [4]='\0';
```

Observação:

...

char str[10] = "Joao";

...

~~str = "maria";~~

Não ultrapasse o tamanho de uma string

Quando você efetua operações com string, precisa garantir de não sobrescrever as localizações de memória da string.

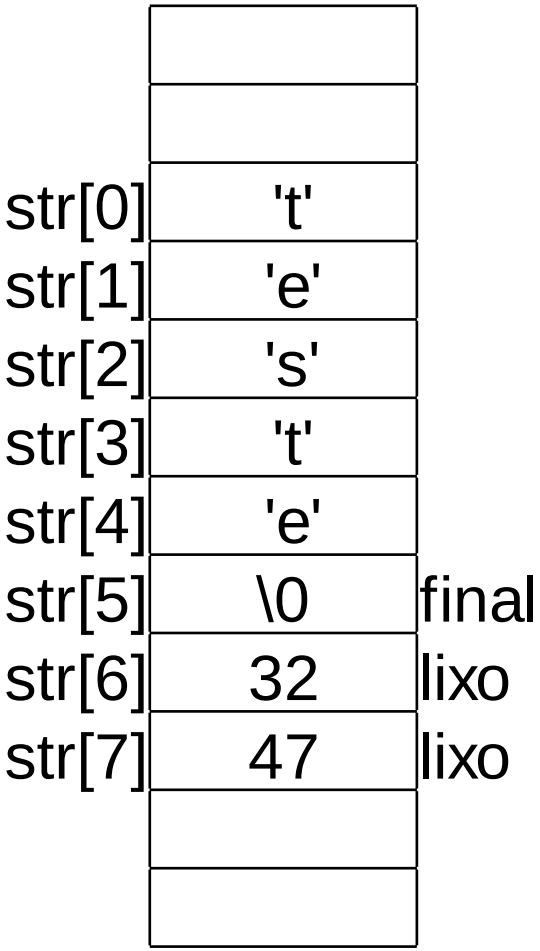
Considere a declaração: **char str[8];**

Se você atribuir mais de 8 caracteres (7 + \0) à string str, o SO pode não identificar o erro.

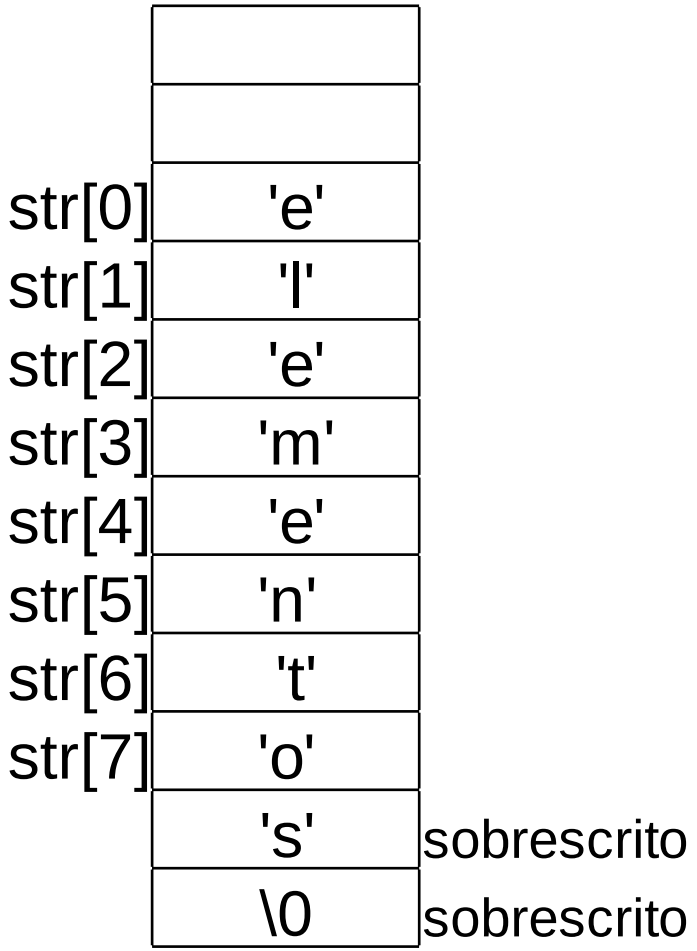
Em vez disso, os caracteres sobrescrevem áreas de memória correspondentes a outras variáveis.

Este tipo de erro é muito difícil de ser detectado e pode fazer seu programa e o SO interromper a execução.

char str[8] = "teste";



char str[8] = "elementos";



Lendo uma string com scanf e escrevendo com printf

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

int main(void)
{
  char nome[30];

  printf("Digite seu nome: ");
  scanf("%s",nome); //lê todos os caracteres até encontrar um caracter
                    igual a espaço ou fim de linha <enter>.
  printf("O nome armazenado foi: %s", nome);

  getch();
 }
```

Lendo uma string com gets e escrevendo com puts

```c
#include<stdio.h>
#include<conio.h>

int main(void)
{
  char nome[30];

  puts("Digite seu nome: ");
  gets(nome);

  puts(nome);

  getch();
 }
```

Função strcpy()

A função é usada para copiar o conteúdo da string "de" para a string "para".

strcpy(para,de);

```
#include <stdio.h>
#include<string.h>
int main(void)
{
  char str [80];
  strcpy(str, “Copiando uma string");
  printf("%s", str);
}
```

Função strcat()

Esta função anexa (concatena) "s2" em " s1"; "s2" não é modificada. As duas strings devem ser terminadas com nulo.

strcat(s1,s2);

```
#include <stdio.h>
#include<string.h>
int main(void)
 {
    char s1[20];
    strcpy(s1, "Alo");
    strcat(s1," para todos");
    printf("%s",s1);
 }
```

Função strcmp()

A função strcmp() compara duas strings e retorna:

strcmp(s1,s2);

valor 0 se elas forem iguais;

```
printf("%d\n",strcmp("AA",“AA"));
```

valor > 0 se s1 é lexicograficamente maior que s2

```
printf("%d\n",strcmp("BBBB","AAAA"));
```

valor < 0 se s1 for menor que s2.

```
printf("%d\n",strcmp("Ana","Andre"));
```

O segredo da utilização da função "strcmp()" é que ela retorna zero (falso) quando as strings são iguais. Portanto, você precisará usar o operador NOT se desejar que alguma coisa ocorra quando a string é igual.

Função strcmp()

```
#include <stdio.h>
#include <string.h>
int main(void)
{
     char s[80];
     printf("informe a senha: ");
     gets(s);
     if(strcmp(s, "senha"))
    {
         printf("senha invalida\n");
         exit(0);
    }
    printf("senha correta\n");
 }
```

Função strlen()

A função "strlen()" retorna o tamanho de s.

strlen(s); onde s é uma string.

```
#include<stdio.h>
#include<string.h>
int main(void)
{
    char str [80];
    printf("digite uma string: ");
    gets(str);
    printf("%d", strlen(str));
}
```

Função sprintf()

A função sprintf() permite que se simule um “printf” em uma string. É usada para concatenar valores provenientes de variáveis de diferentes tipos dentro de uma string.

sprintf(s,”formato”,<lista de variáveis>);

onde s é a string que receberá o “printf”

```
#include<stdio.h>
#include<string.h>
int main(void)
{
        char frase[80];
        sprintf(frase, “Ivoti, %d de %s de %d \n”, 6, “maio”,2011);
        printf(“%s”, frase);
}
```

Percorrendo um vetor de char

```
#include <stdio.h>
#include <conio.h>
int main(void)
{
  int i;
  char texto[7] = "string";
  printf("Valor da variavel texto = %s\n", texto);
  for (i=0; i<6; i++)
  {
       printf("Valor do elemento %d da string =
       %c\n",i,texto[i]);
  }
  getch();
}
```

Outras funções da string.h

strncpy

strncat

strncmp

strlwr

strupr

strchr

strrchr

charcnt

strrev

strset

ctype.h

Na listagem abaixo, podemos visualizar as principais funções de ctype.h.

Funções para conversão de caracteres maiúsculos e minúsculos:

tolower → **char=tolower(char);**
Converte o caracter maiúsculo em minúsculo.

toupper → **char=toupper(char);**
Converte o caracter minúsculo em maiúsculo.

ctype.h

Funções para manipulação de caracteres. A função recebe um char e retorna 1 ou 0.

isalnum(char);
Verifica se o caracter é alfanumérico

isalpha(char);
Verificar se o caracter é uma letra do alfabeto

iscntrl(char);
Verificar se o caracter é um caracter de controle

isdigit(char);
Verificar se o caracter é um digito decimal

ctype.h

Funções para manipulação de caracteres. A função recebe um char e retorna 1 ou 0.

isgraph(char);
Verifica se o caracter tem representação gráfica

islower(char);
Verifica se o caracter é minúsculo

isprint(char);
Verifica se o caracter é imprimível.

ispunct(char);
Verifica se o caracter é um ponto

ctype.h

Funções para manipulação de caracteres. A função recebe um char e retorna 1 ou 0.

isspace(char);
Verificar se o caracter é um espaço em branco

isupper(char);
Verifica se o caracter é uma letra maiúscula

isxdigit(char);
Verifica se o caracter é um dígito hexadecima

Exercícios

1. Escrever um programa que lê uma string e a escreve em ordem inversa.
2. Escrever um programa que lê duas strings e informa o tamanho, a igualdade entre elas e no final escrever as strings concatenadas.
3. Escrever um programa que lê uma string s[30] e escreve cada palavra desta string numa nova linha.
4. Escrever um programa que lê uma string e a escreve em maiúsculo.

Exercícios

5. Escreva um programa em C que leia uma string e informe a quantidade de palavras presentes na string.
6. Escreva um programa em C que leia uma string via teclado. Em seguida, transforme a string lida em vazia, ou seja, a string deverá ficar em branco (sem nenhum caractere).
7. Escreva um programa em C que leia uma string e também um caractere. O seu programa deverá contar o número de ocorrências do caractere lido na string.

Exercícios

8. Faça um programa em C que leia uma palavra pelo teclado e faça a impressão conforme o exemplo a seguir para a palavra OLA.

   O
   OL
   OLA

9. Faça um programa que lê uma certa quantidade de nomes de pessoas e os escreve em ordem crescente.
   Utilizar uma matriz de caracter n[10][30].

Exercícios

10. Elabore uma função (função do usuário) que retorne o tamanho de uma string. Não pode utilizar a função strlen(). Protótipo da função: int tamanho(char str[])

11. Elabore uma função (função do usuário) que copie o conteúdo de uma string para outra. Não pode utilizar a função strcpy().
    Protótipo da função: char copia(char str1[], char str2[])