21 Strings

Ronaldo F. Hashimoto e Carlos H. Morimoto

O objetivo desta aula é introduzir o conceito de strings. Ao final dessa aula você deverá saber:

- Descrever o que são strings.
- Descrever a distinção entre strings e vetores de caracteres.
- Ler e imprimir strings.
- Usar recursos da string.h.
- Utilizar strings em seus programas.

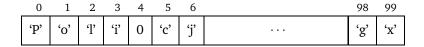
21.1 O que são strings?

Na maioria das vezes que temos que programar nos deparamos com a necessidade de uma estrutura de dados que armazena uma sequência de caracteres. Por exemplo, uma mensagem para o usuário, o nome de uma pessoa, seu endereço, seu email e assim por diante. *Strings* não são nada mais do que **vetores de caracteres** com um código para marcar sua terminação. Então, como vimos na aula de vetores, um *string* pode ser então definido da seguinte forma:

```
char < nome_do_string > [< numero_de_casas >];
```

Exemplo:

char palavra[100]; /* declaração de um vetor de caracteres */



O nosso vetor acima poderia guardar uma frase com no máximo 100 caracteres. No nosso exemplo, o vetor palavra guarda a sequência de caracteres "Poli". Observe que das 100 casas reservadas, a palavra "Poli" usa somente 4 casas. Assim, precisamos designar o término de uma sequência de caracteres armazenada em um string. Para isso, é reservado um caractere especial para ser usado no vetor ao final da palavra designando assim o seu término. O caractere especial designado para indicar fim da sequência é o caractere de código ASCII 0 (zero) também denotado por NULL ou '\0' (barra zero). Obserque que o caractere '\0' não pode fazer parte de nenhuma sequência. Agora, não confunda o caractere '\0' (cujo código ASCII é zero) com o dígito '0' (cujo código ASCII é 48). Note também que todos os caracteres depois do '\0' são considerados lixo dentro do vetor.

21.2 Leitura de Strings

Uma forma de ler um string em C (existem outras formas, mas não serão discutidas aqui) é utilizar o scanf com $%[^n]$ e em seguida coloca-se o nome do vetor.

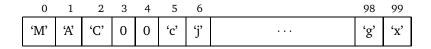
Exemplo:

```
char palavra[100]; /* declaração de um vetor de caracteres */
printf ("Entre com uma palavra: ");
scanf ("%[^\n]", palavra);
```

Na Linha 3, temos um exemplo do uso do scanf para leitura de uma sequência de caracteres que vai ser armazenada no vetor (string) palavra. O usuário vai digitar uma sequência de caracteres e depois, para terminar, digita a tecla "enter" do teclado, que produz o caractere '\n' de código ASCII 10. O comando scanf com %[^\n] lê caractere a caractere e coloca um a um em cada casa do vetor palavra; no final, o caractere '\n' de código ASCII 10 é ignorado (ou seja, não é colocado no vetor) e coloca-se o caractere '\0' de código ASCII 0 (zero). Assim, se o usuário digitar

MAC<enter>

o vetor palavra vai conter:



Note que o número 0 (zero) na casa 3 é o código ASCII do caractere '\0' (proveniente da substituição do caractere '\n' gerado pela tecla "enter" por '\0'). Assim, para representação do string, pode-se optar por colocar o código ASCII (número inteiro zero) ou o caractere entre apóstrofes '\0'. Observe ainda que não é necessário mexer nos outros caracteres depois do primeiro zero , pois eles são considerados como lixo.

21.3 Impressão de Strings

Uma forma de imprimir um string em C (existem outras formas, mas não serão discutidas aqui) é utilizar o scanf com %s e em seguida coloca-se o nome do vetor.

```
char palavra[100]; /* declaração de um vetor de caracteres */
printf ("Entre com uma palavra: ");
scanf ("%[^\n]", palavra);

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

Na Linha 5, temos um exemplo do uso do scanf para impressão da sequência de caracteres armazenada no vetor palavra.

e uma sequência de caracteres que vai ser armazenada no vetor (string) palavra. O usuário vai digitar uma sequência de caracteres e depois, para terminar, digita a tecla "enter" do teclado, que produz o caractere '\n' de código ASCII 10. O comando scanf com %[^\n] lê caractere a caractere e coloca um a um em cada casa do vetor palavra; no final, o caractere '\n' de código ASCII 10 é ignorado (ou seja, não é colocado no vetor) e coloca-se o caractere '\0' de código ASCII 0 (zero). Assim, se o usuário digitar

21.4 Biblioteca < string.h>

Em C existe uma biblioteca de funções para manipular strings. A seguir, algumas funções:

• int strlen (char s[]); devolve via return o comprimento do string armzenado no vetor s. Exemplo:

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

int main () {

char palavra[100]; /* declaração de um vetor de caracteres */
printf ("Entre com uma palavra: ");
scanf ("%[^\n]", palavra);

printf ("%s tem %d caracteres\n", palavra, strlen (palavra));

return 0;
}

return 0;
}
```

• void strcpy (char s1[], char s2[]);

copia o string armazenado em s2 para o vetor s1.

Exemplo:

```
\# include <stdio.h>
     # include <string.h>
2
     int main () {
       char palavra[100]; /* declaração de um vetor de caracteres */
       char texto[200]; /* declaração de um vetor de caracteres */
       printf ("Entre com uma palavra: ");
       scanf ("%[^\n]", palavra);
10
11
       /* copia o string armazenado em palavra para o vetor texto */
       strcpy (texto, palavra);
       printf ("%s tem %d caracteres \n", texto, strlen (texto));
14
17
       return 0;
```

21.5 Exemplo de um Programa que usa String

O programa a seguir lê um string pelo teclado e imprime quantos caracteres tem o string lido.

```
# include <stdio.h>
     # include <string.h>
2
3
      int main () {
        char nome [80]; /* string */
        int cont1, cont2;
        printf ("Entre com um nome: ");
        scanf (" %[^\n]", nome);
Q
10
        for (cont1=0; nome[cont1] != 0; cont1++);
11
       cont2 = strlen (nome);
13
14
       /* cont1 e cont2 sao iguais */
15
16
        printf ("%s tem %d caracteres\n", nome, cont1);
17
        return 0;
19
20
      }
```

21.6 Problema 1

Faça um programa que leia uma frase e imprima esta frase usando apenas letras maiúculas.

```
# include <stdio.h>
int main () {
   char frase [80];
   int i;

   printf ("Entre com uma frase: ");
   scanf ("%[^\n]", frase);

for (i=0; frase[i] != 0; i++) {
    if (frase[i] >= 'a' && frase[i] <= 'z')
      frase[i] = frase[i] - ('d' - 'D');
   }

   printf ("Frase Digitada em Maiusculas: %s\n", frase);
   return 0;
}</pre>
```

Na Linha 8, o usuário deve digitar uma frase seguida por um "enter". Esta frase é armazenada no vetor frase.

Na Linha 10, o vetor frase é percorrido até encontrar o caractere '\0' (código ASCII zero) que indica o fim de string. Para cada casa do vetor, verifica se é uma letra minúscula. Em caso afirmativo, transforma para maiúscula subtraindo da diferença entre uma letra minúscula e sua correspondente maiúscula (note que esta diferença é a mesma para qualquer letra - neste caso, foi escolhida a letra 'd').

21.7 Problema 2

Dados dois strings a e b, verifique quantas são as ocorrências do string b dentro de a.

Exemplo: se a é o string "Raras araras em Araraquara" e b é o string "ara", o seu programa deve responder 5, pois o string "ara" aparece uma vez em "Raras", duas em "araras" e outras duas em "Araraquara" (e não três já

que é feita a distinção entre maiúsculas e minúsculas).

Para resolver este problema, vamos fazer uma função que verifica se um string menor encaixa em um string maior a partir do índice ind.

```
int encaixa (char menor[], char maior[], int ind) {
   int i,j;
   j=ind;
   for (i=0; menor[i] != 0; i++) {
      if (menor[i] != maior[j]) {
        return 0;
      }
      j++;
   }
   return 1;
}
```

Agora é somente usar esta função para contar quantas vezes b encaixa em a.

```
int main () {
 char a[80], b[80];
 int compr_a, compr_b, cont, i;
 printf ("Entre com uma frase a: ");
 scanf (" %[^\n]", a);
  printf ("Entre com uma palavra b: ");
 scanf (" %[^\n]", b);
 /* descobrindo o comprimento da frase a */
 for (compr_a=0; a[compr_a] != 0; compr_a++);
 /* descobrindo o comprimento da palavra b */
 for (compr_b=0; b[compr_b] != 0; compr_b++);
  for (i=cont=0; i < compr_a - compr_b + 1; i++) {
   cont = cont + encaixa (b, a, i);
 printf ("%s aparece em %s %d vezes\n", b, a, cont);
 return 0;
}
```