ESTUDO DIRIGIDO e AULA PRÁTICA: STRINGS

RESUMO

- Uma **string** é um vetor de caracteres cujo conteúdo da última posição ocupada é o '\0'.

- Verificamos que o caractere delimitador na entrada de uma string via scanf é o caracter espaço (branco) ou um enter. Para podermos ler do teclado uma string que inclua caracteres especiais ou o espaço é necessário permitir quaisquer caracteres menos o enter: **scanf(%[^\n], str1).**

- **scanf** com o especificador de formato %[...]
  - %[...] lista entre os colchetes todos os caracteres aceitos na leitura
  - %[^...] lista entre os colchetes todos os caracteres não aceitos na leitura
  - exemplos:
    - %[aeiou]
      - lê seqüências de vogais
      - leitura prossegue até encontrar um caractere que não seja uma vogal
    - %[^aeiou]
      - lê seqüências de caracteres que não são vogais
      - leitura prossegue até encontrar um caractere que seja uma vogal

- Principais funções da biblioteca **<string.h>:** *stlen, strcpy, strcmp, strcat*

  - A função **strlen** retorna o número de caracteres em uma cadeia de caracteres.
    - O caractere '\0' não é contado.
    - O cabeçalho da função é:
      - **int strlen(char str[ ])**;

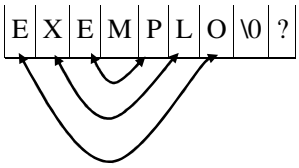
  - A função **strcpy** é usada para copiar o conteúdo de uma cadeia de caracteres (fonte) para outra cadeia (destino).
    - O cabeçalho da função é:
      - **int strcpy(char destino[ ], fonte[ ])**;

  - A função **strcmp** é usada para fazer a comparação lexicográfica de duas cadeias.
    - Se as cadeias são iguais, isto é, se str1 e str2 têm mesmo comprimento e caracteres iguais nos elementos de mesmo índice, a função retorna 0
    - Se str1 for lexicograficamente maior do que str2, a função retorna 1
    - Se str2 for lexicograficamente maior do que str1, a função retorna -1
    - O cabeçalho da função é:
      - **int strcmp(char str1[ ], char str2[ ])**;

  - A função **strcat** anexa o conteúdo de uma cadeia de caracteres (fonte) ao final de outra cadeia (destino).
    - O tamanho da cadeia de destino deve ser suficiente para armazenar o total de caracteres das duas cadeias.
    - O cabeçalho da função é:
      - **int strcat(char destino[ ], char origem[ ])**;

EXERCÍCIOS

1) Faça um programa que leia uma string do teclado e imprima a string invertida. Por exemplo: ao digitar EXEMPLO, o usuário veria na tela após a execução a string OLPMEXE.



2) Uma palavra é denominada um palíndromo se for invertida e a leitura da mesma permanecer sem nenhuma alteração. Algumas palavras que são palíndromos são: aba, radar, reter, rever, rir, rotor, dentre outras. Construir um programa que detecte se uma palavra (string) digitada pelo usuário é ou não um palíndromo.

3) Faça um programa que pede para o usuário:
   (i) Uma string $s_{inicial}$,
   (ii) Um caractere ch1,
   (iii) Um caractere ch2.
   O programa deve criar uma nova string s_{final} substituindo todas as ocorrências do caractere ch1 em $s_{inicial}$ pelo caractere ch2 e exibir a quantidade de caracteres substituídos

4) **Faça uma função** que receba o código de uma disciplina (7 digitos) e retorne o caminho de acesso ao site. Sabe-se que os três primeiros caracteres do código dizem o departamento.
   Exemplo: código da disciplina: inf1005
   código do departamento : inf
   String de retorno: www.**inf**.puc-rio.br/**~inf1005**
   Crie uma main para testar sua função. Utilize as funções adequadas da biblioteca **string.h**

5) Faça uma função que leia uma string e imprima as palavras em ordem inversa à lida.
   - Considere que cada palavra é separada por apenas um espaço
     Exemplo: "Isto é um exemplo" ===> exemplo um é Isto
   - Considere que cada palavra é separada por uma quantidade qualquer de espaços
     Exemplo: "Isto é um exemplo" ===> exemplo um é Isto

6) Faça um programa que leia uma seqüência de nomes completos e para cada nome imprima:

   a) as iniciais de todos os componentes do nome e o sobrenome completo.
      Ex: "João Pedro Albuquerque Cardoso Silva" ===> J P A C Silva
   b) o sobrenome completo e as iniciais de cada nome
      Ex: "João Pedro Albuquerque Cardoso Silva" ===> Silva J P A C

7) Faça duas funções que recebam duas strings e verifiquem se a string2 é substring da string1.

   a) retorna 0 se string2 não for substring da string 1 ou a posição onde a string2 ocorre pela primeira vez na string1.
      Exemplo: substr_pos("banana","ana") ==> 1
   b) retorna 0 se string2 não for substring da string 1 ou o número de vezes que a string2 ocorre na string1.
      Exemplo: substr_qt("banana", "ana") ==> 2

8) Faça uma função que receba uma string e dois caracteres (c1, c2) e substitua na string1 todas as ocorrências do caracter c1 pelo caracter c2, retornando o número de substituições realizadas.

9) Faça uma função que receba 3 strings e substitua todas as ocorrências da string2 na string1 pela string3 retornando o número de substituições realizadas .
   Exemplo: string1:“banana” string2: “ana” string3:”or” ==> string1: “borna” retorna 1
   string1:“banana” string2: “ana” string3:”ora” ==> string1: “borora” retorna 2

10) Construir um programa que armazena em um vetor de inteiros v os índices de onde um caractere ch1 digitado pelo usuário aparece em uma string s também digitada pelo usuário. Depois de obter o vetor v, o mesmo deve ser impresso. Um exemplo de execução do programa é dado por:
    Digite uma string: *Uma longa jornada.*
    Digite o caractere: *a*
    Indices onde ocorrem o caractere *a*: *2 8 14 16*
    Para verificar que o exemplo de execução esta correto, basta ver que:

S=	U	m	**a**		l	o	n	g	**a**		j	o	r	n	**a**	d	**a**	.
i=	0	1	**2**	3	4	5	6	7	**8**	9	10	11	12	13	**14**	15	**16**	17
			↓						↓						↓		↓	
V=	**[2 8 14 16]**																	