UTFPR - Universidade Tecnológica Federal do Paraná
DACOM - Departamento de Computação
BCC2001 :: Algoritmos

Lista de Exercícios no. 06 :: Vetores | Strings

Instruções Gerais

- Os exercícios são de resolução individual.
- Crie um único arquivo .c e faça cada exercício em uma ou mais funções. Teste-as na main().
- Utilize o compilador gcc e o editor VS Code (ou algum outro de sua preferência).
- Você pode utilizar as seguintes funções da biblioteca padrão de C, exceto quando houver restrição imposta pelo enunciado do exercício:
  - rand() **<stdlib.h>**, printf() **<stdio.h>**, strlen(), strcpy(), strcmp() e strcat() **<string.h>**
- **Não é permitido o uso de recursos que ainda não foram abordados na disciplina até o momento da publicação desta lista. Esta lista considera até:**
  - **Vetores e strings em C.**

1. Escreva uma função que recebe uma string como parâmetro e a imprime na tela, com cada caractere separado por um espaço. Você não pode utilizar **strlen()**.

   ```
   void print_string(char str[])
   ```

2. Escreva uma função que recebe uma string e a imprime ao contrário (da direita para esquerda). Faça duas versões da função: (1) utilizando **strlen()** e (2) não utilizando a função.

   ```
   void print_string_reversed(char str[])
   ```

3. Escreva uma função que recebe uma string e imprime:
   a. A quantidade de letras (maiúsculas ou minúsculas);
   b. A quantidade de dígitos;
   c. A quantidade de símbolos.

   OBS: considere apenas a porção dos caracteres imprimíveis da tabela ASCII, isto é, dos índices 32 ao 126.

   ```
   void string_report(char str[])
   ```

4. Escreva uma função que recebe uma string e a converte para letras maiúsculas. Atenção: a string poderá conter letras maiúsculas e símbolos.

   ```
   void string_to_upper(char str[])
   ```

   ```
   Ex: char s[] = “All your BASE are Belong to US!”;
       stringToUpper(s);
       printf(“%s”, s); // saída: ALL YOUR BASE ARE BELONG TO US!
   ```

5. A função strcmp(str1, str2) compara duas strings alfabeticamente. Ela devolve:
    a. res < 0 se str1 < str2 << str1 vem antes de str2
    b. res = 0 se str1 = str2
    c. res > 0 se str1 > st2 << str1 vem depois de str2

    Escreva uma função que compara duas strings independente do caso (maiúsculo ou minúsculo). Ela deve retornar os mesmos tipos de valores que strcmp(). Dica: com a função do exercício anterior, você poderá passar ambas strings para maiúsculas e, então, compará-las com strcmp(). OBS: você não deve modificar as strings **s1** e **s2** ("**const**" garante isso).

    int strcmp_plus(const char s1[], const char s2[])

    ```
    Ex: int res = strcmp_plus(“JOanna”, “joanna”); // res:0 (strings iguais)
    ```

6. Escreva uma função que conta e devolve o número de palavras em uma string. Considere que poderá haver mais de um espaço entre as palavras, bem como, no início e final da string.

    int count_words_plus(const char str[])

    ```
    Ex: char s[] = “  first     things  first, second  things latter “;
        printf(“%d”, count_words_plus(s)); // saída: 6
    ```

7. Escreva uma função que recebe uma string composta de várias palavras. A função deve modificar a letra inicial de cada palavra para maiúscula e, as demais, para minúsculas. Considere que sempre haverá ao menos um espaço entre cada palavra.

    void string_capitalize(const char str[])

    ```
    Ex: char s[] = “welCOME  To     COMPUTER programming!!”;
        string_capitalize(s);
        printf(“%s”, s); // saída: Welcome  To     Computer Programming!!
    ```

8. Escreva uma função que implementa um comportamento similar à strcmp(). Entretanto, você não deve utilizar strcmp() em sua implementação. A função criada deve retornar:
    a. res = -1 se str1 < str2 << str1 vem antes de str2
    b. res = 0 se str1 = str2
    c. res = 1 se str1 > st2 << str1 vem depois de str2

    int string_compare(const char str1[], const char str2[])

9. Escreva uma função que recebe uma string e imprime o número de ocorrências de cada letra da tabela ASCII (26 possibilidades), bem como, seu percentual sobre o todo (desconsiderando espaços). A contagem deve desconsiderar letras maiúsculas e minúsculas.

    void string_count(const char str[])

    ```
    Ex: char s[] = “Hello World of Software Development“;
        string_count(s);

        /* Saída:
          D x2 (6%)
    ```

```
  E x5 (16%)
  F x2 (6%)
  H x1 (3%)
  L x4 (13%)
  O x5 (16%)
  R x2 (6%)
  S x1 (3%)
  T x2 (6%)
  W x2 (6%)
*/
```

10. Escreva uma função que remove os espaços que possam existir antes e depois de uma string. Neste caso, a função deverá, necessariamente, modificar a string de entrada **str**.

    void string_trim(char str[])

    ```
    Ex: char s[] = “    hello world    “;
        string_trim(s);
        printf(“%s”, s); // saída: “hello world”
    ```

11. Escreva uma função que informa, com 1 (true) ou 0 (false), se uma string está contida em outra.

    int find_substring(const char str[], const char sub[])

    ```
    Ex: char s[] = “first things first, second things latter“;
        int check = find_substring(s, “second”);
        // neste caso, deve devolver 1, pois a string contém a palavra “second”
    ```

12. Escreva uma função que corta uma string após uma dada palavra. As posições após a palavra devem ser preenchidas com o caractere NULO ('\0' ou 0) até o final do vetor, isto é, até a posição do NULO anterior.

    void cut_string(char str[], const char word[])

    ```
    Ex: char s[] = “first things first, second things latter“;
        cut_string(s, “second”);
        printf(“%s”, s); // s = “first things first, second”
    ```

13. Escreva uma função que converte um número inteiro (parâmetro de entrada **number**) para uma string (parâmetro de saída **converted**).

    void int_to_string(int number, char converted[])

    ```
    Ex: char num[11];
        int_to_string(512, num);
        printf(“%s”, num); // saída: “512” (string)
    ```

14. Escreva uma função que converte uma string para um inteiro. Utilize a notação posicional para montar o número inteiro. Ex: 2506 = 2 x 10^3 + 5 x 10^2 + 0 x 10^1 + 6 x 10^0.

    int string_to_int(const char number_str[])

    ```
    Ex: int n = string_to_int(“1024”);  // n = 1024
    ```