

# Introdução à Programação

## Ficha Laboratorial 7

### Tópicos da matéria: Strings

### Bibliografia:

“Linguagem C”: Luis Damas  
Capítulo 7

“C programming: A Modern Approach”: K. N. King  
Capítulo 13 (secções 13.1 a 13.5)

**Nota:** Antes da implementação deve desenvolver o algoritmo para cada um dos exercícios propostos.

1. Desenvolva um programa que leia uma frase introduzida pelo utilizador e a escreva invertida.

**Por exemplo:** Hoje e Domingo!

**O programa deverá escrever:** !ognimoD e ejoH

2. Desenvolva uma função, cujo protótipo seja `int contaPrimeiro(char[])`, que receba por argumento uma *string* e devolva o número de vezes que o caracter inicial (o primeiro carácter da frase que não seja um espaço em branco) surge ao longo da frase. O programa não deve distinguir entre letras maiúsculas e minúsculas.

**Por exemplo:** `contaPrimeiro(" Amanha nao e Domingo!");`

**Deve devolver o valor:** 4

3. Desenvolva um programa que leia uma frase introduzida pelo utilizador e escreva cada uma das palavras que constituem a frase numa linha separada. Considere que as palavras estão separadas por um ou mais espaços em branco, podendo também existir espaços no início e fim da frase.

**Por exemplo:** Hoje e Domingo!

**O programa deverá escrever:**

Hoje  
e  
Domingo!

4. Desenvolva um programa que leia uma frase introduzida pelo utilizador e verifique quantas vezes a primeira palavra se repete. O programa não deve distinguir entre letras maiúsculas e minúsculas.

**Por exemplo:** Ter ou nao ter

**O programa deverá escrever:**

A palavra ter repete-se duas vezes.

5.

- a) Desenvolva uma função que insira uma palavra no meio de uma frase. A frase é um conjunto de palavras separadas por um ou mais espaços. A função deve receber por argumento uma frase, uma palavra e um valor inteiro positivo. Este valor indica qual é a posição da tabela *frase* a partir da qual a nova palavra deve ser inserida.

**Exemplo:**

Se função for chamada com o valor 16 e duas tabelas (*frase* e *palavra*) com o conteúdo:

```
"A Pizzaria Pepe serve pizzas muito quentes." e
"Verde",
```

Na primeira tabela (*frase*) deve ficar guardada a seguinte informação:

```
"A Pizzaria Pepe Verde serve pizzas muito quentes."
```

- b) Teste a função anterior. Tanto a frase como a palavra deverão estar armazenadas em tabelas de caracteres locais ao `main()` com a seguinte declaração:

```
#define TAM_FRASE 150
#define TAM_PALAVRA 20
...
void main (void
{
    char frase[TAM_FRASE], palavra[TAM_PALAVRA];
    ...
}
```

A frase armazenada é um conjunto de palavras separadas por um ou mais espaços. Considere que, quando a função a desenvolver for chamada, as duas estruturas já devem estar inicializadas (respectivamente com uma frase e uma palavra, terminadas com `'\0'`).

6. Desenvolva uma função que receba um array de caracteres terminado por `'\0'` (*string*), a respectiva dimensão e um carácter (*c*). Esta função deve transformar a *string* original de tal forma que duplique o carácter *c* de cada vez que ele aparecer na *string*. A função deve devolver o valor inteiro 1 se a *string* for modificada de acordo com os requisitos especificados atrás e 0 caso contrário.

**Por exemplo**, se a *string* passada à função for:

```
Como vai o amigo?
```

e o carácter for *o*, a *string* transformada de acordo com o especificado será:

```
Coomoo vai oo amigoo?
```

7. Escreva uma função que receba uma frase, uma palavra e as dimensões dos respectivos vectores onde estão armazenadas as strings, como argumentos. A função deverá substituir a última palavra da frase pela palavra passada como 2º argumento, devolvendo 1 se a frase for, de facto, modificada e 0 caso contrário. Assuma que as palavras na frase estão separadas por um ou mais espaços, podendo existir espaços no início e no fim da frase.

O protótipo da função será:

```
int func(char frase[], char palavra[], int tamf, int tamp);
```

Considerando a seguinte função main()...

```
#include <stdio.h>
#define TAMFRASE 80
#define TAMPAL 15

void main()
{
    char f[TAMFRASE], p[TAMPAL];

    printf ("Indique frase: "); gets(f);
    printf("Indique palavra: "); gets(p);

    if(func(f,p,TAMFRASE,TAMPAL))
        printf("Frase modificada:\n\t%s\n",f);
    else
        printf("Frase não modificada!\n");
}
```

O resultado de execução deverá ser o seguinte:

```
Indique frase: Hoje é um dia importante
Indique palavra: normal
Frase modificada:
    Hoje é um dia normal
```

#### 8. Elabore uma função que:

- Receba como argumentos uma string *str*, um valor inteiro correspondente ao tamanho do vector onde a string é armazenada *tam*, um caracter *c* e um valor inteiro *x*.
- Coloque o caracter recebido no final da string o número de vezes possível até uma máximo de *x* vezes.
- Devolva o número de caracteres adicionados à string
- A string alterada, deve continuar a ser uma string válida.

O protótipo da função será:

```
int func(char str[], int tam, char c, int x);
```

#### Exemplo de execução:

*str*: ABCDE *c*: 'Z' *x*: 5 *tam*: 10  
após a chamada à função, *str*: ABCDEZZZZ, devolve 4

*str*: AA *c*: 'Z' *x*: 5 *tam*: 10  
após a chamada à função, *str*: AAZZZZZ, devolve 5

*str*: ABCDEFGHI *c*: 'M' *x*: 5 *tam*: 10  
após a chamada à função, *str*: ABCDEFGHI, devolve 0

9. a) Desenvolva uma função que receba como argumentos uma string, um número inteiro **x** e um carácter **c**, e verifique se nessa string surge o carácter **c**, recebido como argumento, **x** vezes consecutivas. Se surgir a função deve devolver 1, caso contrário deve devolver 0.
- b) Desenvolva um programa que, peça ao utilizador um carácter. Depois disso deve pedir ao utilizador um conjunto de strings. A introdução de strings deve terminar quando ele introduzir a string **"fim"**. No final do programa deve ser mostrado ao utilizador qual foi a maior das strings introduzidas onde o carácter dado pelo utilizador surgiu 3 vezes consecutivas. Utilize a função anterior para saber se na string introduzida o carácter dado aparece três vezes consecutivas.

**Exemplo de execução do programa:**

Introduza um caracter: **B**

Introduza uma string: **ABA**

Introduza uma string: **ABBBBBBAAA**

Introduza uma string: **BBBXB**

Introduza uma string: **CCC**

Introduza uma string: **FIM**

A maior string onde o carácter B surgiu 3 vezes consecutivas é: **ABBBBBBAAA**

10. Faça uma **função** que simule o funcionamento de uma calculadora elementar para valores inteiros. A função recebe como argumentos um carácter (**c**) e duas *strings* (**num1**, **num2**). O carácter **c** deve corresponder a uma das 4 operações aritméticas elementares ('+', '-', '\*', '/') e cada uma das *strings* é constituída por um conjunto de algarismos. A função deve calcular e devolver o **número inteiro** correspondente à **operação especificada** considerando como operandos os 2 números inteiros guardados em **num1** e **num2**.

Considere o seguinte protótipo para a função:

```
int calculadoraInteiros(char c, char num1[], char num2[]);
```

Assuma que quer o carácter **c** quer as strings **num1** e **num2** possuem valores válidos quando a função é chamada.

**Por exemplo:**

- a) se **c = '+'**, **num1="234"** e **num2="3218"**  
o valor devolvido pela função deve ser o inteiro 3452
- b) se **c = '\*'**, **num1="234"** e **num2="20"**  
o valor devolvido pela função deve ser o inteiro 4680

**Notas:**

Não pode usar a função `atoi()` nem a `sscanf()`

Caso necessite pode utilizar a função: `int strlen(char str[])`, a função `strlen` devolve o número de caracteres efectivos existentes na string "str" (sem contar com o '\0').