## Algoritmos e Técnicas de Programação II

### Exercícios de Fixação

### (Manipulação de *Cadeias de Caracteres - strings*)

**Observação: Os tipos dos parâmetros deverão ser definidos de acordo com as necessidades estabelecidas nos enunciados.**

1. Faça uma função que receba uma *string* como parâmetro e retorne o tamanho dela. Utilize a função **strlen** disponível no C.
2. Ao serem fornecidas duas cadeias de caracteres, gerar e retornar, em uma terceira a concatenação das duas primeiras, ou seja:

Cadeia 1: *Em casa de ferreiro*

Cadeia 2: *Espeto é de pau.*

Cadeia concatenada: *Em casa de ferreiro Espeto é de pau.*

1. Faça uma função que receba como parâmetros um caractere e uma *string*, e execute a operação de inserção do caractere no final da *string* (concatenação). A função deverá retornar a nova *string*.
2. Faça uma função que receba como parâmetros uma *string*, um valor que represente uma posição da string (início) e outro valor que represente uma quantidade de caracteres (quant), e execute a operação de remover da string a quantidade de caracteres especificada por (quant) a partir da posição (início). A função deve retornar a nova *string*.
3. Faça uma função que receba como parâmetros duas strings e verifique se a *string1* é menor que a *string2* em relação ao tamanho lógico de cada uma, ou seja, quantidade de caracteres válidos. Caso verdade a função deverá retornar *Verdadeiro (qualquer valor)*, senão *Falso (zero)*.
4. Faça uma função que receba como parâmetros duas *strings* e verifique se a *string1* é igual a *string2* com relação ao conteúdo, caso verdade a função retornar *Verdadeiro (qualquer valor)*, senão *Falso (zero)*.
5. Faça uma função que receba como parâmetros uma *string* (frase), uma *substring* (palavra) e um valor que represente uma posição da *string* (posição), e execute a operação de inserção da *substring* na string a partir da posição passada como parâmetro. A função deverá retornar a nova *string*.
6. Faça uma função que receba como parâmetro uma *string* e retorne o número de vogais existente na *string*.
7. Faça uma função que receba como parâmetro uma *string* e retorne o número de palavras existente, independentemente da quantidade de espaços existentes.
8. Escreva uma função que receba uma *string* S como parâmetro e retire todos os brancos (espaços) contidos na mesma.
9. Escreva uma função que receba um número real e retorne uma *string* correspondente ao número recebido, com o mesmo convertido para *string* com tamanho mínimo e 2 casas decimais, e com uma vírgula no lugar do ponto decimal.
10. Escreva uma função que receba uma *string* do teclado e a retorne invertida.

Exemplo: Entrada  **MARIA BONITA**

Saída  **ATINOB AIRAM**

1. Construa uma função que receba uma frase qualquer, e possibilite ao usuário procurar uma palavra nesta frase.
2. Elabore uma função que ao receber uma data no formato “dd/mm/aaaa”, exiba o mês por extenso, conforme exemplo:

* Data conhecida: *26/10/2020*
* Data a ser exibida: *Presidente Prudente, 26 de Outubro de 2020.*

1. Ao serem fornecidas duas cadeias de caracteres (frases), faça em uma terceira cadeia a intersecção das palavras existentes nas duas primeiras.

Frase 1: *Palmeiras – Campeão Brasileiro 2018 – Brasil*

Frase 2: *Palmeiras – Campeão da Copa do Brasil 2012 – Brasil*

Intersecção: ***Palmeiras – Campeão – Brasil***

1. Construa uma função que receba uma matriz de nomes e possibilite, ao Usuário, a consulta por um intervalo estabelecido. Verifique exemplo:

* Intervalo a ser exibido: ***A*** e ***G***

|  |  |  |
| --- | --- | --- |
| Ambrósio | Zobaido | Ziguifrida |
| Tareska | Gorobovsk | Palesquito |
| Florência | Hubalda | Juvêncio |

* Nomes exibidos: ***Ambrósio, Florência, Gorobovski***

1. Construa uma função que receba um texto, e possibilite ao usuário procurar e trocar todas as ocorrências de uma palavra qualquer por outra definida previamente.