Pontifícia Universidade Católica de Goiás
Escola de Ciências Exatas e da Computação
CMP1048 - Técnicas de Programação
Max Gontijo de Oliveira
Lista de Exercícios 1:C - Funções

1. Fazer uma função que receba um caractere como parâmetro e retorne 1 caso o caractere seja uma consoante, ou 0 caso contrário.

2. Fazer uma função para verificar se um caractere é uma vogal minúscula, retornando 1 (verdadeiro) ou 0 (falso). Faça um programa para testar a função.

3. Criar uma função que recebe uma string e verifica se o texto é um palíndromo ou não, retornando 1 para o caso positivo ou 0 para o caso negativo.

4. Diz-se que um número é perfeito se esse número é igual à soma de seus divisores próprios. Divisores próprios de um número positivo N são todos os divisores inteiros positivos de N exceto o próprio N. Por exemplo, 6 é um número perfeito, pois $6 = 1 + 2 + 3$. Escreva uma função que determina se o numero passado como parâmetro é um número perfeito. Utilize essa função em um programa que determina e exibe todos números perfeitos entre 1 e 10000. Exiba os fatores de cada número perfeito confirmando que o número é de fato perfeito. Segue o início do resultado esperado.

```
6 = 1 + 2 + 3
28 = 1 + 2 + 4 + 7 + 14
.
.
.
```

5. Construa uma função chamada `to_lower_case` que receba uma string (pode ser um vetor de `char` ou um objeto `std::string`). A função deverá retornar uma outra string cujo conteúdo seja o mesmo da string original, mas com todas as letras **minúsculas**. Considere que haverão apenas caracteres ASCII e não haverão letras com marcações especiais, como acentuação, trema, cedilha, etc. Note que os caracteres além das letras não devem sofrer alteração.

6. Construa uma função chamada `to_upper_case` que receba uma string (pode ser um vetor de `char` ou um objeto `std::string`). A função deverá retornar uma outra string cujo conteúdo seja o mesmo da string original, mas com todas as letras **maiúsculas**. Considere que haverão apenas caracteres ASCII e não haverão letras com marcações especiais, como acentuação, trema, cedilha, etc. Note que os caracteres além das letras não devem sofrer alteração.

7. Construa uma função chamada `caixa_com_texto_centralizado` que receba por parâmetro uma string S, um número inteiro N e um caractere C. A função deverá imprimir uma caixa cujo contorno seja composto pelo caractere C contendo o texto S centralizado, considerando que a largura da caixa seja dada por N.

   Exemplo de saída para o caso de `S = "Ola Mundo!"`, `N = 30` e `C = '+'`:

```
++++++++++++++++++++++++++++++
+          Ola Mundo!          +
++++++++++++++++++++++++++++++
```

   Note que se a largura da caixa for um número ímpar e o tamanho da string for um número par (ou vice-versa), o texto não caberá certinho no centro. Nesse caso, a função deverá imprimir um caractere em branco a menos do lado esquerdo. Por exemplo, se os parâmetros fossem `S = "Ola Mundo!"`, `N = 29` e `C = '+'` a saída deverá ser assim:

```
++++++++++++++++++++++++++++++
+         Ola Mundo!          +
++++++++++++++++++++++++++++++
```

Por fim, caso o tamanho da string seja maior do que a largura da parte interna da caixa (lembre-se de que a borda ocupa dois espaços - um à esquerda e outro à direita), o texto deverá ser suprimido para caber. Portanto, por exemplo, se os parâmetros fossem `S = "Ola Mundo!"`, `N = 10` e `C = '+'` a saída deverá ser assim:

```
++++++++++
+Ola Mund+
++++++++++
```