Ficha de Trabalho N.º 3

Objetivos: Funções e *arrays*.

1 - Escreva as seguintes funções sobre o tipo char:

	Função	Devolve
a)	**int isdigit(char c)**	Verdade quando c é um dígito e Falso c.c.
b)	**int isalpha(char c)**	Verdade quando c é uma letra e Falso c.c.
c)	**int isalnum(char c)**	Verdade quando c é um caracter alfanumérico e Falso c.c.
d)	**char tolower(char c)**	Devolve c transformado na minúscula correspondente
e)	**char toupper(char c)**	Devolve c transformado na maiúscula correspondente

Nota: Obtém-se acesso a estas funções através da diretiva
#include <ctype.h> // Funções sobre o tipo char (*ctype -> char type*)

2 - a) Escreva as funções a seguir indicadas de modo que devolvam os resultados descritos:

	Função	Devolve
a)	**int resto (int a, int b)**	O resto da divisão de a por b
b)	**int impar (int x)**	Verdade se x for ímpar e Falso c.c.
c)	**int perfeito (int n)**	Verdade se n for "perfeito" (igual à soma dos divisores de n, inferiores a n) e Falso c.c.
d)	**int primo (int n)**	Verdade se n for "primo" (apenas divisível por 1 e por n) e Falso c.c.

b) Crie um main() que permita testar as funções criadas nos exercícios 1 e 2.

3 - No século I d.C. os números naturais dividiam-se em três categorias:

REDUZIDOS: os superiores à soma dos seus divisores;
ABUNDANTES: os inferiores à soma dos seus divisores;
PERFEITOS: os que são iguais à soma dos seus divisores.

NOTA: Nesta definição exclui-se o próprio número do conjunto dos seus divisores.

Escreva uma função que liste os inteiros entre a e b, a<b, classificando-os de acordo com esse critério, e que escreva também o total de cada uma das categorias.
Crie um main() eu permita testar a função criada.

4 - Elabore funções que determinem:

a. O cubo de um número inteiro **n**. O número **n** deve ser pedido ao utilizador através de uma função (denominada **leitura**) e o seu **cubo** deve ser calculado através de outra função (de nome **cubo**).

b. Copiar a função cubo criada na alínea a., e alterá-la, criando a função **exponenciacao**, de forma a torná-la mais genérica: calcular **x^exp**. Os números **x** e **exp** devem ser solicitados ao utilizador através da função **leitura**. De igual modo, **x^exp** deve ser calculado através da função **exponenciacao**.

Obs. No final, teste e corrija as funções, criando um main() para o efeito.

5 - Escreva uma função que determine o maior de dois números dados. Teste a função num pequeno programa.

6 - Elabore um programa que:

a) Leia as *n* componentes de um vetor;
b) Escreva as n componentes de um vetor;
c) Determine a posição em que se encontra a menor componente.

Cada uma destas tarefas deve ser realizada por uma função.

7 - Escreva uma função que determine o produto interno de dois vetores de n componentes inteiras.

8 - Dado um vetor x com n componentes inteiras, escreva funções que permitam realizar as seguintes operações:

a) Trocar a componente da posição p com a da posição q.
b) Efetuar a permutação circular do vetor dado;

Obs.: Crie um main() que lhe permita testar as funções criadas nos exercícios 7 e 8.

9 - Considere uma matriz quadrada com nxn elementos inteiros.

Elabore um programa que lhe permita:

- Ler os *n*x*n* elementos da matriz;
- Mostrar no monitor os *n*x*n* elementos da matriz;
- Determinar o valor mínimo da matriz;
- Verificar se a matriz é ou não simétrica;
- Determinar a transposta da matriz;
- Calcular a soma de duas matrizes dadas.

Cada tarefa deve ser realizada por uma função.