# MEFT - Programação

# 1º Ano - 1º Semestre de 2015/2016

# Série 4 (02/11/2015)

**1.** Considere a função de Bernoulli[1] $x_{n+1} = 2\,x_n \pmod 1$.
a) Construa um programa em C que calcule as primeiras 100 iteradas da função começando com uma condição inicial fornecida pelo utilizador. O programa, para além de escrever os valores no ecran, deve igualmente escrever os resultados num ficheiro externo chamado "*Bernoulli.dat*" e o formato deve ser o seguinte: "(número da iterada) (6 espaços) (valor de x com 10 casas decimais)".
b) Calcule à mão as 30 primeiras iteradas e compare os resultados obtidos com os fornecidos pelo seu programa.

**2.** Construir um programa que executa as seguintes operações com números complexos: **módulo**, **adição**, **diferença**, **produto** e **divisão**. A entrada de dados deve ser na forma:

$$./programa < Operacao > < Re\ z_1 > < Im\ z_1 > < Re\ z_2 > < Im\ z_2 >$$

A representação dos complexos no programa deve ser feita através de estruturas. Quando definir a estrutura crie também um nome alternativo usando 'typedef'. Deverá ser criada uma função para executar cada uma das operações pedidas. A transferência das estruturas para as funções (e das funções) deve ser feita com ponteiros. A impressão dos resultados deverá ser feita na função main e não nas funções que execução as operações. Caso o programa seja executado sem argumentos, deverá apresentar um pequeno menu explicativo.

**Nota:** No caso do módulo o programa deve receber apenas um complexo.

**3.** Tabela Periódica de Elementos. Construa um programa que contém a informação referente aos 18 primeiros elementos da tabela periódica (do Hidrogénio ao Argon). A cada elemento deve associar o nome, o símbolo químico, o número atómico, a massa atómica, os pontos de fusão e de ebulição, o raio atómico e a energia de 1ª ionização. Pretende-se:
a) Dado o nome de um elemento, receber toda a informação que lhe está associada;
b) Dado o nome de um elemento e uma propriedade específica, receber o seu valor para esse elemento.
c) Dada uma propriedade específica, receber essa informação para todos os elementos.

**Atenção:** Utilize um vector de estruturas para a organização interna da informação no programa. Faça uma legenda suficientemente clara para a sua utilização. Pesquise na *net* 'links' para encontrar a informação em causa. As ordens devem ser transmitidas ao programa a partir da linha de comandos.

(v.s.f.f.)

---

[1] (mod 1) significa eliminar a parte inteira de um número real.

Para decompor um real (double) nas partes inteira e fraccionária pode usar-se a função:

double modf (double *value*, double **integer-part*);

em que *value* é o número que se pretende dividir, *integer-part* é o ponteiro para a parte inteira e o retorno é a parte fraccionária. Funções idênticas são igualmente definidas para *float* (**modff**) para *long double* (**modfl**).

**4.** Construa um programa que lê um ficheiro de texto e o reescreve noutro ficheiro. Podendo executar as seguinte operações de acordo com o pedido efectuado (ignore as cedilhas e vogais acentuados):
a) Passar todo o texto a minúsculas ('M2m');
b) Passar todo o texto a maiúsculas ('m2M');
Escreva o programa de maneira a poder executar estas operações do seguinte modo:

*./programa* M2m *file_de_leitura* *file_de_escrita*
*./programa* m2M *file_de_leitura* *file_de_escrita*

**Nota:** Para a leitura do ficheiro do problema **4**, poderá ser usada a função **fgets**:

char * fgets (char *s, int count, FILE *stream)

A função **fgets** lê no máximo "count - 1" duma linha de texto do ficheiro (incluindo o caracter nova linha), guarda-os na string "s" e coloca no final o terminador de string. Caso se encontre no final do ficheiro, é retornado o ponteiro NULL e o conteúdo de "s" não é alterado.