Ficha de Trabalho n.º 1 (Extensão)

Objectivos: Revisão sobre ponteiros, memória dinâmica; utilização de ficheiros binários, métodos de ordenação e pesquisa.

Observação: Para fazer os exercícios abaixo, devem previamente ter feito a Filha 1, **pretende-se que os alunos façam pesquisas na net de modo a serem mais autónomos na chegada à solução**.

Problema: Considere a estrutura de dados **REGISTO_UTILIZADORES** e um ficheiro binário com a designação "Utilizadores.dat" que contêm registos de utilizadores de um Site. Elabore um programa organizado em funções que disponibilize um conjunto de funcionalidades que terão como dados de trabalho um ficheiro de registos de utilizadores com muitos registos do tipo REGISTO_UTILIZADORES, com a informação apresentada na tabela abaixo.

Obs. A informação dos utilizadores deve depois ser registada em memória num vector/array, alocando somente o espaço necessário!

```
typedef struct
{
        int dia;
        int mes;
        int ano;
}DATA;
```

```
typedef struct
{
        char nome[50];
        char utilizador[20];
        char password[20];
        float joia;
        DATA data_registo;
        char email[30];
        char pagina_web_pessoal[50];
        int telemovel;
        int numero_acessos;
        DATA data_ultimo_acesso;
}REGISTO_UTILIZADORES, USER;
```

Implemente as seguintes funcionalidades, criando as funções com o nome indicado:

1. Função para gravar para XML os dados do vector; (Os alunos devem pesquisar o que é o formato XML e conversar com o docente acerca da importância de Standarts….)
   a. int **GravarXML**(USER *V, int N, char *ficheiro)

2. Função para converter todos os nomes dos utilizadores para maiúsculas;
   a. void **ToLetrasGrandes**(USER *V, int N)

3. Listar os USERS e devolver o seu número onde o seu nome contem outro nome; Exemplo (ListarUsersContain(V, N, "Miguel") // lista e devolve todos os nomes que contêm "miguel", maiúsculas ou minúsculas é indeferente!
   a. int **ListarUsersContain**(USER *V, int N, char *subnome)

4. Em cada função que implementaram seria interessante gravar num ficheiro o tempo de execução de cada. (usar a função clock()!) Esse ficheiro poderia chamar-se "performance.csv" onde estaria por exemplo o nome de cada função e o seu tempo de execução.
   Ordenacao; 123 ms
   Listar; 23 ms

5. Este exercício tem um grau de dificuldade elevado!: Determinar qual o nome (parte do nome) é mais comum!
   a. char ***NomeMaisComum**(USER *V, int N)