Laboratórios de Informática III (LI3)

LEI – 2º ano – 2º semestre – Universidade do Minho – 2013/2014

F. Mário Martins, João Luís Sobral

Projecto de C:

Criação, Gestão e Consulta de um Catálogo de Publicações

1.- Introdução e Objectivos.

O projecto de C da disciplina de LI3 de LEI tem por objectivo fundamental ajudar à consolidação experimental dos conhecimentos teóricos e práticos adquiridos nas UCs de Programação Imperativa, onde foi leccionado o paradigma imperativo de programação usando a linguagem C, de Algoritmos e Complexidade, onde foram leccionadas as mais importantes estruturas de dados e respectivos algoritmos associados à sua manipulação, e da disciplina de Arquitectura de Computadores, na qual questões de optimização de código foram salientadas.

Os objectivos deste trabalho (e de LI3) não se restringem apenas a aumentar os conhecimentos dos alunos na linguagem C, o que seria até questionável, mas, e talvez fundamentalmente, apresentar aos alunos, de forma pragmática, quais os desafios que se colocam a quem concebe e programa aplicações software (em qualquer linguagem), quando passamos a realizar a designada **programação em larga escala**, ou seja, aplicações com grandes volumes de dados e com mais elevada complexidade algorítmica e estrutural.

De facto, quando passamos para tais patamares de complexidade, torna-se imperioso conhecer e usar os melhores princípios da Engenharia de Software, de modo a que tais projectos de software, em geral realizados por equipas, possam ser concebidos com melhor estrutura, de modo a que sejam mais facilmente modificáveis, e sejam, apesar da complexidade, o mais optimizados possível a todos os níveis.

Para que tal seja possível, teremos que introduzir novos princípios de programação, mais adequados à programação em grande escala, designadamente:

- Modularidade e encapsulamento de dados;
- Criação de código reutilizável;
- Escolha optimizada das estruturas de dados;
- Testes de *performance*.

O projecto a desenvolver em trabalho de grupo (de no máximo 3 alunos) visa a experimentação e aplicação destas práticas de desenvolvimento de software usando a linguagem C (mas que são extensíveis a outras linguagens e paradigmas).

2.- Requisitos do programa a desenvolver.

O projecto a desenvolver em C (usando **gcc**), de nome **GESTAUTS**, tem como fonte de dados um ficheiro de texto no qual cada linha representa os nomes dos autores de uma dada publicação científica publicada no ano indicado.

Cada linha do ficheiro designado por publicx.txt conterá os nomes dos autores de tal publicação separados por vírgulas, sendo terminada pelo ano da respectiva publicação. Para todos os efeitos, deve considerar-se que todas as linhas foram "bem formadas" pelo programa que as criou, ou seja, os nomes dos autores estão correctos, não existem caracteres especiais entre estes a não ser as vírgulas e espaços, o ano poderá ser correctamente convertido para um inteiro (caso não o seja a linha deverá ser aceite e o ano considerado 0) e a linha é terminada pelo carácter de fim de linha do sistema onde foi criada.

Apresentam-se em seguida alguns exemplos destas linhas:

```
Kim-Xang, Alan C. Carter, 2001
John Kalmus, Gustavo Paz, João C. Lopes, 1991
K. Dix, 2013
```

Em termos estritamente numéricos, teremos portanto um ficheiro com um número garantidamente elevado de linhas, certamente mais de 123.000, correspondendo a um número total de nomes a ler da ordem dos 330.000, dos quais cerca de 150.000 serão nomes distintos.

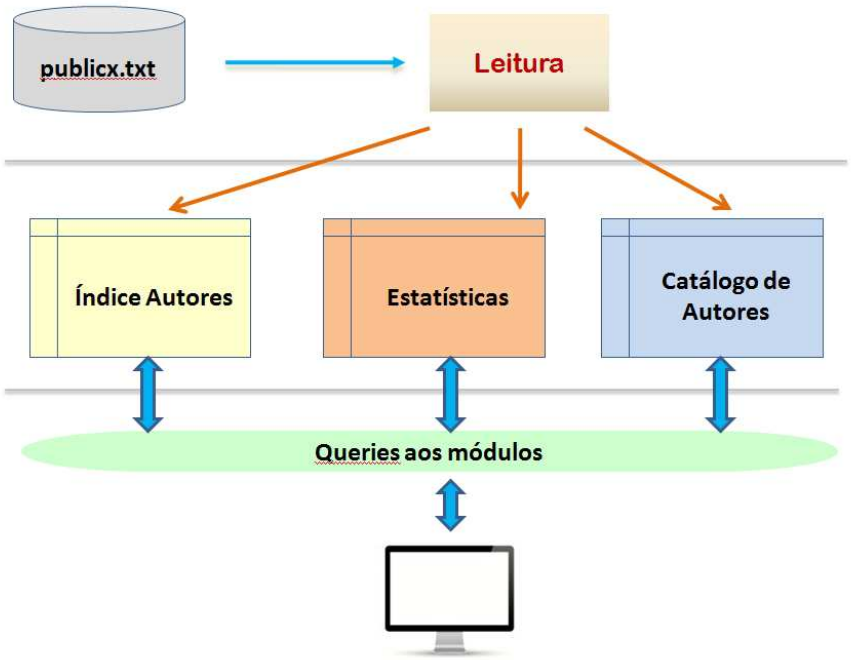
Quantitativamente, estes são os números iniciais quanto ao volume de informação a tratar. Porém, o importante são as decisões quanto à forma do seu armazenamento em memória em função do que pretendermos realizar com tais dados.

2.1.- Arquitectura da aplicação.

No sentido de se deixar desde já uma orientação de base quanto à arquitectura possível da aplicação a desenvolver, pretende-se de facto que a aplicação possua uma arquitectura na qual, tal como apresentado na figura seguinte, se identifiquem de forma clara os seguintes módulos:

- **Leitura:** função ou parte do código de `main()` no qual é realizada a leitura e eventual contabilização e tratamento adequado dos dados das linhas do ficheiro publicx.txt;

- **Índice de Autores:** módulo de dados onde deverão ser guardados os nomes de todos os autores, organizados por índice alfabético, que irá permitir, de forma eficaz, saber quais são os autores cujos nomes começam por uma dada letra do alfabeto, quantos são, comprimentos dos nomes, etc.;
- **Estatísticas:** módulo de dados que irá conter as estruturas de dados responsáveis pela resposta eficiente a questões quantitativas que relacionam anos e número de publicações, tais como número de artigos publicados em cada ano, número de artigos com um dado número de co-autores em cada ano, etc.;
- **Catálogo de Autores:** módulo de dados que conterá as estruturas de dados adequadas à representação quantitativa do total de artigos de um dado autor (por ano), e dos relacionamentos existentes entre autores, designadamente, questões como, quais os co-autores de um dado autor, quantos artigos dois autores publicaram em conjunto no total, total de co-autores de dado autor, etc.;



2.2.- Queries interactivas.

Tendo sido apresentada a arquitectura genérica da aplicação, a efectiva estruturação de cada um dos módulos depende, naturalmente, da funcionalidade esperada de cada um deles. Tal é, naturalmente, completamente dependente das *queries* que a aplicação deve implementar para o utilizador final.

Tal como a figura indica, vamos nesta secção apresentar todas as funcionalidades que o utilizador da aplicação pode invocar, ou seja, toda a funcionalidade que a aplicação deverá implementar nos seus módulos.
Deste modo, e fornecida que foi uma arquitectura de referência, deixa-se ao critério dos grupos de trabalho a concepção das soluções, módulo a módulo, para a satisfação da implementação de cada uma das *queries* que podem ser realizadas pelo utilizador e, até, a sua adequada estruturação sob a forma de menus, etc.

Assim, e sem qualquer ordem em particular, o **GESTAUTS** deve ser capaz de dar ao utilizador respostas eficazes às seguintes solicitações:

1. Ler um ficheiro de texto de nome indicado (cf. **publicx.txt**), em que cada linha tem a estrutura anteriormente indicada, apresentando de imediato o nome do ficheiro lido, o número total de linhas lidas (ou seja, publicações), o número total de nomes lidos e o intervalo fechado dos anos das publicações (cf., por exemplo, Anos = [1990 a 2012]);
2. Apresentar uma tabela com o número total de publicações registadas em cada ano a considerar, por ordem crescente do ano em questão.
3. Dado um ano e um autor existente, determinar o número de publicações desse autor em tal ano;
4. Determinar o número total de autores que nunca publicaram com outros autores, ou seja, apenas publicaram a solo;
5. Dado um autor, criar uma tabela com o número de artigos publicados ano a ano (para anos em que não publicou a entrada deverá ficar a 0);
6. Determinar a lista de todos os autores cujo nome se inicia pela letra dada como parâmetro (maiúscula ou minúscula deverá ser indiferente);
7. Dado um intervalo fechado de anos, por exemplo de 2000 a 2006, determinar o total de publicações registadas nesse intervalo;
8. Dado um nome de um autor, determinar o nome ou nomes dos co-autores com quem mais publicou;
9. Dado um intervalo fechado de anos, determinar a lista de nomes dos autores que publicaram em todos esses anos;
10. Determinar, para um dado ano, uma tabela que contenha o número total de publicações de 1, 2, ou 3 autores; Tal tabela deverá ser guardada num ficheiro de nome dado e também apresentada em ecrã ao utilizador;

11. Criar um ficheiro em formato CSV (compatível com Excel), contendo para cada ano em que há publicações registadas, o número de artigos com 1, 2, 3, etc. autores, satisfazendo assim o formato final exemplo (com linha de cabeçalho):

```
"Ano","#Autores","#Artigos"
"1971","1","3"
"1971","2","10"
....
"2012","10","2"
```

12. Determinar, para um dado ano, a lista dos N autores (N dado pelo utilizador) com maior número de publicações, com indicação de tal número;
13. Dado um nome de autor e um ano, determinar a sua % de publicações relativamente ao total de publicações desse ano;
14. Determinar a média dos comprimentos dos nomes dos autores, bem como os nomes de maior e menor comprimento.

3.- Testes de performance.

Depois de desenvolver e codificar todo o seu projecto tendo por base o ficheiro **publix.txt**, deverá realizar alguns testes de *performance* e apresentar os respectivos resultados. Pretende-se comparar os tempos de execução dos *queries* 8, 9 e 12, usando os ficheiros, **publicx.txt**, **publicx_x4.txt** e **publicx_x6.txt**. Todos os ficheiros serão fornecidos numa pasta disponibilizada via BB.

4.- Requisitos para a codificação final.

A codificação final deste projecto deverá ser realizada usando a linguagem C e o compilador **gcc**. O código fonte deverá compilar sem erros usando o *switch* **-ansi**. Podem também ser utilizados *switches* de optimização. Para a correcta criação das *makefiles* do projecto aconselha-se a consulta do utilitário **GNU Make** no endereço **www.gnu.org/software/make.**

Qualquer utilização de bibliotecas de estruturas de dados em C deverá ser sujeita a prévia validação por parte da equipa docente. Não são aceitáveis bibliotecas genéricas tais como LINQ e outras semelhantes.

O código final de todos os grupos será sujeito a uma análise usando a ferramenta **JPlag**, que detecta similaridades no código de vários projectos, e, quando a percentagem de similaridade ultrapassar determinados níveis, os grupos serão chamados a uma clara justificação para tal facto.

5.- Apresentação do projecto e Relatório.

O projecto será submetido por via electrónica num *site* do DI a indicar oportunamente (bem como o formato da pasta e a data e hora limite de submissão). Tal *site* garantirá quer o registo exacto da submissão quer a prova da mesma a quem o submeteu (via e-mail). Tal garantirá extrema segurança para todos.

O código submetido na data de submissão será o código efectivamente avaliado. A *makefile* deverá gerar o código executável, e este deverá executar correctamente. Projectos com erros de *makefile*, de compilação ou de execução serão de imediato rejeitados.

Para além do código do projecto, cada grupo deverá elaborar um relatório no qual descreverá o projecto e todas as suas decisões de concepção do mesmo, designadamente:

- Descrição de cada módulo em termos de API (.h), respectivas estruturas de dados (apresentar um desenho destas) e justificação para a escolhas das mesmas;
- Arquitectura final da aplicação e razões para tal modularidade;
- Complexidade das estruturas e optimizações realizadas;
- Resultados dos testes realizados.

O relatório será entregue aquando da apresentação presencial do projecto e servirá de guião para a avaliação do mesmo.

A avaliação presencial do projecto implica a presença de todos os membros do grupo de trabalho, sob pena de reprovação imediata em caso de ausência injustificada. Para além da análise e avaliação da solução, em termos estruturais e de eficácia de execução, e do relatório, alguma avaliação individual poderá ser realizada quando tal se justificar.

Uma avaliação final inferior a 10 valores neste primeiro projecto implica a reprovação imediata à UC de LI3.

Todas as questões adicionais deverão ser esclarecidas ao longo das aulas junto da equipa docente.

F. Mário Martins