Programação

Folha Prática 9

Lab. 9

Departamento de Informática
Universidade da Beira Interior
Portugal
2015

LAB. 9

9ª semana

FICHEIROS E ENTRADA/SAÍDA (I/O)

1. Revisão.
2. Objectivos.
3. Exercícios

Lab. 9

FICHEIROS E ENTRADA/SAÍDA (I/O)

1. Revisão

1. Biblioteca de funções de I/O em C:
   1.1. Função de abertura de ficheiro:

   fopen

   1.2. Funções de leitura de ficheiro:

   fread, fgetc, fscanf, fgets

   1.3. Funções de escrita em ficheiro:

   fwrite, fputc, fprintf, fputs

   1.4. Funções de posicionamento em ficheiro:

   fseek, ftell, frewind

   1.5. Função de fecho de ficheiro:

   fopen

2. Ligações Web úteis:
   - http://en.wikibooks.org/wiki/C_Programming/File_IO
   - http://www.codingunit.com/category/c-tutorials
   - http://www.codingunit.com/c-tutorial-file-io-using-text-files
   - http://www.codingunit.com/c-tutorial-binary-file-io

2. Objectivos

No final deste módulo prático, o aluno deve ser capaz de:

1. Guardar dados em ficheiros através de um programa em C, quer na forma de texto quer na forma binária, bem como ler aqueles mesmos dados a partir dos referidos ficheiros e carregá-los em memória.

2. Programar pequenas aplicações de software em C com o recurso à utilização de ficheiros.

3. Exercícios

Exercício 9.1 (Cópia de ficheiros)

Escreva um programa que copie um ficheiro de texto para outro. Os nomes dos dois ficheiros são passados como argumentos do programa principal. O ficheiro original é aberto no modo de leitura, ao passo que o ficheiro destino é aberto no modo de escrita.

Exercício 9.2 (Saída das últimas *n* linhas dum ficheiro de texto)

Escreva um programa que escreva no ecrã as últimas n linhas de um ficheiro de texto. O valor de n e o nome do ficheiro deverão ser passados como argumentos do programa principal. Por exemplo,

```
last –n file.txt
```

em que *last* é o nome do programa executável.

Exercício 9.3 (Linhas de ficheiro com uma dada palavra)

Escreva um programa que escreva no ecrã as linhas de um dado ficheiro que contêm uma dada palavra. Esta palavra e o ficheiro são passados como argumentos do programa principal. Este programa é, afinal de contas, uma versão simples do comando *grep* do Linux.

Exercício 9.4 (Comparação de ficheiros)

Escreva um programa que compare dois ficheiros de texto e que escreva no ecrã as linhas distintas de cada um deles.

Sugestão: Utilize as rotinas de bibliotecas de manipulação de strings e de ficheiros. O programa não será seguramente muito longo.

Exercício 9.5 (Listagem de ficheiro com pausa)

Escreva um programa que escreva no ecrã as linhas de um dado ficheiro, parando de 20 em 20 linhas após premir na tecla <space> ou na tecla <return>.

Este programa é, afinal de contas, uma versão simples do comando *more* do Linux.

Exercício 9.6 (Concatenação de ficheiros de texto)

Escreva um programa que faça a concatenação e fusão de dois ficheiros de texto num terceiro ficheiro a ser criado durante a execução do programa.

Exercício 9.7 (Armazenamento de reais em ficheiro binário)

Escreva um programa que guarde uma sequência de números reais num ficheiro binário, pelo que se assume que estes números são introduzidos via teclado pelo utilizador. Num passo subsequente, esta sequência de números deve ser lida a partir do ficheiro binário e enviada para o ecrã, mas agora na forma de texto.

OBRIGATORIEDADE: use as funções de I/O não-formatada.

Exercício 9.8 (Aplicação para gestão de pessoal duma empresa)

Desenhe e construa uma aplicação interactiva que permita fazer a gestão do pessoal duma empresa. A aplicação deve, através de um menu, executar as seguintes tarefas:

1. Cadastrar os funcionários, sendo que a informação do registo de cada funcionário deve incluir pelo menos o nome, a categoria, o salário, o seu número de entrada na empresa e a data de entrada na empresa;
2. Apresentar os registos dos funcionários pelo número de entrada na empresa;
3. Procurar um funcionário pelo seu número de entrada;
4. Apresentar por ordem crescente os registos dos funcionários que recebem salários superiores a um dado valor em euros;
5. Apresentar por ordem crescente os registos dos funcionários que recebem salários inferiores a um dado valor em euros;
6. Sair da aplicação de cadastro.

Este exercício é uma extensão ao Exercício 9.8 na medida em que a informação é guardada de forma

permanente em ficheiro. Ou seja, os registos dos funcionários são agora guardados em disco. Portanto, sempre que o programa é executado, deve copiar a informação dos funcionários em disco para o array de funcionários existentes em memória RAM. Quando se efectua a operação de SAVE na aplicação, faz-se a cópia inversa, ou seja, do array de funcionários para um ficheiro em disco.

OBRIGATORIEDADE: use as funções de I/O não-formatada.

Exercício 9.9 (Aplicação de cálculo da média final de licenciatura)

Escreva uma aplicação em C que permita a um aluno de licenciatura calcular a sua média final de curso. A aplicação deve incluir um menu a partir o qual de pode inserir e eliminar disciplinas do plano de estudos, bem como introduzir as notas obtidas nas diferentes unidades curriculares.

Este exercício é uma extensão ao Exercício 9.9 na medida em que a informação é guardada de forma permanente em ficheiro. O plano curricular do curso é guardado em ficheiro segundo o formato <nome_disciplina, ano, semestre, créditos>. Quando o utilizador/aluno pretende calcular a sua média final, deve carregar este plano de estudos em memória, e inserir depois a nota de cada disciplina.

OBRIGATORIEDADE: use as funções de I/O não-formatada.

Exercício 9.10 (Aplicação interactiva para gestão de livros)

Considere o tipo struct **LIVRO** e um array de structs do tipo LIVRO com tamanho 100 identificado por biblioteca:

```
typedef struct
{
        char    titulo[40];
        char    autor[20];
        float   numeroDoCatalogo;
        int     anoDePublicacao;
        int     numeroDaCopia;
} LIVRO;

#define numeroLivros 100

LIVRO biblioteca [numeroLivros];
```

As tarefas principais a desenvolver nesta aplicação são as seguintes:

1. Altere a estrutura LIVRO por forma a incluir dois campos adicionais: dataEmprestimo (tipo DATE a definir) e dataRetorno (tipo DATE).

2. Escreva uma função de saída do seguinte menu:

```
BIBLIOTECA DA UBI
1 – Inserir novo livro
2 – Procurar livro por título.
3 – Procurar livro por autor.
4 – Alterar nome do livro
5 – Apagar livro
6 – Registar data de empréstimo de um livro.
7 – Registar data de retorno de um livro.
8 – Listar livros existentes na biblioteca.
9 – Sair
```

3. Coloque todas as opções do menu a funcionar (o programa só deverá terminar quando o utilizador seleccionar a opção "Sair").

   Para isso escreva funções para:

   - Ler um livro.
   - Mostrar os dados de um livro.
   - Inserir um livro.
   - Procurar um livro através do seu título.
   - Procurar um livro através do nome do seu autor.
   - Alterar o título de um livro.
   - Eliminar um livro dos registos da biblioteca.
   - Registar a data de empréstimo de um livro.
   - Registar a data de retorno de um livro.
   - Listar todos os livros existentes.

Este exercício é uma extensão ao Exercício 9.10 na medida em que a informação dos livros deve ser guardada de forma permanente em ficheiro.

OBRIGATORIEDADE: use as funções de I/O não-formatada.