Sistemas Operativos 2006/2007

Ficha Prática Nº1 – Programação em C

Introdução

A linguagem de programação C é caracterizada por apresentar uma gramática simples e de dimensões reduzidas, por permitir o acesso directo a funções de chamada ao sistema, ser uma linguagem estruturada, permitir a manipulação de dados a baixo nível (*bit level*) e fazer um uso extensivo de ponteiros de memória. O C é uma linguagem capaz de produzir programas extremamente eficientes mas, por ser fracamente tipada e não possuir mecanismos robustos para detecção e tratamento de erros, o C pode muito facilmente permitir a introdução de faltas em programas que poderão vir a originar erros fatais durante a execução.

Objectivos

Após a conclusão deste trabalho o aluno deverá ser capaz de:

- Executar um conjunto de comandos essenciais para o desenvolvimento de programas em sistemas operativos Linux.
- Utilizar diversas ferramentas de desenvolvimento e programação para a linguagem C, e.g. gcc, gdb, ddd.
- Possuir conhecimentos de programação em C com ponteiros.
- Possuir conhecimentos sobre os mecanismos de acesso (leitura e escrita) a ficheiros em C.
- Saber implementar e manipular uma lista ligada.
- Opcional: saber escrever uma *makefile* e executar o comando `make`.

Material de Apoio

- Do livro *"Unix Systems Programming: Communication, Concurrency, and Threads"*, Kay A. Robbins e Steven Robbins editado pela Prentice Hall ler:
  - Capítulo 4 - *Unix I/O*
  - Apêndice A - *Unix Fundamentals*
- Do livro *"The C Programming Language, 2nd Edition (Ansi C)"* de by Brian W. Kernighan, Dennis M. Ritchie editado pela Prentice Hall ler:
  - Capítulo 5 - *Pointers and Arrays*
  - Capítulo 7 - *Input and Output*
  - Capítulo 8 - *The Unix System Interface*
- Do livro "*Programming in C and Unix"*, disponível em http://gd.tuwien.ac.at/languages/c/programming-dmarshall/ ler:
  - *C/C++ Program Compilation*
  - *Pointers*
  - *Dynamic Memory Allocation and Dynamic Structures*
  - *Advanced Pointer Topics*
  - *Input and Output (I/O):stdio.h*
- Apontamentos sobre comandos básicos de Unix
- Apontamentos sobre acesso a ficheiros em C
- Apontamentos sobre *Makefiles*

```c
#include <stdio.h>
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>

#define N 10
#define FILE_NAME "list.dat"

struct item {
   int value;
   struct item *next;
};
typedef struct item element;

void create_random_list(element **list_head);
void free_list(element *list_head);
void save_list(element *list_head, char* file_name);
void read_list(element **list_head, char* file_name);
void print_list(element* list_head);
void reverse_list(element** list_head);

int main() {
        element *list_head;
        list_head = NULL;
        create_random_list(&list_head);
        print_list(list_head);
        save_list(list_head, FILE_NAME);
        free_list(list_head);
        list_head = NULL;
        read_list(&list_head, FILE_NAME);
        print_list(list_head);
        free_list(list_head);
        exit(0);
}

void create_random_list(element **list_head) {
        element *curr;
        int i;
        srandom(getpid());
        for(i = 0; i < N; i++) {
                curr = (element *)malloc(sizeof(element));
                curr->value = random()%N;
                curr->next  = *list_head;
                *list_head = curr;
        }
        return;
}
```

O código apresentado na caixa de texto no topo da página pertence a um conjunto de ficheiros que implementam diversas funcionalidades para gestão de listas ligadas em C.

Na caixa de texto é apenas visível a implementação de dois métodos, o método `int main()` e o métod `void create_random_list(element **list_head)`. O seu objectivo para este trabalho será implementar os restantes métodos da biblioteca de forma a conseguir executar o código em `int main()`.

1. Comece por copiar o código aqui apresentado para um ficheiro com o nome `llist.c`. De seguida implmente o método `void free_list(element *list_head)` de forma a que este liberte o espaço de memória reservado para conter os elementos da lista ligada com início em `list_head`.
2. Implemente o método `void print_list(element* list_head)` de forma a que este exiba no ecrã o conteúdo da lista liga de uma forma perceptível para os utilizadores.

3. Implemente o método `void reverse_list(element** list_head)` de forma a que este realize a inversão da lista ligada referênciada por `list_head`.
4. Implemente o método `void save_list(element *list_head, char* file_name)`. Este método deve escrever todos os elementos da lista ligada apontada por `list_head` num ficheiro com o nome definido em `file_name`. Se o ficheiro em questão não existir deve ser criado, caso contrário, a informação existente deve ser eliminada antes da nova lista ser escrita.
5. Implemente o método `void read_list(element **list_head, char* file_name)`. Este método deve ler todos os elementos de uma lista ligada a partir de um ficheiro com o nome `file_name` e construir a lista em memória, sendo que, list_head será o ponteiro para o primeiro elemento da lista.

   Nota: Tenha em consideração a ordem pela qual os elementos da lista foram escritos no ficheiro (ou a ordem pela qual os lê) pois pode ser necessário inverter a lista após a sua leitura.

Tarefa Final

De forma a testar as funções que acabou de implementar execute a função `main()`. O output obtido deverá ser semelhante a:

```
$ ./llist
4 5 1 5 7 4 2 9 8 6
4 5 1 5 7 4 2 9 8 6
$ ./llist
7 3 1 0 4 7 5 5 1 5
7 3 1 0 4 7 5 5 1 5
```

Instruções para a submissão das respostas

- Deve guardar o código em C que escrever num ficheiro com o nome llist.c.
- No mesmo ficheiro, de preferência no início, deve indicar o nome dos elementos do grupo e o tempo gasto na realização do trabalho.
- O ficheiro que enviar será compilado pelos professores e executado. Todos os requisitos extra (bibliotecas, *makefiles*, outros ficheiros, …) que entenda serem necessários para se realizar uma compilação e execução com sucesso do programa, devem ser indicados no início do ficheiro.
- O ficheiro deve ser enviado por e-mail para ambos os docentes das práticas (bcabral@dei.uc.pt;nuno@dei.uc.pt).
- O *subject* do e-mail deverá ser: **SO_F1_P.X**, sendo que, **X** deverá ser substituído pelo número de identificação da turma prática a que os alunos pertencem.
- O prazo limite para a entrega é 25 de Setembro, trabalhos recebidos posteriormente não serão aceites.
- Qualquer tentativa de fraude será punida de acordo com os estatutos da FCTUC podendo resultar na reprovação à disciplina, de todos os alunos envolvidos, com a classificação de 0 valores.