UM–SJTU Joint Institute

Introduction to Operating Systems
(VE482)

Laboratory Report

Lab 4

Name: Ji Xingyou ID: 515370910197
Date: 26 October 2017

Contents

1 Layer programming

- **The program can be divided into three layers, what are they?**

  These three layers are separated as follows:

  1. Implement the list.
  2. Read the argv[] and generate the correct file name. Traverse the content of the file, inserting into list and sort them according to the sorting mode. Generate the output file name and write the sorting result into the output file
  3. Implement the main function.

- **Split the program into files according to the defined layers.**

  Done.

- **Create the appropriate corresponding header files.**

  For the list implementing layer, all the codes are put in list.h. Its implementation is in list.c.

  For the reading and generating file name layer, all the codes are put in readName.h. Its implementation is in readName.c.

  For the sorting layer, all the codes are put in sort.h. Its implementation is in sort.c.

  For the writing into output file layer, all the codes are put in writeFile.h. Its implementation is in writeFile.c.

- **If necessary rewrite functions such that no call is emitted from lower level functions to upper level functions.**

  Luckily, I did not need to rewrite any functions. But in order to make the division among all layers clear, I reorganize the order of my codes and made some new combinations.

- **The initial program implements a command line interface, write a Menu interface which (i) welcomes the user, (ii) prompts him for some task to perform, and (iii) runs it. When a task is completed the user should (i) be informed if it was successful and then (ii) be displayed the menu. From the menu he should be able to exit the program.**

  When I finish dividing the layer, it is very easy to write another main.c in order to support the "Menu" function. For the prompt information, I intentionally grab the lines from the famous horror movie Saw in order to tense the atmosphere.

- **Write two main functions, one which will dispatch the work to another function which will run the command line user interface and a second one which will dispatch the work to the Menu user interface.**

  Already done.

2 Libraries

- **What are the three stages performed when compiling a file?**

  The three stages are listed as follows:

  1. Preprocessing: generate the preprocessing file.

  2. Building: generate the assembly code file.

  3. Link: generate the executable file.

- **Briefly describe each of them.**

  Already do that part in the previous question.

- **Explain the difference between the static and dynamic libraries.**

  For the static libraries, they will be linked to the target codes when compiling. When the program is running, they will no longer be needed.

  For the dynamic libraries, they will not be linked to the target codes when compiling. Instead, they are loaded into the target codes when the program is running. Therefore, it is obvious they are needed when the program is running.

- **Create two static libraries, one for each of the two lowest layers in the previous program.**

```
add_library(lab4_list_static STATIC list.c)
add_library(lab4_sort_static STATIC sort.c)
```

- **Compile the command line version of the program using these two static libraries.**

```
add_executable(lab4_main_static main.c)
target_link_libraries(lab4_main_static lab4_list_static lab4_sort_static)
```

- **Generate two dynamic libraries, one for each of the two lowest layers in the previous program.**

```
add_library(lab4_list_dynamic SHARED list.c)
add_library(lab4_sort_dynamic SHARED sort.c)
target_link_libraries(l4_list_dynamic l4_sort_dynamic)
```

- **Compile the whole program.**

```
add_executable(lab4_1 main.c list.c sort.c)
add_executable(lab4_2 main.c list.c sort.c)
```

- **Compile the Menu version of the program using these two dynamic libraries.**

```
add_executable(lab4_2 main.c)
target_link_libraries(lab4_2 lab4_sort_dynamic lab4_list_dynamic)
```

- **What is the difference between a library and the API.**

  Library is some basic functions supported by some certain programming languages.

  API is some functions provided by the operating system in order to make it easy and convenient for programmers to implement certain functions.