CS 2050 – Lab 1 – Fall 2013

File I/O, C Structures, Command line argument

**Directions:**

Complete the following lab assignment using the description given in each section.

**Purpose:**

* Use File I/O
* Use command line argument
* Use C structures and arrays

**Submission information:**

Submit this assignment by following the instructions given by your lab assistant. It must be submitted before the end of the lab USING THE CODE THE TA GIVES YOU. Not doing so will result in a zero.

**Description:**

In this assignment you have to use file as input, and write your response to another file. C structures must be used in the form of array. For reading and writing files we recommend to use **fscanf()** and **fprintf()**. These two built-in functions are similar to reading and writing input from console.

Imagine you work for the department of history. Currently the department is studying history of roman emperors. They have initial input about various emperors and need your help, to identify the longest living emperor, the youngest to die, and the longest to rule. For simplicity we only consider emperors born Anno Domini (AD) according to Julian Calendar. The input is provided in the text file. First line of the file contains the number of emperors listed, rest of the text is formatted as follows

Name of emperor , Year of birth, Year of death, Years to rule

Your assignment is to read the list from file, store them into the structure, that holds all the fields described above. Then scan the list and find emperor that lived longest, lived shortest, and ruled longest. Write the summery into the output file.

Data structure, C constants and function headers are provided below. The history department has asked you to stick with the exact format, since they might hire more programmers later to work on the project. In other words, you will lose points for not following exact format of function headers.

**#define MAX 15** – maximum number of emperors listed in a single file

**#define MAX\_LENGTH 30** – maximum length of the emperor’s name

Define structure to store emperors information

**typedef struct{**

**char name[MAX\_LENGTH];**

**int birth;**

**int death;**

**int reign;**

**} data;**

**data emperors[MAX]; -** Create global array to have access from any function

**FILE \*in, \*out; -** Create two global file pointers for input and output files

**int openFiles(char\* input, char\* output)-** This function should take argv[1] and argv[2] as input and output file names. Return -1 if files couldn’t be opened. Open input in read mode, respectively - output in write mode.

**int readContent() -** function reads whole file and returns the number of emperors listed in the file. This number is stored on the first line of the text file.

**int emperor\_to\_live\_longest (int size)-** Scan through the list of emperors, and find the oldest emperor to die. Return the index of the array (NOT the age) back to main function.

**int youngest\_emperor\_to\_die** **(int size)-** Scan the array and find the youngest emperor to die. Similarly to previous function, return the index, do not return age of the emperor.

**int longest\_lifetime\_reign(int size)-** Scan the list, find the emperor that spent most of his lifetime in reign. (hint: Age – Years of Reign is smallest) Return the index in the array. With the index you will be able to extract name, and age as well.

**int emperor\_to\_rule\_longest(int size)-** Emperor who spent most of years in reign. As usual only return index to main function.

**void writeFile(int oldest, int youngest, int longest, int successful, int size)**

Once you calculate all the numbers, print the list of emperors into the output file, and lastly the summery required by the history department. It is worth noting first four arguments, passed to the function are the indexes, last argument is the size of the array. Example output will give you better understanding on how to format the output file.

**void closeFiles() -** Lastly close both files and quit the program.

**int main(int argc, char\*\* argv)** – everything you write in the output file, print it on the screen as well. Call functions from main and use command line arguments for input/output.**./a.out**

Enter Input and Output File Names

**./a.out emperors.txt summery.txt**

|  |  |
| --- | --- |
| Example Input  *emperors.txt* | Example Output  *summery.txt* |
| 11  Caligula  12  41  3  Vespasian  9  79  9  Nero  37  68  13  Titus  39  81  2  Trajan  53  117  19  Marcus\_Aurelius  121  180  19  Commodus  161  192  12  Septimius\_Severus  145  211  17  Macrinus  165  218  1  Severus\_Alexander  208  235  13  Diocletianus  244  311  20 | Caligula Birth: 12, Death: 41, Reign: 3  Vespasian Birth: 9, Death: 79, Reign: 9  Nero Birth: 37, Death: 68, Reign: 13  Titus Birth: 39, Death: 81, Reign: 2  Trajan Birth: 53, Death: 117, Reign: 19  Marcus\_Aurelius Birth: 121, Death: 180, Reign: 19  Commodus Birth: 161, Death: 192, Reign: 12  Septimius\_Severus Birth: 145, Death: 211, Reign: 17  Macrinus Birth: 165, Death: 218, Reign: 1  Severus\_Alexander Birth: 208, Death: 235, Reign: 13  Diocletianus Birth: 244, Death: 311, Reign: 20  ---------------  Oldest emperor to die Vespasian, Age: 70  Youngest emperor to die Severus\_Alexander, Age: 27  Emperor to rule longest: Diocletianus, 20 years  Emperor to spend most lifetime in reign: Severus\_Alexander, 13 years in reign. Died at age of 27 |

**Guidelines for Grading Lab 1 (30 Points Possible)**

If your program does not compile, does not link, loops infinitely, segfaults, or fails to produce any output, your lab will receive a grade of ZERO. You might have noted we eliminated spaces in the text file, to support fscanf() function and make file processing easier for you.

**3 points** - int openFiles(char\* input, char\* output);

**5 points** - int readContent();

**3 points** - int emperor\_to\_live\_longest (int size);

**3 points** - int youngest\_emperor\_to\_die(int size);

**3 points** - int longest\_lifetime\_reign(int size);

**2 points** - int emperor\_to\_rule\_longest(int size);

**4 points** - void writeFile(int oldest, int youngest, int longest, int size);

**3 points** - void closeFiles();

**4 points** - int main(int argc, char\*\* argv);

**Bonus (5 points extra):**

Create function **sort(int size).** Under this function print the list of emperors in a sorted order to the output file, from the oldest to the youngest to die. Append the sorted list to the summery. Call this function right before closing input and output files.

Vespasian (70) Birth: 9, Death: 79, Reign: 9

Diocletianus (67) Birth: 244, Death: 311, Reign: 20

Septimius\_Severus (66) Birth: 145, Death: 211, Reign: 17

Trajan (64) Birth: 53, Death: 117, Reign: 19

Marcus\_Aurelius (59) Birth: 121, Death: 180, Reign: 19

Macrinus (53) Birth: 165, Death: 218, Reign: 1

Titus (42) Birth: 39, Death: 81, Reign: 2

Commodus (31) Birth: 161, Death: 192, Reign: 12

Nero (31) Birth: 37, Death: 68, Reign: 13

Caligula (29) Birth: 12, Death: 41, Reign: 3

Severus\_Alexander (27) Birth: 208, Death: 235, Reign: 13