CS 162 LAB #5 – File I/O

In order to get credit for the lab, you need to be checked off by the end of lab. You can earn a maximum of 3 points for lab work completed outside of lab time, but you must finish the lab before the next lab. For extenuating circumstances, contact your lab TAs and the instructor.

This lab is worth 15 points total. Here's the breakdown:

- Part 1: Worksheet (5 pts)
- Part 2: Design for the problem (Group Work: 4 pts)
- Part 3: Implement your design (Individual Work: 6pts)
- Part 4: Extra credit (Optional: 2 pts)

(5 pts) Part 1: Worksheet

This session will be led by your lab TAs. Please follow their instructions, participate, and complete worksheet 5:

https://classes.engr.oregonstate.edu/eecs/winter2024/cs162-001/labs/WS5.docx (pdf version)

(4 pt) Part 2: Design

In this lab activity, you will practice reading and writing from/to a file, a.k.a. File I/O. You can copy and paste or download this example file using wget command:

wget https://classes.engr.oregonstate.edu/eecs/winter2024/cs162-001/labs/input.txt

The input file provides details for a book database in the following format:

Number_of_books
Book_title Pages Author_Name Year_Of_Release
Book_title Pages Author_Name Year_Of_Release
…<Repeats n number of times>…
Book_title Pages Author_Name Year_Of_Release

As a group, create a design for a program that does the following:

1. Open the file (input.txt) using a file object.

2. From the file, read the number of books and each book's information into your program. (Hint, you need to store all books into a dynamic array of struct book, whose size is determined by the number of books from the file)

3. Provide the following two options to the user, and ask for the input:

- Search books by title
- Print all books that have more than 300 pages

4. Based on the user input, process the data from the array, and write the result **to an output file** named `output.txt`. The result should be **appended** to the existing content. For each

book that meets the requirements, use the following format to store/print:

```
Book Title:
Pages:
Author Name:
Year of Release:
```

Each section of information should be labeled in the output file in all capital letters. A book struct should be used to store and manipulate the file information between reading and writing the file. You must include the follow three functions with the exact prototypes:

(Feel free to add more functions when necessary)

- `Book * create_books(int);`
  This function will create the array of books based on the number of books in the file

- `void populate_one_book(Book *, int, ifstream &);`
  This function should read through the `ifstream` that represents your opened file and storing the information of **a single book** at the provided index of the pre-allocated array of books that's passed in.
  Parameters:
  Book* : a pointer to the book array
  int: the index of book to be populated
  ifstream &: file to read from

- `void delete_books(Book *&);`
  This function will delete all dynamic memory created in your program.
  Note: don't forget to set the original 1D array pointer to NULL/nullptr inside this function.
  Question: Why do we need to pass the pointer by reference?

Your **main function needs to check to make sure the file you open exists** before moving forward. If the file doesn't exist, then you need to provide an error message and exit the program.

- (1 pt) Write a **design** for the main function in the driver file, `driver.cpp`.
- (3 pts) Write a **design** for the `create_books()`, `populate_one_book()`, **and** `delete_books()` **as well as the functions needed to satisfy the above bulleted output** functions in the implementation file, `book.cpp`

Here's some documentation that will help you get going with File I/O:

- C++ Basic file I/O: http://www.learncpp.com/cpp-tutorial/186-basic-file-io/
- ifstream: http://www.cplusplus.com/reference/fstream/ifstream/
- ofstream: http://www.cplusplus.com/reference/fstream/ofstream/
- fstream: http://www.cplusplus.com/reference/fstream/fstream/

(6 pts) Part 3: Implementation

(5 pts) Now, implement the `driver.cpp`, `book.cpp`, and `book.h` files. (1 pt) Create a `Makefile` to manage the compilation of all these files. You can adapt the Makefile that was posted on the Calendar page in Canvas.

(2 pts) Extra Credit: Sorting

In addition to the two options listed above (searching by book title and books with more than 300 pages), implement a third option to **sort the books by year of release** (from the oldest to the most recent), and write the result to the output file.

Hint: you may use bubble sort algorithm: https://www.geeksforgeeks.org/bubble-sort/

Show your completed work and answers to the TAs for credit. You will not get points if you do not get checked off!

Submit your work to TEACH for our records **(Note: you will not get points if you don't get checked off with a TA!!!)**

1. Create a **zip file** that contains all files you've created in this lab:

```
zip lab5.zip book.h book.cpp driver.cpp Makefile
```

2. Transfer the zip file from the ENGR server to your local laptop.
3. Go to TEACH.
4. In the menu on the right side, go to **Class Tools → Submit Assignment**.
5. Select **CS162 Lab5** from the list of assignments and click "**SUBMIT NOW**"
6. Select your files and click the Submit button.