# Assignment 2: A Library Catalog

## Summary

This program manages the catalog of an old-fashioned library, i.e., one that actually has books. A catalog entry for a book consists of a unique catalog number, the author's last name, the author's first name, the name of the book, the genre, and whether it is available for checkout. The program is not very useful or robust because that would be too hard for a second project, but it will give you practice in many different aspects of computing. The program will open two input files specified on the command line, one containing a database of library holdings (books), and a second containing a list of requested books. The list of requested books is something like a batch processing system for the library. Imagine that at the end of every day, all requests made that day, which were recorded in a file, are then processed. This is what that file represents.

The program will produce three reports, one for each of the computational tasks described below, and these will be saved in three different output files whose names are also specified on the command line. The order of the file names on the command line is as follows:

1. library holdings input file
2. book request input file
3. author-sort output file
4. genre-sort output file
5. checkout output file

## Input

There are two input files. If an input file is missing, the program should output an error message and quit. You can assume the input files will have the formats described below; there is no need to error check the format.

### Library Holdings File

The ***library holdings file*** contains a list of all of the library's books in a specific format. Each line represents a single book, and will have six tokens (fields). Tokens are separated by tab characters in each line. Tokens can have non-tab whitespace characters in them. Let <T> denote a tab character. Then a line has the form

*catalog number*<T>*author last name*<T>*author first name*<T>*book title*<T>*genre*<T>*availability*

A sample input file might look like the following, in which tabs are also denoted by <T>:

```
C0594<T>Woolf<T>Virginia<T>A Room of One's Own<T>essay<T>available
C7495<T>Gibson<T>William<T>Neuromencer<T>science fiction<T>available
C1934<T>Heidegger<T>Martin<T>Being and Time<T>philosophy<T>unavailable
C8394<T>Kundera<T>Milan<T>The Unbearable Lightness of Being<T>novel<T>available
```

There will be no more than 10000 entries in the library holdings file.

### Book Request File

The ***book request file*** contains a list of catalog numbers, one per line. These are the books being requested by various library users, and the program should check if they are available. A sample input file might look like the following:

```
C0594
C7495
C1934
C2237
```

There will be no more than 500 entries in the book request file.

## Computational Tasks

There are three different tasks that the program must perform on the input files.

**author-sort.** List all of the books in the library in ascending alphabetical order of author last name. If two authors have the same last name, they should be listed in ascending order by first name.

**genre-sort.** List all the books in the library in ascending alphabetical order of genre. Within a genre, books should be ordered by author last name in ascending alphabetical order.

**checkout.** Given a list of requested catalog numbers, check if those books are available for checkout, and if so, for each one, indicate availability.

## User Interface

There is no user interface to this program. The arguments are given on the program command line, and they are the names of files that the program uses for its input.

## Output

The program will produce three output files with the names specified on the command line. If those files exist already they will be overwritten by the program.

### Author-Sort Output File

The output file for the author-sort task will list the library holdings sorted by author last name, in ascending alphabetical order. The format is the same as that of the "library holdings input file". If two authors have the same last name, the books should be listed in ascending order by first name within that set of last names.

### Genre-Sort Output File

The output file for the genre-sort task will list the library holdings sorted by genre, in ascending alphabetical order. The format is the same as that of the "library holdings input file". Within a genre, books should be ordered by author last name in ascending alphabetical order.

### Checkout Output File

The output file for the checkout task will list the catalog numbers appearing in the "book request input file" followed by the corresponding title and availability. The books should be listed in the same order in which they appear in the input file. If a requested catalog number does not appear in the library holdings, the program should list the catalog number followed by the word unavailable.

A sample output file for computational task 3 might be:

```
C0594 A Room of One's Own available
C7495 Neuromencer available
C1934 Being and Time unavailable
C2237 unavailable
```

## Programming Rules

Your program must conform to the programming rules described in the Programming Rules document on the course website. Remember in particular that:

- You must write your program yourself. I will on occasion ask you to explain the code to me, so you need to make sure that you understand the code that you write. If you want to use features of C++ that we did not cover in class, you need to understand them thoroughly.
- You must document your program (preamble, function pre- and post-conditions, comments throughout the code, appropriately choosing variable and function names).
- All input and output must be performed by the `main` function only, i.e. no other functions should perform any I/O operations. The functions that perform the computational tasks must pass all data through their parameter lists.
- Late programs lose 20% for each 24 hours they are late.

## Grading

The program will be graded based on the following rubric.

- Does the program compile (on a cslab machine): 15%
- Correctness and implementation of the book class: 10%
- Correctness and implementation of the sort and search functions: 15%
- Correctness and implementation of I/O, `main()`, and any other features in the program: 35%
- Correctness of multiple file implementation and modularity: 10%
- Documentation: 10%
- Style and proper naming: 5%

## Submitting the Assignment

This assignment is due by the end of the day (i.e. 11:59PM, EST) on November 5, 2012. You need to submit multiple files for this assignment, so they must be placed into a directory and zipped up as the instructions below describe. The name of your directory must be ***username*`_hwk2.`** where ***username*** is to be replaced by your username on our system. Do not name it anything else. If it is named anything else, you lose 3% of the program's value. Before you submit the assignment, make sure that it compiles and runs correctly on one of the cslab machines. Do not enhance your program beyond this specification. Do not make it do anything except what is written above.

You are to create the above named directory and place all of your source code files into it. Do not place any executable files or object files into this directory. You will lose 1% for each file that does not belong there. With all files in your directory, run the command

```
zip -r username_hwk2.zip ./username_hwk2
```

This will compress all of your files into the file named `username_hwk2.zip`. Put this zip file into the directory

```
/data/biocs/b/student.accounts/cs135_sw/cs135projects/project2
```

Give it permission 600 so that only you have access to it. To do this, `cd` to the above directory and run the command

```
chmod 600 username_hwk2.zip
```

If you put a file there and then decide to change it before the deadline, just replace it by the new version. Once the deadline has passed, you cannot do this. I will grade whatever version is there at the end of the day on the due date.