Programming Assignment #3 - Commands, Sorted Array, Binary Search, Separate C file(40 pts)

This assignment is similar to #2 except

* You must sort the Book Array. Print it before sorting, after sorting, and after all of the customers have been processed.
* You must use a **binary search** to find a book id.
* Instead of an customer file, we will have a **command** file.
* You will be provided with a driver program (cs1713p3Driver.c) (see below)
* Your code must be created in a separate C file (p3*abc123*.c). I provided a file named p3abc123.c. Rename it using your abc123 ID. (see below)
* There is a new include file, cs1713p3.h
* The output is different due to the driver printing the commands as they are encountered. Please see the sample output.

Input:

Book same as Programming Assignment #2 although there is different data.

Command This is different from the previous assignment. The file contains text in the form of commands.

CUSTOMER BEGIN szBookId szCustomerId szCheckedOutDt dLateFeePerDay dMaxLateFee szTitle

specifies the beginning of customer request and includes all the identification information from program 2.

CUSTOMER ADDRESS szStreetAddress,szCity,szStateCd,szZipCd

specifies the address for a customer (separated by commas)

CUSTOMER TRANS cTransType szBookId szTransDt

specifies a single book transaction. Steps:

* Print the Transaction Type, Book Id, and Transaction Date
* Lookup the book ID using a binary search. If not found, print a warning (but do not terminate your program) and return.
* If the transaction date is invalid, show a message stating "invalid date", but do not terminate. Use the **validateDate** function.
* If the transaction type is C (meaning checkout):
  + If the book is already checked out, show a message stating "already checked out", but do not terminate.
  + Otherwise, check out the book to this customer, setting the book's customer ID. The book's checked out date needs to be set to the transaction's date.
* If the transaction type is R (meaning return):
  + Use dateDiff to subtract the book's szCheckOutDt from the transaction's szTransDt
  + If the difference is more than 14:
    - Determine the number of days late by subtracting 14 from that difference.
    - Compute the late fee by multiplying the number of days late by the book's dLateFeePerDay.
    - If that late fee is greater than the book's dMaxLateFee, change it to dMaxLateFee.
    - Print the late fee.
    - Add the computed late fee to the customer's dFeeBalance
  + Set the book's customer ID to "NONE".
  + Set the book's check out date to "0000-00-00".

CUSTOMER COMPLETE

specifies the completion of a customer. Print the total fees for this customer.

BOOK CHANGE szBookId dLateFeePerDay dMaxLateFee

change the Late Fee Per Day and Max Late Fee to the specified values

BOOK SHOW szBookId

requests a display of a particular book. Show all of its information.

**Larry provided:**

**cs1713p3\_h.txt** - include file for program #3; please rename it to cs1713p3.h

**cs1713p3Driver\_c.txt** - driver C file which will invoke some of your functions; please rename to cs1713p3Driver.c

**p3abc123\_c.txt** - skeletal code which you must modify; please rename to p3abc123.c using your abc123 ID.

**p3Command.txt** - command input file

**p3Book.txt** - book data input file

**Driver program** (green highlighting means you have done most of this previously, yellow highlighting means new functionality):

You will be provided with a driver program, cs1713p3Driver.c which

1. invokes the driver's processCommandSwitches
2. invokes your getBooks (in p3abc123.c) to read the original book information.
3. invokes printBooks (in p3abc123.c) to print the original book information
4. invokes your sortBooks (in p3abc123.c) to sort the original book information
5. invokes printBooks to print the sorted book information
6. invokes a driver-provided processCommands which
   * reads input lines from the command file until eof:
     + prints the input line
     + determines command and subcommand
     + invokes either
       - your processCustomerCommand to process a CUSTOMER subcommand (you should be able to reuse some of your program 2 code, although you must change it) Note that your sscanf will use **pzRemainingInput** instead of **szInputBuffer**.
       - your processBookCommand to process a BOOK subcommand. Note that your sscanf will use **pzRemainingInput.**
7. invokes printBooks to print the resulting book information

**Your p3*abc123*.c code:**

* You have been furnished with skeletal code p3abc123.c (rename it from p3abc123\_c.txt). Your code will be in the file **p3*abc123***.c. You must **not** place it in the cs1713p3Driver.c file.
* It does the following includes:

#include <stdio.h>

#include <string.h>

#include "cs1713p3.h"

* It must **not** include cs1713p3Driver.c within your **p3*abc123***.c file. Look at the notes below on **compiling**. The "link" step is where the functions you call in the driver and the functions the driver calls in your code get resolved.
* You must create/complete the following routines (see the include file):
  + getBooks- reads and returns the books and returns a count of books (most of this is in program #2)
  + sortBooks - sorts the book information using a bubble sort (see your course notes)
  + searchBooks - find a book ID in the sorted book array using a binary search algorithm. You will use this in your routines.
  + processCustomerCommand - processes the various CUSTOMER commands (most of this is in program #2; however, use pszRemainingInput instead of szInputBuffer). Skeletal code is provided.
  + processBookCommand - processes the various BOOK commands

Please review the cs1713p3.h include file.

Compiling:

* (Before doing these steps, **make a copy of your code** called p3Saved.c in case you incorrectly type them and wipe out your .c file.)
* Compile the driver using

gcc -g -c -o cs1713p3Driver.o cs1713p3Driver.c

(you can do this instead: gcc -g -c cs1713p3Driver.c

causing it to automatically create the cs1713p3Driver.o)

* Compile your code using

gcc -g -c -o p3abc123.o p3abc123.c

(you can do this instead: gcc -g -c p3abc123.c

causing it to automatically create the p3abc123.o)

* Link them together using:

gcc -g -o p3 cs1713p3Driver.o p3abc123.o

* You only have to compile code that changes. Since the driver shouldn't change, you only have to compile it once.

Executing the p3 executable:

./p3 -c p3Command.txt -b p3Book.txt

Turn in:

Your include file (if it changed)

Your C code

Your output based on the data provided.

Sample Output:

Initial Books

Book Id Title Customer Ck Out Dt Late Fee Max Late

JOYPGM001 The Joys of Programming NONE 0000-00-00 0.25 50.00

JOYPGM002 The Joys of Programming 333333 2016-01-05 0.25 50.00

JOYPGM003 The Joys of Programming NONE 0000-00-00 0.25 50.00

JAVADD001 Java Isn't an Addiction 777777 2016-02-01 0.30 60.00

PYTHON001 Learn Python Without Getting Bit 111111 2016-01-02 0.30 60.00

PYTHON002 Learn Python Without Getting Bit NONE 0000-00-00 0.30 60.00

TECHDR001 My Technical Dream Job NONE 0000-00-00 0.25 50.00

JOYPGM004 The Joys of Programming NONE 0000-00-00 0.25 50.00

LINUXX004 Learning Linux 333333 2016-01-05 0.30 60.00

COBOLL001 How your Grandpa Coded in COBOL NONE 0000-00-00 0.10 10.00

EXCELL001 Excel at Excell 444444 2015-09-01 0.80 65.00

PERLLL001 Is PERL the Jewel of Programming? NONE 0000-00-00 0.10 10.00

SQLDBB001 Making Your DB Queries SQueeL 555555 2016-02-16 0.30 60.00

PRANKS001 Software Pranks NONE 0000-00-00 0.90 60.00

ARTINT001 A.I. Practical Algorithms 333333 2015-11-15 0.30 55.00

Sorted Books

Book Id Title Customer Ck Out Dt Late Fee Max Late

ARTINT001 A.I. Practical Algorithms 333333 2015-11-15 0.30 55.00

COBOLL001 How your Grandpa Coded in COBOL NONE 0000-00-00 0.10 10.00

EXCELL001 Excel at Excell 444444 2015-09-01 0.80 65.00

JAVADD001 Java Isn't an Addiction 777777 2016-02-01 0.30 60.00

JOYPGM001 The Joys of Programming NONE 0000-00-00 0.25 50.00

JOYPGM002 The Joys of Programming 333333 2016-01-05 0.25 50.00

JOYPGM003 The Joys of Programming NONE 0000-00-00 0.25 50.00

JOYPGM004 The Joys of Programming NONE 0000-00-00 0.25 50.00

LINUXX004 Learning Linux 333333 2016-01-05 0.30 60.00

PERLLL001 Is PERL the Jewel of Programming? NONE 0000-00-00 0.10 10.00

PRANKS001 Software Pranks NONE 0000-00-00 0.90 60.00

PYTHON001 Learn Python Without Getting Bit 111111 2016-01-02 0.30 60.00

PYTHON002 Learn Python Without Getting Bit NONE 0000-00-00 0.30 60.00

SQLDBB001 Making Your DB Queries SQueeL 555555 2016-02-16 0.30 60.00

TECHDR001 My Technical Dream Job NONE 0000-00-00 0.25 50.00

CUSTOMER BEGIN 111111 0.75 petem@xyz.net Pete Moss

CUSTOMER ADDRESS 123 Boggy Lane,New Orleans,LA,70112

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Library Receipt \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

111111 petem@xyz.net Pete Moss (previously owed 0.75)

123 Boggy Lane

New Orleans, LA 70112

Trans Book Date

CUSTOMER TRANS C JOYPGM001 2016-01-25

C JOYPGM001 2016-01-25

CUSTOMER TRANS C TECHDR001 2016-01-25

C TECHDR001 2016-01-25

CUSTOMER TRANS R PYTHON001 2016-01-25

R PYTHON001 2016-01-25 Late Fee = 2.70

CUSTOMER COMPLETE

Total Fees = 3.45

CUSTOMER BEGIN 222222 0.00 pcorn@abc.net Pop Corn

CUSTOMER ADDRESS 456 Kernel,San Antonio,TX,78210

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* Library Receipt \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

222222 pcorn@abc.net Pop Corn (previously owed 0.00)

456 Kernel

San Antonio, TX 78210

Trans Book Date

CUSTOMER TRANS C TECHDR001 2016-01-25

C TECHDR001 2016-01-25 \*\*\* Already checked out

CUSTOMER TRANS C TECHDR002 2016-01-25

C TECHDR002 2016-01-25 \*\*\* Book Not Found

CUSTOMER TRANS C JOYPGM004 2016-01-25

C JOYPGM004 2016-01-25

CUSTOMER COMPLETE

Total Fees = 0.00