CS3743 Program #1 Intro to Hash Files (20 points)

This is the first part of a two part programming assignment. In this part, we are reading and writing to a hashed file. In the next part, we will be implementing chaining to handle synonyms.

I have provided a driver program, include file, and input file. The driver is provided to reduce your effort on this programming assignment.

cs3743p1Driver.c - driver program which invokes your functions. It also provides a hash function.

cs3743p1.h - include file which contains constants, HashHeader and Book typedefs, and function prototypes.

p1Input.txt - stream input file used by the driver to specify what needs to be invoked

Functions you must code:

int **hashCreate**(char szFileNm[], HashHeader \*pHashHeader)

This function creates a hash file containing only the HashHeader record.

* If the file already exists, return RC\_FILE\_EXISTS
* Create the binary file by opening it.
* Set pHashHeader->iHighOverflowRBN to the value of pHashHeader->iNumPrimary
* Write the HashHeader record to the file at RBN 0.
* fclose the file.

FILE \***hashOpen**(char szFileNm[], HashHeader \*pHashHeader)

This function opens an existing hash file which must contain a HashHeader record and returns a file pointer to it.

* Use fopen to open the file. If it doesn't exist, return NULL.
* Use fread to read the HashHeader record and return it through the parameter. If the read fails, return NULL.
* Return the opened FILE pointer as the function value when successful.

int **readRec**(FILE \*pFile, int iRBN, void \*pRecord, int iRecSize)

This function reads a record of the specified size at the specified RBN in the specified file.

* Determine the RBA based on iRBN and iRecSize.
* Use fseek to position the file in that location.
* Use fread to read that record and return it through pRecord.
* If the location is not found, return RC\_LOC\_NOT\_FOUND. Otherwise, return RC\_OK.
* Note: if the location is found, that does NOT imply that a book was written to that location. Why?

int **writeRec**(FILE \*pFile, int iRBN, void \*pRecord, int iRecSize)

This function writes a record of the specified size at the specified RBN in the specified file.

* Determine the RBA based on iRBN and iRecSize.
* Use fseek to position the file in that location.
* Use fwrite to write that record to the file.
* If the fwrite fails, return RC\_LOC\_NOT\_WRITTEN. Otherwise, return RC\_OK.

int **bookInsert**(FILE \*pFile, HashHeader \*pHashHeader, Book \*pBook)

This function inserts a book into the specified file.

* Determine the RBN using the driver's hash function.
* Use readRec to read the record at that RBN.
* If that location doesn't exist or the record at that location has a szBookId[0] == '\0':
  + Sets its iNextChain to 0.
  + Write this new book record at that location using writeRec.
* If that record exists and that book's szBookId matches, return RC\_REC\_EXISTS. (Do not update it.)
* Otherwise, return RC\_SYNONYM.

int **bookRead**(FILE \*pFile, HashHeader \*pHashHeader, Book \*pBook)

This function reads the specified book by its szBookId.

* Determine the RBN using the driver's hash function.
* Use readRec to read the record at that RBN.
* If the book at that location matches the specified szBookId, return the book via pBook and return RC\_OK.
* Otherwise, return RC\_REC\_NOT\_FOUND

Note:

1. Your code must be written based on my programming standards and placed in cs3743p1.c
2. Do not modify either cs3743p1.h or cs3743p1Driver.c.
3. Turn in your cs3743p1.c and p1out.txt (output) using a zip file named (*LastFirst*.zip) via BlackBoard.

Sample Output (partial):

\* CS3743 p1Input.txt

\* Nuke the Hash file if it exists

>> NUKE BOOK book.dat

\* Opening of a non-existent file should cause an error

>> OPEN BOOK book.data

\*\*\*\* ERROR: File does not exist or has invalid format: 'book.data'

\* Create the hash file

>> CREATE BOOK book.dat 19

Record size is 84

>> DUMP BOOK book.dat

Primary=19, HighOverflow=19, RecSize=84

\* Creating it again should cause an existence error

>> CREATE BOOK book.dat 17

Record size is 84

\*\*\*\* ERROR: file already exists

\* Open it

>> OPEN BOOK book.dat

\* Insert some books

>> INSERT BOOK JOYPGM001,The Joys of Programming,TECH,PGMING,100

Hash RBN is 8

>> DUMP BOOK book.dat

Primary=19, HighOverflow=19, RecSize=84

8 Next=0

JOYPGM001 TECH PGMING 100 The Joys of Programming

>> INSERT BOOK PYTHON001,Learn Python Without Getting Bit,S PRESS,PGMING,200

Hash RBN is 1

>> DUMP BOOK book.dat

Primary=19, HighOverflow=19, RecSize=84

1 Next=0

PYTHON001 S PRESS PGMING 200 Learn Python Without Getting Bit

8 Next=0

JOYPGM001 TECH PGMING 100 The Joys of Programming

>> INSERT BOOK SQLDBB001,Making Your DB Queries SQueeL,XYZ,DB,300

Hash RBN is 16

>> INSERT BOOK TECHDR001,My Technical Dream Job,TECH,PGMING,400

Hash RBN is 18

>> DUMP BOOK book.dat

Primary=19, HighOverflow=19, RecSize=84

1 Next=0

PYTHON001 S PRESS PGMING 200 Learn Python Without Getting Bit

8 Next=0

JOYPGM001 TECH PGMING 100 The Joys of Programming

16 Next=0

SQLDBB001 XYZ DB 300 Making Your DB Queries SQueeL

18 Next=0

TECHDR001 TECH PGMING 400 My Technical Dream Job

\* Read an existing book

>> READ BOOK TECHDR001

.. Next=0

TECHDR001 TECH PGMING 400 My Technical Dream Job

\* Read should not find this one

>> READ BOOK JAVADD001

\*\*\*\* ERROR: record not found

\* Insert an existing book - should cause an error

>> INSERT BOOK TECHDR001,My Technical Dream Job Again,TECH,PGMING,444

Hash RBN is 18

\*\*\*\* ERROR: record already exists

>> READ BOOK TECHDR001

.. Next=0

TECHDR001 TECH PGMING 400 My Technical Dream Job