CS3743 Program #2 Hash File Using Probing for Synonyms (40 points) Spring 2020 (updated 2020-02-13)

**© Copyright 2020 Larry Clark, this document must not be copied to any other website**

This is the **second part** of a two-part programming assignment. In this part, we are implementing **probing 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.

Files I provided (can be copied from /usr/local/courses/clark/cs3743/2020Sp/Pgm2):

**cs3743p2Driver.c** - driver program which invokes your functions. It also provides a **hash** function.

**cs3743p2.h** - include file which contains constants, HashHeader, HashFile, and Book typedefs, and function prototypes.

**p2Input.txt** - stream input file used by the driver to specify what needs to be invoked

Your hash file will have a header record at Record Block Number (RBN) 0. All other records will either be empty or contain books.

This program will use your code from the first programming assignment, making changes to some of the functions, and adding some new ones.

Functions you must code:

int **hashCreate**(char szFileNm[], HashHeader \*pHashHeader) – reused from program #1

int **hashOpen**(char szFileNm[], HashFile \*pHashFile) – reused from program #1

int **readRec**(HashFile \*pHashFile, int iRBN, void \*pRecord) – reused from program #1

int **writeRec**(HashFile \*pHashFile, int iRBN, void \*pRecord) – reused from program #1

int **insertBook**(HashFile \*pHashFile, Book \*pBook) – you must modify this

This function inserts a book into the specified file.

* Determine the RBN using the driver's hash function.
* Use readRec to read a record at that RBN.
* If that location doesn't exist or the record at that location has a szBookId[0] == '\0':
  + Write the new book record (using pBook) at that location using writeRec. (Note this is different if doing the extra credit.)
* If that record exists and that book's szBookId matches pBook->szBookId, return RC\_REC\_EXISTS. (Do not update it.)
* Otherwise, it is a synonym to the book in the hashed location:
  + Determine if it exists by probing. We use a probing K value of 1. If it does already exist, return RC\_REC\_EXISTS. (Do not update it.)
  + Limit the probing to pHashFile->hashHeader.iMaxProbes. For example, **if iMaxProbes is 3**, you can look at the original hash location and at most two additional records. We are only looking at adjacent records below it.
  + if there isn't an empty slot and we have probed a total of iMaxProbes times (including looking at the hashed location), return RC\_TOO\_MANY\_COLLISIONS. If it doesn't exist and there is an empty slot (maybe because we haven't yet written to that slot), write it to that empty slot.
  + Also, we will not probe past the header record’s iMaxOvRBN. This would also cause you to return RC\_TOO\_MANY\_COLLISIONS.

int **readBook**(HashFile \*pHashFile, Book \*pBook, int \*piRBN) – you must modify this

This function reads the specified book by its szBookId.

* We are returning the RBN for where it was written and also handling synonyms.
* Determine the primary 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, set \*piRBN, and return RC\_OK.
* Otherwise, it is a synonym to the book in the hashed location:
  + Determine if it exists as a synonym using probing with a K value of 1. If it does exist, return the book via pBook, set \*piRBN to its actual RBN (where it was found), and return RC\_OK.
  + If you read past the end of the file, return RC\_REC\_NOT\_FOUND.
  + If you have read for the maximum probes and it wasn’t found, return RC\_REC\_NOT\_FOUND.

int **updateBook**(HashFile \*pHashFile, Book \*pBook) – this is new

* This function reads the specified book by its szBookId. If found, it updates the contents of the book in the hash file. If not found, it returns RC\_REC\_NOT\_FOUND. Note that this function must understand probing.

int **deleteBook**(HashFile \*pHashFile, Book \*pBook) – this is new

* If you did not do the extra credit, create a simple function that just returns RC\_NOT\_IMPLEMENTED.
* This function finds the specified book and deletes it by simply setting all bytes in that record to '\0'. Once deleted, this may impact your readBook, insertBook, and updateBook since there can now be empty records along a synonym list even though the needed book **could be after it**.
* Memset to whole record to ‘\0’

Notes:

1. When looking at your output, your display tool needs to use a fixed font; otherwise with a proportional font, the output will not be aligned properly.
2. The data in your book.dat file will not be easily visible since it contains some binary (e.g., integer) data.
3. Your code must be written based on my **programming standards** and placed in **cs3743p2.c**
4. Do **not** modify either cs3743p2.h or cs3743p2Driver.c.
5. You must run your code on a fox server.
6. Turn in your cs3743p2.c and p2out.txt (output) using a zip file named (*abc123.*zip) via BlackBoard. The zip file must not contain any directories. **In the notes in BlackBoard, also specify whether you did the extra credit program.**
7. Copying this assignment to another website will be considered scholastic dishonesty.

**Extra Credit** (5 pts + 200/ N)

* To be **eligible** for extra credit, your code **must meet** **all** the requirements including the programming standards and your solution **must not be late**.
* N is the number of students (in the combination from both sections) who met all the requirements.
* Implement the ability to delete books from the hash file. When a book is deleted, we simply zero out all of its information.

int **deleteBook**(HashFile \*pHashFile, Book \*pBook)

* A deleted book will **not be replaced** by a subsequent synonym of the list already in the file.
* This will impact your code for **readBook**:
  + You may encounter empty szBookId values along a synonym list (including in the hashed location). This doesn't mean that the book wasn't found. Instead, it might be later in the list.
* This will impact your code for **insertBook**:
  + Similar to readBook, you must look throughout a synonym list regardless of whether an empty szBookId is encountered.
  + If the book doesn't already exist in the synonym list, place it in the **first empty slot** along the list if there is one. (This might be in the hashed location or in subsequent probed location.) If there isn't an empty slot and we have probed a total of iMaxProbes times (including looking at the hashed location), return RC\_TOO\_MANY\_COLLISIONS.
* This will impact your code for **updateBook**:
  + Similar to readBook, you must look throughout a synonym list regardless of whether an empty szBookId is encountered.
* Run your code using **p2Input.txt**. It must work for all cases.

Sample Output (partial):

\*

\* 8. Insert more books

\*

>> INSERT BOOK PYTADW001,Programming in Python,ADWES,PGMING,500

Hash RBN is 11, id='PYTADW001'

>> INSERT BOOK DSTRUC001,Data Structures in C,S PRESS,PGMING,240

Hash RBN is 7, id='DSTRUC001'

\*

\* 9. Insert a book that is a synonym

\*

>> INSERT BOOK EXCELL001,Excel at Excell,MS PRESS,ACCOUNT,200

Hash RBN is 2, id='EXCELL001'

>> READ BOOK EXCELL001

Hash RBN is 2, id='EXCELL001'

3 EXCELL001 MS PRESS ACCOUNT 200 Excel at Excell Hash=2

>> READ BOOK JAVADD001

Hash RBN is 2, id='JAVADD001'

2 JAVADD001 S PRESS PGMING 600 Java Isn't an Addiction Hash=2

>> PRINTALL BOOK book.dat

MaxPrim=19, RecSize=80, MaxOv=25, MaxProbes=5

1 PYTHON001 S PRESS PGMING 200 Learn Python Without Getting Bit Hash=1

2 JAVADD001 S PRESS PGMING 600 Java Isn't an Addiction Hash=2

3 EXCELL001 MS PRESS ACCOUNT 200 Excel at Excell Hash=2

4 ARTINT001 S PRESS PGMING 400 A.I. Practical Algorithms Hash=4

7 DSTRUC001 S PRESS PGMING 240 Data Structures in C Hash=7

8 JOYPGM001 TECH PGMING 100 The Joys of Programming Hash=8

10 LINUXX004 XYZ OS 700 Learning Linux Hash=10

11 PYTADW001 ADWES PGMING 500 Programming in Python Hash=11

16 SQLDBB001 XYZ DB 300 Making Your DB Queries SQueeL Hash=16

\*

\* 10. Insert more books; however, some will have synonyms.

\*

>> INSERT BOOK PERLLL001,Is PERL the Jewel of Programming,S PRESS,PGMING,500

Hash RBN is 16, id='PERLLL001'

>> INSERT BOOK FUNDBS001,Fundamentals of Database Systems,PEARSON,DB,900

Hash RBN is 7, id='FUNDBS001'

>> INSERT BOOK PGMLAN001,Progamming Language Concepts,X PRESS,PGMING,700

Hash RBN is 4, id='PGMLAN001'

>> INSERT BOOK COBOLL001,How Your Grandpa Coded in COBOL,S PRESS,PGMING,800

Hash RBN is 19, id='COBOLL001'

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

Hash RBN is 18, id='TECHDR001'

>> PRINTALL BOOK book.dat

MaxPrim=19, RecSize=80, MaxOv=25, MaxProbes=5

1 PYTHON001 S PRESS PGMING 200 Learn Python Without Getting Bit Hash=1

2 JAVADD001 S PRESS PGMING 600 Java Isn't an Addiction Hash=2

3 EXCELL001 MS PRESS ACCOUNT 200 Excel at Excell Hash=2

4 ARTINT001 S PRESS PGMING 400 A.I. Practical Algorithms Hash=4

5 PGMLAN001 X PRESS PGMING 700 Progamming Language Concepts Hash=4

7 DSTRUC001 S PRESS PGMING 240 Data Structures in C Hash=7

8 JOYPGM001 TECH PGMING 100 The Joys of Programming Hash=8

9 FUNDBS001 PEARSON DB 900 Fundamentals of Database Systems Hash=7

10 LINUXX004 XYZ OS 700 Learning Linux Hash=10

11 PYTADW001 ADWES PGMING 500 Programming in Python Hash=11

16 SQLDBB001 XYZ DB 300 Making Your DB Queries SQueeL Hash=16

17 PERLLL001 S PRESS PGMING 500 Is PERL the Jewel of Programming Hash=16

18 TECHDR001 TECH PGMING 400 My Technical Dream Job Hash=18

19 COBOLL001 S PRESS PGMING 800 How Your Grandpa Coded in COBOL Hash=19