University of Toronto

CSCB28F—File Structures, Fall 1998

Assignment 3: Searchable Library Catalog

Due:	*Part A: Wednesday November 25 by 12:00 p.m.(electronic submission only),*
	Part B: Wednesday December 9 10:00 am.
Partners:	*groups up to 2 people.*
Weight:	*15%.*

Objectives and the catalog features

The purpose of this assignment is to give you experience with multi-index files and B+ trees. To learn about B+ trees, read Chapter 9 (also Chapter 8) in the textbook by Folk and Zoellick.
You are also going to gain experience on working in groups.
You are to write a program that maintains information on books in a set of libraries, similarly to the UTCat catalogue for UofT (but a much simpler version). Users can search for book information by author or title. The catalog should be an on-line interactive package. The user is prompted for the following options for searching through the catalog or for doing other operations:
1) Search by author and title:
The author should be searched using last name first, and for simplicity search only for first-author names. For author and title searches, you should return a list of all the keys that have its first characters matching the given search key. For instance, if you search for "Knuth", your program should show a list of first-author's names that start with the characters in "Knuth", displayed in alphabetical order; for instance UTCat would show something like:

```
1   [3] Knuth Alice
2  [36] Knuth Donald Ervin
3   [7] Knuth Reinhard
```

After that, the user can select an author name for seeing a list of books by the author, and then select a book for having its detailed information displayed. A search for "Knuth A" would only show the first item in the above list. A search for title "The art of" would show the titles starting with these characters, in alphabetical order, for instance:

```
1   [2] The art of Budism
2   [5] THE ART OF COMPUTER PROGRAMMING
3   [2] The art of the theatre
```

Note that UTCat does not work quite like that for title, but your program should. Again, selecting the appropriate item would lead to a list of books with that title that the user could choose from in order to access the complete book information.
2) Insert and delete books:
Insertions and deletions should be done only by authorized librarians. Before performing this operation, you should ask for a 4-character password and check against the system's password. For simplicity, let us have a constant password in your program, which is "9999". To delete a book the user should provide both the first-author name and the ISBN number; to insert a book the user should be prompted for all the fields.
3) Other options: end the program, print the whole catalog (author and title only) sorted by author, and print the whole catalog (author and title only) sorted by title.

Contest

Check the web page for information on a contest for outstanding assignments, its rules and prizes!!!

File Organization

Although your test file may be small, this application is designed for large files, and you should organize your files having this in mind.
A text master file (`catmaster.txt`) consists of the following fields (the number of characters per field is inside parenthesis): ISBN (10), first-author name (30), book title (60), other-author name (30), call number (30), subject (30), publication information (20). A sample file with a few records is available in your course web page; note that in the author fields, the last name comes first.
Your program should create a searchable catalog by organizing the information in the given master file into a catalog formed by two B+ trees (two files):

1. Main B+ tree: This B+ tree is organized using a key that is a combination of first-author name and ISBN. This should be used for operations involving first-author name (searching by author and printing catalog sorted by author). The full records are stored as leaves of this B+ tree.
2. Title B+ tree: This B+ tree is organized using a key that is a combination of book title and ISBN. This should be used for operations involving title (searching by title and printing catalog sorted by title). Besides additional fields required for the B+ tree implementation, the leaves of this B+ tree should contain only book title, ISBN and the page number for the complete record in the main B+ tree.

The two keys described above (as a combination of two fields) should be understood as follows. You only have to look at ISBN when the first field is the same, in order to break ties. Note that in both B+ trees the key to be stored is formed by two fields.
You should use a page size of 700 bytes. The pointers for other pages (integer number indicating the position of another page) should be 10 characters long. Important: deletions and insertions should be done so that both B+ trees are updated!

Your program

Part A: Submitting a file with 10 records (5%)
Search through UTCat for author name starting with the first two characters of your last name. Type the data obtained from any 10 records into a file according to the sample file format provided in the course web page. All submissions by students will be merged to form a big common sample file. **Submit this part by November 25!**
Part B: The catalog program (95%)
You must create two programs (the executable files should be called `createcat` and `searchcat`) and a `makefile`. You may also submit a `README` file that the tutor will read.
The program `createcat` should read the file `catmaster.txt` (in the same format as in part A) and create the file `catalog.txt` organized as described in the previous section. The program `searchcat` is the program for searching the catalog that will offer the features described in the first section (search, deletion, insertion, etc.).
It is important that your program is correct, that the user interface is simple to use, and that your program is modular and well documented. The user interface can be simple, providing a list of options the user can choose from.
We suggest you write your program in stages following the parts described below. Part B2 depends on B1. Once your part B1 is working, save it in a different directory. In case part B2 combined with B1 doesn't work, you should submit Part B1 only. Say in your report which parts you did submit. If both parts are working, a single program with all features should be submitted.

- **Part B1: Main B+ tree**. Implement the main B+ tree, and all the features described previously except the search by title and the printing of the catalog sorted by title.

- **Part B2: Title B+ tree**. If your Part B1 is working, save your source program in another directory. Modify your program from part B1 to include the title B+ tree, and to allow for search by title and printing of the catalog sorted by title. Notice that insertions/deletions on the Main B+ tree should produce changes in the title B+ tree whenever a record in the main B+ tree is moved from one page to another. In addition, insertions/deletions will also produce insertions/deletions in the title B+ tree.

Divide different aspects of the program into different files to be linked by your makefile. For instance, procedures for user interface and the ones for file structure operations should be in different files. You should discuss the assignment with your partner, design the main parts, and discuss how to divide up the work. You may adopt a "top-down" approach: after the main parts are decided upon, you decide together on all the procedures you need, and then you may code individual procedures separately. Also make sure you decide on style issues (such as the way you choose variable names, indentation, etc.), so that all the parts of the program have a similar look. Of course, it's up to you whether you code everything together.

Report, hardcopies and submission information

Your report should contain the following parts:

1. User reference guide (maximum 1 page): this should be a simple guide to the user (and also to the TA marking it) on how to use the catalog. It should explain your user interface and features available.
2. System information (up to 2 pages): this is to let your TA know some implementation decisions you made; explain in high level terms (the TA can refer to your program for details). You should include which features are working and if any feature is not working; also include "known bugs" which can be limitations of your implementation or other known errors you couldn't solve.
3. Attached hardcopies: all programs (*.c and *.h files), and a test file. Note that the input comes from standard input and the tutor can redirect the input to come from the test file in order to test your program. The test file should be complete in order to convince your tutor that all the features of your program work (add pencil comments about the purpose of each part of your test file as usual).

Submit electronically: `makefile`, all your programs (*.c and *.h), the test file and a `README` file, if any, as usual:

```
submit -N a3 cscb28f makefile README <file1> ... <fileN>
```

```
Marking guide for Assignment 3
==============================

Part A  (5%)                                    ___ /  5

Part B  (95%)

   User interface:                              ___ /  5

   Correctness and efficiency:
      Part B1                                   ___ / 40
      Part B2                                   ___ / 20

   Style                                        ___ / 10

   Report and quality of test file              ___ / 20

                                 Total          ___ /100
```