Data Structure
Programming Assignment #3

Due Date: Oct. 24th, 2010 11:55pm

Shumin Guo

October 12, 2010

1 Overview

For this assignment, you will extend the employee database that we constructed in Lab #1. While the actual database records will still be stored in a LinkedSortedList of Employee objects, there will now be an Employee ID index implemented as a binary search tree of pointers to Employee records. Employee IDs will be unique (no duplicates). For example, consider the following set of employees:

```
Name             ID
John Doe         523921
Jane Doe         239182
Michael Johnson  387721
Bill Anderson    768321
```

The database for this information would appear as follows:

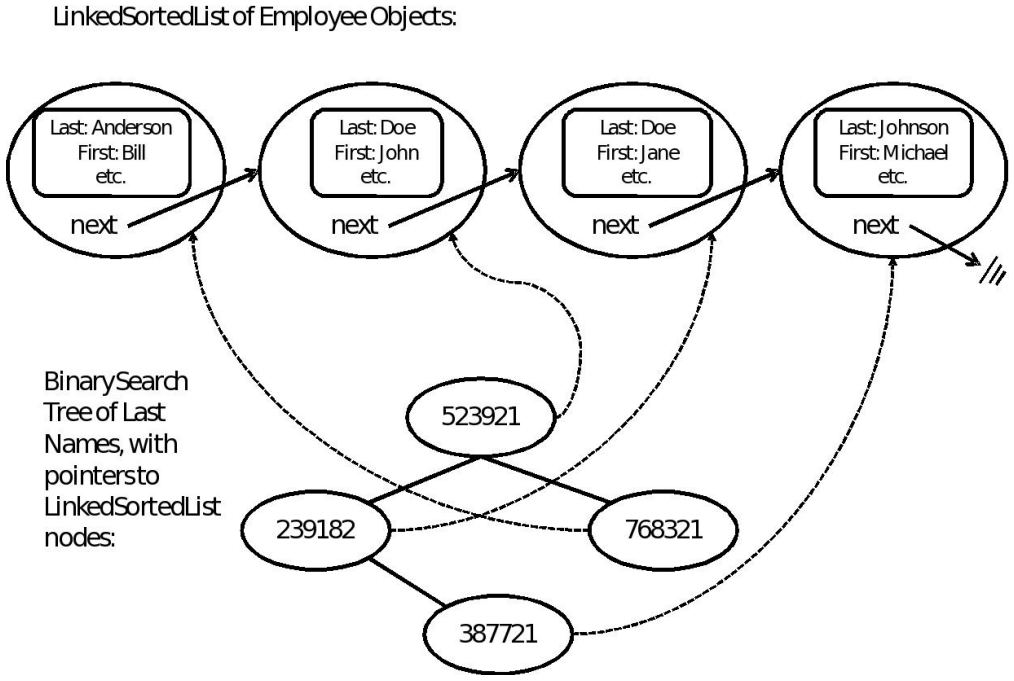

Figure 1: Structure of Linked Sorted List.

Your database should support the following operations:

- Insert new records (prompt user for all fields)
- Search on Last Name (print ALL matching records)
- Search on Employee ID (print matching record)
- Delete records (by employee ID only)
- Save database to disk (not including indexes)
- Load database from disk (rebuilding indexes)

Saving the database should store records in last name order. When loading a database from the disk, all current records should be deleted, and the database should be loaded from a file, and the indexes should be rebuilt. An example of a saved database file will be provided it will have the same format as the previous lab. Your program should be able to read this file and write files in this format.

The search operations should report the number of tree nodes searched, in addition to the results. For example, an ID query for 387721 would return results similar to the following:

```
MENU
(I)nsert new record
(D)elete record
(L)ast name search
(E)mployee ID search
(S)ave database to a file
(R)ead database from a file

 Enter choice: E

 Enter Employee ID:  387721
 Searching.

 3 index nodes searched.  Found 1 record:

 Last:  Johnson
 First: Michael
 EID:   387721
 Salary:$105,000
 Dept:  Legal
```

2 Requirements

1. Your code should follow the Code Standards handed out during the first day of class. Your code will be graded according to its correctness, efficiency, organization, and readability.
2. Your main program should be called lab3.cpp. You should implement your binary search tree in the files bst.h and bst.cpp.
3. Make sure that each .cpp file includes your name in the header comments.
4. Turn in all files needed to compile and execute your code (including any needed files from the previous lab) via webCT. If for some reason WebCT is unavailable, submit your source code by email tow lodarski.4 AT wright.edu. If you want, you can also cc to the instructo r Meilin Liu, whose email address is meilin.liu AT wright.edu.

5. If there are any special instructions that I need to know about, be sure to include a file named README.TXT in your lab2 directory detailing them.

6. The grader will test your programs under the schools UNIX environment, e.g., unixapps1.wright.edu. It is YOUR responsibility to make your programs workable and runnable by others under schools UNIX environment.

7. The programming assignment is individual. You must do the project by yourself. If you allow others to copy your programs or answers, you will get the same punishment as those who copy yours.