University of Toronto

csc228S— File Structures and Data Management, Winter 2002

Assignment 2 — On-line Dictionary

Deadline A :	*Beginning of your tutorial section, Friday February 15, 2002.*
Deadline B :	*Beginning of your tutorial section, Friday March 8, 2002.*
Total Weight:	*15% of your course grade.*
	Part A is worth a small fraction of this total weight.

Introduction

Your company is building an on-line dictionary, spell checker and ultimately a thesaurus tool, and you are asked to design and generate the master file which will hold the words and their definitions and related information. You will also be responsible to create the the functions that will interface with the dictionary files. Someone else will write a nice graphical front end which will use your functions to manage the actual data.

Your Job

As described above, there are two aspects to your job:

Database Design You need to design the file structure for a database in which each record contains at least:

- A dictionary word.
- The word's definition.
- A list of synonyms for that word.

Also since you will need to access this big database efficiently (i.e. do searches and insertions), so you are responsible to design and implement an index to the dictionary master file. The index should be in the form of a binary search tree[1] (BST). Each node in the BST stores a word and a pointer to the dictionary entry for that word. The index is kept in a file; you are to assume that it will not fit into memory. You may keep at most a small constant number of nodes in memory but notice that you are not required to implement paging of your BST.

Function Design and Implementation In order for this dictionary to be useful you must implement code that interfaces with it. Namely, you need to create functions that:

- Search for a word: search quickly for a string in the dictionary and determine if it is already there. If it is there you would like to return the word, its definition and a list of related words

[1] A B-tree, would be more suitable, but we're using a BST in order to make the assignment simpler.

(synonyms) along with their definitions. If it isn't there return something appropriate, maybe a suggestion of possible matches based on words that begin with similar letters. [2]

- Insert a word: adds a new word and its definition to the dictionary.
- Specify a synonym: specifies when two words are synonyms, this potentially modifies two records.

These functions don't use any prompts or command lines or printing; assume that the client code will provide the fancy interface to obtain information from the user and then simply call your functions providing all the information in the parameters.

A small example

One possible way an index and a dictionary are related is shown in Figure 1. Some comments about the diagram:

- The diagram shows the *logical* structure of the files. Of course their physical layout is linear. So for example, the nodes of the tree would be laid out one after the other in the file. (The order in which they occur in the file would depend on the order in which they were created.) It is the pointers from parent to child that make them into a tree.
- Notice that the synonyms are indicated by pointers to the dictionary record for the synonymous word. You are required to implement it this way.

Implementation Details

First, design file structure layouts for the dictionary file and index file. It may seem that everything has been decided, but the specifications above only describe the *logical* structure of the files. There are a number of design decisions left, including the following:

- Variable length records or fixed length records?
- For "pointers" into a file, will you use a byte offset or relative record number?
- Inside the dictionary file records will you store the word or just get it from the index record?

There are also a number of decisions left about the services your functions will offer. What will your function to specify a synonym do if one or both of the words are not already in the dictionary? Will it make both words register the other as a synonym? What will your search function return when the word is not found? Will you have a limited number of synonyms? If you do, then what will you do when the user asks for another and you've already reached the maximum. How will you design your file structure based on this decision?

[2] Just for the fun of it we will provide you with a C++ function called `soundex`, which transform a string into a 4 letter code. If two words sound the same they will be assigned the same code, this will allow you (in theory) to provide a list of words that are similar (in they way they are pronounced) to another. **You are free to add this feature, however there will be no extra marks for it, just simple fun and satisfaction**

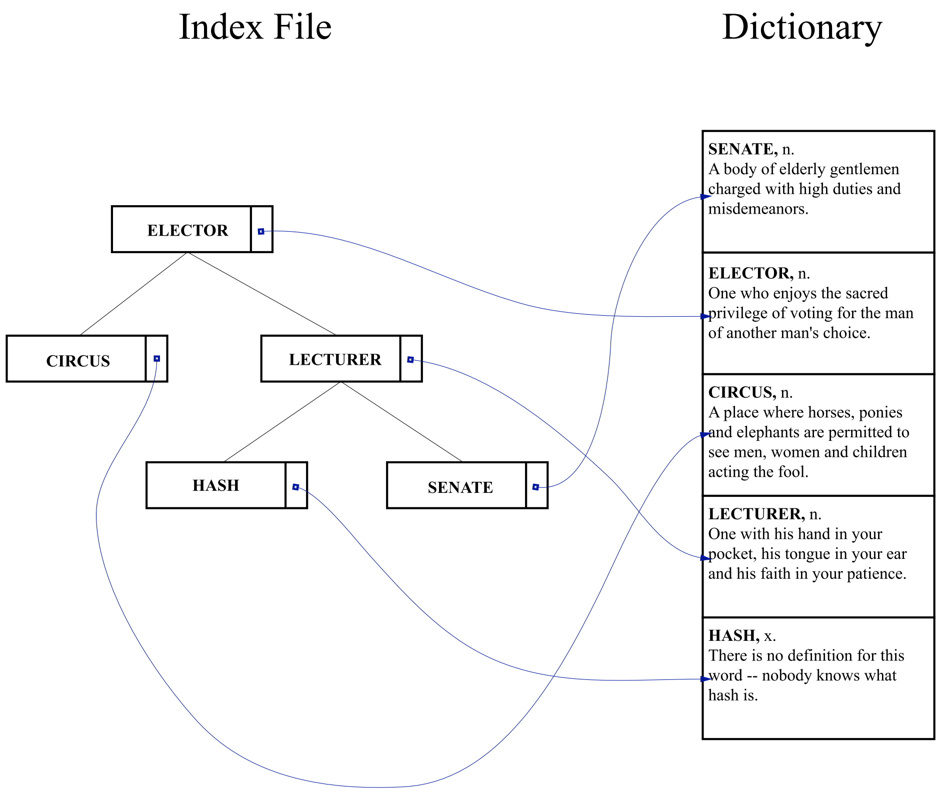

Figure 1: Logical structure of the Dictionary file and index file that might result from a number of calls to the add function.

All these decisions are left for you. Remember that you must explain and justify your decisions in your report.
Notice that because we never want to remove a word from the dictionary, there are no deletions!
You will need to write a small program to test your code. Do not bother building a fancy interface just keep it simple to demonstrate that your code provides the required functionality.

Some advice about the code development

Take time to design the code structure before you start writing. This means much more than a rough idea of what the classes are. For each class, you should design its interface precisely, giving careful specifications of what each public method does, including preconditions.

Also take time to plan the *process* of developing your code. In what order will you do it? What sort of testing can you do along the way? Plan to write and test the code incrementally, and to keep a record of such testing.

What to hand in for deadline A

For deadline A, you should have some code running. Exactly what is up to you, but it should be something non-trivial that is along the way to completing the assignment. Don't hand in a report with part A. Just hand a printout of your code (including any driver programs), and some test runs that demonstrate what works. Explain your test results with brief, hand-written comments on your printout. Try to make your submission as brief as possible.

What to hand in for deadline B

For deadline B, hand in your assignment in the form of a report, as described in the 228 Course Guide. Your report must include all of your code and all relevant input and output test files. Remember to justify the decisions you made in designing your program. This is a very important part of your report.

Your report should be thorough, well-written, concise, and honest. It should be no more than about five pages long, not including code and tests. Make sure that you run your report through a spelling checker.

You are also required to hand in your assignment electronically. Submit all your source files and include a makefile. You must ensure that your assignment will compile and run on the appropriate school computer facility for your campus.