Homework 3

Data Structures and Algorithms in C++
Shlomo Hershkop
Department of Computer Science
Columbia University
Summer 2006

Due Wednesday, July 26 11pm

Theory (50 points)

1. Show (Draw) the results of inserting 2, 1, 4, 5, 9, 13, 3, 17 into an initially empty AVL tree.

2. Draw an example of an AVL tree such that a single remove operation could require \Theta(log n) rotations from a leaf to the root in order to restore the height-balance property. (Use triangles to represent subtrees that are not affected by this operation.)

3. Draw the B-tree with M=5, L=7 resulting from inserting the following keys (in this order) into an initially empty tree: 4, 40, 23, 50, 11, 34, 62, 78, 66, 22, 90, 59, 25, 72, 64, 77, 10, 12.

4. Given input {3471, 3123, 1673, 1499, 3444, 6979, 8989} and a hash function h(x)=x(mod 10), show the resulting:
   - Separate chaining hash table.
   - Open addressing hash table using linear probing.
   - Open addressing hash table using quadratic probing.
   - Open addressing hash table with second hash function $h_2(x) = 7 - (x \bmod 7)$.

5. Show the result of rehashing the last two hash tables in the previous question.

6.
   - Show the result of inserting 23, 25, 14, 27, 19, 18, 21, 28, 24, 22, 20, 17, 24, 26, 15, one at a time, into an initially empty binary heap.
   - Show the result of using the linear-time algorithm to build a binary heap using the same input.
   - Show the result of performing four deleteMin operations in the heap of the previous heap.

7. Prove that for binary heaps, buildHeap does at most 2N-2 comparisons between elements

8. Consider the sequence of integers S = {8, 9, 7, 9, 3, 2, 3, 8, 4, 6}. For each of the following algorithms, draw a sequence of diagrams that traces the execution of the algorithm as it sorts the sequence: insertion sort, bubble sort, heapsort, mergesort.

9. A sorting algorithm is *stable* if duplicates retain their relative positions in the sorted sequence. For example, consider the following list of names:

```
George Bush
Al Gore
Hillary Clinton
Ronald Reagan
Rick Lazio
George W. Bush
Ralph Nader
```

   If this list is sorted by last name *only*, then a stable sort will produce this list:

```
George Bush
George W. Bush
Hillary Clinton
Al Gore
Rick Lazio
Ralph Nader
Ronald Reagan
```

   whereas an unstable sort might produce this list:

```
George W. Bush
George Bush
Hillary Clinton
Al Gore
Rick Lazio
Ralph Nader
Ronald Reagan
```

   Which of the following sorts are stable:
   insertion sort, selection sort, bubble sort, mergesort, heapsort, quicksort? For each one, describe why it is or is not stable.

Programming (50 points)

We will be building a rudimentary search engine in C++ using your knowledge of data structures. We will be using adopted hash tables to implement the search engine. Except for string, and vector, you can only use code from the book.

We want to be able to read either a single file or set of files into the search engine which we specify. That is the user will pass in the following information to the program

1) name of search engine
2) if we want to add data, say either a single file or set of files
3) if we want to search for a string in the search engine.

So I should be able to program something like:

```
SearchEngine se("testing123");        //will load or create the testing123 search engine
se.addDoc("test.txt");                //will be adding a single document
se.addDocs("somedirectory");          //will add all files under some dir
cout << se.getDocumentCount() << endl;     //print out how many docs we have indexed
vector<ResultSet>  results = se.search("Swimming");
//can iterate over the vector and print out the score and document name of matching doc

se.close();       //shut down the search engine
```

So that is from an outside point of view, how and where do you start ??

1) First build the outside shell which allows you to house a search engine and read file/files/directories in.
2) Make sure you can keep track of how many docs have been added to your search engine
3) You will need to check if a document has been seen in the past before adding it in…it is illegal
    a. Any ideas ??
    b. Hashtable ?
4) Think of a data structure which will allow you to scale the search engine…haha!! Hashtables again

5) An inverted index is the data structure of choice for search engines, you can get more info from :
   http://en.wikipedia.org/wiki/Inverted_index
   a. The idea is to create a hash table which will map strings (search terms) to document id's and locations within those documents.
   b. Also need to keep track of documents and document ids for use in 'a'
   c. NOTE:
      if you are running into problems creating 5, you can take less credit by implementing a vector model to index the data
6) Build a front end which asks you for search terms (if relevant) and allows you to see results.