BSTs and Balanced BSTs

1. Draw the resulting BST after inserting: 5, 9, 14, 7, 3, 4, 23, 18, 1, 2, 11, 10, 16, 21, 44, 75, 51.

2. Draw the sequence of BSTs that results after deleting items 21, 23, 14, 1, 5, 18, in this order, from the tree in problem 1.

3. Draw BSTs of minimum and maximum heights that store all the integers in the range from 1 to 7, inclusive.

4. Suppose that a BST $T$ is constructed by inserting integers 1 to $n$, in this order. Give a big-O characterization of the number of comparisons that were done to construct $T$.

5. Consider a BST $T$. Let the height of an empty tree be $-1$ and the balance of a node $x$ be the difference between the heights of the left and right subtrees of $x$, as calculated by: $\text{height}(v->left) - \text{height}(v->right)$. Write an algorithm for computing the balance of each node of $T$.

6. Consider the insertion of items with the following keys (in the given order) into an initially empty AVL tree: 30, 40, 24, 58, 26, 11, 13, 58, 62, 70, 94, 2. Draw the status of the tree after each insertion (i.e. draw the sequence of trees).

7. A certain Professor Amongus claims that the order in which a fixed set of elements is inserted into an AVL tree does not matter (i.e. the same tree results every time). Give a small example that proves Professor Amongus wrong.

8. What is the minimum number of nodes in an AVL tree of height 8?

Priority Queues and Heaps

1. Let $T$ be a complete binary tree with $n$ nodes (each key is a double) that is realized as an array $S$ of doubles (starting at index 0). Give pseudocode descriptions of each of the following methods:
(a) double root()
(b) int parent(int index)
(c) int leftChild(int index)
(d) int rightChild(int index)
(e) bool isInternal(int index)
(f) bool isExternal(int index)
(g) bool isRoot(int index)

2. Is there a heap $T$ storing seven distinct elements such that a preorder traversal of $T$ yields the elements of $T$ in sorted order? How about an inorder traversal? How about a postorder traversal? Give an example of each or state that none exist for any given traversal.

3. Given the following input sequence: 2, 5, 16, 4, 10, 23, 39, 18, 26, 15, illustrate the performance of the heap-sort algorithm. You must show the results of each iteration in array form, beginning with an array that contains these elements.

Hash Tables

1. Alice says that a hash table with collisions handled using separate chaining can have a load factor greater than 1. Bob says that this is impossible. Who is right and why?

2. Draw the 11-item hash table resulting from inserting 12, 44, 13, 88, 23, 94, 11, 39, 20, 16, 5 using the hash function $h(i) = (2i + 5) \pmod{11}$ and assuming that collisions are handled by chaining.

3. Draw the resulting hash table of Exercise 2, assuming that collisions are handled using linear probing.

4. Draw the resulting hash table of Exercise 2, assuming that collisions are handled using quadratic probing. Stop if no empty slot is found.

5. Draw the resulting hash table of Exercise 2, assuming that collisions are handled by double hashing using a secondary hash function $h'(k) = 7 - (k \pmod{7})$.

6. Dr. Wayne has a new way to do open addressing, where, for a key $k$, if the cell $h(k)$ is occupied, then he suggests trying $(h(k) + i \cdot f(k)) \pmod{N}$, for $i = 1, 2, 3, \ldots$, until finding an empty cell, where $f(k)$ is a random hash function returning values from 1 to $N - 1$. Explain what can go wrong with Dr. Wayne’s scheme if $N$ is not prime.
7. Suppose you would like to build a hash table for images, where the key for each image is a “thumbnail” image of 75x75 pixels, with each pixel being one of 256 possible colors. Describe a hash function for a collection of such images. Your hash function should be fast to compute and it should strive to map different images to different hash values. In particular, reflections and 90° rotations of the same image should, in general, map to different hash values.