CSE 326 DATA STRUCTURES
HOMEWORK 5

Due: **Friday, Feb 19, 2010** at the beginning of class. Please put your quiz section in addition to your name at the top of your homework.

Problem 1. B Trees

For the following questions about B trees, you only need to show the final result, but note that if you do this it will be hard to award partial credit if the final result is incorrect. [Note that we've referred to these as B-trees, though you'll often see them called B+ trees.]

(a) Show the result of inserting 12, 10, 15, 4, 1, 17, 3, 13, and 8 into an initially empty B tree with M = 3 and L = 2.

To maintain consistency in answers, please follow the following rules:

(a) You should *split* nodes whenever there is an overflow due to insertion. Another option, discussed in the textbook, is to put a child up for adoption to avoid splitting; this is a viable option, but we ask that you not pursue it and instead split the overflowed node.

(b) When splitting a leaf node due to insertion overflow, you keep the smallest `ceil((L+1)/2)` elements in the original node and put the largest `floor((L+1)/2)` elements in the new node. When splitting an internal node, you keep the `ceil((M+1)/2)` children with the smaller values attached to the original node and attach the `floor((M+1)/2)` children with the larger values to the new node. So, after splitting a node into a "left" node and a "right" node, the number of elements (or children) in the left node will always be greater than or equal to the number of elements (or children) in the right node.

(b) Now show the result of deleting 12, 13, and 15.

Problem 2. Binary Search Trees

Write a method to generate a perfectly balanced binary search tree of height h with keys 1 through $2^{h+1} - 1$. Your solution should have a linear runtime. Use pseudo code (or Java) for your answer.

Problem 3. Hash Tables

(a) Weiss, problem 5.1. Assume the table size (array size) is 10.

(b) Weiss, problem 5.2. Choose the new table size to be 19, which is prime and roughly twice as big. Naturally, when rehashing, you should start with the corresponding input hash table and rehash elements from top to bottom; i.e., after hashing, you no longer know the insertion order and should simply iterate through the existing hash table. Also, if any items were not successfully inserted the first time, they should be inserted in this pass, at the end, in the order of failure. (In a real situation, if you failed to insert an element, you would probably rehash to a larger table immediately.)