CSE 326 DATA STRUCTURES
HOMEWORK 2

Due: **Friday, January 19, 2007** at the beginning of class in the lecture section you are registered in
Please put your quiz section (AA,AB,BA,BB) in addition to your name at the top of your homework

Problem 1. Fun with Binary Min Heaps

In this problem, you'll perform the basic operations on a binary min heap. Make sure to check your work!

(a) Starting with an empty binary min heap, show the result of inserting, in the following order, 10, 12, 1, 14, 6, 5, 8, 15, 3, 9, 7, 4, 11, 13, and 2, one at a time (using percolate up each time), into the heap. By *show* here we mean "draw the resulting binary tree with the values at each node." Check your work!

(b) Now imagine that for the binary min heap you constructed in part (a), you perform two `deleteMin`'s. What is the resulting binary min heap? ("draw the resulting binary tree with values at each node").

(c) Instead of inserting the elements in part (a) into the heap one at a time, suppose that you use the linear time worst case algorithm described on page 211 of Weiss (Floyd's algorithm). What would your binary min heap then look like?

Problem 2. d-Heap Arithmetic

Binary heaps implemented using an array have the nice property of finding children and parents of a node using only multiplication and division by 2 and incrementing by 1. This arithmetic is often very fast on most computers, especially the multiplication and division by 2, since this corresponds to simple bitshift operations. In *d*-heaps, the arithmetic is also fairly straightforward, but is no longer necessarily as fast. In this problem you'll figure out exactly what this math is.

(a) Weiss problem 6.14.

Problem 3. Min-Max Heaps

One problem with binary min heaps is that finding the maximum element in the heap cannot be done in logarithmic time. A solution to this is to use what is called a Min-Max heap. In this problem you'll explore the min-max heap and give algorithms for insertion and deletion of the min and max into the min-max heap.

(a) Weiss 6.18 parts (a), (b), and (c).