ECE-250 – Algorithms and Data Structures (Winter 2012)

Assignment 4

Due on Friday, March 16, in class.

All questions will have equal weight for the assignment grade (except for the bonus marks questions).

1 – (a) Insert the following values (in the given order) into an initially empty min heap: 46, 34, 83, 75, 25, 57, 93, 27, 17.

You need to justify the steps, but a single sentence describing the steps suffices — provided that all the key details are indeed described (you still need to show the step-by-step contents of the heap, after each insertion).

(b) Dequeue the elements, showing the contents of the heap after each removal.

2 – (a) Dequeue each of the elements from the following min heap represented as an array (you must show the procedure with the processing as an array, including obtaining the subscripts for parents and children, and not drawing the "tree view" of the heap).

	2	5	7	6	12	8	18	11	31	13	21	29	9	19

(b) Convert the following array of unsorted values into a min heap (in-place, with a linear time procedure): $\{48, 23, 17, 26, 14, 87, 53, 89, 32, 49, 95, 12\}$

3 – Describe a traversal strategy for a min-heap that visits every element that is less than a given value. The run time must be $O(m)$, where m is the number of elements that satisfy the condition.

2% Bonus Marks: Write a C++ function that implements that traversal to print the values (you may assume a `Binary_node` class as described in class — of course, with the assumption of the min heap constraints)

4 – For the following array of values: $\{48, 23, 17, 26, 14, 87, 53, 89\}$:

(a) Sort the values (showing the procedure, step-by-step), using merge sort. For the recursion's base case, you may use the point where the array size reaches two, at which point sorting boils down to a conditional swap of the values.

(b) Run the first iteration of quick sort, using median of three as the approximation for the median (that is, the step that ends with every value in the first chunk being less than any of the values in the second chunk).

Show the step-by-step procedure.

5% Bonus Marks – Write a C++ function that determines whether a given binary tree is a min heap. The prototype of the function should be:

```
template <typename Type>
bool is_min_heap (const Binary_node<Type> * tree);
```

5% Bonus Marks – Prove that removal (of the root node) from a heap has worst-case run time $\Omega(\log n)$; that is, prove that removal from a heap can not have a worst-case run time faster than $\Theta(\log n)$.