CS 410 Summer 2000
Homework 3
Due 11:30 AM, Friday July 14

Important notes about all exercises for this course:
- No late homework will be accepted.
- Justify your answers. Answers without brief (but adequate) justification may be considered incomplete.
- You may use pseudocode, Java, C, or C++ to write your algorithms. You do not need to compile or run your code for written exercises.
- When presenting algorithms, also provide a brief English description of the pseudocode and an explanation of why the algorithm (and your analysis of it) is correct.

Written Exercises

1. Consider the following 3 sets of data:

   (a) 1 2 3 4 5

   (b) 5 4 3 2 1

   (c) 3 1 4 2 5

   Show step by step how each of these data sets will be sorted by each of insertionsort, mergesort, and heapsort, respectively. Show what gets changed after the execution of each step. Use the algorithms as detailed in CLR. Create a table which gives the total number of array element comparisions made in each case.

2. Is the sequence $\langle 23, 17, 14, 6, 13, 10, 1, 2, 5, 7, 12 \rangle$ a heap?

3. In the *Partition* subroutine of *Quicksort*, it would be problematic if the pointer j (which starts at the back) ended pointing to the last element of the array, since then the two "subarrays" formed would be of size n and 0, respectively. Explain carefully why this can never happen with the pseudocode given.

4. CLR 8.3 on page 169.

5. CLR 9.1-1.

6. CLR 9.1-3.

7. Demonstrate your understanding of linear time sorting algorithms:

   (a) Using CLR Figure 9.2 as a model, illustrate the operation of Counting-Sort on the array $A = \langle 7, 1, 7, 4 \rangle$.

   (b) Using CLR Figure 9.3 as a model, illustrate the operation of Radix-Sort on the following list of English words:
   BOX, COW, BAR, FOX, EAR.

   (c) Using CLR Figure 9.4 as a model, illustrate the operation of Bucket-Sort on the array $A = \langle .71, .58, .41, .17, .14 \rangle$.

8. Give an algorithm which sorts any array of input size 5 with no more than 7 comparisions. ($\lceil \lg 5! \rceil = 7$, hence 7 comparisions are needed.) This algorithm (not others) should be given in decision tree format.

9. Consider a 2 dimensional array A[1..m,1..n]. Each row of the array is sorted in nondecreasing order and also each column is sorted in nondecreasing order. Given an element x, give an algorithm which detects whether x is in A in time O(m+n).

10. You have m arrays $A_1, A_2, ...A_m$. A_i has n_i elements. $n_1 + n_2 + ... + n_m = n$. The numbers from 1..n are distributed among these "m" arrays. sort all the "m" arrays in time O(n).

11. CLR 12.4-1.