CS3230 : Design and Analysis of Algorithms (Spring 2015)
Tutorial Set #4

(Decision Tree Modeling, Counting Sort, Radix Sort, Find Min/Max, Selection)
[For discussion during Week 6]

OUT: 09-Feb-2015 **Tutorials:** Mon & Fri, 16, 20 Feb 2015

Topics: Lower Bound, Linear Time Sorting, Selection, Max, Min, Max-Min, Selection

IMPORTANT: Read "Remarks about Homework" – also applies to tutorials.

Prepare your answers to all the D-Problems in every tutorial set.

When preparing to present your answers,

- First explain clearly what is the problem,
- Think of a CLEAR EXPLANATION,
- Illustrate with a good worked example,
- Describe the main ideas,
- Can you sketch why the solution works,
- Give analysis of running time, if appropriate,
- Can you think of other (perhaps simpler) solutions?

Helpful Hints Series: Because of busy schedule and in his haste to send out this tutorial, Prof. Leong is taking a break on this hint series. Will restart this series after the semester break.

Routine Practice Problems *-- do not turn these in -- but make sure you know how to do them.*

R1. (Exercise 8.2-4 on p. 197 of [CLRS])
Describe an algorithm that, given n integers in the range 0 to k, preprocesses its input and then answers any query about how many of the n integers fall into a range $[a..b]$ in $O(1)$ time. Your algorithm should use $O(n + k)$ preprocessing time.

R2. (Exercise 8.3-2 on p. 200 of [CLRS] modified)
Which of the following sorting algorithms are stable: bubble sort, selection sort, insertion sort, merge sort, heapsort, and quicksort? Give a *simple scheme* that makes *any sorting algorithm* stable. How much additional time and space does your scheme entail?

D-Problems: Solve these D-problems and prepare to discuss them in tutorial class. You may be called upon to present your solution *or your best attempt at a solution*. Your solution presentation does NOT need to be fully correct, given your best attempt. The TA will help clarify and correct any issues or errors.

D1. (Exercise 8.3-1 on p. 199 of [CLRS]) [Exercising Radix-Sort]
[First, read up on Radix-Sort in text or from lecture notes]
Using Figure 8.3 (or lecture note diagram) as a model, illustrate the operation of RADIX-SORT on the following list of English words: COW, DOG, SEA, RUG, ROW, MOB, BOX, TAB, BAR, EAR, TAR, DIG, BIG, TEA, NOW, FOX.

D2. [About Finding Largest and related problems]

(a) [Lower Bound on Find-Max]
Show that *any* method/algorithm for finding the *largest number* in a table of *n* numbers requires at least (n–1) comparisons.
(*Hint*: Each comparison has only one loser.)

(b) Exercise 9.1-1 (p.215) [Second Largest]
Second largest of *n* elements in ($n + \lceil \lg n \rceil - 2$) comparisons in the worst case.
(*Hint:* Also find the largest element.)

D3. [Finding Min-and-Max simultaneously]
This problem deals with finding both the *minimum and maximum* of *n* elements using as few comparisons as possible. One simple way is to just call Find-Min and Find-Max separately, which will take $2(n–1)$ comparisons. But, we can do better.

Given an algorithm to find both the *minimum and maximum* of *n* elements using as most $3\lceil n/2 \rceil$ comparisons (in the worst case).

D4. [Weighing Problem – Contributed by Tan Lee Eng, 1991]
You are given 27 coconuts labeled 1, 2, . . . , 27, and *only* a beam balance. Among these 27 coconuts, 26 of them have exactly the *same* weight, but one of them is *lighter*. We want an algorithm to identify the lighter coconut using *only* the beam balance (which gives three possible outcomes: left-side is heavy, both-sides balanced, right-side is heavy).

(a) Before solving this, how about solving the smaller 3 coconuts problem. Give a simple algorithm for doing this. Specify which coconuts to put onto the beam balance and what to do for each possible outcome of the weighing.

(b) Now, give an algorithm to solve the original 27 coconuts problem, namely to identify the lighter coconut using the least number of weighing with the beam balance.

(c) [**Question:** What, if anything, does this problem have to do with Analysis of Algorithms? Or, why is Prof. Leong asking this weird question in CS3230?] (Short answers please.)

Advanced Problems – Try these for challenge and fun. There is no deadline for A-problems. *Turn in your attempts DIRECTLY to Prof. Leong. Do not combine it with your HW solutions.*)

A7: **[More Lower Bounds]** Prove the lower bound of $(\lceil 3n/2 \rceil - 2)$ comparisons in the worst case to find both the maximum and the minimum of n numbers.
(*Hint:* Consider how many numbers are potentially either the maximum or minimum, and investigate how a comparison affects these counts. Adopt an *adversary argument* to force *any algorithm* to do as many comparisons as possible before finding the answer.)