CS3933 Assignment 9
Summer 2007

Due: Thursday, July 19, in class or by 10:00am in the box on E-level. Please try the questions before Tuesday, and ask questions then if you want to.

- Read the rest of chapter 6.

- Once again, a repeated question on Divide-and-Conquer. Do question 3 from chapter 5 again, but this time find a linear time algorithm. The algorithm does not look so much like a divide-and-conquer algorithm, because it only requires one recursive call to itself, like binary search.

  Hint: The recurrence relation satisfied by the time-complexity function is $T(n) = T(n/2)+O(n)$. Remember that we use worst case complexity; the recursive call can use fewer than $n/2$ elements.

  Actually the recursive call could use $(n + 1)/2$ cards, but I made a simplifying assumption so that you can do the following question easily. Prove that the algorithm requires only linear time. I want a detailed proof using the definition of big O.

- Two linear dynamic programming problems. Do questions 3 (page 314) and 7 (page 318) in chapter six.

  To solve a dynamic programming problem, you must define a mathematical function *in terms of the inputs*, then give a recurrence relation that it must satisfy (including a base case), and explain how you can calculate the function iteratively.

  Next week we will move to two dimensional problems.