# CS 577: Introduction to Algorithms — Homework 3b

**Out: 10/26/06** — **Due: 11/02/06**

## Ground Rules

- Grading. You will be graded on the correctness as well as clarity of your solutions. You are required to prove any claims that you make. In particular, when you are asked to design an algorithm, you must argue its correctness and running time.
- Collaboration. You are allowed to discuss questions with other people in the class. However, **you must solve and write your answers yourself without any help**. You must also give explicit citations to any sources besides the textbook and class notes, including discussions with classmates. Solutions taken from external sources such as the WWW, even if cited, will receive no credit unless there is significant "value added". In cases of doubt, you may be asked to explain your answer to the instructor and this will determine your grade.
- Lateness. Please see the class webpage for details on the lateness policy.
- This homework is due in *one week*. Start working early. Plan your work in such a way that you have the opportunity to put some problems on the back burner for a while and revisit them later. Good luck!

## Problems

1. **(Got change?)** You are given coins of denominations $a_1, a_2, \cdots, a_n$ (in cents), and are required to make change for a certain amount $x$ (also in cents). In all of the below questions, your job is to design a dynamic programming based polynomial-time algorithm. In each case, you should give either a (top-down) recursive procedure for the DP, or a bottom-up array-filling procedure. Explain what each entry of the array stands for, and give the algorithm's running time. A proof of correctness is not required.

   (a) Give an algorithm for determining whether you can make change for $x$ using the given denominations.

   (b) Give an algorithm for determining whether you can make change for $x$ using at most one coin of each given denomination.

   (c) Give an algorithm for determining whether you can make change for $x$ using at most $k$ coins in total from among the given denominations.

2. **(Winning Strategy.)** Consider the following card game between two players. The dealer deals $n$ cards face up in a row. Each card has a value visible to both players. The players move in sequence. Each player, at her turn, can pick either the rightmost or the leftmost of the remaining cards on the table. The game ends when no cards are left. The goal of each player is to accumulate as much value as possible.

   For example, suppose that $n = 4$ and the cards have values $4, 2, 1, 7, 5$. Then if the players pick the cards in the order $5, 7, 4, 2, 1$, then the first player gets total value $5 + 4 + 1 = 10$ and the second gets value $7 + 2 = 9$. On the other hand, if the players pick the cards in the order $4, 5, 7, 2, 1$, then the first player gets total value $4 + 7 + 1 = 12$ and the second gets value $5 + 2 = 7$.

   Design an $O(n^2)$ time algorithm for computing the optimal strategy for the first player. In particular, given a list of values, your algorithm should output whether the first player should pick the leftmost or the rightmost card in order to maximize her total value. Assume that the second player also plays optimally.

3. Problem 6.23 in the textbook (p. 331).

4. Problems 7.2 and 7.3 in the textbook (p.415–416).