CSE 101 - ALGORITHMS - SUMMER 2000
Homework 2 - Solutions

Problem 1: Problem 3-6 in Skiena

(10 points) In the United States, coins are minted with denominations of 1,5,10,25, and 50 cents. Now consider a country whose coins are minted with denominations of $\{d_1, ..., d_k\}$ units. They seek an algorithm that will enable them to make change of n units using the minimum number of coins.

1. The greedy algorithm for making change repeatedly uses the biggest coin smaller than the amount to be changed until it is zero. Show that the greedy algorithm does not always give the minimum number of coins in a country whose denominations are $\{1, 6, 10\}$.
2. Give an efficient algorithm that correctly determines the minimum number of coins needed to make change of n units using denominations $\{d_1, ..., d_k\}$. Analyze its running time.

Proof: This problem consists of b) and c) of Exercise 4.9 in the *revised* Lecture Notes 6. Go there for a solution for this problem.

Problem 2: Problem 3-9 in Skiena

(10 points) Consider the following data compression technique. We have a table of m text strings, each of length at most k. We want to encode a data string D of length n using as few text strings as possible. For example, if our table contains $(a, ba, abab, b)$ and the data string is $bababbaababa$, the best way to encode it is $(b, abab, ba, abab, a)$ - a total of five code words. Give and $O(nmk)$ algorithm to find the length of the best encoding. You may assume that the string has an encoding in terms of the table.

Proof: This is Exercise 4.8 in the *revised* Lecture Notes 6. Check Lecture Notes 6 for a dynamic programming solution for this problem.

Problem 3: Blackjack Hand Card Counting

(10 points) You are given an array A of n positive integers (cards with face values) with values from 1 to k, and positive integers $l < n, v < kn$. Count the number of sets of l array positions (hands of l cards) whose total value is equal to v. Give the best algorithm you can for this problem. Analyze your algorithm in terms of n, k and l. Your algorithm should take time polynomial in all 3 parameters.

Proof: This is Exercise 4.10 in Lecture Notes 6. Go there for a solution.