CS4040/5040
Homework # 6

Due Friday, December 3, by 11:59pm. (CS4040 75 pts / CS5040 85 pts)
(plus a possible 10point bonus)

1. (10 pts.) **Coin Changing**. Give a set of coins of size at least 5 for which the greedy algorithm does not produce an optimal solution. Provide the optimal solution for this set, showing all work for a *dynamic programming solution*. (i.e. show the required table(s)). Your set of coins must include a 1 cent coin.

2. (10 pts.) **Huffman encoding**. Given that frequencies of letters to encode via Huffman encoding are given by the following recurrence relation: $f(n) = f(n-1) + f(n-2) + 2f(n-3)$, and that the initial three letters (a, b, and c) have initial frequencies 1, 1, and 2, what is the code for the first 8 letters? (i.e. a:1, b:1, c:2, d:5, e:9, f:18, g:37, h:73). Ties should be broken by keeping the deeper tree on the left. If two trees have equal depth, keep the one with the earliest letter in the alphabet on the left. Can this solution be generalized to the first n letters? If so, provide the generalization.

3. (10 pts.) Given n points in the plane, develop an algorithm that can find all line segments that contain 4 or more of the points in $O(n^2 \log n)$ time.

4. (10 pts.) Give a formal definition for the problem of finding the longest simple cycle in an undirected graph. Give the formal definition of the related decision problem. Give the language corresponding to the decision problem.

5. (10 pts.) Provide a dynamic programming solution (psuedocode) to the $0-1$ knapsack problem. Your algorithm should run in $O(nP)$ time where n is the number of items and P is the maximum weight in pounds that the knapsack can hold.

6. (15 pts.)

   (a) State the formal decision version of the $0-1$ knapsack problem.

   (b) Describe how the solution to problem 5 could be used to solve the decision version of the $0-1$ knapsack problem. What is the time complexity of this algorithm that solves the decision problem?

   (c) Is the dynamic programming solution to the $0-1$ knapsack problem a polynomial time algorithm? You should assume that we are talking about the decision version of the 0-1 knapsack problem. Explain your answer.

7. (10 pts.) Consider the language GRAPH-ISOMORPHISM $= \{\langle G_1, G_2\rangle : G_1$ and G_2 are isomorphic graphs$\}$. Prove that GRAPH-ISOMORPHISM $\in$ NP by describing a polynomial-time algorithm that verifies the language.

8. (10 pts.) This problem is for extra credit. Given an $m \times n$ matrix A composed of all integers and an m-vector c also composed of all integer values, consider the problem of determining whether there exists an n-vector y whose entries are either 1 or 0 and whose product with A is less than c, i.e. $Ay \leq c$. This notation means that each entry of the vector on the left is $\leq$ to each corresponding value of the vector on the right. Prove or disprove that this problem is in NP.

CS5040 only:

9. (10 pts.) Exercise 34.2-9, page 1066.