CS 473G: Combinatorial Algorithms, Fall 2005
Homework 2

Due Thursday, September 22, 2005, by midnight (11:59:59pm CDT)

Name:	
Net ID:	Alias:

Name:	
Net ID:	Alias:

Name:	
Net ID:	Alias:

Starting with Homework 1, homeworks may be done in teams of up to three people. Each team turns in just one solution, and every member of a team gets the same grade.

Neatly print your name(s), NetID(s), and the alias(es) you used for Homework 0 in the boxes above. Staple this sheet to the top of your homework.

Required Problems

1. (a) Suppose Lois has an algorithm to compute the shortest common supersequence of two arrays of integers in $O(n)$ time. Describe an $O(n \log n)$-time algorithm to compute the longest common subsequence of two arrays of integers, using Lois's algorithm as a subroutine.

   (b) Describe an $O(n \log n)$-time algorithm to compute the longest increasing subsequence of an array of integers, using Lois's algorithm as a subroutine.

   (c) Now suppose Lisa has an algorithm that can compute the longest increasing subsequence of an array of integers in $O(n)$ time. Describe an $O(n \log n)$-time algorithm to compute the longest common subsequence of two arrays $A[1..n]$ and $B[1..n]$ of integers, **where $A[i] \neq A[j]$ for all $i \neq j$**, using Lisa's algorithm as a subroutine.[1]

[1] For extra credit, remove the assumption that the elements of A are distinct. This is probably impossible.

2. In a previous incarnation, you worked as a cashier in the lost 19th-century Antarctican colony of Nadira, spending the better part of your day giving change to your customers. Because paper is a very rare and valuable resource on Antarctica, cashiers were required by law to use the fewest bills possible whenever they gave change. Thanks to the numerological predilections of one of its founders, the currency of Nadira, called Dream Dollars, was available in the following denominations: \$1, \$4, \$7, \$13, \$28, \$52, \$91, \$365.[2]

   (a) The greedy change algorithm repeatedly takes the largest bill that does not exceed the target amount. For example, to make \$122 using the greedy algorithm, we first take a \$91 bill, then a \$28 bill, and finally three \$1 bills. Give an example where this greedy algorithm uses more Dream Dollar bills than the minimum possible.

   (b) Describe and analyze an efficient algorithm that computes, given an integer n, the minimum number of bills needed to make n Dream Dollars.

3. Scientists have branched out from the bizarre planet of Yggdrasil to study the vodes which have settled on Ygdrasil's moon, Xryltcon. All vodes on Xryltcon are descended from the first vode to arrive there, named George. Each vode has a color, either cyan, magenta, or yellow, but breeding patterns are *not* the same as on Yggdrasil; every vode, regardless of color, has either two children (with arbitrary colors) or no children.

   George and all his descendants are alive and well, and they are quite excited to meet the scientists who wish to study them. Unsurprisingly, these vodes have had some strange mutations in their isolation on Xryltcon. Each vode has a *weirdness* rating; weirder vodes are more interesting to the visiting scientists. (Some vodes even have negative weirdness ratings; they make other vodes more boring just by standing next to them.)

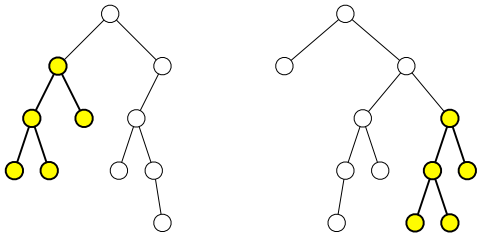
   Also, Xryltconian society is strictly governed by a number of sacred cultural traditions.

   - No cyan vode may be in the same room as its non-cyan children (if it has any).
   - No magenta vode may be in the same room as its parent (if it has one).
   - Each yellow vode must be attended at all times by its grandchildren (if it has any).
   - George must be present at any gathering of more than fifty vodes.

   The scientists have exactly one chance to study a group of vodes in a single room. You are given the family tree of all the vodes on Xryltcon, along with the wierdness value of each vode. Design and analyze an efficient algorithm to decide which vodes the scientists should invite to maximize the sum of the wierdness values of the vodes in the room. Be careful to respect all of the vodes' cultural taboos.

[2] For more details on the history and culture of Nadira, including images of the various denominations of Dream Dollars, see http://www.dream-dollars.com. Really.

4. A *subtree* of a (rooted, ordered) binary tree T consists of a node and all its descendants. Design and analyze an efficient algorithm to compute the *largest common subtree* of two given binary trees T_1 and T_2, that is, the largest subtree of T_1 that is isomorphic to a subtree in T_2. The *contents* of the nodes are irrelevant; we are only interested in matching the underlying combinatorial structure.



Two binary trees, with their largest common subtree emphasized

5. Let $D[1..n]$ be an array of digits, each an integer between 0 and 9. An *digital subsequence* of D is an sequence of positive integers composed in the usual way from disjoint substrings of D. For example, $3, 4, 5, 6, 23, 38, 62, 64, 83, 279$ is an increasing digital subsequence of the first several digits of π:

$$\boxed{3}, 1, \boxed{4}, 1, \boxed{5}, 9, \boxed{6}, \boxed{2,3}, 4, \boxed{3,8}, 4, \boxed{6,2}, \boxed{6,4}, 3, 3, \boxed{8,3}, \boxed{2,7,9}$$

The *length* of a digital subsequence is the number of integers it contains, *not* the number of digits; the previous example has length 10.

Describe and analyze an efficient algorithm to compute the longest increasing digital subsequence of D. *[Hint: Be careful about your computational assumptions. How long does it take to compare two k-digit numbers?]*

*6. *[Extra credit]* The *chromatic number* of a graph G is the minimum number of colors needed to color the nodes of G so that no pair of adjacent nodes have the same color.

   (a) Describe and analyze a *recursive* algorithm to compute the chromatic number of an n-vertex graph in $O(4^n \operatorname{poly}(n))$ time. *[Hint: Catalan numbers play a role here.]*

   (b) Describe and analyze an algorithm to compute the chromatic number of an n-vertex graph in $O(3^n \operatorname{poly}(n))$ time. *[Hint: Use dynamic programming. What is $(1+x)^n$?]*

   (c) Describe and analyze an algorithm to compute the chromatic number of an n-vertex graph in $O((1+3^{1/3})^n \operatorname{poly}(n))$ time. *[Hint: Use (but don't regurgitate) the algorithm in the lecture notes that counts all the maximal independent sets in an n-vertex graph in $O(3^{n/3})$ time.]*

Practice Problems

*1. Describe an algorithm to solve 3SAT in time $O(\phi^n \operatorname{poly}(n))$, where $\phi = (1+\sqrt{5})/2$. *[Hint: Prove that in each recursive call, either you have just eliminated a pure literal, or the formula has a clause with at most two literals.]*

2. Describe and analyze an algorithm to compute the longest increasing subsequence in an n-element array of integers in $O(n \log n)$ time. *[Hint: Modify the $O(n^2)$-time algorithm presented in class.]*

3. The *edit distance* between two strings A and B, denoted $\text{Edit}(A, B)$, is the minimum number of insertions, deletions, or substitutions required to transform A into B (or vice versa). Edit distance is sometimes also called the *Levenshtein distance*.

   Let $\mathbf{A} = \{A_1, A_2, \ldots, A_k\}$ be a set of strings. The *edit radius* of $\mathbf{A}$ is the minimum over all strings X of the maximum edit distance from X to any string A_i:

$$\text{EditRadius}(\mathbf{A}) = \min_{\text{strings } X} \max_{1 \le i \le k} \text{Edit}(X, A_i)$$

   A string X that achieves this minimum is called an *edit center* of $\mathbf{A}$. A set of strings may have several edit centers, but the edit radius is unique.

   Describe an efficient algorithm to compute the edit radius of three given strings.

4. Given 5 sequences of numbers, each of length n, design and analyze an efficent algorithm to compute the longest common subsequence among all 5 sequences.

5. Suppose we want to display a paragraph of text on a computer screen. The text consists of n words, where the ith word is $W[i]$ pixels wide. We want to break the paragraph into several lines, each exactly L pixels long. Depending on which words we put on each line, we will need to insert different amounts of white space between the words. The paragraph should be fully justified, meaning that the first word on each line starts at its leftmost pixel, and *except for the last line*, the last character on each line ends at its rightmost pixel. (Look at the paragraph you are reading right now!) There must be at least one pixel of white space between any two words on the same line. Thus, if a line contains words i through j, then the amount of *extra* white space on that line is $L - j + i - \sum_{k=i}^{j} W[k]$.

   Define the *slop* of a paragraph layout as the sum, over all lines *except the last*, of the *cube* of the extra white space in each line. Describe an efficient algorithm to layout the paragraph with minimum slop, given the list $W[1..n]$ of word widths as input. You can assume that $W[i] < L/2$ for each i, so that each line contains at least two words.

6. A *partition* of a positive integer n is a multiset of positive integers that sum to n. Traditionally, the elements of a partition are written in non-decreasing order, separated by + signs. For example, the integer 7 has exactly twelve partitions:

$$\begin{array}{ccc} 1+1+1+1+1+1+1 & 3+1+1+1+1 & 4+1+1+1 \\ 2+1+1+1+1+1 & 3+2+1+1 & 4+2+1 \\ 2+2+1+1+1 & 3+2+2 & 4+3 \\ 2+2+2+1 & 3+3+1 & 7 \end{array}$$

The *roughness* of a partition $a_1 + a_2 + \cdots + a_k$ is defined as follows:

$$\rho(a_1 + a_2 + \cdots + a_k) = \sum_{i=1}^{k-1} |a_{i+1} - a_i - 1| \; + \; a_k - 1$$

A *smoothest* partition of n is the partition of n with minimum roughness. Intuitively, the smoothest partition is the one closest to a descending arithmetic series $k + \cdots + 3 + 2 + 1$, which is the only partition that has roughness 0. For example, the smoothest partitions of 7 are $4 + 2 + 1$ and $3 + 2 + 1 + 1$:

$$\begin{array}{ccc} \rho(1+1+1+1+1+1+1) = 6 & \rho(3+1+1+1+1) = 4 & \rho(4+1+1+1) = 4 \\ \rho(2+1+1+1+1+1) = 4 & \rho(3+2+1+1) = 1 & \rho(4+2+1) = 1 \\ \rho(2+2+1+1+1) = 3 & \rho(3+2+2) = 2 & \rho(4+3) = 2 \\ \rho(2+2+2+1) = 2 & \rho(3+3+1) = 2 & \rho(7) = 7 \end{array}$$

Describe and analyze an algorithm to compute, given a positive integer n, a smoothest partition of n.