National Taiwan University of Science and Technology
Department of Electrical Engineering

Algorithm Design and Application

Homework #2 (Units 3 & 4)

1. Exercise 12.2-1.
2. Search trees.
   (a) Give the binary search tree that results from successively inserting the keys 7, 2, 1, 5, 4, 6, 8 into an initially empty tree.
   (b) Label each node in the tree with R or B denoting the respective colors RED and BLACK so that the tree is a legal red-block tree.
   (c) Give the red-black tree that results from inserting the key 3 into the tree of (b).
   (d) Give the red-black tree that results from deleting the key 1 from the tree of (c).
3. Exercise 15.2-1.
4. Exercise 15.4-1.
5. Determine the cost and structure of an optimal binary search tree for a set of $n = 4$ keys with the following probabilities:

i	0	1	2	3	4
p_i		0.10	0.15	0.10	0.10
q_i	0.05	0.10	0.20	0.10	0.10

6. Lamigo Monkeys and Brother Elephants are competing for the 2016 Taiwan Professional Baseball Championship. Both teams play a series of games until one of the teams wins n games. Assume that the probability of the Lamigo Monkeys Team winning a game is the same for each game and equal to p and the probability of losing a game is $q = 1 - p$. (Hence, there are no ties.) Let $P(i, j)$ be the probability of Lamigo Monkeys winning the series under the condition that Lamigo Monkeys needs i more games to win the championship and Brother Elephants needs j more games to win the championship.
   (a) Find the optimal substructure (a recurrence relation for $P(i, j)$).
   (b) The probability of the Lamigo Monkeys team winning a game is only 0.4. Find the probability of Lamigo Monkeys winning a 3-game series (i.e., 2 wins to get the championship), **based on the recurrence found in (a).**
   (c) Give a dynamic programming algorithm for solving the problem (please write down your pseudo-code). What are the time and space complexity of your algorithm?
7. Suppose you are given three strings of characters: $X = x_1x_2 \cdots x_m$, $Y = y_1y_2 \cdots y_n$, and $Z = z_1z_2 \cdots z_{m+n}$. Z is said to be a *shuffle* of X and Y if Z can be formed by interspersing the characters from X and Y in a way that maintains the left-to-right ordering of the characters from each string. For example, *cchocohilaptes* is a shuffle of *chocolate* and *chips*, but *chocochilatspe* is not a shuffle of the two strings. Let s be a Boolean table where each entry $s[i, j]$ is TRUE if and only if Z_{i+j} is a shuffle of X_i and Y_j, where $X_i = x_1x_2 \cdots x_i$, $Y_j = y_1y_2 \cdots y_j$, $1 \leq i \leq m$, and $1 \leq j \leq n$.
   (a) Find the optimal substructure (i.e., derive the recurrence of $s[i, j]$). (**Hint**: Recall what we do for the LCS problem discussed in class.)
   (b) Design a dynamic programming algorithm **Is_Shuffle** that takes as inputs X, Y, Z, m, and n, and determines whether Z is a shuffle of X and Y (**utilize the recurrence derived in (a) instead of directly adopting LCS**).
   (c) What is the running time of your algorithm?