Homework Assignment #4

Due Time/Date

1:20PM Tuesday, October 8, 2024. Late submission will be penalized by 20% for each working day overdue.

How to Submit

Please write or type your answers on A4 (or similar size) paper. Put your completed homework on the table in front of the instructor's desk before the class on the due date starts. For early or late submissions, please drop them in Yih-Kuen Tsay's mail box on the first floor of Management College Building 2. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

There are five problems in this assignment, each accounting for 20 points. (Note: problems marked with "(X.XX)" are taken from [Manber 1989] with probable adaptation.)

1. Design an efficient algorithm that, given a sorted array A of n integers and an integer x, determine whether A contains two integers whose sum is exactly x. Please present your algorithm in adequate pseudocode and make assumptions wherever necessary. Give also an analysis of its time complexity. The more efficient your algorithm is, the more points you will be credited for this problem.

2. (5.17) The Knapsack Problem that we discussed in class is defined as follows. Given a set S of n items, where the ith item has an integer size $S[i]$, and an integer K, find a subset of the items whose sizes sum to exactly K or determine that no such subset exists.

   We have described in class an algorithm to solve the problem. Modify the algorithm to solve a variation of the knapsack problem where each item has an *unlimited* supply. In your algorithm, change the type of $P[i,k].belong$ into integer and use it to record the number of copies of item i needed.

3. (5.20 adapted) You are given a set of n integers, stored in an array X. Design an algorithm to partition the set into two subsets of equal sum, or determine that it is impossible to do so. Please present your algorithm in adequate pseudocode and make assumptions wherever necessary. Give also an analysis of its time complexity.

4. Suppose that you are given an algorithm/function called *subsetSum* as a *black box* (you cannot see how it is designed) that has the following property: if you input any sequence X of real numbers and an integer k, $subsetSum(X,k)$ will answer "yes" or "no", indicating whether there is a subsequence of the numbers whose sum is exactly k. Show how to use this black box to find the subsequence whose sum is k, if it exists.

   Please present your algorithm in adequate pseudocode and make assumptions wherever necessary. You should use the black box $O(n)$ times (where n is the size of the sequence). Note that a sequence of n numbers are stored in an array of n elements. Note also that the input k must be an integer.

5. (5.23) Write a non-recursive program (in suitable pseudocode) that prints the moves of the solution to the towers of Hanoi puzzle. The three pegs are respectively named A, B, and C, with n (generalizing the original eight) disks of different sizes stacked in decreasing order on peg A.