Instructor Name: Siddhartha Nath
Email: s.nath@northeastern.edu

Program Structure and Algorithms (INFO 6205)
Homework #5 – 100 points

Student NAME:

Student ID:

Question 1 (30 points). *A **palindrome** is a non-empty string over some alphabet that reads the same forward and backward. Examples of palindromes are all strings of length 1, civic, racecar, and aibohphobia (fear of palindromes).*
Give an efficient algorithm to find the longest palindrome that is a subsequence of a given input string of length n. For example, given the input $\{character\}$, your algorithm should return $\{carac\}$.

(a) (8 points) Please describe your subproblems.

(b) (8 points) Please describe the decisions.

(c) (10 points) Please write down the recursion and base cases.

(d) (4 points) Please describe the running time of your algorithm in terms of the number of subproblems and the running time per subproblem.

Question 2 (30 points). *You have to store n books $(b_1, b_2, \ldots, b_n)$ on shelves in a library. The order of books is fixed by the cataloging system and cannot be rearranged. A book b_i has a thickness t_i and height h_i, where $1 \leq i \leq n$. The length of each bookshelf at the library is L. The values of h_i are not necessarily assumed to be all the same. In this case, we have the freedom to adjust the height of each bookshelf to that of the tallest book on the shelf. Thus, the cost of a particular layout is the sum of the heights of the largest book on each shelf.*

***(a)** (10 points) Give an example to show that the greedy algorithm of stuffing each shelf as full as possible does not always give the minimum overall height.*

***(b)** (20 points) Describe in English an algorithm based on dynamic programming to achieve the minimum overall height. No need to provide pseudocode, but you must state your subproblems, decisions, recursion, base case(s) and analyze the running time complexity.*

Question 3 (40 points). *You are given an unlimited supply of coins of a given denomination S and a target amount N. Your goal is to find the minimum number of coins required to make change for N using the given denominations.*

Example #1. *Suppose the denominations as $S = \{1, 3, 5, 7\}$ and $N = 18$, the minimum number coins is 4 (i.e., $\{7, 7, 3, 1\}$ or $\{5, 5, 5, 3\}$ or $\{7, 5, 5, 1\}$).*

Example #2. *Suppose the denominations are $S = \{25, 10, 1\}$ and $N = 40$, the minimum number of coins is 4 (i.e., $\{10, 10, 10, 10\}$).*

(a) *(6 points) Can you show that the greedy choice (used in HW4) for coin changing is suboptimal for Example #2?*

(b) *(4 = 2 + 2 points) Can you please describe a brute-force (non-DP) approach to solve Example #2? What will be the running time of the brute-force approach?*

(c) *(2 points) Suppose we use dynamic programming, which problem that we have solved in the class is the closest to this problem?*

(d) *(6 points) Please describe your subproblems based on your answer in (c)?*

(e) *(6 points) Please explain the decisions you need to make to solve each of your subproblems.*

(f) *(6 points) Please write down the recursion for this problem and the base case(s).*

(g) *(4 points) Please describe the running time of your algorithm in terms of the number of subproblems and the running time per subproblem.*

(h) *(6 points) For Example #2, please describe the value of each of your subproblems that your dynamic programming algorithm (based on previous steps) will compute in each iteration. In how many iterations will your algorithm terminate?*