CS 5720 Design and Analysis of Algorithms
Homework #6.5

Submission requirements:

- Submit your work in PDF format to the appropriate assignment on Canvas.
- **5% extra credit** if your writeup is *typed*.

Assignment: Other canonical DP algorithms

There are many famous problems which can be solved using DP approaches. Here is a small list of some of these:

- Edit Distance
- Rod Cutting
- Dice Throw
- Minimum Partition
- Subset Sum
- Shortest Common Supersequence
- Longest Common Subsequence

Select 3 problems from the above list, and search the internet for a DP-style algorithm which solves each problem. For each of the 3 algorithms you find, your assignment is to:

(a) Write down the algorithm clearly in pseudocode

(b) Identify the "DP Idea" that underlies the algorithm. That is, be clear about exactly how the optimal solution of the problem is composed of (or derived from) optimal solutions of subproblems. One way to think about this is to ask "what information do I need that would make solving this problem really easy?"

(c) For each problem, create an instance of the problem, solve it using the algorithm, and use the optimal solution to illustrate your answer to part (b) *on the last step of the algorithm.*

(See next page for an example!)

Example: Consider the `{0,1}`-Knapsack problem and the algorithm we did in class.

(a) Pseudocode: given in class and book.

(b) The "DP Idea" which underlies this algorithm is based on the fact that an optimal subset is composed of optimal subsets for appropriately-defined subproblems. In particular, we can say that

> The optimal subset for a Knapsack problem with elements $\{1, \ldots, n\}$ and capacity C is either
>
> - The optimal subset for a Knapsack problem with elements $\{1, \ldots, n-1\}$ and capacity C, or
> - The optimal subset for a Knapsack problem with elements $\{1, \ldots, n-1\}$ and capacity $C - w_n$, plus element n.
>
> That is, "if only we knew" the answers to those two subproblems, we'd easily be able to figure out the answer to the main problem.

(c) Consider the problem instance and solution from class. The optimal subset is $\{3, 4\}$, with a total value of 65. The *last step* of the DP algorithm compares the following two scenarios:

- The optimal subset for a Knapsack problem with elements $\{1, 2, 3\}$ and capacity 10. For this subproblem, the optimal subset is $\{1, 2\}$, with total value 54.
- The optimal subset for a Knapsack problem with elements $\{1, 2, 3\}$ and capacity 5, plus element 4. For this subproblem, the optimal subset is to only take $\{3\}$, with value 40. Adding in element 4, we have a total value of 65.

The actual optimal subset for the overall problem is derived from the 2nd scenario: items $\{3, 4\}$ with total value 65.