CS 401: Assignment #3

These are the two problems for Assignment 3.

For regular students, the deadline is **November 24** in class.

For special needs students, the deadline is **December 1** in class.

No late assignments will be accepted.

Special note: Any answer that is not sufficiently clear even after a reasonably careful reading will not be considered a correct answer, and only what is written in the answer will be used to verify accuracy. No vague descriptions or sufficiently ambiguous statements that can be interpreted in multiple ways will be considered as a correct answer, nor will the student be allowed to add any explanations to his/her answer after it has been submitted.

Problem 1 [50 points] Design a dynamic programming algorithm for the following problem:

> *given an amount $n > 0$ and unlimited quantities of each of the coins $d_1, d_2, \ldots, d_m$, find the smallest number of coins that add up to n or indicate that the problem does not have a solution.*

Your algorithm should run in $O(mn)$ time. If you cannot get $O(mn)$ running time, provide the best possible running time you can get.

Problem 2 [50 points] Consider the following **more general version** of the Knapsack problem. There are p **groups of objects** $\mathcal{O}_1, \mathcal{O}_2, \ldots, \mathcal{O}_p$ and a knapsack capacity W. Each object x has a *weight* w_x and a *value* v_x. Our goal is to select a subset of objects such that:

- the total weights of selected objects is at most W,
- **at most one object is selected from any group**, and
- the total value of the selected objects is *maximized*.

Suppose that n is the **total number of objects in all the groups** Give an $O(nW)$ time algorithm for this general Knapsack problem. Explain why your algorithm is correct and analyze the running time of your algorithm.

Hint: Do a dynamic programming with increasing Knapsack capacity.