Problem 1 (Knapsack, Take II, 50 pts)

Alice and Bob are siblings who share a collection of items. There are n items, and each item i has a size s_i, as well as separate valuations for Alice v_i^A and for Bob v_i^B. Each sibling has their own knapsack with a capacity S. Alice and Bob can pick disjoint subsets of items to maximize their combined valuation, with the constraint that the total size of the items they pick does not exceed the capacity of their respective knapsacks. Notice their subsets of items are disjoint because they cannot both pack the same item.

Put formally, we want to maximize $\sum_{i \in I_A} v_i^A + \sum_{i \in I_B} v_i^B$ subject to the constraints $\sum_{i \in I_A} s_i \leq S$ and $\sum_{i \in I_B} s_i \leq S$. $I_A, I_B \subseteq \{1, 2, \ldots, n\}$ represent the subsets of items that Alice and Bob each pick, and we require I_A and I_B to be disjoint ($I_A \cap I_B = \phi$). For simplicity, we only need to output the maximum combined valuation for Alice and Bob, not the subsets.

(a) Find the max combined value Alice and Bob can get for the following input:

- $n = 3$
- Item 1: $s_1 = 2, v_1^A = 3, v_1^B = 5$
- Item 2: $s_2 = 1, v_2^A = 4, v_2^B = 2$
- Item 3: $s_3 = 3, v_3^A = 7, v_3^B = 6$
- Capacity of knapsacks $S = 4$.

(b) Give a DP algorithm to solve this variant of the knapsack problem in $O(nS^2)$. Justify the correctness and runtime of your proposed algorithm.

(c) How would you adapt your algorithm in part (b) if Alice and Bob have different knapsack capacities, S_A and S_B, respectively? And how will the runtime of your algorithm change? You do not need to justify correctness for this part.

Problem 2 (Make Change, 50 points)

Given a set of coin denominations $S = \{s_1, s_2, \ldots, s_n\}$, consider the problem of making change for m cents using the fewest number of coins. Assume that the set of coin denominations consists of only positive integers and that for each denomination you can use an unlimited number of them. For example, if $S = \{1, 5, 10\}$ and $m = 17$, we can make the change of 17 cents by taking one 10, one 5, and two 1's, or by taking seventeen 1's, but the former is preferred, as it uses a minimal number of coins (4 in this case).

(a) Let $S = \{1, 5, 10, 25\}$, and $m = 83$, what is the minimal number of coins to make the change? Also, give the number of coins for each denomination used.

(b) Let $S = \{1, 7, 10, 15\}$, and $m = 36$, what is the minimal number of coins to make the change? Also, give the number of coins for each denomination used.

(c) Give a DP algorithm to solve the make change problem in $O(nm)$ time. Your algorithm only needs to output the minimal number of coins, not the detailed allocation. You may assume the denominations always include 1, so there is always a way to make the change. Justify the correctness and runtime of your proposed algorithm. *(Hint: Let $DP[i, j]$ denote the optimal solution for making a change of j cents using denominations $\{s_1, s_2, \ldots, s_i\}$)*