1. Activity Scheduling Problem: Consider $n$ activities with fixed start times $s_1, \ldots, s_n$ and fixed finish times $f_1, \ldots, f_n$, which must use the same resource (such as lectures in a lecture hall, or jobs on a machine.) At any given time only one activity may take place. Two activities are considered incompatible if they overlap. Your objective is to schedule activities so as to maximize the number of activities which can be completed, while respecting the compatibility constraint. In other words determine a set of mutually compatible activities of maximum size. Consider the following greedy strategies:

   a. Order the activities by increasing total duration. Schedule the activities with the shortest duration first, satisfying the compatibility constraint. If there is a tie, choose the one which starts first.
   b. Order the activities by increasing start time. Schedule the activities with the earliest start times first, subject to the compatibility constraint. If there is a tie, choose the one which is of shortest duration.
   c. Order the activities by increasing finish times. Schedule the activities with the earliest finish times first, subject to the compatibility constraint. If there is a tie, pick one arbitrarily.

Which, if any, of these strategies provides a correct solution to all instances of the problem? If your answer is yes, state and prove a theorem which establishes the correctness of the proposed strategy. If your answer is no, provide a counterexample (i.e. specific start and end times) showing that the strategy can fail, and compute the solution given by the proposed strategy as well as the true optimal solution.

2. Recall the coin changing problem: Given denominations $d = (d_1, d_2, \ldots, d_n)$ and an amount $N$, determine the number of coins in each denomination necessary to disburse $N$ units using the fewest possible coins. Assume that there is an unlimited supply of coins in each denomination.

   a. Write pseudo-code for a greedy algorithm which attempts to solve this problem. (Recall that the greedy strategy doesn’t necessarily produce an optimal solution to this problem. Whether it does or not depends on the denomination set $d$.) Your algorithm will take the array $d$ as input and return an array $g$ as output, where $g[i]$ is the number of coins of type $i$ which are to be disbursed. Assume that the denominations are indexed by increasing value $d_1 < d_2 < \cdots < d_n$, so that your algorithm will step through array $d$ in reverse order. You may also assume that $d_1 = 1$ so that it is possible to pay any amount.
   b. Suppose $d_i = b^{i-1}$ for some integer $b > 1$, and $1 \leq i \leq n$, i.e. $d = (1, b, b^2, \ldots, b^{n-1})$. Does the greedy strategy always produce an optimal solution in this case? Either prove that it does, or give a counter-example.
   c. Suppose $d = (1, 5, 10, 25)$. Prove that the greedy strategy produces an optimal solution for any amount $N$.
   d. Suppose that $d_1 = 1$ and $2d_i \leq d_{i+1}$ for $1 \leq i \leq n-1$. Does the greedy strategy always produce an optimal solution in this case? Either prove that it does, or give a counter-example.