1. (20 Points) **Canoe Rental Problem.** There are $n$ trading posts numbered 1 to $n$ as you travel downstream. At any trading post $i$ you can rent a canoe to be returned at any of the downstream trading posts $j > i$. You are given an array $R[i, j]$ defining the cost of a canoe which is picked up at post $i$ and dropped off at post $j$, for $1 \leq i < j \leq n$. Assume that $R[i, i] = 0$ and that you can’t take a canoe upriver (so perhaps $R[i, j] = \infty$ when $i > j$). Your problem is to determine a sequence of rentals which start at post 1 and end at post $n$, and which has a minimum total cost. As usual there are really two problems: determine the cost of a cheapest sequence, and determine the sequence itself.

Design a dynamic programming algorithm for both problems. What are the dimensions of your table and what are its entries? How is it initialized and what is the recurrence which determines table entries? Why is the principle of optimality satisfied in this problem? Write algorithms which fill the table given the cost array $R$, and which determine an optimal sequence given the filled table. Determine the asymptotic run time of your algorithms.

2. (30 Points) **Activity Scheduling Problem:** Consider $n$ activities with start times $s_1, \ldots, s_n$ and 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 time only one activity can be scheduled. Two activities are 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.

Consider the following greedy strategies:

a. (10 Points) Order the activities in increasing order of 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. (10 Points) Order the activities in increasing order of start time. Schedule the activities with the earliest start times first, satisfying the compatibility constraint. If there is a tie, choose the one which is of shortest duration.

c. (10 Points) Order the activities in increasing order of finish times. Schedule the activities with the earliest finish times first, satisfying the compatibility constraint. If there is a tie, pick one arbitrarily.

Which, if any, of these strategies provide 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. Compute the solution given by the proposed strategy as well as the true optimal solution.
3. (40 Points) 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. (10 Points) 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. (10 Points) 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. (10 Points) Suppose \( d = (1, 5, 10, 25) \). Prove that the greedy strategy produces an optimal solution for any amount \( N \).

d. (10 Points) Suppose that \( d_i = 1 \) 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.