CSE 102
Homework Assignment 6

1. **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, where $j \geq i$. You are given an array $R[i, j]$ defining the cost of a canoe that is picked up at post i and dropped off at post j, for i and j in the range $1 \leq i \leq j \leq n$. 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 canoe rentals that start at post 1, 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 this problem. First, define a 1-dimensional table $C[1 \cdots n]$, where $C[i]$ is the cost of an optimal (i.e. cheapest) sequence of canoe rentals that starting at post 1 and ending at post i. Show that this problem, with subproblems defined in this manner, satisfies the principle of optimality, i.e. state and prove a theorem that establishes the necessary optimal substructure. Second, write a recurrence formula that characterizes $C[i]$ in terms of earlier table entries. Third, write an iterative algorithm that fills in the above table. Fourth, alter your algorithm slightly so as to build a parallel array $P[1 \cdots n]$ such that $P[i]$ is the trading post preceding i along an optimal sequence from 1 to i. In other words, the last canoe to be rented in an optimal sequence from 1 to i was picked up at post $P[i]$. Write a recursive algorithm that, given the filled table P, prints out the optimal sequence itself. Determine the asymptotic runtimes of your algorithms.

2. **Moving on a checkerboard** (This is problem 15-6 on page 368 of the 2nd edition of CLRS.)
   Suppose that you are given an $n \times n$ checkerboard and a single checker. You must move the checker from the bottom (1st) row of the board to the top (n^{th}) row of the board according to the following rule. At each step you may move the checker to one of three squares:

   - the square immediately above,
   - the square one up and one to the left (unless the checker is already in the leftmost column),
   - the square one up and one right (unless the checker is already in the rightmost column).

   Each time you move from square x to square y, you receive $p(x, y)$ dollars. The values $p(x, y)$ are known for all pairs (x, y) for which a move from x to y is legal. Note that $p(x, y)$ may be negative for some (x, y).

   Give an algorithm that determines a set of moves starting at the bottom row, and ending at the top row, and which gathers as many dollars as possible. Your algorithm is free to pick any square along the bottom row as a starting point, and any square along the top row as a destination in order to maximize the amount of money collected. Determine the runtime of your algorithm.

3. (Read the **Rod-Cutting Problem** in 15.1 pp. 360-369 of CLRS 3rd edition. This is 15.1-2 on p. 370.) Show, by means of a counterexample, that the following "greedy" strategy does not always determine an optimal way to cut rods. Define the *density* of a rod of length i to be p_i/i, that is, its value per inch. The greedy strategy for a rod of length n cuts off a first piece of length i, where $1 \leq i \leq n$, having maximum density. It then continues by applying the greedy strategy to the remaining piece of length $n - i$.