Algorithms Assignment 3

This assignment has 5 problems plus a problem for extra credit (no penalty if you do not attempt it, additional bonus marks if you solve it). Some hints and tips:

- Read the assignment early, so you have time to think about the problems!
- Concise answers written in clear English text are perfectly acceptable, as are answers written in *clear* pseudocode.
- If you are unable to provide an algorithm with the required time complexity, part marks may be awarded for providing a **correct** algorithm with a higher time complexity, for as long as the algorithm still runs in polynomial time. Incorrect algorithms that are the required time complexity but far removed from the correct solution will be penalised much more heavily, at the discretion of the marker.

1. There is a row of n items, numbered from 1 to n. Each item has an integer value: item i has value $A[i]$, where $A[1..n]$ is an array. You wish to pick some of the items but you cannot pick two adjacent items (that is, you cannot pick both items i and $i+1$ for *any* i). Subject to this constraint, you wish to maximise the total sum of values of the items you have picked.

   (a) **[2 marks]** Provide a counterexample to show that it is not always optimal to pick the largest item.

   (b) **[3 marks]** Provide a counterexample to show that the optimal solution may require selecting a combination of odd and even numbered elements.

   (c) **[10 marks]** Give an $O(n)$ algorithm that determines this maximum total sum of values.

2. **[15 marks]** A company is organising a retreat for its employees. The organisers of the retreat want to keep costs low, and know the cost of inviting each employee. Note that this may be different for different employees.

   The employees are organised into a strict hierarchy, i.e. a tree rooted at the CEO. There is one restriction, though, on the guest list to the retreat: due to successful union bargaining, it has been established that for every employee x, if x's immediate supervisor (parent in the tree) is *not* attending the retreat, then x *must* attend the retreat. Give an algorithm that finds the minimum total cost of hosting the retreat.

3. **[15 marks]** We say that a sequence of Roman letters A occurs as a subsequence of a sequence of Roman letters B if we can obtain A by deleting some of the symbols of B. Design an algorithm which for every two sequences A and B gives the number of different occurrences of A in B, i.e., the number of ways one can delete some of the symbols of B to get A. For example, the sequence

ba has three occurrences in the sequence baba: **ba**ba, **b**ab**a**, ba**ba**.

4. You are playing a game on an $n \times n$ grid of cells. Each cell contains a single integer: the cell in row r, column c contains the integer $A[r][c]$, where $A[1..n][1..n]$ is a two-dimensional array of integers. You start at the top-left cell $(1, 1)$ and will move to the bottom-right cell (n, n). At each step, you can only move to the cell *immediately* below or to the right. Specifically, from cell (r, c) you can move to $(r + 1, c)$ or $(r, c + 1)$ only. Your score is equal to the sum of the integers among all cells you have passed through: you would like to maximise this score.

   (a) **[10 marks]** Give an $O(n^2)$ algorithm that computes the maximum possible score.

   (b) **[5 marks]** Describe how to extend your algorithm from part (a) to also output any path that produces this maximum possible score. This path should be output as a series of `D` (down) and `R` moves, for instance `DDRDRR` is a valid path on a 4×4 grid. The overall time complexity of the algorithm should still be $O(n^2)$.

   (c) **[5 marks]** Suppose as input you are given the array $A[1..n][1..n]$ in *read-only* memory. This means that you can freely read its values, but cannot modify it in any way.

   Design an algorithm that runs in $O(n^2)$ time that outputs any path with maximum possible score, just as in part (b). However, your algorithm may only use $O(n\sqrt{n})$ additional memory on top of the input array. This additional memory can be freely read from or written to.

5. **[15 marks]** Assume that you are given a network flow graph G with a source s, a sink t and two other distinct vertices u and v. Design an algorithm which returns a smallest capacity cut among all cuts for which the vertex u is in the same side of the cut as the source s and vertex v is in the same side as the sink t and which runs in polynomial time.

6. (EXTRA CREDIT!) At a certain theme park there are n rides, numbered from 1 to n. Before you can go on ride i you must pay its cost $C[i]$ dollars, plus a deposit, $D[i]$ dollars. This means you must have at least $C[i] + D[i]$ dollars to go on the ride. Once you have finished the ride, $D[i]$ dollars are returned to you, as long as you did not damage the ride!

   Careful Cheryl never damages any rides, and currently has T dollars on her. Cheryl would like to know the maximum number of *different* rides she can go on. She can go on rides in *any order* she chooses.

   You may assume $C[1..n]$ and $D[1..n]$ are both positive integer arrays of length n.

(a) **[5 marks]** Prove that Cheryl can go on all the rides if and only if she can go on them in non-increasing order of deposit $D[i]$ (breaking ties arbitrarily).

(b) **[10 marks]** Hence, design an $O(n \log n + nT)$ algorithm to determine the maximum number of different rides Cheryl can go on.

(c) **[2 marks]** Does an $O(n \log n + nT)$ algorithm for this problem run in polynomial time? Why or why not?

(d) **[5 marks]** Design an algorithm for the same task, but which runs in time $O(n^2)$.