Dynamic Programming Problems

1. You plan to attend a college fest that is n days long and has a_i number of activities on the ith day. Attending the fest on one day will leave you tired and unable to attend the next day. However you wish to take part in as many activities as possible. For example, if $n = 5$ and the numbers of activities are 8,3,1,7,4 respectively, you would attend day 1 and day 4 to maximize the number of activities that you can participate in (15 in this case). Suggest an efficient algorithm to solve this problem, i.e. to find the set of days that maximize the number of activities.

2. Given two strings S and T, we say that S is a **supersequence** of T (or equivalently, T is a subsequence of S) if T can be obtained by deleting some letters of S. For example, MONOLITHIC is a supersequence of MOTH.

   Given two strings $A = A_1 \ldots A_m$ and $B = B_1 \ldots B_n$, a common supersequence of A and B is a string that is a supersequence of both A and B. For example, MATCHES is a supersequence of both MATE and ACHES.

   Describe an efficient algorithm to find a **shortest** common supersequence of two given strings.

3. Consider a rectangular grid where some of the cells have some gold coins. An example is shown below, where the number in a cell indicates the number of gold coins it has.

	7		
8			2
		1	
		10	

   The goal is to start at the cell in the bottom-left corner and reach the top-right corner, with each step being to a cell up or right of the current cell, and to collect the maximum number of coins.

   In the above example, the maximum number that can be collected is 15, while a greedy choice (eg: navigating to the cell with 10 first) may not be optimal.

Describe an efficient algorithm to find a route to collect the maximum number of coins. The input is given as a matrix M, where $M_{i,j}$ denotes the number of coins in cell (i, j).

4. India and Australia are playing a series of n one-day cricket matches against each other across different venues. Their chances of winning a particular match depends on the venue, and we suppose that the probability of India winning the ith match is p_i, with Australia's probability of winning being $(1 - p_i)$; there are no draws/ties.

   Given that n is odd, describe an efficient algorithm that accepts the probability values $p_1, \ldots, p_n$ and finds the probablity that India wins the series, i.e. scores mores than $n/2$ wins.