I500/B609: Fundamental Computer Concepts of Informatics
Discussion problems (week 6, Dynamic programming algorithms)

1. Develop dynamic programming algorithm for the maximum-subarray problem, i.e., finding the contiguous subarray in a given array of numbers with the maximum sum.

2. (Two eggs problem) You are given two eggs, and access to a 100-storey building. Both eggs are identical. The aim is to find out the highest floor from which an egg will not break when dropped out of a window from that floor. If an egg is dropped and does not break, it is undamaged and can be dropped again. However, once an egg is broken, it cannot be dropped again. If an egg breaks when dropped from floor *n*, then it would also have broken from any floor above that. If an egg survives a fall, then it will survive any fall shorter than that. The question is: What strategy should you adopt to minimize the number egg drops it takes to find the solution (and guarantees to find it), and what is the worst case for the number of drops it will take?

3. (Scheduling to maximize profit, 20pts) Suppose you have a machine and a set of n jobs, a_1, a_2, …, a_n to process on that machine. Each job a_j has a processing time t_j, a profit p_j and a deadline d_j. The machine can process only one job at a time, and job a_j must run uninterruptedly for t_j consecutive time units. If job a_j is completely by its deadline d_j, you receive a profit p_j; but if it is completed after its deadline, you receive a profit of 0. Give an algorithm to find the schedule that obtains the maximum amount of profit, assuming that all processing times are integers between 1 and n. What is the running time of your algorithm?
4. (Balanced partition problem) Suppose you are given an array of n integers {a_1, a_2, …, a_n} between 0 and M. Give an algorithm for dividing these integers into two sets x and y such that he difference of the sum of the integers in each set is minimized. For example, given the set {2, 3, 2, 7, 9}, you can divide it into {2, 2, 7} (sums to 11) and {3, 9} (sums to 12) for a difference of 1.
5. You are given an ordered sequence of n cities, and the distances between every pair of cities. Design an algorithm to partition the cities into two subsequences (not necessarily contiguous) such that person A visits all cities in the first subsequence (in order), person B visits all cities in the second subsequence (in order), and the sum of the total distances travelled by A and B is minimized. Assume that person A and person B start initially at the first city in their respective subsequences.
6. You are given a string of n characters s[1..n], which you believe to be a corrupted text document in which all punctuation has vanished (so that it looks something like "itwasthebestoftimes…"). You wish to reconstruct the document using a dictionary, which is available in the form of a Boolean function dict(), for any string w, dict(w)= true, if w is a valid word, and false if it is not. Devise a dynamic programming algorithm that determines whether the string s can be

reconstituted as a sequence of valid words. The running time should be at most O(n^2), assuming calls to dict() take only constant time. In the event that a string is valid, your algorithm should output the corresponding sequence of words.