CS524
HW#5

DUE: Monday, November 27

1. (6 points) Consider the array $V[0..n, 0..W]$ computed by the dynamic programming algorithm to solve the 0/1-KNAPSACK PROBLEM. Either prove or give a counter example to each of the following.

<u>CONJECTURE 1</u>: For any instance of the problem and any j, $0 \le j \le n$, and any x,y, $0 \le x < y \le W$, $V[j, x] \le V[j, y]$.

<u>CONJECTURE 2</u>: For any instance of the problem and any j, k, $0 \le j < k \le n$, and any x, $0 \le x \le W$, $V[j, x] \le V[k, x]$.

2. (6 points) Show how to solve the 0/1-KNAPSACK PROBLEM and return both the value of an optimal solution **and** the number of optimal solutions.

3. (6 points) A typical dynamic programming algorithm provides the cost of a solution or establishes the existence of a solution without actually constructing the solution. To see how to construct a solution by using an efficient mechanism which tests for the existence of a solution, solve the following:

You are given a boolean function *BlackBox* of two inputs:

-a list of integers $x_1, ..., x_n$,

-an integer q,

and you are told that, in time $O(1)$, *BlackBox* will return true if there is some subset of $x_1, ..., x_n$ whose sum is q and false otherwise. Design an algorithm (a program is not needed) with the same input which will return an actual subset of $x_1, ..., x_n$ whose sum is q, if such a subset exists, or else it should return "failure".

For example, *BlackBox*((23,27,41,72,-4,6), 29) would return "true", but your algorithm with the same input would return (27,-4,6) or (23,6). Your algorithm may call *BlackBox* as often as it wishes and it should work in time $O(n)$.