**Dynamic programming Problems:**

1. [Maximum Value Contiguous Subsequence](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_1.swf). Given a sequence of n real numbers A(1) ... A(n), determine a contiguous subsequence A(i) ... A(j) for which the sum of elements in the subsequence is maximized.
2. [Making Change](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_2.swf). You are given n types of coin denominations of values v(1) < v(2) < ... < v(n) (all integers). Assume v(1) = 1, so you can always make change for any amount of money C. Give an algorithm which makes change for an amount of money C with as few coins as possible. [on problem set 4]
3. [Longest Increasing Subsequence](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_3.swf). Given a sequence of n real numbers A(1) ... A(n), determine a subsequence (not necessarily contiguous) of maximum length in which the values in the subsequence form a strictly increasing sequence. [on problem set 4]
4. [Box Stacking](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_5.swf). You are given a set of n types of rectangular 3-D boxes, where the i^th box has height h(i), width w(i) and depth d(i) (all real numbers). You want to create a stack of boxes which is as tall as possible, but you can only stack a box on top of another box if the dimensions of the 2-D base of the lower box are each strictly larger than those of the 2-D base of the higher box. Of course, you can rotate a box so that any side functions as its base. It is also allowable to use multiple instances of the same type of box.
5. [Building Bridges](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_6.swf). Consider a 2-D map with a horizontal river passing through its center. There are n cities on the southern bank with x-coordinates a(1) ... a(n) and n cities on the northern bank with x-coordinates b(1) ... b(n). You want to connect as many north-south pairs of cities as possible with bridges such that no two bridges cross. When connecting cities, you can only connect city i on the northern bank to city i on the southern bank. (Note: this problem was incorrectly stated on the paper copies of the handout given in recitation.)
6. [Integer Knapsack Problem (Duplicate Items Forbidden)](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_7.swf). This is the same problem as the example above, except here it is forbidden to use more than one instance of each type of item.
7. [Balanced Partition](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_4.swf). You have a set of n integers each in the range 0 ... K. Partition these integers into two subsets such that you minimize |S1 - S2|, where S1 and S2 denote the sums of the elements in each of the two subsets.
8. [Edit Distance](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_8.swf). Given two text strings A of length n and B of length m, you want to transform A into B with a minimum number of operations of the following types: delete a character from A, insert a character into A, or change some character in A into a new character. The minimal number of such operations required to transform A into B is called the edit distance between A and B.
9. [Counting Boolean Parenthesizations](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_9.swf). You are given a boolean expression consisting of a string of the symbols 'true', 'false', 'and', 'or', and 'xor'. Count the number of ways to parenthesize the expression such that it will evaluate to true. For example, there are 2 ways to parenthesize 'true and false xor true' such that it evaluates to true.
10. [Optimal Strategy for a Game](http://people.cs.clemson.edu/%7Ebcdean/dp_practice/dp_10.swf). Consider a row of n coins of values v(1) ... v(n), where n is even. We play a game against an opponent by alternating turns. In each turn, a player selects either the first or last coin from the row, removes it from the row permanently, and receives the value of the coin. Determine the maximum possible amount of money we can definitely win if we move first.
11. Given a matrix consisting of 0's and 1's, find the maximum size sub-matrix consisting of only 1's.
12. Coin Change - Given a value N, if we want to make change for N cents, and we have infinite supply of each of S = { S1, S2, .. , Sm} valued coins, how many ways can we make the change?
13. Longest Palindromic Subsequence - The question is same as above but the subsequence should be palindromic as well.
14. Minimum Number of Jumps - Given an array of integers where each element represents the maximum number of steps that can be made forward from that element, find the minimum number of jumps to reach the end of the array (starting from the first element).