**Algorithmic Thinking in Problem Solving**

**Dynamic Programming**

1. Watch the following video on Dynamic Programming

<https://www.youtube.com/watch?v=7dPnb_lJywc>

1. In your own words, explain what Dynamic Programming is and why it is such an important algorithm design technique.
2. Solve the following coding problem

Write a method that, given a binary tree as input, finds the maximum sum you can compute by adding all the values from the root to one of the leaves. Note: Unlike a regular binary tree, for this problem, assume that nodes in the tree SHARE children (see example). Feel free to write helper methods. You have to use dynamic programming to get credit.

Example:  
   
       3  
     /   \  
    10    6  
   /  \  /  \  
  -5  -70   -4  
 /  \ /  \ /  \  
-1   2   50   -8  
   
Output: 3 + 6 + (-4) + 50 = 55

class Node:

def \_\_init\_\_(self, left=None, right=None, item=None):

self.left = left

self.right = right

self.item = item

def max\_sum(root):

return 0

Test your implementation by calling it multiple times with different inputs and comparing the output produced by your method and the expected output. For each test, add a short comment explaining why you think that test is appropriate. Do not write an excessive amount of tests; just write the number of tests you think you need and justify your decisions.

Spend at most 1 hour trying to solve this problem. If you are unable to solve it after 1 hour, Google the problem and find a solution, then do the following:

⦁ Trace the solution using a concrete instance of the problem (include your trace in your submission).

⦁ What prevented you from solving the problem?

⦁ What did you learn? Did you have to Google a little more to understand the solution?

⦁ What would you do differently in the future if you were presented with a similar problem?