Design and Analysis of Algorithm (CS 412)
Instructor: Dr. Ayesha Enayet
Date: _____
CS 6th

SIS ID:_____ Name:_____

Instructions: Attempt all the questions. Use of device(s) is not allowed. Use a Blue/Black pen.

A. Longest Common Substring: Given two strings X and Y design a dynamic programming solution to find the length of longest common substring. Note that this is not the same as the Longest Common Subsequence problem, in which characters are not necessarily contiguous. For example, in "week" and "weakest" the length of the longest common substring is 2. [2]

	j	T	O	O	F	O	O	D	I	E
i	0	0	0	0	0	0	0	0	0	0
T	0	1	0	0	0	0	0	0	0	0
O	0	0	2	1	0	1	1	0	0	0
O	0	0	1	3	0	1	2	0	0	0
D	0	0	0	0	0	0	0	3	0	0
Y	0	0	0	0	0	0	0	0	0	0

Formulate the dynamic programming recurrence for the given problem:

$$M[i,j] = \{\ \text{M[i-1][j-1]+1}, \quad \text{X[i]=Y[j]} \quad \}$$

Hint: for X[i]≠Y[j], M[i,j]=0

B. [True/False] The shortest path problem can be solved using both Dynamic programming and greedy algorithms. [1]
True

C. Write down an algorithm for 0/1 knapsack problem and identify its worst-case time complexity. [2]

Input: Integer W, arrays w[1..n], v[1..n]

Output: Maximum value that can be achieved without exceeding capacity W

Initialize dp[0..n][0..W] = 0

for i from 1 to n:

for w from 0 to W:

if w_i > w:

```
            dp[i][w] = dp[i-1][w]              // can't include item i
        else:
            dp[i][w] = max(
                dp[i-1][w],                    // don't take item i
                dp[i-1][w - w_i] + v_i          // take item i
            )
return dp[n][W]
```