DSC 190 - Homework 05

Due: Thursday, February 18

Write your solutions to the following problems by either typing them up or handwriting them on another piece of paper. Unless otherwise noted by the problem's instructions, show your work or provide some justification for your answer. Homeworks are due via Gradescope on Thursday at 11:59 p.m.

Problem 1.

Consider the following set of activities.

Activity #	Start	Finish	Weight
0	0	3	12
1	1	2.5	10
2	2	3	22
3	2.75	4.25	14
4	4	8	33
5	4.5	11.5	11
6	5	6	12
7	5.5	6.5	7
8	8	10	5
9	9	12	19

a) Suppose the dynamic programming solution to the weighted activity selection problem is run on this data, and let `cache` be an array storing the memoized solution to each subproblem. In particular, `cache[i]` is the weight of the max weight schedule considering only the activities $i, i+1, \ldots, 9$. Note that we will consider two activities `x` and `y` to be compatible if `y.start` $\geq$ `x.finish` or `x.start` $\geq$ `y.finish`.

What are the entries of `cache`? You do not need to include the "dummy" entry of zero at the end of the array.

b) What is the weight of the optimal solution to the weighted activity selection problem for this data?

Problem 2.

In lecture, we designed a top-down dynamic programming solution for the longest common subsequence problem. In this problem, you will create a bottom-up iterative solution.

In a file named `lcs_bup.py` create a function named `lcs_bup(a, b)` which accepts two strings, a and b, and returns the length of a longest common subsequence of a and b. Your code should take a bottom-up dynamic programming approach to solving the problem. Namely, it should be iterative and have $\Theta(mn)$ time complexity, where m and n are the lengths of a and b, respectively.

Hint: see the last discussion, as well as the lecture on the weighted scheduling problem, for strategies on converting a top-down DP solution to a bottom-up algorithm.