CSCI 3104-Spring 2016: Assignment #6.

Assigned date: Wednesday, 3/9/2016,
Due date: Tuesday, 3/29/2016, before class
Maximum Points: 40 points (includes 5 points for legibility).

Note: This assignment *must be turned in on paper, before class.* Please do not email: it is very hard for us to keep track of email submissions. Further instructions are on the moodle page.

P1 (25 points) Write a dynamic programming algorithm to solve the longest increasing subsequence (LIS) problem. You are given an array a of n integers. An *increasing subsequence* is a sequence k indices

$$0 \leq i_1 < i_2 < \cdots < i_k \leq (n-1)$$

such that

$$a[i_1] < a[i_2] < \cdots < a[i_k]$$

Given a find the *longest increasing subsequence.* I.e, an increasing subsequence $i_1, \ldots, i_k$ whose length k is the longest possible.

Example: Input is $a : [1,5,2,3,8]$. The longest increasing subsequence has length 4. $i_1 = 0,\ i_2 = 2,\ i_3 = 3,\ i_4 = 4$ yielding the elements $1,2,3,8$.

Let us define the function $\textsc{lis}(a, j, M)$ as the length of the LIS for the first j elements of the array $(a[0] \cdots a[j-1])$, where j ranges from 0 to n and all elements of this subsequence are less than M.

(A) For the array $a : [1,5,2,3,8]$ write down the values of $\textsc{lis}(a,3,2)$ and $\textsc{lis}(a,5,\infty)$.

(B) Write down the base cases for $\textsc{lis}(a,0,M)$.

(C) Write down a recurrence for $\textsc{lis}(a,j,M)$ for any j with $1 \leq j \leq n$.

(D) Describe a bottom-up scheme to memoize the recurrence $\textsc{lis}$ above.

Describe the memo table itself: how many rows? how many columns? what do the rows and columns stand for? How are the base cases filled out in the table? How is each entry of the memo table filled out?

Also, show how the filled out memo table for the example array $a : [1,5,2,3,8]$.

(E, Extra Credit) Program your dynamic programming scheme in Python. Input to your function is the list a and output should be the LIS itself.

Solution. (A) $\textsc{lis}(a,3,2) = 1$ and $\textsc{lis}(a,5,\infty) = 4$.

(B) We have base case:

$$\textsc{lis}(a,0,M) = 0$$

(C) The recurrence for $j \geq 2$ is

$$\textsc{lis}(a,j,M) = \max \begin{cases} \textsc{lis}(a,j-1,M) \\ 1 + \textsc{lis}(a,j-1,a[j-1]) & \text{If } a[j-1] < M \end{cases}$$

(D) For convenience, let us assume that the array $a[n] = \infty$. Noting that the argument M of $\textsc{lis}(a, j, M)$ is always some $a[k]$ where $j \leq k$, we write the recurrence instead as $\textsc{lis}(a, j, k)$, where $M = a[k]$. Also, we take $a[n] = \infty$ and allow k to range from 0 to n.

	0	1	2	$\cdots$	n
$a[1]$	0				
$a[2]$	0				
$\vdots$	$\vdots$	$\vdots$	$\ddots$		
$a[n](= \infty)$	0				

The columns indicate the value of j and range from $j = 0$ to n, the first argument of the recursive call.

The rows indicate the value of k range from 1 to n.

The result is at the bottom right corner of the table $T[n, n]$.

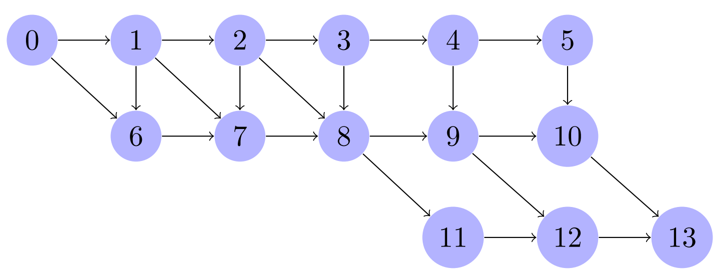
Following the recurrence, each entry $T[j, k]$ is filled out using one of two cases: (a) If $a[j-1] < M$, then compute maximum of the $T[j-1, k]$ and $1 + T[j-1, j-1]$. (b) If $a[j-1] \geq M$ then $T[j, k] = T[j-1, k]$.

To reconstruct the maximum, we additionally add an annotation to each entry $T[j, k]$ of the table: We annotate through $\leftarrow$ if the value of $T[j-1, k]$ equals that of $T[j, k]$, and use $\nwarrow$ if the entry $T[j, k]$ equals $1 + T[j-1, j-1]$.

Start from $T[n, n]$. If for each entry $T[j, k]$ we visit the annotation is $\leftarrow$, we omit $a[j-1]$ and move to $T[j-1, k]$. Otherwise, if it is annotated $\nwarrow$, we add $a[j-1]$ to the LIS and move to the entry $T[j-1, j-1]$.

(E) Attached file lis.py.

P2 (15 points) We are given a network below. Calculate the number of paths from node 0 to node 13.



Solution. We can clearly see that number of paths from a node n to 13 is simply the sum of the number of paths from each of its successors to 13. The base case is 13 itself has 1 path to 13. We fill out the following memo table.

Node	# Paths to 13
13	1
12	1
11	1
10	1
9	2
8	3
7	3
6	3
5	1
4	3
3	6
2	12
1	18
0	21

We conclude 21 paths.