CSC 321/Design and Analysis of Algorithms Assignment 4

Introduction

Please complete each problem below. A problem asking you to implement an algorithm means that you should write it as a program in either Java, version 8 or later, or Python 3.7 or later.

Problems [60 points]

Problem A. [20 points] I presented the pseudocode for finding the length of the longest common subsequence in class on Wednesday, 10/26. As requested during the class, please implement this algorithm according to the requirements mentioned above. Also, your program must at least consist of a method implementing what's given in the pseudocode AND a main method or section that:

- Declares and initializes two strings to "GTTAATGTAGCTTAATAACAAAGCAAAGCACTGAAAATGCTTAGATGGATAATTGTATCCCATAA" and "GTTAATGTAGCTTAATTACAAAGCAAGGCACTGAAAATGCCTAGATGAGTACGCGCTACTCCATA";
- Calls the LCS method and stores the return value (an integer);
- Prints the strings and the length of the LCS returned.

Call the program LCS.java or lcs.py. Write the program so that, when run, it prints the above output. In other words, it should not require any action other than running it. It should not request input from the user.

If you want to test your program on other, perhaps shorter, strings, that's fine just be sure that the version you submit uses the strings mentioned above.

Here's the pseudocode in case you didn't write it down.

$LCS(x, y)$
  $m = x.size$
  $n = y.size$
  $c = new\ m \times n\ table$
  $for\ i = 0\ to\ m$
    $c[i, 0] = 0$
  $for\ j = 0\ to\ n$
    $c[0, j] = 0$
  $for\ i = 1\ to\ m$
    $for\ j = 1\ to\ n$
      $if\ x[i] = y[j]\ then$
        $c[i, j] = c[i - 1. j - 1] + 1$
      $else$
        $c[i, j] = \max\{c[i - 1, j], c[i, j - 1]\}$
  $return\ c[m, n]$

Problem B. [20 points] Also on this past Wednesday, I presented the edit distance problem from the textbook (section 3.7). While discussing it, I put on the board a table showing how to compute the minimum edit distance between the strings "LOYOYA" and "DEPAUL". I filled in some of the entries of that table. For this problem, please write or draw the table with all of the entries filled. Use my lecture and the book to determine how to do this.

The arrows I drew are not necessary in your answer. You only need to fill in the integers in the table. But feel free to add them if you wish.

Problem C. [20 points] Finally, at the end of Wednesday's lecture, I presented the maximum profit path problem (MPP). I provided a 5×5 table p with profits filled in. Recall that the problem is to find the path from the top row (row 1) to the bottom row (row 5) that has the maximum total profit. A path's profit is the sum of the profits in each of its square. A path must start in some square in row 1 and end in row 5. A path currently at square $p[i,j]$ must go to one of the squares $p[i+1,j-1]$, $p[i+1,j]$, or $p[i+1,j+1]$ and it cannot go off the left or right side of the table.

Using the p table I wrote, you are to write or draw a table m where an entry $m[i,j]$ is the maximum profit for every path that comes to square $p[i,j]$. To compute the values for the m table, use this recurrence:

$$m[i,j] = \begin{cases} p[i,j] & if\ i = 1 \\ \max\begin{Bmatrix} m[i-1,j-1] \\ m[i-1,j] \\ m[i-1,j+1] \end{Bmatrix} + p[i,j] & otherwise \end{cases}$$

So the square at i,j has a value computed by adding $p[i,j]$ to the maximum value among:

- the square one above and to the left ($m[i-1,j-1]$)
- the square directly above ($m[i-1,j]$)
- the square one above and to the right ($m[i-1,j+1]$)

If there is no square because one of the indices is less than 1 or greater than 5, the value used is 0.

Submission

Submit the program for Problem A into its slot. Place the tables from Problems B and C in a PDF or Word document called Assignment4.pdf or Assignment4.docx and submit it into its slot.