**Fall 2019**

**CSE-5311-004 – Design and Analysis of Algorithms**

**Programming Assignment 1 - Report**

|  |  |
| --- | --- |
| **Name** | **UTA ID** |
| Goutami Padmanabhan | 1001669338 |

**Java Version**: 1.8

**Packages/Libraries:**

import java.io.\*;

import java.util.LinkedList;

import java.util.List;

**Run Instructions:**

Program was run using IntelliJ IDE Ultimate edition (you may use any other IDE) using Java version 1.8.0

It can also be run from command prompt using the following commands one after another in the same order.

***Note****: Kindly replace the path below to your respective input path.*

Run Command Prompt

cd \mywork

dir

set path=%path%;C:\Program Files\Java\jdk1.8.0\_181\bin

javac LongestCommonSubsequence.java

dir

java LongestCommonSubsequence \C:\mywork\input.txt

**Input:**

AGGTAB

GXTXAYB

ABCDAF

ACBCF

**Expected Output:**

Longest Common Subsequence of AGGTAB and GXTXAYB is

GTAB

Length of Longest Common Subsequence is 4

Longest Common Subsequence of ABCDAF and ACBCF is

ABCF

Length of Longest Common Subsequence is 4

**Complexity:**

Time Complexity: O(n2)

Space Complexity: O(n2)

**Description:**

The program submitted follows the Dynamic programming Bottom-up approach.

**Main Method:**

The Class *LongestCommonSubsequence* has a main method which creates a list of input strings *inputStringList* by reading each and every line of the input file. The input file is read and parsed using the method *fileReader*.

For each string in the *inputStringList*, the strings s1 and s2 are assigned. These two strings are made into One-dimensional Character arrays X and Y for s1 and s2 respectively.

A two-dimensional character array B is created and obtains the return value of the method *longestCommonSubseq*. The character arrays X and Y are passed to the method *longestCommonSubseq* and the Longest Common Subsequence is displayed using *printLcs* method. The array B, array X, Length of array X and Length of array Y is passed to the method *printLcs*.

The method *lcsLength* is called to determine the length of Longest Common Subsequence found. The strings s1 and s2 are passed to the method *lcsLength* to obtain this. The length is later displayed.

**fileReader Method:**

Using the BufferedReader class, we read the input file contents line by line and store it in the List inputStringList in our program and it is returned to the main method.

**longestCommonSubseq Method:**

For each string in the List *inputStringList* taking two at a time, the Longest Common Subsequence is found. The first row and the first column of C[i][j] is set to 0 as there will be no matching sequence present. Later the remaining array is built based on matching the sequences of the two strings. It stores the C[i][j] values in a table C[0...m][0...n] and it computes the entries in row-major order. (That is, the method fills in the first row of C from left to right, then the second row, and so on.) The method also maintains the table B[1…m][1…n] to help us construct an optimal solution. B[i][j] points to the table entry corresponding to the optimal subproblem solution chosen when computing C[i][j]. The method returns the B table. The running time of this method is θ(mn).

The B table returned by *longestCommonSubseq* enables us to construct an LCS of strings s1 and s2. We begin at B[m][n] and trace through the table by following the symbols (\\,|,-). Whenever we encounter a “\\” in entry B[i][j], it implies that X[i] = Y[j] is an element of the LCS that *longestCommonSubseq* found. With this method, we encounter the elements of this LCS in reverse order.

**printLcs Method:**

This method prints the Longest Common Subsequence in the proper forward order. The call has parameters B, X, X.length, Y.length that is being passed to this method. The running time for this method is O(m+n).

**lcsLength Method:**

This method constructs the array by determining the maximum of the diagonal and adjacent previous index. Every time, there is a match in the two sequences, the value is added by 1 and written in that index. This method thus determines the length of the subsequence by the last element in the two-dimensional array. The running time for this method is O(n2).

Finally, the Longest Common subsequence is displayed, and the Length of the subsequence found is also printed.