Bhavyansh Mishra

Data Structures and Algorithms II

Project 5

Functional Decomposition

Longest Common Subsequence

*(Using Dynamic Programming)*

**Part 1**

main.c

Displays the instructions of the program and outputs the results of the Longest Common Subsequence algorithms for both Part 1 and Part 2.

lcs.c

**int lcs(char\* X, char\* Y, char\* LCS);**

The method that performs the full 2D array top-down Dynamic Programming algorithm to find the Longest Common Subsequence for the strings X and Y.

**int opt\_lcs(char\* X, char\* Y);**

The memory optimized Dynamic Programming algorithm which uses O(2n) memory to compute the length of the Longest Common Subsequence for the strings X and Y

**void construct(int dp[rows][cols], int i, int j);**

The method that uses the ‘c’ array to recursively generate the Longest Common Subsequence of two strings given the 2D array containing the lengths of the intermediate LCSs. This method is called by the lcs(X,Y,LCS) method as a routine.

**Part 2**

similarity.c

**char sim(int lcs, int a, int b);**

This method calculates the measure of similarity between two given string lengths and the length of their LCS, and returns the character representing the degree of similarity.

**void getString(int n, char\* name, int\* starts, char\* str)**

This is a tokenizing function which reads the file ‘name’ and finds the string in the nth line of the file, with the help of the OFFSET value for that string given in the ‘starts’ array of offsets, and places the string in the ‘str’ variable.

**void display(int n, FILE\* fp, int\* starts);**

This method reads a file containing a number n, and n subsequent strings of varying lengths, calculates the length of the LCS of each of the several unique pairings of the strings, and outputs the degree of their similarity in form of a table containing H, M, L and D characters in the upper triangle.

**void print(int\* a, int n);**

A helpful routine used to print the contents of the array ‘a’ of size ‘n’ in a linear format.

**void startings(int n, FILE\* fp, int\* s);**

This method finds the OFFSET values for each of the strings present in the file “multipleSequences.txt” and returns these offsets in form of the array ‘s’ to the calling function. This method is required for the display method which needs the OFFSET values in order to read the strings from the file properly.

This method requires also uses the #define NL 1 macro to use a total of 1 byte currently to represent the size of a newline character. This #define NL attribute can be changed to 2 for running the program with a ‘multipleSequences.txt’ file created on a Windows environment.