**Time Complexity of Pseudo Code**

Pseudo code

// In the main function we get text files from user. If there are more than two files, we compare first file with second and their result with third so on.

//Function to convert text file to string

Convert\_to\_String(File)

j=File.length **O(1)**

Let A[0….j] be a string **O(1)**

for i=0 to j **O(n) (n=j)**

A[i]= character of file **O(n)**

return A **O(1)**

**Total Complexity: O(n)**

//Function to clear sensitivity of string

Clear\_Sensitivity(A)

for i =0 to A.length **O(n)**

if A[i] >= 65 and A[i] <= 92 **O(nx1)**

A[i] = A[i] +32 **O(nx1)**

return A **O(1)**

**Total Complexity: O(3n+1) = O(n)**

//Function to remove Spaces

Remove\_Spaces(A)

i = 1 **O(1)**

size = A.length **O(1)**

while i <= A.length **O(n+1)**

if A[i] = “ ” **O(nx1)**

A[i] = A[i+1] **O(nx1)**

size = size- 1 **O(nx1)**

return A, size **O(1)**

**Total Complexity: O(4n+4) = O(n)**

// Function to find common characters between two strings

LCS\_length( X, Y)

m = X.length **O(1)**

n = Y.length **O(1)**

let b[1…m,1…n] and c[0…m,0...n] be new tables **O(1)**

for i = 1 to m **O(m+1)**

c[i,0] = 0 **O(m)**

for j = 0 to n **O(n)**

c[0][j] = 0 **O(n)**

for i = 1 to m **O(m+1)**

for j = 1 to n **O((m+1)(n+1))**

if x[ i] == y[ j] **O(m+1)n**

c[i][j] = c[i - 1][j - 1] + 1 **O(m+1)n**

b[i][j] = '/' **O(m+1)n**

else if c[i - 1][j] >= c[i][j - 1] **O(m+1)n**

c[i][j] = c[i - 1][j] **O(m+1)n**

b[i][j] = '|' **O(m+1)n**

else c[i - 1][j] = c[i][j - 1] **O(m+1)n**

c[i][j] = c[i][j - 1] **O(m+1)n**

b[i][j] = '-' **O(m+1)n**

return c and b **O(1)**

**Total Complexity : 9mn+4m+11n+6 = O(mn)**

//Getting common characters and their index

Print\_LCS( b, X, i, j)

int q=0

let D[0…i] be the array to keep index

if i == 0 or j == 0

return

if b[i][j] == '/'

Print\_LCS(b, x, i - 1, j - 1)

print x[i]

D[q] = i

q = q+1

return D

else if b[i][j] == '|'

Print\_LCS(b, x, i - 1, j)

else Print\_LCS(b, x, i, j - 1)

**Total Complexity**: **O(m+n)**

// Getting percentage of Plagiarism

Plagiarism\_Percentage (D,A)

// A is given file here

i=A.length **O(1)**

x=D.length **O(1)**

p=x/i \*100 **O(1)**

print p **O(1)**

**Total Complexity: O(1)**

**Complexity of all functions:**

Total Complexity = O(n) + O(n)+ O(n) + O(mn) + O(m+n) + O(1)

= O(mn)