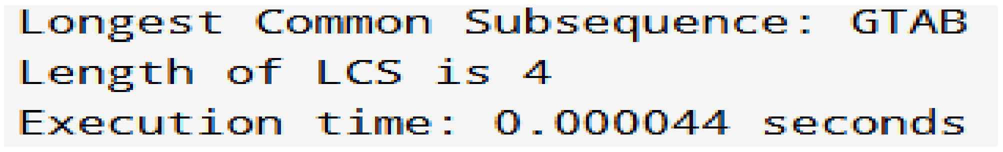
Description:

C program that solves the Longest Common Subsequence problem, and reports execution time using clock().

Source Code:

```
#include <stdio.h>

#include <string.h>

#include<time.h>

// Function to find the maximum of two integers

int max(int a, int b) {

 return (a > b) ? a : b;

}

// Function to find the Longest Common Subsequence (LCS)

int lcs(char *X, char *Y, int m, int n) {

 int L[m + 1][n + 1]; // DP table to store lengths of LCS

 // Fill the L table in bottom-up manner

 for (int i = 0; i <= m; i++) {

 for (int j = 0; j <= n; j++) {

 if (i == 0 || j == 0) {

 L[i][j] = 0; // Base case: if either string is empty, LCS is 0

 } else if (X[i - 1] == Y[j - 1]) {

 L[i][j] = L[i - 1][j - 1] + 1; // Characters match, add 1 to diagonal

 } else {

 L[i][j] = max(L[i - 1][j], L[i][j - 1]); // Characters don't match, take max of above or left

 }

 }
 }
 // L[m][n] contains the length of LCS. Now, reconstruct the LCS string.
 int index = L[m][n];
 char lcs_str[index + 1];
 lcs_str[index] = '\0'; // Null-terminate the string
 int i = m, j = n;
 while (i > 0 && j > 0) {
 if (X[i - 1] == Y[j - 1]) {
 lcs_str[index - 1] = X[i - 1]; // Character is part of LCS
 i--;
 j--;
 index--;
 } else if (L[i - 1][j] > L[i][j - 1]) {
 i--; // Move up
 } else {
 j--; // Move left
 }
 }
 printf("Longest Common Subsequence: %s\n", lcs_str);
 return L[m][n]; // Return the length of LCS
}
int main() {
clock_t start, end;
double cpu_time_used;
 char X[] = "AGGTAB";
 char Y[] = "GXTXAYB";
 int m = strlen(X);
 int n = strlen(Y);
 start=clock();
 printf("Length of LCS is %d\n", lcs(X, Y, m, n));
 end = clock();
cpu_time_used = ((double)(end - start) / CLOCKS_PER_SEC);
printf("Execution time: %f seconds\n", cpu_time_used);
return 0;
}
```

Output:



```
Longest Common Subsequence: GTAB
Length of LCS is 4
Execution time: 0.000044 seconds
```