Vashisth Zatakia
60003200163 B3

Aim: Implementation of Longest common subsequence.

Theory:

Longest Common Subsequence

Here longest means that the subsequence should be the biggest one. The common means that some of the characters are common between the two strings. The subsequence means that some of the characters are taken from the string that is written in increasing order to form a subsequence.

Let's understand the subsequence through an example.

Consider two strings:

X= a b a a b a

Y= b a b b a b

The following steps are followed for finding the longest common subsequence.

1. Create a table of dimension n+1*m+1 where n and m are the lengths of X and Y respectively. The first row and the first column are filled with zeros.

	^	b	a	b	b	a	b
^	0	0	0	0	0	0	0
a	0	0					
b	0						
a	0						
a	0						
b	0						
a	0						

2. Fill each cell of the table using the following logic.
3. If the character corresponding to the current row and current column are matching, then fill the current cell by adding one to the diagonal element. Point an arrow to the diagonal cell.
4. Else take the maximum value from the previous column and previous row element for filling the current cell. Point an arrow to the cell with maximum value. If they are equal, point to any of them.

Vashisth Zatakia
60003200163 B3

The value in the last row and the last column is the length of the longest common subsequence.

	^	b	a	b	b	a	b
^	0	0	0	0	0	0	0
a	0	0	1	1	1	1	1
b	0	1	1	2	2	2	2
a	0	1	2	2	2	3	3
a	0	1	2	2	2	3	3
b	0	1	1	3	3	3	4
a	0	1	2	2	2	4	4

In order to find the longest common subsequence, start from the last element and follow the direction of the arrow.

Complexity Analysis:

If we use the dynamic programming approach, then the number of function calls are reduced. The dynamic programming approach stores the result of each function call so that the result of function calls can be used in the future function calls without the need of calling the functions again.

In the above dynamic algorithm, the results obtained from the comparison between the elements of x and the elements of y are stored in the table so that the results can be stored for the future computations. The time taken by the dynamic programming approach to complete a table is **O(mn),** since we are using two for loops for both the strings ,therefore the time complexity of finding the longest common subsequence using dynamic programming approach is O(n * m) where n and m are the lengths of the strings and the time taken by the recursive algorithm is $2^{max(m, n)}$.

Source Code:

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int max(int num1, int num2)

{

   return (num1 > num2) ? num1 : num2;

}
```

Vashisth Zatakia
60003200163 B3

```
void lcs(int len1, int len2, char *str1, char *str2)
{
   int LCS[len1 + 1][len2 + 1];
   int k = 0;
   for (int i = 0; i <= len1; i++)
   {
      for (int j = 0; j <= len2; j++)
      {
         if (i == 0 || j == 0)
         {
            LCS[i][j] = 0;
         }
         else if (str1[i - 1] == str2[j - 1])
         {
            LCS[i][j] = 1 + LCS[i - 1][j - 1];
         }
         else
         {
            LCS[i][j] = max(LCS[i - 1][j], LCS[i][j - 1]);
         }
      }
   }
   int ind = LCS[len1][len2];
   char sequence[ind + 1];
   sequence[ind] = '\0';
   printf("The length of longest commen subsequnece: %d", ind);
   while (len1 > 0 && len2 > 0)
   {
      if (str1[len1-1] == str2[len2 - 1])
```

```
        {
            sequence[ind - 1] = str1[len1 - 1];
            len1--;
            len2--;
            ind--;
        }
        else if (LCS[len1 - 1][len2] > LCS[len1][len2 - 1])
        {
            len1--;
        }
        else
        {
            len2--;
        }
    }
    printf("\nLongest Common Subsequnce is: ");
    puts(sequence);
}
int main()
{
    int len1 = 0, len2 = 0;
    char dummy[100], _tempnam;
    printf("Enter string 1: ");
    gets(dummy);
    len1 = strlen(dummy);
    char str1[len1];
    strcpy(str1, dummy);
    printf("Enter string 2: ");
    gets(dummy);
```

```
    len2 = strlen(dummy);
    char str2[len2];
    strcpy(str2, dummy);
    lcs(len1, len2, str1, str2);
    return 0;
}
```

Output:

```
PS C:\Users\zatak\Downloads> cd "c:\Users\zatak\Downloads\" ; if ($?) { gcc LCS.c -o LCS } ; if ($?) { .\LCS }
Enter string 1: abaaba
Enter string 2: babbab
The length of longest commen subsequnece: 4
Longest Common Subsequnce is: baba
```

Conclusion: Thus, we have implemented longest common subsequence using Dynamic Programming Approach and the time complexity of the program is O(mn) where m and n are length of both the strings respectively.