Name	Kinnari Shah
UID no.	2021700058
Experiment No.	4

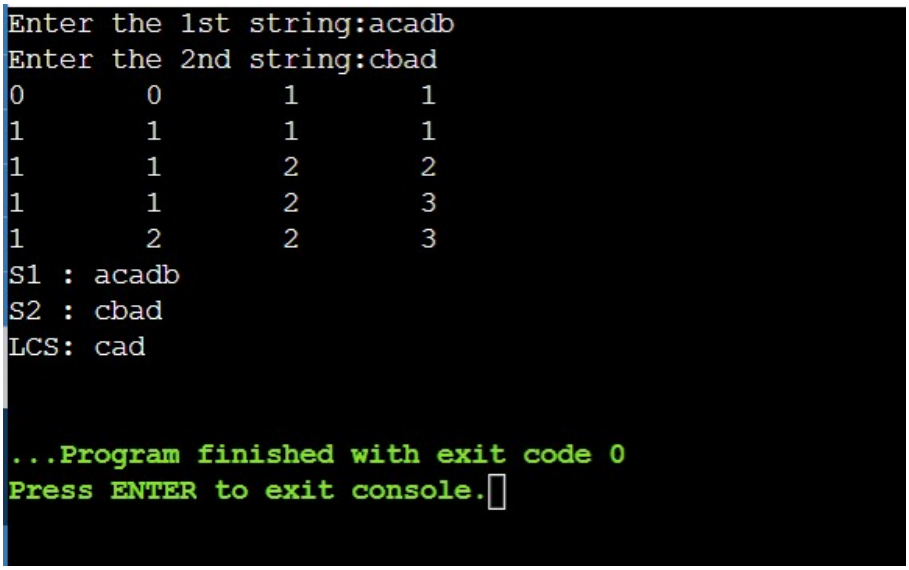
AIM:	Dynamic Programming -Longest Common Subsequence
Program 1	
PROBLEM STATEMENT :	Find the longest common subsequence from the 2 given strings.
ALGORITHM:	The longest common subsequence (LCS) is defined as the longest subsequence that is common to all the given sequences, provided that the elements of the subsequence are not required to occupy consecutive positions within the original sequences. 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. 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. 5. **Step 2** is repeated until the table is filled. 6. The value in the last row and the last column is the length of the longest common subsequence. 7. In order to find the longest common subsequence, start from the last element and follow the direction of the arrow.

PROGRAM:

```
#include <stdio.h>
#include <string.h>
int i, j, m, n, LCS_table[20][20];
char b[20][20];

void lcsAlgo(char S1[],char S2[]) {
  m = strlen(S1);
  n = strlen(S2);

 // Filling 0's in the matrix
  for (i = 0; i <= m; i++)
    LCS_table[i][0] = 0;
  for (i = 0; i <= n; i++)
    LCS_table[0][i] = 0;
 // Creating the mtrix in bottom-up way
  for (i = 1; i <= m; i++)
  {
    for (j = 1; j <= n; j++)
    {
      if (S1[i - 1] == S2[j - 1])
      {
        LCS_table[i][j] = LCS_table[i - 1][j - 1] + 1;
      }

      else
      {
        if (LCS_table[i - 1][j] >= LCS_table[i][j - 1])
        {
           LCS_table[i][j] = LCS_table[i - 1][j];
        }

        else
        {
        LCS_table[i][j] = LCS_table[i][j - 1];
        }
      }
     printf("%d\t",LCS_table[i][j]);
    }
  printf("\n");
  }
```

```
 int index = LCS_table[m][n];

 char lcsAlgo[index + 1];
 lcsAlgo[index] = '\0';

 int i = m, j = n;
 while (i > 0 && j > 0) {
  if (S1[i - 1] == S2[j - 1]) {
    lcsAlgo[index - 1] = S1[i - 1];
    i--;
    j--;
    index--;
  }

  else if (LCS_table[i - 1][j] > LCS_table[i][j - 1])
    i--;
  else
    j--;
 }

 // Printing the sub sequences
 printf("S1 : %s \nS2 : %s \n", S1, S2);
 printf("LCS: %s", lcsAlgo);
}

int main() {
   char S1[20] , S2[20];
   printf("Enter the 1st string:");
   scanf("%s",S1);

   printf("Enter the 2nd string:");
   scanf("%s",S2);
 lcsAlgo(S1,S2);
 printf("\n");
}
```

RESULT:



CONCLUSION:	In this experiment, we found the longest common subsequence of the 2 strings whose input was taken from the user.