|  |  |
| --- | --- |
| **Name** | Manish Shashikant Jadhav |
| **UID** | 2023301005 |
| **Subject** | Design and Analysis of Algorithms (DAA) |
| **Experiment No.** | 4 |
| **Aim** | To implement Dynamic Algorithms.   1. Assembly Line Scheduling. 2. Longest Common Subsequence. |
| **Code:** | **Longest Common Subsequence (LCS):**  #include <stdio.h>  #include <string.h>  *// Function to find the maximum of two integers*  int max(int *a*, int *b*)  {    return (*a* > *b*) ? *a* : *b*;  }  *// Function to find the length of longest common subsequence*  *// and print one of the common subsequences*  void lcs(char \**X*, char \**Y*, int *m*, int *n*)  {    int L[*m* + 1][*n* + 1];    int i, j;  *// Building the L[m+1][n+1] in bottom-up fashion*    for (i = 0; i <= *m*; i++)    {      for (j = 0; j <= *n*; j++)      {        if (i == 0 || j == 0)          L[i][j] = 0;        else if (*X*[i - 1] == *Y*[j - 1])          L[i][j] = L[i - 1][j - 1] + 1;        else          L[i][j] = max(L[i - 1][j], L[i][j - 1]);      }    }  *// Following code is used to print one of the common subsequence*    int index = L[*m*][*n*];    char lcs[index + 1];    lcs[index] = '\0';  *// Start from the right-most-bottom-most corner and*  *// one by one store characters in lcs[]*    i = *m*;    j = *n*;    while (i > 0 && j > 0)    {  *// If current character in X[] and Y are same, then*  *// current character is part of LCS*      if (*X*[i - 1] == *Y*[j - 1])      {        lcs[index - 1] = *X*[i - 1]; *// Put current character in result*        i--;        j--;        index--; *// reduce values of i, j and index*      }  *// If not same, then find the larger of two and*  *// go in the direction of larger value*      else if (L[i - 1][j] > L[i][j - 1])        i--;      else        j--;    }  *// Print the lcs*    printf("Longest Common Subsequence: %s\n", lcs);  }  int main()  {    char X[50], Y[50];    printf("Enter first sequence: ");    scanf("%s", X);    printf("Enter second sequence: ");    scanf("%s", Y);    int m = strlen(X);    int n = strlen(Y);    lcs(X, Y, m, n);    return 0;  } |
| **Output** |  |
| **Conclusion** | Hence, by completing this experiment I came to know about implementation of Dynamic Algorithms.   1. Assembly Line Scheduling. 2. Longest Common Subsequence. |