// A Dynamic Programming based C++ program for LPS problem

// Returns the length of the longest palindromic subsequence in seq

#include<stdio.h>

#include<string.h>

// A utility function to get max of two integers

int max (int x, int y) { return (x > y)? x : y; }

// Returns the length of the longest palindromic subsequence in seq

int lps(char \*str)

{

   int n = strlen(str);

   int i, j, cl;

   int L[n][n];  // Create a table to store results of subproblems

   // Strings of length 1 are palindrome of lentgh 1

   for (i = 0; i < n; i++)

      L[i][i] = 1;

    // Build the table. Note that the lower diagonal values of table are

    // useless and not filled in the process. The values are filled in a

    // manner similar to Matrix Chain Multiplication DP solution (See

    // <https://www.geeksforgeeks.org/archives/15553>). cl is length of

    // substring

    for (cl=2; cl<=n; cl++)

    {

        for (i=0; i<n-cl+1; i++)

        {

            j = i+cl-1;

            if (str[i] == str[j] && cl == 2)

               L[i][j] = 2;

            else if (str[i] == str[j])

               L[i][j] = L[i+1][j-1] + 2;

            else

               L[i][j] = max(L[i][j-1], L[i+1][j]);

        }

    }

    return L[0][n-1];

}

/\* Driver program to test above functions \*/

int main()

{

    char seq[] = "GEEKS FOR GEEKS";

    int n = strlen(seq);

    printf ("The lnegth of the LPS is %d", lps(seq));

    getchar();

    return 0;

}