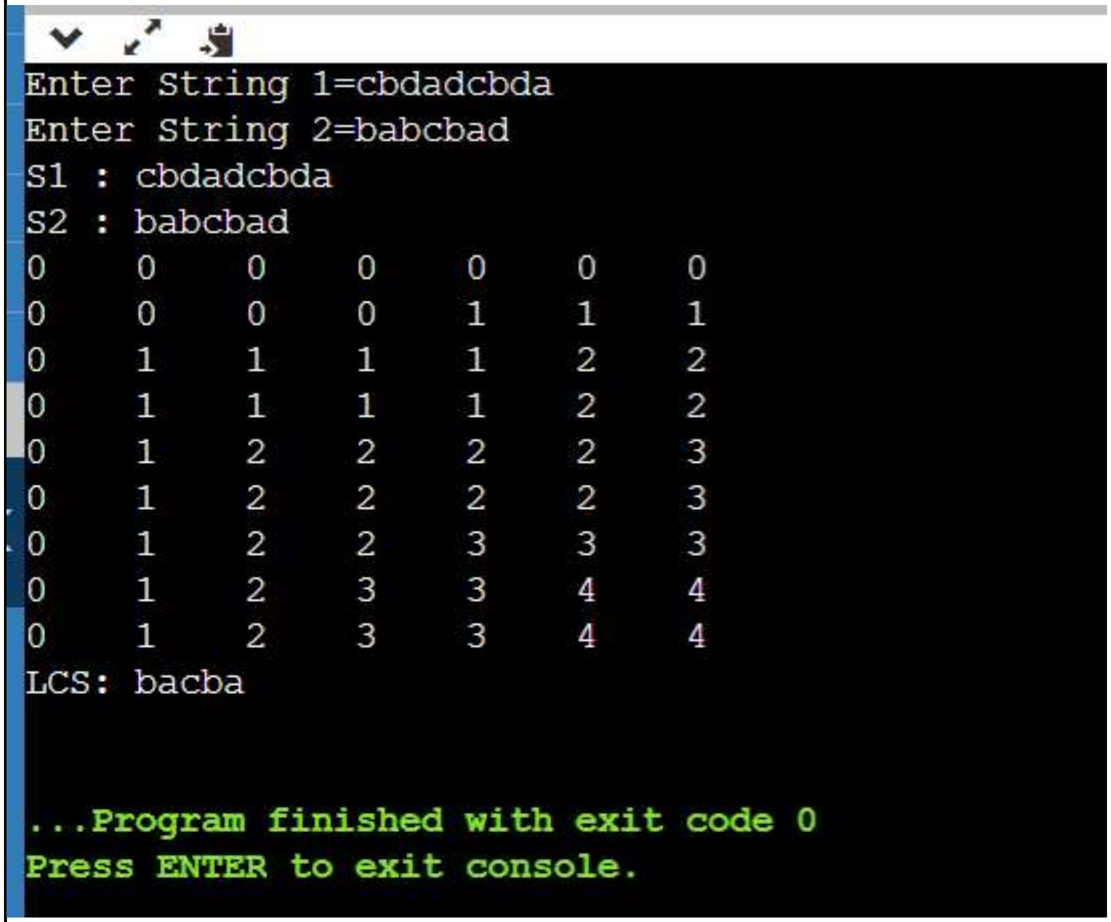
NAME:	Harshal Chawan
UID:	2021300019
SUBJECT	Design and Analysis of Algorithm
EXPERIMENT NO :	04
DATE OF PERFORMANCE	05/03/2023
DATE OF SUBMISSION	14/03/2023
AIM:	To find the longest common substring.
PROBLEM STATEMENT 1:	**Find the LCS of two strings.**
ALGORITHM and THEORY:	A subsequence of a string is a sequence that is generated by deleting some characters (possibly 0) from the string without altering the order of the remaining characters. For example, “abc”, “abg”, “bdf”, “aeg”, ‘”acefg”, etc are subsequences of the string “abcdefg”. **Input:** S1 = “AGGTAB”, S2 = “GXTXAYB” **Output:** 4 **Explanation:** The longest subsequence which is present in both strings is “GTAB”. **First step:** Initially create a 2D matrix (say dp[][]) of size 8 x 7 whose first row and first column are filled with 0. **Second step:** Traverse for i = 1. When j becomes 5, S1[0] and S2[4] are equal. So the dp[][] is updated. For the other elements take the maximum of dp[i-1][j] and dp[i][j-1]. (In this case, if both values are equal, we have used arrows to the previous rows). **Third step:** While traversed for i = 2, S1[1] and S2[0] are the same (both are ‘G’). So the dp value in that cell is updated. Rest of the elements are updated as per the conditions. **Fourth step:** For i = 3, S1[2] and S2[0] are again same. The updations are as follows.

	Fifth step: For i = 4, we can see that S1[3] and S2[2] are same. So dp[4][3] updated as dp[3][2] + 1 = 2. **Sixth step:** Here we can see that for i = 5 and j = 5 the values of S1[4] and S2[4] are same (i.e., both are ‘A’). So dp[5][5] is updated accordingly and becomes 3. **Final step:** For i = 6, see the last characters of both strings are same (they are ‘B’). Therefore the value of dp[6][7] becomes 4.

PROGRAM:

```c
#include<stdio.h>
#include<math.h>
#include<limits.h>
int MCM(int a[],int i,int j)
{
    if(i==j)
    {
        return 0;
    }
    int k;
    int min=INT_MAX;
    int count;
    for(k=i;k<j;k++)
    {
        count=MCM(a,i,k)+MCM(a,k+1,j)+a[i-1]*a[k]*a[j];
        if(count<min)
        {
            min=count;
        }
    }
    return min;
}
int main()
{
    int n,i,j;
    printf("Enter the size of the array: ");
    scanf("%d",&n);
    int a[n];
    for(i=0;i<n;i++)
    {
        a[i]=rand()%50;
    }
    printf("Array: ");
    for(i=0;i<n;i++)
```

	`{`
	`    printf("%d ",a[i]);`
	`}`
	`int N=sizeof(a)/sizeof(a[0]);`
	`printf("\nMinimum number of multiplications is %d.\n",MCM(a,1,n-1));`
	`        return 0;`
	`}`

OUTPUT:	
CONCLUSION:	By performing above experiment I have understood longest common subsequence. This dynamic programming approach reduces time complexity of the calculation of longest common subsequence.

```
Enter String 1=cbdadcbda
Enter String 2=babcbad
S1 : cbdadcbda
S2 : babcbad
0    0    0    0    0    0    0
0    0    0    0    1    1    1
0    1    1    1    1    2    2
0    1    1    1    1    2    2
0    1    2    2    2    2    3
0    1    2    2    2    2    3
0    1    2    2    3    3    3
0    1    2    3    3    4    4
0    1    2    3    3    4    4
LCS: bacba


...Program finished with exit code 0
Press ENTER to exit console.
```