Marwadi University	**Marwadi University** **Faculty of Technology** **Department of Information and Communication Technology**

Subject: DAA (01CT0512)	**AIM: Longest Common Subsequence using Dynamic Programming**	
Experiment No: 18	**Date: 26/9/2023**	**Enrolment No: 92100133020**

Longest Common Subsequence using Dynamic Programming:

Dynamic programming finds the longest common subsequence (LCS) of two sequences by building a 2D table where **dp[i][j]** stores the length of LCS of the first **i** characters of one sequence and the first **j** characters of the other.

Algorithm:

1. Create a 2D array **dp[m+1][n+1]** where **m** and **n** are the lengths of the two sequences.
2. Initialize the array with zeros.
3. Iterate through sequences and fill up the **dp** table based on whether the characters match or not.

Code:

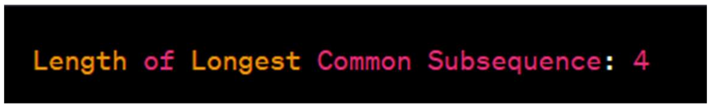
```
#include <iostream>
#include <cstring>
using namespace std;

int longestCommonSubsequence(string X, string Y) {
   int m = X.length();
   int n = Y.length();
   int dp[m + 1][n + 1];
   memset(dp, 0, sizeof(dp));

   for (int i = 1; i <= m; i++) {
      for (int j = 1; j <= n; j++) {
         if (X[i - 1] == Y[j - 1])
            dp[i][j] = dp[i - 1][j - 1] + 1;
         else
            dp[i][j] = max(dp[i - 1][j], dp[i][j - 1]);
      }
   }
   return dp[m][n];
}

int main() {
   string X = "AGGTAB";
   string Y = "GXTXAYB";
   cout << "Length of Longest Common Subsequence: " << longestCommonSubsequence(X, Y);
   return 0;
}
```

Output:



```
Length of Longest Common Subsequence: 4
```

Space complexity: __________

Justification:____
____

Time complexity:

Best case time complexity: __________

Justification:____
____

Worst case time complexity: __________

Justification:____