**=> LCS : O(n\*m) dp**

-> Brute Force Explanation : <https://youtu.be/diEIHwT_0kY?t=115>

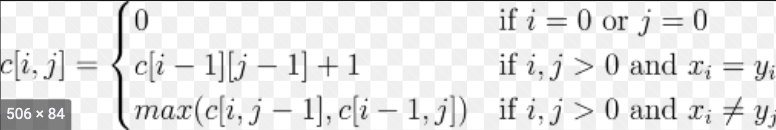
Code : <https://ide.geeksforgeeks.org/TCO0tBP5FF>

-> Let’s store overlapping subproblem value in a dp table.

**DP Bottom Up Recursive APPROACH:**

Code : <https://ide.geeksforgeeks.org/u9FtDRw5HL>

-> **Iterative DP + Print all LCS:**

****

Code : <https://ide.geeksforgeeks.org/W20qk5BwbU>

**=> LCS of 3 strings : O(n\*m\*n) dp**

**Problem :** [**https://practice.geeksforgeeks.org/problems/lcs-of-three-strings0028/1**](https://practice.geeksforgeeks.org/problems/lcs-of-three-strings0028/1)

**Approach :**

-> **If all 3 letters same then increment all 3 , ‘i’,’j’ and ‘k’,**

-> If any 2 same e.g s[i] and s[j] same, then we can’t simply increment i and j and keep k as it is bcoz this means now you will not be able to compare the s[i] and s[j] with any s[k] but actually it might be forming a common subsequence with some subsequence of s3

E.g

**abcd**

**abcd**

c**abcd**

If i and j are incremented just bcoz s1[0]=s2[0] its wont give correct answer as now the 0th letter ‘a’ of s1 and s2 can never be compared with letter ‘a’ of s3,so instead of correct answer=abcd. We will get answer=bcd

**-> SO if s[‘i’] , s[‘j’] , s[‘k’] don’t match , recur for all 6 possibilities:**

(i,j+1,k) , (i,j,k+1),(i,j+1,k+1)

(i+1,j,k), (i+1,j+1,k),(i+1,j,k+1)

No need to try (i,j,k) as we are already in this state currently and that is being compared already here,and don't recur for (i+1,j+1,k+1), as it would be visited by one of the states above.

-> So this nested 3 loop will do the job:

for(int l=i;l<=i+1;l++){

for(int m=j;m<=j+1;m++){

for(int n=k;n<=k+1;n++){

**if(!( (l==i && m==j && n==k) || (l==i+1&&m==j+1&&n==k+1) )){**

lcs(s1,s2,s3,l,m,n,n1,n2,n3);

dp[i][j][k]=max(dp[i][j][k],dp[l][m][n]);

}

}

}

}

**Code :** [**https://ideone.com/i6qpQ1**](https://ideone.com/i6qpQ1)

**=> LONGEST REPEATING SUBSEQUENCE : O(n^2) dp**

[**https://practice.geeksforgeeks.org/problems/longest-repeating-subsequence2004/1#**](https://practice.geeksforgeeks.org/problems/longest-repeating-subsequence2004/1#)

**LOGIC :** same as lcs just call **lcs(str,str)** , only change is that you can compare only if

Index i!=j, where i moves through first string and j moves through the second.