Lec-45

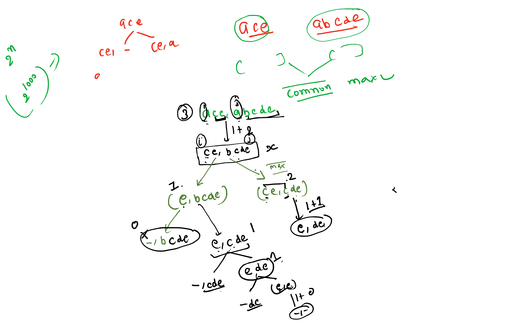
26/12/19 10:24 AM

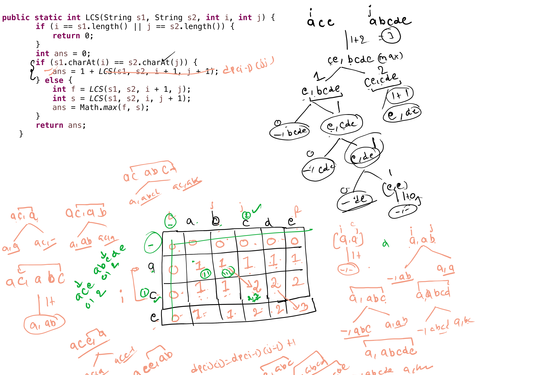
Given two strings `text1` and `text2`, return *the length of their longest* **common subsequence**. If there is no **common subsequence**, return `0`.

A **subsequence** of a string is a new string generated from the original string with some characters (can be none) deleted without changing the relative order of the remaining characters.

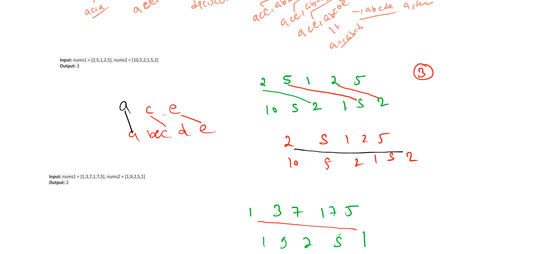
- For example, "ace" is a subsequence of "abcde".

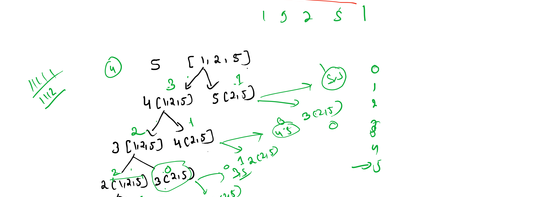
A **common subsequence** of two strings is a subsequence that is common to both strings.



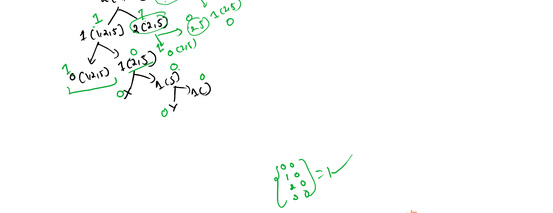


```
public static int LCS(String s1, String s2, int i, int j) {
        if (i == s1.length() || j == s2.length()) {
            return 0;
        }
        int ans = 0;
        if (s1.charAt(i) == s2.charAt(j)) {
            ans = 1 + LCS(s1, s2, i + 1, j + 1);
        } else {
            int f = LCS(s1, s2, i + 1, j);
            int s = LCS(s1, s2, i, j + 1);
            ans = Math.max(f, s);
        }
        return ans;
    }
```

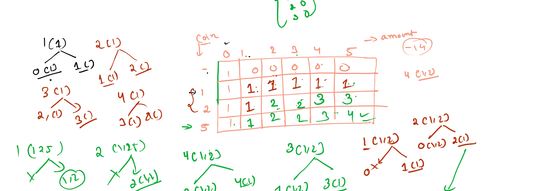


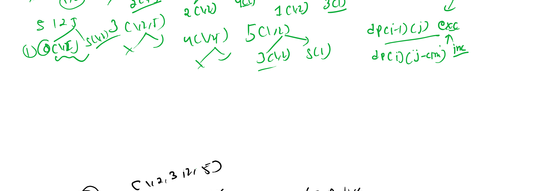


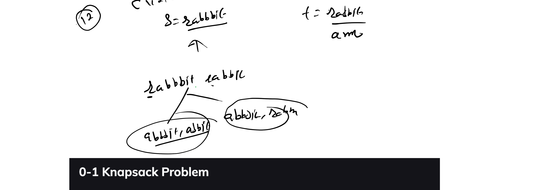
Input: nums1 = [2,5,1,2,5], nums2 = [10,5,2,1,5,2]
Output: 3

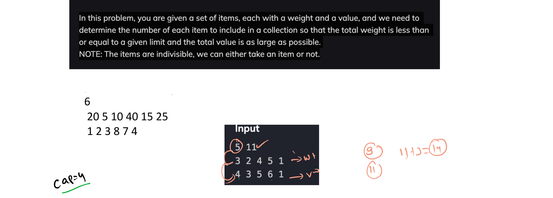


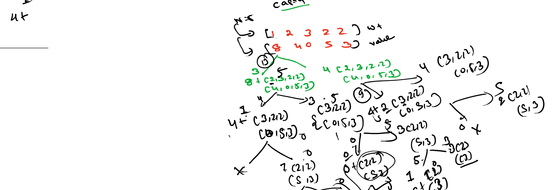
Input: nums1 = [1,3,7,1,7,5], nums2 = [1,9,2,5,1]
Output: 2











0-1 Knapsack Problem

In this problem, you are given a set of items, each with a weight and a value, and we need to determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.
NOTE: The items are indivisible, we can either take an item or not.

6
20 5 10 40 15 25
1 2 3 8 7 4

Input

5 11
3 2 4 5 1 → w1
4 3 5 6 1 → v1