



Minimum Deletions & Insertions to Transform a String into another

We'll cover the following

- Problem Statement
- Solution
- Bottom-up Dynamic Programming

Problem Statement

Given strings s1 and s2, we need to transform s1 into s2 by deleting and inserting characters. Write a function to calculate the count of the minimum number of deletion and insertion operations.

Example 1:

```
Input: s1 = "abc"
    s2 = "fbc"
Output: 1 deletion and 1 insertion.
Explanation: We need to delete {'a'} and insert {'f'} to s1 to transform it into s
2.
```

Example 2:

Example 3:

Solution

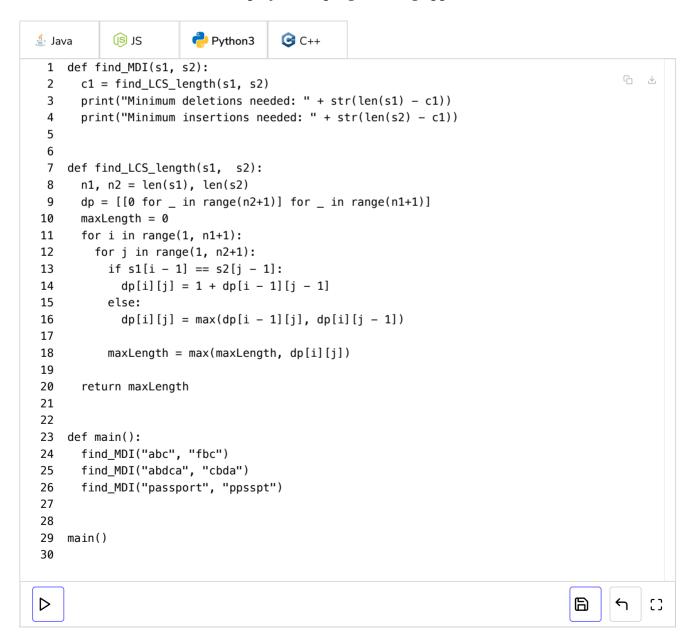
This problem can easily be converted to the Longest Common Subsequence (https://www.educative.io/collection/page/5668639101419520/5633779737559040/565753520167 (LCS). If we can find the LCS of the two input strings, we can easily find how many characters we need to insert and delete from s1. Here is how we can do this:

- 1. Let's assume len1 is the length of s1 and len2 is the length of s2.
- 2. Now let's assume c1 is the length of LCS of the two strings s1 and s2.
- 3. To transform s1 into s2, we need to delete everything from s1 which is not part of LCS, so minimum deletions we need to perform from s1 => len1 c1
- 4. Similarly, we need to insert everything in s1 which is present in s2 but not part of LCS, so minimum insertions we need to perform in s1 => len2 c1

Let's jump directly to the bottom-up dynamic programming solution:

Bottom-up Dynamic Programming

Here is the code for our bottom-up dynamic programming approach:



The time and space complexity of the above algorithm is O(m*n), where 'm' and 'n' are the lengths of the two input strings.

