**Print Shortest Common Supersequence**

**Hard**

Given two strings str1 and str2, return *the shortest string that has both*str1*and*str2*as****subsequences***. If there are multiple valid strings, return **any** of them.

A string s is a **subsequence** of string t if deleting some number of characters from t (possibly 0) results in the string s.

**Example 1:**

**Input:** str1 = "abac", str2 = "cab"

**Output:** "cabac"

**Explanation:**

str1 = "abac" is a subsequence of "cabac" because we can delete the first "c".

str2 = "cab" is a subsequence of "cabac" because we can delete the last "ac".

The answer provided is the shortest such string that satisfies these properties.

**Example 2:**

**Input:** str1 = "aaaaaaaa", str2 = "aaaaaaaa"

**Output:** "aaaaaaaa"

**Constraints:**

* 1 <= str1.length, str2.length <= 1000
* str1 and str2 consist of lowercase English letters.

class Solution {

public:

    string shortestCommonSupersequence(string str1, string str2) {

        int m=str1.length(), n=str2.length();

        string str="";

        int dp[m+1][n+1];

        memset(dp, 0, sizeof(dp));

        for (int i=0; i<n+1; i++) dp[0][i]=i;   // Tabulation DP

        for (int i=1; i<m+1; i++) dp[i][0]=i;

        for (int i=1; i<m+1; i++) {

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

                if (str1[i-1]==str2[j-1]) dp[i][j]=1+dp[i-1][j-1];

                else dp[i][j]=min(1+dp[i-1][j], 1+dp[i][j-1]);

            }

        }

        int i=m, j=n;

        while (i and j) {

            if (str1[i-1]==str2[j-1]) {

                str+=str1[i-1];

                i--; j--;

            }

            else {

                if (1+dp[i-1][j]<1+dp[i][j-1]) {

                    str+=str1[i-1];

                    i--;

                }

                else {

                    str+=str2[j-1];

                    j--;

                }

            }

        }

        if (i) {

            while (i) {

                str+=str1[i-1];

                i--;

            }

        }

        if (j) {

            while (j) {

                str+=str2[j-1];

                j--;

            }

        }

        reverse(str.begin(), str.end());

        return str;

    }

};