Data Structure and Algorithm Practicals

14. Implementation of Dynamic Programming- LCS

```
//by Situ Zhengmei
function LCS(str1, str2){
	var rows =  str1.split("")
	rows.unshift("")
	var cols =  str2.split("")
	cols.unshift("")
	var m = rows.length
	var n = cols.length
	var dp = []
	for(var i = 0; i < m; i++){
	   dp[i] = []
	   for(var j = 0; j < n; j++){
	      if(i === 0 || j === 0){
	         dp[i][j] = 0
	         continue
	      }

	      if(rows[i] === cols[j]){
	         dp[i][j] = dp[i-1][j-1] + 1 //Diagonal +1
	      }else{
	         dp[i][j] = Math.max( dp[i-1][j], dp[i][j-1]) //To the left, take the
largest from the top
	      }
	   }
	   console.log(dp[i].join(""))//debugging
	}
	return dp[i-1][j-1]
   }


//by Situ Zhengmei
function LCS(str1, str2){
   var m = str1.length
   var n = str2.length
   var dp = [new Array(n+1).fill(0)] //The first line is all 0
   for(var i = 1; i <= m; i++){ //A total of m+1 lines
      dp[i] = [0] //The first column is all 0
      for(var j = 1; j <= n; j++){//A total of n+1 columns
         if(str1[i-1] === str2[j-1]){
            //Note here, the first character of str1 is in the second column, so
you have to subtract 1, and str2 is the same
            dp[i][j] = dp[i-1][j-1] + 1 //Diagonal +1
         } else {
             dp[i][j] = Math.max( dp[i-1][j], dp[i][j-1])
```

```
            }
        }
    }
    return dp[m][n];
}


//by Situ Zhengmei, print an LCS
function printLCS(dp, str1, str2, i, j){
    if (i == 0 || j == 0){
        return "";
    }
    if( str1[i-1] == str2[j-1] ){
        return printLCS(dp, str1, str2, i-1, j-1) + str1[i-1];
    }else{
        if (dp[i][j-1] > dp[i-1][j]){
            return printLCS(dp, str1, str2, i, j-1);
        }else{
            return printLCS(dp, str1, str2, i-1, j);
        }
    }
}
//by Situ Zhengmei, convert the target string into regular, verify whether it is
the LCS of the previous two strings
function validateLCS(el, str1, str2){
   var re =  new RegExp( el.split("").join(".*") )
   console.log(el, re.test(str1),re.test(str2))
   return re.test(str1) && re.test(str2)
}
function LCS(str1, str2){
    var m = str1.length
    var n = str2.length
    //.... Omit, add it yourself
   // var s = printLCS(dp, str1, str2, m, n)
    validateLCS(s, str1, str2)
    return dp[m][n]
}
var c1 = LCS( "ABCBDAB","BDCABA");
console.log(c1) //4 BCBA?BCAB?BDAB
var c2 = LCS("13456778" , "357486782" );
console.log(c2) //5 34678
var c3 = LCS("ACCGGTCGAGTGCGCGGAAGCCGGCCGAA"
,"GTCGTTCGGAATGCCGTTGCTCTGTAAA" );
console.log(c3) //20 GTCGTCGGAAGCCGGCCGAA
```