*Example:*  
Justin wrote "love you".  
Then he launched his program and got:"qwlerotyvuei ypopu".  
Now it is totally not obvious what the original message was!

In order to decode it, Jessica should find the Longest Common Subsequence of the received message and her secret key.

Subsequence of a string a1..aN is a stringak1..akN, where  1 ≤ ki < ki + 1 ≤ N for everyi.

Your task is to write a program which can decode Justin's message using the secret key.  
It is guaranteed that the decoded string is unique in terms of symbol equity.

**Example:**

Let's say Jessica's secret key was"asldfgohjvkze xcybonmu".  
Now we take the Longest Common Subsequences of those two strings and we get exactly "love you", believe it or not.

*Illustration:*  
"qw**L**er**O**ty**V**u**E**i **Y**p**O**p**U**  
"as**L**dfg**O**hj**V**kz**E** xc**Y**b**O**nm**U**"  
*Note*: The space between "love" and "you" wasn't ignored as well. It exists in both strings at appropriate positions.  
All letters in both string are lower case. The upper case was used only for illustration purposes.

* **[input] string message**
  + Justin's encoded message of length in[0,100].
* **[input] string secret\_key**
  + Jessica's secret key of length in[1,500]
* **[output] string**
  + Decoded message.

<https://codefights.com/challenge/T65ab6QcMerEo65qJ>

--MI SOLUCION ACEPTADA-

-ADAPTACION DE : <http://www.geeksforgeeks.org/printing-longest-common-subsequence/>

std::string decode\_LCS\_cipher(std::string message, std::string secret\_key) {

int m = message.length();

int n = secret\_key.length();

int L[m+1][n+1];

for (int i=0; i<=m; i++)

{

for (int j=0; j<=n; j++)

{

if (i == 0 || j == 0)

L[i][j] = 0;

else if (message[i-1] == secret\_key[j-1])

L[i][j] = L[i-1][j-1] + 1;

else

L[i][j] = std::max(L[i-1][j], L[i][j-1]);

}

}

int index = L[m][n];

std::string lcs = "";

lcs[index] = '\0';

int i = m, j = n;

while (i > 0 && j > 0)

{

if (message[i-1] == secret\_key[j-1])

{

lcs.insert(lcs.begin(), message[i-1]);

i--; j--; index--;

}

else if (L[i-1][j] > L[i][j-1])

i--;

else

j--;

}

return lcs;

}