1. 2 longest common subsequences of the 2 strings are: TCAGTTG and TCGATGC. There are more.

𝑋= “TCAAAGATTAAGC”

𝑌= “TCGATGTCTCGTTG”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | T | C | G | A | T | G | T | C | T | C | G | T | T | G |
|  | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T | 0 | ***1*** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| C | 0 | 1 | ***2*** | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| A | 0 | 1 | 2 | 2 | **3** | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| A | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| A | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| G | 0 | 1 | 2 | *3* | 3 | 3 | **4** | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| A | 0 | 1 | 2 | 3 | *4* | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| T | 0 | 1 | 2 | 3 | 4 | *5* | 5 | **5** | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| T | 0 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 6 | **6** | 6 | 6 | 6 | 6 | 6 |
| A | 0 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| A | 0 | 1 | 2 | 3 | 4 | 5 | 5 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| G | 0 | 1 | 2 | 3 | 4 | 5 | *6* | 6 | 6 | 6 | 6 | **7** | 7 | 7 | 7 |
| C | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 6 | *7* | 7 | 7 | 7 | 7 | 7 | 7 |



printLCS(int[][] L, String X, Y)

String s = “”

for (int n = X.length, m = Y.length; n > 0 && m > 0;) {

while (L[n][m] == L[n-1][m] && n > 0) n--;

while (L[n][m] == L[n][m-1] && m > 0) m--;

if (n > 0 && m > 0) s += L[n][m];

n--;

m--;

}

return s.reverse()



Proof by contradiction: Given X[n] != Y[m] and Z[k] != X[n], assume Z is not an LCS of X[1]...X[n-1] and Y. We know that Z is an LCS of X and Y. For Z not to be an LCS of X[1]...X[n-1] and Y, but to be an LCS of X and Y, the difference between the 2 pairs of strings, or X[n], must be in Z. However, since X[n] is at the end of X, if it were to appear in Z, it must be the last character of Z, or Z[k], for Z to be an LCS of X and Y. This is a contradiction, therefore Z must be an LCS of X[1]...X[n-1] and Y given the above assumptions.

1. 1. [10, 8, 4, 6, 9]

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| A | 1 | 1 | 1 | 4 | 1 | 6 | 1 | 8 | 1 | 1 | 11 |
| B | 0 | 2 | 0 | 0 | 2 | 0 | 2 | 0 | 9 | 0 | 0 |
| C | 0 | 0 | 0 | 0 | 5 | 0 | 0 | 0 | 0 | 0 | 0 |
| D | 0 | 0 | 0 | 0 | 0 | 0 | 7 | 0 | 0 | 0 | 0 |
| R | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 10 | 0 |



consecutiveBlanks(String txt, int M) {

for (int i = 0, j = 0; i < txt.length(); i++) { //n loops max

if (txt.charAt(i) == “ “) { //1 time

j++; //1 time

if (j >= M) return i-(j-1); //1 time

}

else j=0; //1 time

}

return txt.length();

}

This algorithm is O(3n) = O(n), in every case.