public class Memoization {

private static String longer (String a, String b){

if (a.length() > b.length()) return a;

else return b;

}

private static int llcs( String u, String v ) {

if ( u.equals("") || v.equals("") ) return 0;

else if ( u.charAt(0) == v.charAt(0) ) return 1 + llcs( u.substring(1), v.substring(1) );

else return Math.max( llcs(u.substring(1),v), llcs(u,v.substring(1)) );

}

public static int llcsDP(String u, String v){

int m = u.length();

int n = v.length();

int[][] mem = new int[m+1][n+1];

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

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

if ( (i == 0) || (j == 0) ) mem[i][j] = 0;

else if ( u.charAt(m-i) == v.charAt(n-j) ) mem[i][j] = 1 + mem[i-1][j-1];

else

mem[i][j] = Math.max(

mem[i][j-1],

mem[i-1][j]

);

}}} // FINE FOR

return mem[m][n]; }

\* @param u parola1

\* @param v parola2

\* @return Longest Common Subsequence (MEMOIZATION = TOP-DOWN)

\*/

public static String lcsMem(String u, String v){

int m = u.length();

int n = v.length();

String[][] mem = new String[m+1][n+1];

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

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

if ( (i == 0) || (j == 0) ) mem[i][j] = "";

else if ( u.charAt(m-i) == v.charAt(n-j) ) {

mem[i][j] = u.charAt(m-i) + mem[i-1][j-1];

else

mem[i][j] = longer(

mem[i][j-1],

mem[i-1][j]

);

}}}

return mem[m][n]; }

\* @param u parola1

\* @param v parola2

\* @return LCS (DYNAMIC PROGRAMMING = BOTTOM-UP)

public static String lcsDP(String u, String v){

int m = u.length();

int n = v.length();

int[][] mem = new int[ m+1 ][ n+1 ];

String lcs = ""; // Percorso attraverso la matrice mem

int i = m;

int j = n;

while ( mem[i][j] > 0 ) {

if ( u.charAt(m-i) == v.charAt(n-j) ) {

lcs = lcs + u.charAt(m-i);

i = i - 1;

j = j - 1;

} else if ( mem[i-1][j] < mem[i][j-1] ) {

j = j - 1;

} else if ( mem[i-1][j] > mem[i][j-1] ) {

i = i - 1;

} else if ( Math.random() < 0.5 ) {

j = j - 1;

} else {

i = i - 1;

}}

return lcs; }

public static String lcsDp2( String u, String v ) {

int m = u.length();

int n = v.length();

String[] mem = new String[ n+1 ];

for ( int j=n; j>=0; j=j-1 ) {

mem[j] = "";

}

for ( int i=m-1; i>=0; i=i-1 ) {

String w = "";

for ( int j=n-1; j>=0; j=j-1 ) {

if ( u.charAt(i) == v.charAt(j) ) {

String t = u.charAt(i) + w;

w = mem[j];

mem[j] = t;

} else {

w = mem[j];

mem[j] = longer( mem[j], mem[j+1] );

}

}}

return mem[0];

}

}