15.4-2 : a version of this without b table.

```
PRINT-LCS(b, X, i, j)
1  if i == 0 or j == 0
2      return
3  if b[i, j] == "↖"
4      PRINT-LCS(b, X, i − 1, j − 1)
5      print x_i
6  elseif b[i, j] == "↑"
7      PRINT-LCS(b, X, i − 1, j)
8  else PRINT-LCS(b, X, i, j − 1)
```

This gives a different LCS from normal LCS algorithms.

① ↓

```
Print_LCS(c, X, i, j)
1  if i==0 or j==0
       return
if c[i][j] == c[i-1][j]   # up ↑
    Print_LCS(c, X, i-1, j)
elseif c[i][j] == c[i][j-1]   # Left ←
    Print-LCS(c, X, i, j-1)
else   # upper left ↖
    Print-LCS(c, X, i-1, j-1)
    Print x_i
```

(2)

```
Print_LCS(c, X, Y, i, j):
    if i==0 or j==0:
        return
    if X[i-1] == Y[j-1]:  # upper left ↖
        Print_LCS(c, X, Y, i-1, j-1)
        print x;
    else:
        if c[i-1][j] >= c[i][j-1]:  # up ↑
            Print_LCS(c, X, Y, i-1, j)
        else:  # left ←
            Print_LCS(c, X, Y, i, j-1)
```