Algorithm 1 Backtrack the longest common subsequence

```
function LCSQ($S_1$: Array($n$), $S_2$: Array($m$))
    $M, P \leftarrow$ LCSQ_MATRIX($S_1$, $S_2$)
    $L \leftarrow Array(M[n][m])$
    $k \leftarrow 0$
    $i \leftarrow n$
    $j \leftarrow m$
    while $i > 0$ and $j > 0$ do
        if $P[i][j] ='\nwarrow'$ then
            $L[k] \leftarrow S_1[i]$
            $i - -$
            $j - -$
            $k + +$
        else if $P[i][j] ='\leftarrow'$ then
            $j - -$
        else
            $i - -$
        end if
    end while
    return $L$
end function
```

Algorithm 2 Recursive reconstruction of the longest common subsequence

```
procedure LCSQ($S_1$: Array($n$), $S_2$: Array($m$))
    $M, P \leftarrow$ LCSQ_MATRIX($S_1$, $S_2$)
    $i \leftarrow n$
    $j \leftarrow m$
    AUX($P$, $S_1$, $i$, $j$)
end procedure
procedure AUX($P$: Array($n+1$, $m+1$), $S_1$: Array($n$), $i$, $j$)
    if $P[i][j] = '\nwarrow'$ then
        $l \leftarrow S_1[i]$
        AUX($P$, $S_1$, $i-1$, $j-1$)
        print($l$)
    else if $P[i][j] = '\leftarrow'$ then
        AUX($P$, $S_1$, $i$, $j-1$)
    else
        AUX($P$, $S_1$, $i-1$, $j$)
    end if
end procedure
```