Algorithm 1 Construct a longest common subsequence matrix keeping the path in memory

```
1: function LCSQ_Matrix_Path(S_1: Array(n), S_2: Array(m))
2:     M ← Array(m + 1, n + 1)
3:     P ← Array(m + 1, n + 1)
4:     for (i = 0; i < n + 1, i + +) do
5:         M[i][0] ← 0
6:     end for
7:     for (j = 0; j < m + 1; j+) do
8:         M[0][j] ← 0
9:     end for
10:    for (i = 1; i < n + 1; i + +) do
11:        for (j = 1; j < m + 1; j + +) do
12:            if i = 1 or j = 0 then
13:                M[i][j] = 0
14:            else
15:                if S_1[i − 1] = S_2[j − 1] then
16:                    M[i][j] ← M[i − 1][j − 1] + 1
17:                    P[i][j] ←'↖'
18:                else if M[i][j − 1] ≥ M[i − 1][j] then
19:                    M[i][j] ← M[i][j − 1]
20:                    P[i][j] ←'←'
21:                else
22:                    M[i][j] ← M[i − 1][j]
23:                    P[i][j] ←'↓'
24:                end if
25:            end if
26:        end for
27:    end for
28:    return M, P
29: end function
```