Algorithm 2 Symbolic LU Factorization.

```
Require: A m × n matrix A.
procedure LU(A, k)                                          ▷ Symbolic full-pivoting LU procedure
    M ← A                                                   ▷ Initialize the matrix M
    rnk ← min(m, n)                                         ▷ Initialize the rank of M
    for k from 1 to rnk do                                  ▷ Perform Gaussian elimination
        p, q, l ← SymbolicPivoting(M, k)                    ▷ Find the best pivot for the k-th step
        if p = 0 then                                       ▷ Check for null pivot
            rnk ← k − 1                                     ▷ The rank of M is k − 1
            break                                           ▷ The matrix is singular
        end if
        r_k, c_k ← q, l                                     ▷ Store the pivot row and column indices
        M ← SwapRows(M, k, q)                               ▷ Swap the k-th and q-th rows
        M ← SwapColumns(M, k, l)                            ▷ Swap the k-th and l-th columns
        for i from k + 1 to m do                            ▷ Compute the k-th column of L
            M_kk ← Veil(M_kk)                               ▷ Veil the k-th pivot
            M_ik ← Veil(Normalizer(M_ik/M_kk))              ▷ Normalize the k-th pivot
            for j from k + 1 to n do                        ▷ Compute the k-th row of U
                M_ij ← Veil(Normalizer(M_ij − M_ik M_kj))   ▷ Finalize the Schur complement
            end for
        end for
    end for
    P, Q ← PermutationMatrices(r, c)                        ▷ Compute the permutation matrices
    L ← LowerTriangular(M)                                  ▷ Extract the lower-triangular part of M
    U ← UpperTriangular(M)                                  ▷ Extract the upper-triangular part of M
    return L, U, P, Q, r, c, rnk                            ▷ Return the factors and the rank of A
end procedure
```