Procedure 2 Generate Pattern database using BFS.

```
Initialize a pattern database as dict( )
Initialize matrix X ← [1,0,-1,0] and Y ← [0,1,0,-1]
Set num_of_patterns ← $i$ for $i$ in initial pattern if $i$ is not equal to zero
Set QUEUE ← deque([(initial pattern, 0)])
Initialize the string of initial pattern as the list of the pattern database ← 0
while QUEUE:
    Set current state, current move ← QUEUE.popleft( )
    For num_of_pattern ∈ num_of_patterns :
        Initialize num_move_tile ← current state of pattern number as index
        Put $i, j$ ← num_move_tile // 4, num_move_tile % 4
        For $x, y$ ∈ zip(X, Y) :
            Initialize $r, p \leftarrow i + x, j + y$
            Initialize new state ← np.array(np.array(current state).reshape(4, 4))
            If $0 \leq r < 4$ and $0 \leq p < 4$ and new state[$r, p$] $==$ 0:
                Set new state[$i, j$], new state[$r, p$] ← new state[$r, p$], new state[$i, j$]
                Put new state = new state.flatten( ).tolist( )
                If new state ∉ pattern database :
                    QUEUE.append((new state, current move + 1))
                    Set pattern database[str(new state)] = current move + 1
Return pattern database
```