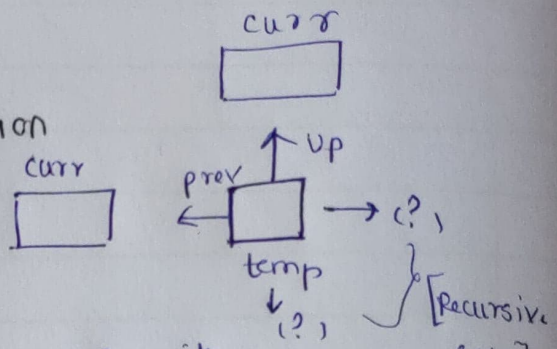
Algorithm for 2D Linked List Implementation



global matrix [MAXROW] [MAXCOL]

Parameters

Function[]: CREATE_2D_LL (int i, ← making Node for i^{th} row

int j, ← & j^{th} column element in matrix

int row, ← maxm no. of row in matrix

int col, ← maxm no. of cols in matrix

struct node * curr_node ← current node (pointer to))

(A) Base case: if (i >= row OR j >= col) (Out of Bounds in 0-based matrix)

return NULL;

Step 1: allocate memory for new node "temp"

Step 2: Initialise the data in "temp" node to mat[i][j]

{ temp_node → data = mat[i][j] }

Step 3: Initialize (Link) the 'prev' & 'up' links of "temp" node to current node [curr_node]

{ temp_node → prev = curr_node;
temp_node → up = curr_node; }

(B) Step 4: // Recursive Step

// Recursive step to link the left side Node with current node

temp_node → next = CREATE_2D_LL (i, j+1, row, col, temp_node);

// Recursive step to link the down side Node with current node

temp_node → down = CREATE_2D_LL (i+1, j, row, col, temp_node);

return temp_node // All 4 links complete