Strivers-A2Z-DSA-Sheet-main\02.Binary Search\2D Arrays\1.Row_with_maximum_number_of_1's.cpp

```cpp
/*
QUESTION:-
Given a boolean 2D array of n x m dimensions where each row is sorted. Find the 0-based index of the first row that has the maximum number of 1's.

Example 1:

Input:
N = 4 , M = 4
Arr[][] = {{0, 1, 1, 1},
           {0, 0, 1, 1},
           {1, 1, 1, 1},
           {0, 0, 0, 0}}
Output: 2
Explanation: Row 2 contains 4 1's (0-based indexing).
*/

/*
APPROACH:-
-> We can use two pointer i and j which indicates current row and col
-> As we know the matrix is row-wise sorted we can intilaize j=m-1 i.e. last col and i=0 i.e. first row
-> Now, the idea is we will keep moving left j while we occur 1 and if 0 is found we will check in next row
-> The last row where we encountered 1 will be our ans

                {*0, *1, *1, *1}
                {*0,  0,  1,  1}
out of matrix  *{*1,  1,  1,  1} ---> ans
                {0,   0,   0,   0}



*/

// CODE:-
int rowWithMax1s(vector<vector<int>> arr, int n, int m)
{
    int j = m - 1;
    int i = 0;
    int ans = -1;
    while (j >= 0 && i < n)
    {
        while (arr[i][j] == 1)
        {
            ans = i;
            j--;
        }
        if (i < n)
            i++;
    }
    return ans;
}

// TIME COMPLEXITY = O(N+M)
// SPACE COMPLEXITY = O(0)
```