Row with max 1s

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).
```

Example 2:

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

Your Task:
You don't need to read input or print anything. Your task is to complete the function **rowWithMax1s()** which takes the array of booleans **arr[][], n** and **m** as input parameters and returns the 0-based index of the first row that has the most number of 1s. If no such row exists, return -1.

Expected Time Complexity: O(N+M)
Expected Auxiliary Space: O(1)

Constraints:

$1 \leq N, M \leq 10^3$
$0 \leq Arr[i][j] \leq 1$

```
#User function Template for python3
class Solution:

    def rowWithMax1s(self,arr, n, m):
        # code here
        res = -1
```

```
        ones = 0
        lo = 0
        hi = 0
        while lo < n and hi < m:
            if arr[lo][hi] == 0:
                if hi!=m-1:
                    hi = hi+1
                else:
                    hi = 0
                    lo = lo+1
            else:
                temp = m-hi
                if temp>ones:
                    ones = temp
                    res = lo
                lo = lo+1
                hi = 0

        return res
```