Search in a sorted Matrix

Difficulty: **Medium** Accuracy: **56.27%** Submissions: **126K+** Points: **4**

Given a strictly sorted 2D matrix **mat[][]** of size **n x m** and a number **x.** Find whether the number **x** is present in the matrix or not.

Note: In a strictly sorted matrix, each row is sorted in strictly increasing order, and the first element of the $\mathbf{i}^{th}$ row (**i**!=0) is greater than the last element of the $(\mathbf{i-1})^{th}$ row.

Examples:

Input: mat[][] = [[1, 5, 9], [14, 20, 21], [30, 34, 43]], x = 14
Output: true
Explanation: 14 is present in the matrix, so output is true.

Input: mat[][] = [[1, 5, 9, 11], [14, 20, 21, 26], [30, 34, 43, 50]], x = 42
Output: false
Explanation: 42 is not present in the matrix.

Input: mat[][] = [[87, 96, 99], [101, 103, 111]], x = 101

« Prev
Next »

Java (1.8) | Start Timer

```
1  // } Driver Code Ends
36 class Solution {
37     public boolean searchMatrix(int[][] mat, int x) {
38         int n=mat.length;
39         int m=mat[0].length;
40         int left=0;
41         int right=n*m-1;
42         while(left<=right){
43             int mid=(left+right)/2;
44             int row=mid/m;
45             int col=mid%m;
46             if(mat[row][col]==x){
47                 return true;
48             }
49             else if(mat[row][col]<x){
50                 left=mid+1;
51             }
52             else{
53                 right=mid-1;
54             }
55         }
56         return false;
57     }
58 }
59
```

Custom Input | Compile & Run | Submit