1198. Find Smallest Common Element in All Rows

Medium 310 22 Add to List Share

Given an `m x n` matrix `mat` where every row is sorted in **strictly increasing** order, return *the* ***smallest common element*** *in all rows*.

If there is no common element, return `-1`.

Example 1:

```
Input: mat = [[1,2,3,4,5],[2,4,5,8,10],[3,5,7,9,11],[1,3,5,7,9]]
Output: 5
```

Example 2:

```
Input: mat = [[1,2,3],[2,3,4],[2,3,5]]
Output: 2
```

Constraints:

- `m == mat.length`
- `n == mat[i].length`
- `1 <= m, n <= 500`
- `1 <= mat[i][j] <= 10^4`
- `mat[i]` is sorted in strictly increasing order.

Accepted 24,007 | Submissions 31,504

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Walmart Labs | 2

Related Topics

Array Hash Table Binary Search Matrix Counting

Hide Hint 1

Notice that each row has no duplicates.

Hide Hint 2

Is counting the frequency of elements enough to find the answer?

Hide Hint 3

Use a data structure to count the frequency of elements.

Hide Hint 4

Find an element whose frequency equals the number of rows.

Java Autocomplete

```java
public int smallestCommonElement(int[][] mat) {
    int n = mat.length, m = mat[0].length;
    for (int j = 0; j < m; ++j) {
        boolean found = true;
        for (int i = 1; i < n && found; ++i) {
            found = Arrays.binarySearch(mat[i], mat[0][j]) >= 0;
        }
        if (found) {
            return mat[0][j];
        }
    }
    return -1;
}


public int smallestCommonElement(int[][] mat) {
    int n = mat.length, m = mat[0].length;
    int pos[] = new int[n], cur_max = 0, cnt = 0;
    while (true) {
        for (int i = 0; i < n; ++i) {
            while (pos[i] < m && mat[i][pos[i]] < cur_max) {
                ++pos[i];
            }
            if (pos[i] >= m) {
                return -1;
            }
            if (mat[i][pos[i]] != cur_max) {
                cnt = 1;
                cur_max = mat[i][pos[i]];
            } else if (++cnt == n) {
                return cur_max;
            }
        }
    }
}
```