2D Arrays-part-1

Assignment Solutions

Relevel
by Unacademy

Assignment Solutions

1. Write an efficient algorithm to search a value in a 2D matrix in which the rows are in sorted order and the first integer of each row is greater than the last integer of the previous row.
Solution - https://jsfiddle.net/rs6fkeob/

```
var M = 3;
var N = 4;


function binarySearch(arr, K)
{
 var low = 0;
 var high = N - 1;
 while (low <= high) {
  var mid = low + parseInt((high - low) / 2);


  if (arr[mid] == K)
   return true;


  if (arr[mid] < K)
   low = mid + 1;
  else
   high = mid - 1;
 }


 return false;
}


function searchMatrix(matrix, K)
{
 var low = 0;
 var high = M - 1;
 while (low <= high) {
  var mid = low + parseInt((high - low) / 2);


  if (K >= matrix[mid][0]
   && K <= matrix[mid][N - 1])
   return binarySearch(matrix[mid], K);
```

```
 if (K < matrix[mid][0])
  high = mid - 1;
 else
  low = mid + 1;
}


return false;
}

var matrix = [ [ 1, 3, 5, 7 ],
     [ 10, 11, 16, 20 ],
     [ 23, 30, 34, 50 ] ];
var K = 3;
if (searchMatrix(matrix, K))
 document.write( "Found" );
else
 document.write( "Not found" );
```

2. Find the row index which has maximum no. of unique elements in a matrix efficiently.
Solution - https://jsfiddle.net/x1ug3596/

```
function get(n, m, v) {

 let s = new Set();


 let max_ans = Number.MAX_VALUE;
 let cnt = -1;

 for (let i = 0; i < n; i++) {
  for (let j = 0; j < m; j++) {
   s.add(v[i][j]);
  }
  let size = s.size;
  if (cnt < size) {
   size = cnt;
   max_ans = Math.min(max_ans, i);
  }
  s.clear();
 }
 return max_ans;
}
```

```
 let arr
 = [[1, 2, 13, 4, 5],
 [1, 2, 2, 4, 17],
 [1, 3, 11, 3, 1]];
let n = arr.length;
let m = arr[0].length;
document.write(get(n, m, arr));
```