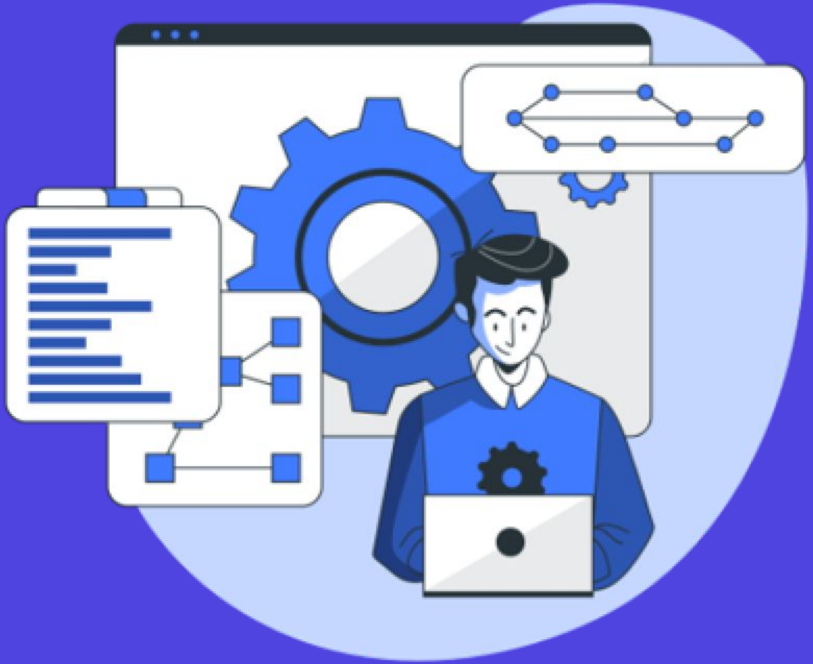
Problem Solving on 2D Array



Relevel
by Unacademy

Problem - 1

Given a 2D array A, return a boolean value indicating that the given matrix is a diagonal matrix or not. A matrix is said to be diagonal if all the elements of the array other then the diagonal is 0

- If there will be zero in the diagonal then also, we will print output as true if all remaining cells have zero. Having zeros in diagonal does not affect our output

Input:

```
A=[  [1,0,0,0],
     [0,2,0,0],
     [0,0,3,0],
     [0,0,0,4] ]
```

Output:

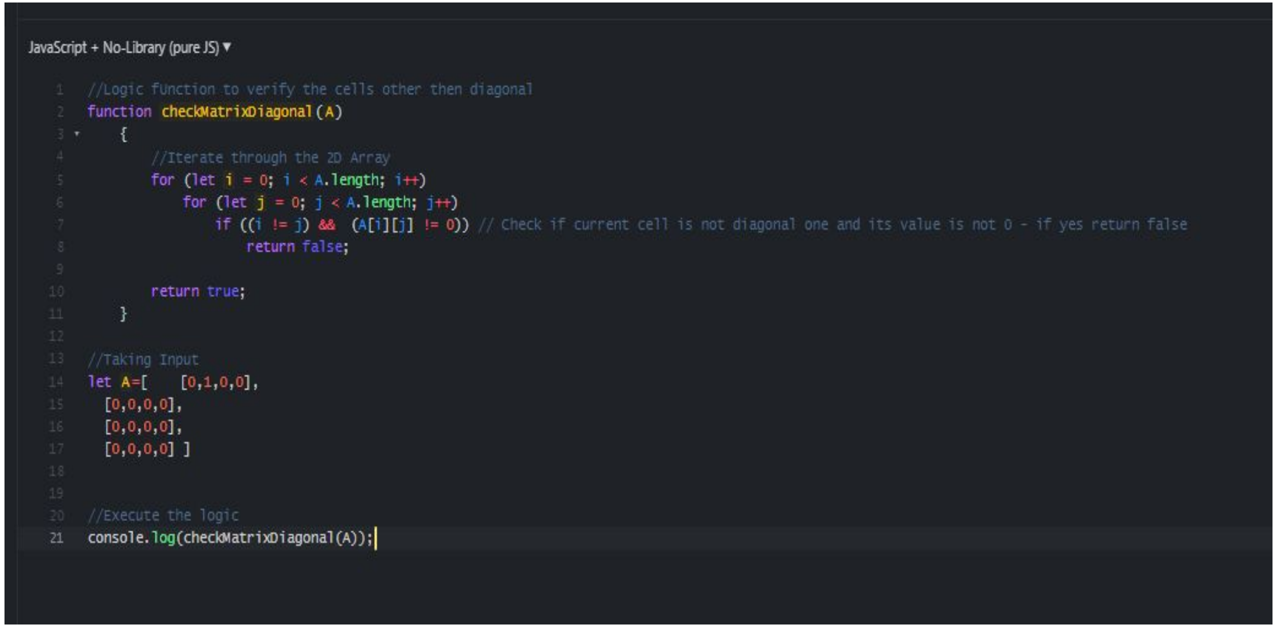
true

Explanation:

Since all the elements of the array other then the diagonal element are 0, it can be considered as a diagonal matrix

Solution-

Code Link - https://jsfiddle.net/4juazeph/



JavaScript + No-Library (pure JS) ▾

```javascript
//Logic function to verify the cells other then diagonal
function checkMatrixDiagonal(A)
    {
        //Iterate through the 2D Array
        for (let i = 0; i < A.length; i++)
            for (let j = 0; j < A.length; j++)
                if ((i != j) && (A[i][j] != 0)) // Check if current cell is not diagonal one and its value is not 0 - if yes return false
                    return false;

        return true;
    }

//Taking Input
let A=[    [0,1,0,0],
  [0,0,0,0],
  [0,0,0,0],
  [0,0,0,0] ]


//Execute the logic
console.log(checkMatrixDiagonal(A));
```

#180DaysofPurpose

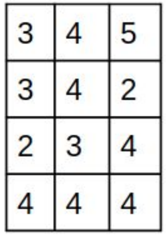
Relevel
by Unacademy

Problem - 2

Given a 2D array of size MxN, you need to display N integers which denotes the column wise addition of the 2D array

Input:

M=4,N=3



3	4	5
3	4	2
2	3	4
4	4	4

Output:

12,15,15

Explanation:

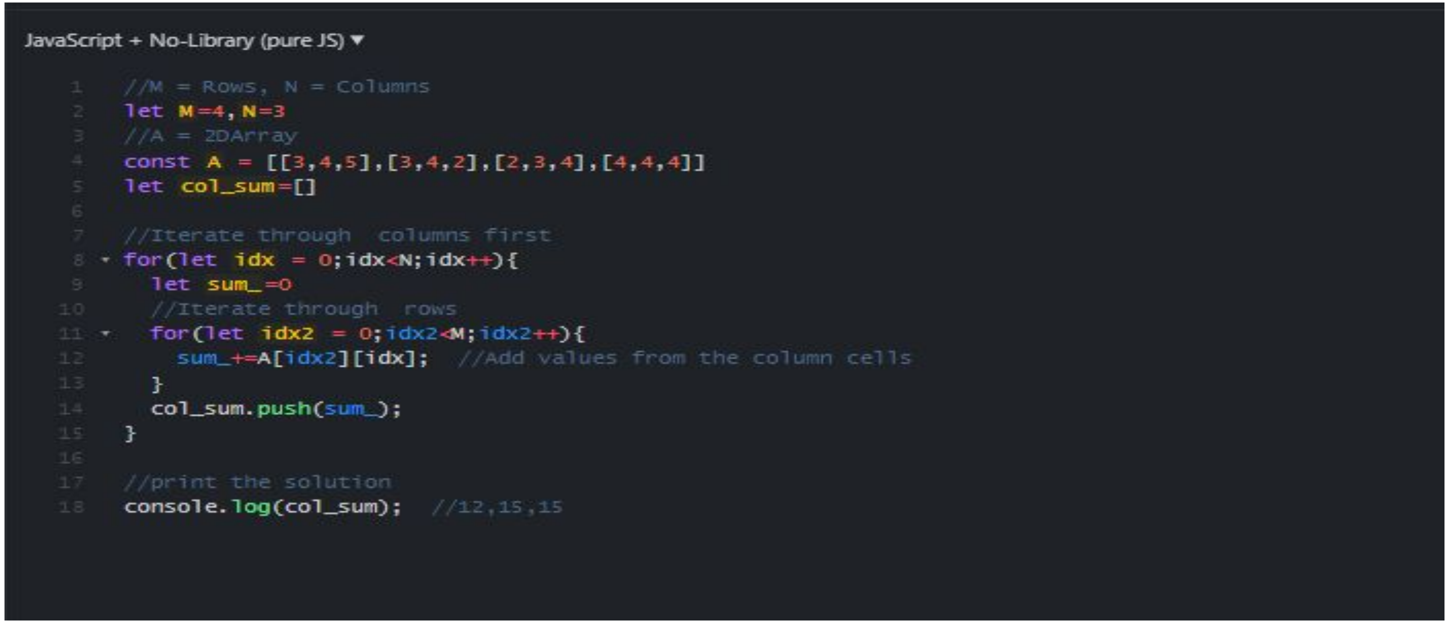
Sum of column-1 : 3+3+2+4= 12

Sum of column-2 : 4+4+3+4 = 15

Sum of column-3 : 2+3+4+4 = 15

Solution-

Code Link - https://jsfiddle.net/2mgetfps/



```
//M = Rows, N = Columns
let M=4, N=3
//A = 2DArray
const A = [[3,4,5],[3,4,2],[2,3,4],[4,4,4]]
let col_sum=[]

//Iterate through  columns first
for(let idx = 0;idx<N;idx++){
  let sum_=0
  //Iterate through  rows
  for(let idx2 = 0;idx2<M;idx2++){
    sum_+=A[idx2][idx];  //Add values from the column cells
  }
  col_sum.push(sum_);
}

//print the solution
console.log(col_sum);  //12,15,15
```

Approach - 2

Intuition:

We can reduce it by applying column wise addition on two rows at a time. I.e.

3	4	5
3	4	2

On using reduce, we will apply same logic with the next row

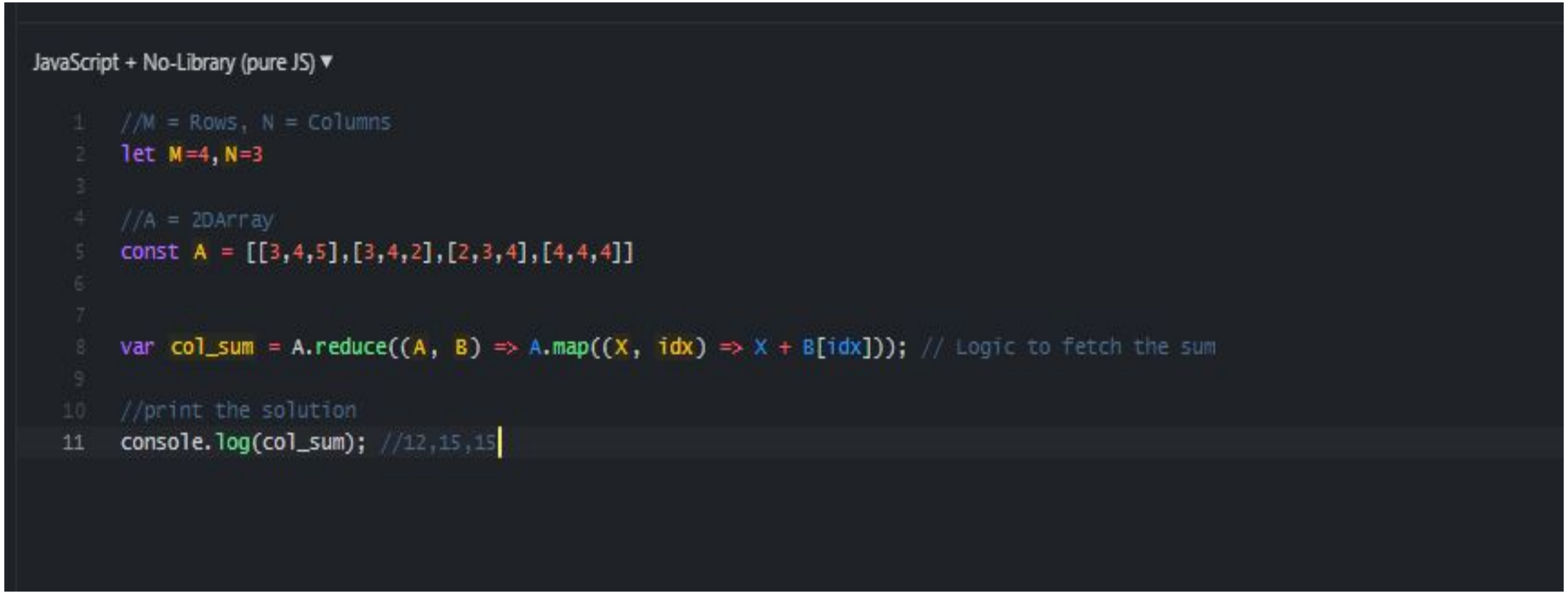
 [6,8,7]
+[2,3,4]
—-------
[8,11,11]

Again using reduce we add this with next row

[8,11,11]
+[4,4,4]
—---------
[12,15,15]

Solution - 2

Code Link - https://jsfiddle.net/yapc5v2d/



JavaScript + No-Library (pure JS) ▾

```
//M = Rows, N = Columns
let M=4,N=3

//A = 2DArray
const A = [[3,4,5],[3,4,2],[2,3,4],[4,4,4]]


var col_sum = A.reduce((A, B) => A.map((X, idx) => X + B[idx])); // Logic to fetch the sum

//print the solution
console.log(col_sum); //12,15,15
```

Problem - 3

Problem Statement - Given a 2D array. Our task is to find the sum of lower and upper triangles in the array.
Lower triangle sum = sum of diagonal + lower triangle elements
Upper triangle sum = sum of diagonal + upper triangle elements

Input -

Input : {2, 15, 4}
{1, 2, 15}
{2, 10, 7}

Output :
Upper sum is 45
Lower sum is 24

Approach -

Intuition - Since we need to find the sum of the upper and lower triangle, we will use loops and conditionals here. We will use row and column index comparison to check the position of the element

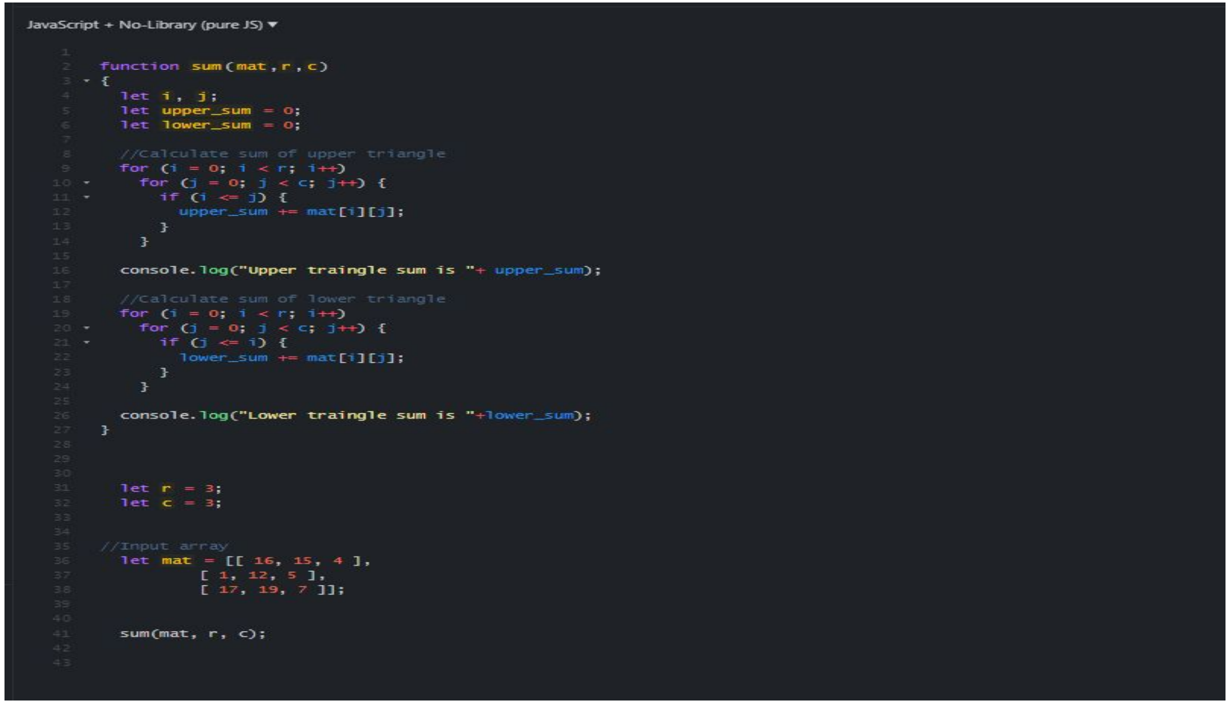
Let's have a look step by step -

1) To check upper triangle element we will use - (i <= j)
2) To check lower triangle element we will use - (j <= i)
   Where i and j are row and column index respectively

Code link - https://jsfiddle.net/juf5szb7/

Solution:

Code link - https://jsfiddle.net/juf5szb7/



JavaScript + No-Library (pure JS) ▾

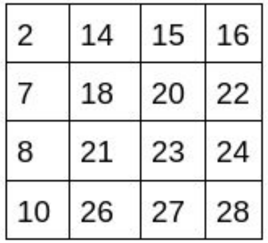
```

function sum(mat,r,c)
{
  let i, j;
  let upper_sum = 0;
  let lower_sum = 0;

  //Calculate sum of upper triangle
  for (i = 0; i < r; i++)
    for (j = 0; j < c; j++) {
      if (i <= j) {
        upper_sum += mat[i][j];
      }
    }

  console.log("Upper traingle sum is "+ upper_sum);

  //Calculate sum of lower triangle
  for (i = 0; i < r; i++)
    for (j = 0; j < c; j++) {
      if (j <= i) {
        lower_sum += mat[i][j];
      }
    }

  console.log("Lower traingle sum is "+lower_sum);
}



  let r = 3;
  let c = 3;


//Input array
  let mat = [[ 16, 15, 4 ],
            [ 1, 12, 5 ],
            [ 17, 19, 7 ]];


  sum(mat, r, c);


```

Relevel
by Unacademy

Problem - 4

Given a 2D array where the value of any row element is greater than the previous row. Our task is to find if element x is present in the array.

Input:



2	14	15	16
7	18	20	22
8	21	23	24
10	26	27	28

X= 21

Output:
True

Explanation:
Position of element 15 in the array is 2,1, so it exists hence true

Approach

In the problem statement, it is mentioned that every row element is greater than the previous row element. So we can start from the top right element and can move either left or downward based upon the comparison of the value.

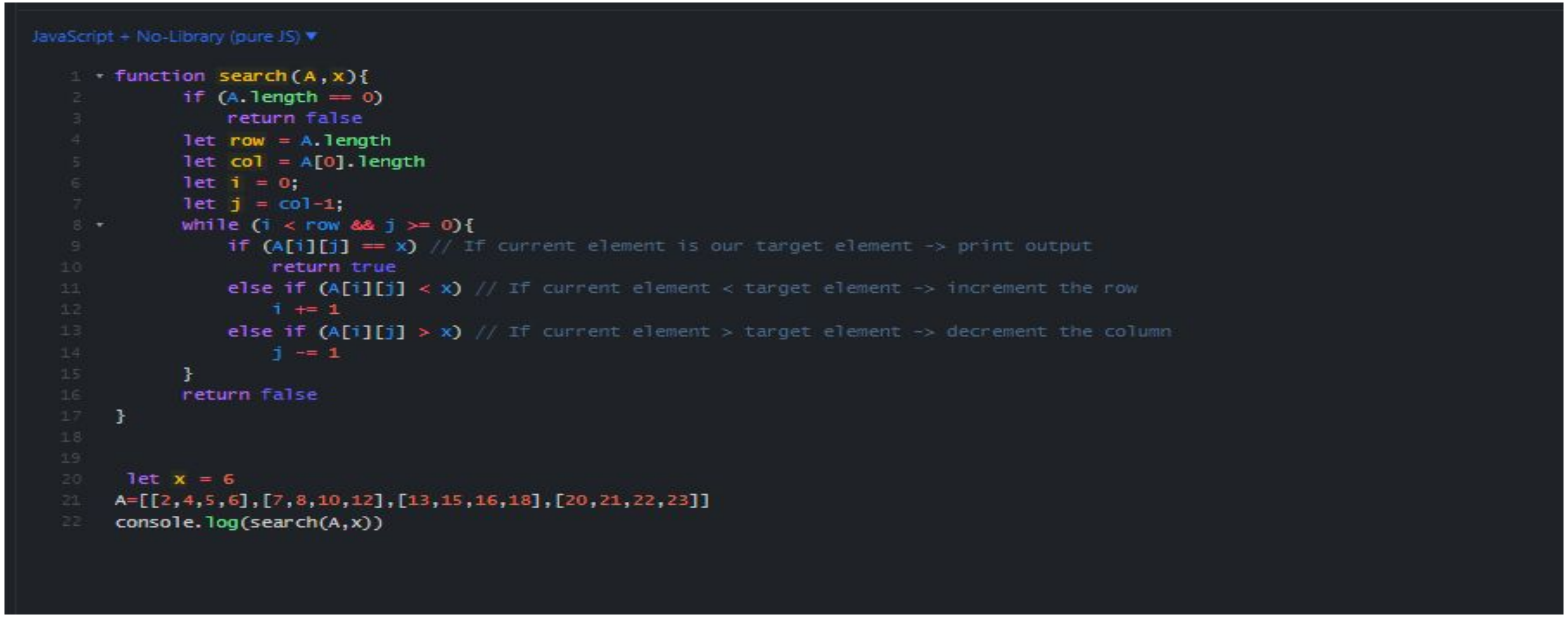
Intuition: Since, we need to start from the top right element, we will initialize our pointers to the index of the top right cell. Then we will continuously check if current cell element is equal to the required element and if not we will move either left or down based on the comparison

Steps -

1) Initialize i and j to top right element of matrix i.e. -> i=0, j=col-1
2) Iterate through the matrix and check if current element is our target element - if yes print output
3) If current element < target element -> increment the row
4) If current element > target element -> decrement the column

Solution

Code Link - https://jsfiddle.net/wh8jureg/



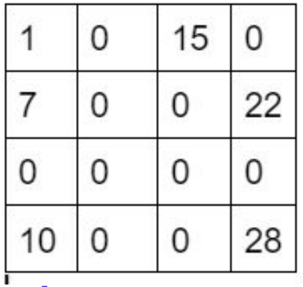
JavaScript + No-Library (pure JS) ▾

```
function search(A,x){
        if (A.length == 0)
            return false
        let row = A.length
        let col = A[0].length
        let i = 0;
        let j = col-1;
        while (i < row && j >= 0){
            if (A[i][j] == x) // If current element is our target element -> print output
                return true
            else if (A[i][j] < x) // If current element < target element -> increment the row
                i += 1
            else if (A[i][j] > x) // If current element > target element -> decrement the column
                j -= 1
        }
        return false
}


 let x = 6
A=[[2,4,5,6],[7,8,10,12],[13,15,16,18],[20,21,22,23]]
console.log(search(A,x))
```

Problem - 5

Given a 2D array. Our task is to check if the array is sparse or not.
Sparse Matrix - A matrix is said to be sparse if the number of zeros is more than half of the total elements.

Input:



1	0	15	0
7	0	0	22
0	0	0	0
10	0	0	28

Output:
True

Approach :

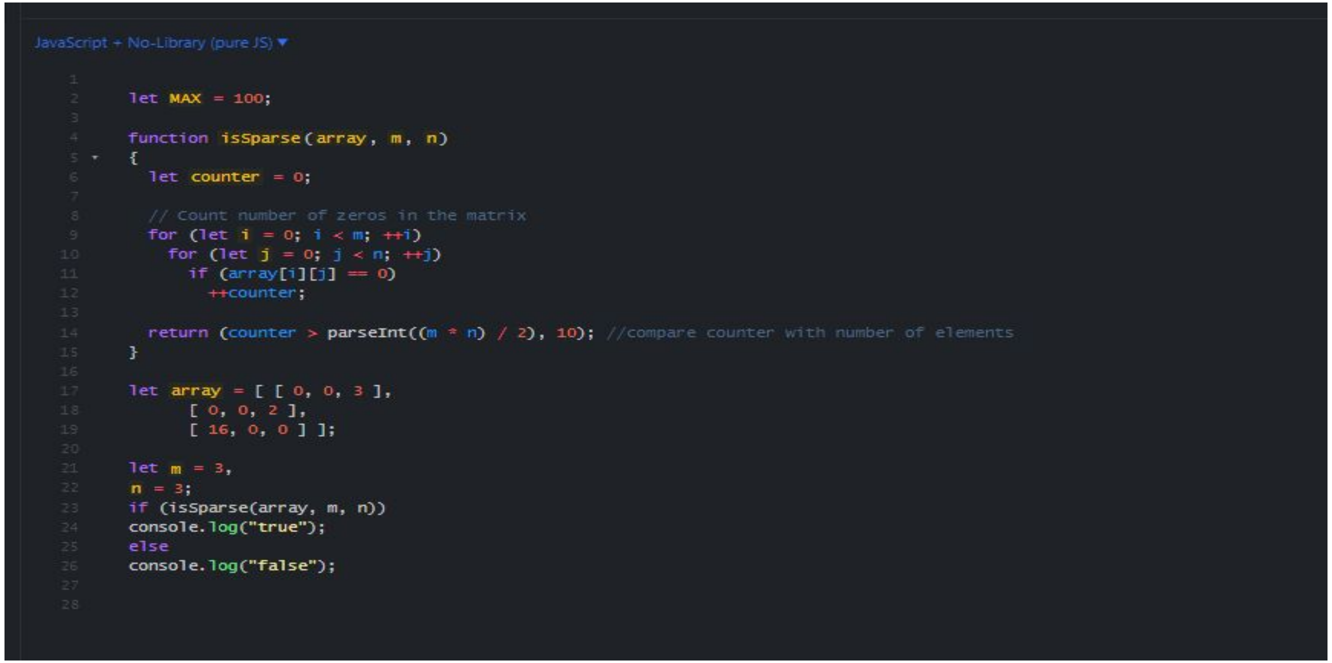
Intuition - Since it is mentioned that if half of the elements in a given array are zero, then it will be a sparse matrix. So we need to find the total number of zeros in the array. We will compare it with total elements and if it is greater then we will print *true* else *false*

Steps -

1) Let m = number of rows and n = number of columns, counter = 0
2) Total elements = m*n
3) Iterate through the array and increment the counter if element is 0
4) Compare counter value with (total elements) / 2

Solution:

Code link -https://jsfiddle.net/L0the93n/



```

let MAX = 100;

function isSparse(array, m, n)
{
  let counter = 0;

  // Count number of zeros in the matrix
  for (let i = 0; i < m; ++i)
    for (let j = 0; j < n; ++j)
      if (array[i][j] == 0)
        ++counter;

  return (counter > parseInt((m * n) / 2), 10); //compare counter with number of elements
}

let array = [ [ 0, 0, 3 ],
      [ 0, 0, 2 ],
      [ 16, 0, 0 ] ];

let m = 3,
n = 3;
if (isSparse(array, m, n))
console.log("true");
else
console.log("false");


```

Problem - 6

Given a 2D array. Our task is to find the unique elements. A unique element is an element whose frequency is 1 i.e. it is not repeating in the whole 2D Array.
If there is no any unique element, print message as “No unique element found”

Input:

2	14	15	18
10	18	14	22
8	21	22	15
10	14	21	28

Output:

```
2
8
```

Approach :

In the problem statement, it is mentioned that we need to find the unique element. We can use a count array which is storing count of elements. If count = 1, we will print those elements in output

Steps -

1) FInd maximum element of the given array as max
2) Initialize 1D array having size = max
3) Iterate through input array and increment value in count array for index = element
4) Iterate through count array and print index if value is equal to 1

Solution:

Code link - https://jsfiddle.net/j38uhwve/

Practice Question

Problem-1:

Given a 1D array of size 10 , convert it into a 2D array of size 2x5

Input - {1,2,3,4,5,6,7,8,9,10}

Output -

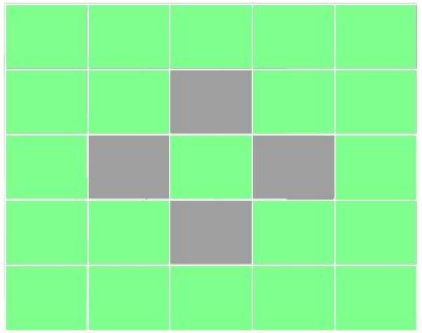
[[1,2,3,4,5],

[6,7,8,9,10]]

Problem-2:

Given a 2D array, find the sum of the diagonal and the boundary elements of it.

In the given matrix of size 5x5 the colored cell marks the diagonal and the boundary elements



Input:

A=[[1,2,3,4,1],

[5,6,7,8,2],

[9,10,11,12,13],

[13,14,15,16,15],

[11,12,15,19,15],

]

Output

195

Thank You!