Naga Rajasekhar

Arrays - 5th Assignment

**1st Solution**

function convert1DTo2D(original, m, n) {

if (original.length !== m \* n) {

return [];

}

const result = [];

for (let i = 0; i < m; i++) {

const row = [];

for (let j = 0; j < n; j++) {

const k = i \* n + j;

if (k < original.length) {

row.push(original[k]);

} else {

return result;

}

}

result.push(row);

}

return result;

}

const original = [1, 2, 3, 4];

const m = 2;

const n = 2;

const output = convert1DTo2D(original, m, n);

console.log(output);

**2nd Solution**

function arrangeCoins(n) {

let left = 0;

let right = n;

while (left <= right) {

let k = Math.floor((left + right) / 2);

let curr = k \* (k + 1) / 2;

if (curr === n) {

return k;

}

if (curr > n) {

right = k - 1;

} else {

left = k + 1;

}

}

return right;

}

const n = 5;

const result = arrangeCoins(n);

console.log(result);

**3rd Solution**

function sortedSquares(nums) {

const result = [];

for (let i = 0; i < nums.length; i++) {

const square = nums[i] \* nums[i];

result.push(square);

}

result.sort((a, b) => a - b);

return result;

}

const nums = [-4, -1, 0, 3, 10];

const squaredSortedArray = sortedSquares(nums);

console.log(squaredSortedArray);

**4th Solution**

function findDisappearedNumbers(nums1, nums2) {

const set1 = new Set(nums1);

const set2 = new Set(nums2);

const answer1 = [];

const answer2 = [];

for (let num of nums1) {

if (!set2.has(num)) {

answer1.push(num);

}

}

for (let num of nums2) {

if (!set1.has(num)) {

answer2.push(num);

}

}

return [answer1, answer2];

}

const nums1 = [1, 2, 3];

const nums2 = [2, 4, 6];

const answer = findDisappearedNumbers(nums1, nums2);

console.log(answer);

**5th Solution**

function findTheDistanceValue(arr1, arr2, d) {

let count = 0;

for (let num1 of arr1) {

let satisfied = false;

for (let num2 of arr2) {

if (Math.abs(num1 - num2) <= d) {

satisfied = true;

break;

}

}

if (!satisfied) {

count++;

}

}

return count;

}

const arr1 = [4, 5, 8];

const arr2 = [10, 9, 1, 8];

const d = 2;

const distanceValue = findTheDistanceValue(arr1, arr2, d);

console.log(distanceValue);

**6th Solution**

function findDuplicates(nums) {

const result = [];

for (let num of nums) {

const index = Math.abs(num) - 1;

if (nums[index] < 0) {

result.push(Math.abs(num));

} else {

nums[index] = -nums[index];

}

}

return result;

}

const nums = [4, 3, 2, 7, 8, 2, 3, 1];

const duplicates = findDuplicates(nums);

console.log(duplicates);

**7th Solution**

function findMin(nums) {

let left = 0;

let right = nums.length - 1;

while (left < right) {

const mid = Math.floor((left + right) / 2);

if (nums[mid] > nums[right]) {

left = mid + 1;

} else {

right = mid;

}

}

return nums[left];

}

const nums = [3, 4, 5, 1, 2];

const minElement = findMin(nums);

console.log(minElement);

**8th Solution**

function findOriginalArray(changed) {

const n = changed.length;

if (n % 2 !== 0) {

return [];

}

changed.sort((a, b) => a - b);

const original = [];

const freqMap = new Map();

for (let num of changed) {

freqMap.set(num, (freqMap.get(num) || 0) + 1);

}

for (let num of changed) {

if (freqMap.get(num) > 0) {

freqMap.set(num, freqMap.get(num) - 1);

if (!freqMap.has(num / 2) || freqMap.get(num / 2) === 0) {

return [];

}

freqMap.set(num / 2, freqMap.get(num / 2) - 1);

original.push(num / 2);

}

}

return original;

}

const changed = [1, 3, 4, 2, 6, 8];

const original = findOriginalArray(changed);

console.log(original);