Naga Rajasekhar

Arrays - 4th Assignment

**1st Solution**

function arraysIntersection(arr1, arr2, arr3) {

const result = [];

let i = 0, j = 0, k = 0;

while (i < arr1.length && j < arr2.length && k < arr3.length) {

if (arr1[i] === arr2[j] && arr2[j] === arr3[k]) {

result.push(arr1[i]);

i++;

j++;

k++;

} else if (arr1[i] < arr2[j]) {

i++;

} else if (arr2[j] < arr3[k]) {

j++;

} else {

k++;

}

}

return result;

}

const arr1 = [1, 2, 3, 4, 5];

const arr2 = [1, 2, 5, 7, 9];

const arr3 = [1, 3, 4, 5, 8];

const result = arraysIntersection(arr1, arr2, arr3);

console.log(result);

**2nd Solution**

function findDisjoint(nums1, nums2) {

const distinctNums1 = new Set(nums1);

const distinctNums2 = new Set(nums2);

const answer = [];

const distinctInNums1 = [];

const distinctInNums2 = [];

for (let num of distinctNums1) {

if (!distinctNums2.has(num)) {

distinctInNums1.push(num);

}

}

for (let num of distinctNums2) {

if (!distinctNums1.has(num)) {

distinctInNums2.push(num);

}

}

answer.push(distinctInNums1);

answer.push(distinctInNums2);

return answer;

}

const nums1 = [1, 2, 3];

const nums2 = [2, 4, 6];

const result = findDisjoint(nums1, nums2);

console.log(result);

**3rd Solution**

function transpose(matrix) {

const rows = matrix.length;

const cols = matrix[0].length;

const transposed = [];

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

const newRow = [];

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

newRow.push(matrix[i][j]);

}

transposed.push(newRow);

}

return transposed;

}

const matrix = [

[1, 2, 3],

[4, 5, 6],

[7, 8, 9]

];

const result = transpose(matrix);

console.log(result);

**4th Solution**

function arrayPairSum(nums) {

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

let sum = 0;

for (let i = 0; i < nums.length; i += 2) {

sum += nums[i];

}

return sum;

}

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

const result = arrayPairSum(nums);

console.log(result);

**5th 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);

**6th Solution**

function sortedSquares(nums) {

const result = [];

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

const square = nums[i] \*\* 2;

result.push(square);

}

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

return result;

}

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

const result = sortedSquares(nums);

console.log(result);

**7th Solution**

function maxCount(m, n, ops) {

for (const [ai, bi] of ops) {

m = Math.min(m, ai);

n = Math.min(n, bi);

}

return m \* n;

}

const m = 3;

const n = 3;

const ops = [[2, 2], [3, 3]];

const result = maxCount(m, n, ops);

console.log(result);

**8th Solution**

function shuffle(nums, n) {

const result = [];

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

result.push(nums[i]);

result.push(nums[i + n]);

}

return result;

}

const nums = [2, 5, 1, 3, 4, 7];

const n = 3;

const result = shuffle(nums, n);

console.log(result);