**Filter**

Write a function that removes all items that are not numbers from the array. The function should modify the existing array, not create a new one.

For example, if the input array contains values [1, 'a', 'b', 2], after processing, the array will contain only values [1, 2].

function filterNumbersFromArray(arr) {

// Write the code that goes here

}

var arr = [1, 'a', 'b', 2];

filterNumbersFromArray(arr);

// At this point, arr should have been modified in place

// and contain only 1 and 2.

for (var i = 0; i < arr.length; i++)

console.log(arr[i]);

**Array Search**

The program uses a data structure with an array that can contain items and other arrays. Write a function *numberOfItems* that recursively passes through all arrays and counts the number of occurrences of a given item. Keep in mind that arrays can be nested within each other.

For example, *numberOfItems(arr, 25)* and *numberOfItems(arr, "apple")*for the array below should both return 2.

var arr = [

25,

"apple",

["banana", "strawberry", "apple", 25]

];

function numberOfItems(arr, item) {

// Write the code that goes here

}

var arr = [

25,

"apple",

["banana", "strawberry", "apple", 25]

];

console.log(numberOfItems(arr, 25));

console.log(numberOfItems(arr, "apple"));

## Vectors

Write a function that takes an array of 3D vectors and returns the shortest one. Each vector is represented with an array that contains 3 elements (x, y and z). If multiple vectors have the same length, the function should return any one of them.

To determine the length of a vector use the formula: The length formula.

For example, for the array of 3D vectors [ [1, 1, 1], [2, 2, 2], [3, 3, 3] ] findShortest should return the first vector (array [1, 1, 1]) because it is the shortest.

function findShortest(vectors) {

**// Write the code that goes here**

}

var vectors = [[1, 1, 1], [2, 2, 2], [3, 3, 3]];

var shortest = findShortest(vectors);

console.log(shortest);

## Hobbies

Implement the findAllHobbyists function that takes a hobby, and an object consisting of peoples names mapped to their respective hobbies. The function should return an Array containing the names of the people (in any order) that enjoy the hobby.

For example, the following code should display the name 'Chad'.

var hobbies = {

"Steve": ['Fashion', 'Piano', 'Reading'],

"Patty": ['Drama', 'Magic', 'Pets'],

"Chad": ['Puzzles', 'Pets', 'Yoga']

};

console.log(findAllHobbyists('Yoga', hobbies));

function findAllHobbyists(hobby, hobbies) {

return [];

}

var hobbies = {

"Steve": ['Fashion', 'Piano', 'Reading'],

"Patty": ['Drama', 'Magic', 'Pets'],

"Chad": ['Puzzles', 'Pets', 'Yoga']

};

console.log(findAllHobbyists('Yoga', hobbies));

## Snapshot

Modify the implementation of the Snapshot class so that an array stored in the snapshot is not affected by modifications to either the original or restored array.

class Snapshot {

constructor(array) {

this.array = array;

}

restore() {

return this.array;

}

}

var array = [1, 2];

var snap = new Snapshot(array);

array[0] = 3;

array = snap.restore();

console.log(array.join()); //It should log "1,2"

array.push(4);

array = snap.restore();

console.log(array.join()); //It should log "1,2"