Array practice

1. Shuffle the array
Imagine the supplied array is like a pack of cards. Write a function to shuffle this array in to a completely new and entirely random order, much like shuffling a pack of cards.

```
const shuffle = (someArray) => {
    // your code
}

console.log(shuffle([10, 12, 15])) // [12, 15, 10]
```

2. Evenly mix two arrays
Write a function that receives two arrays, then mix the two arrays evenly –starting with the first item of the first array. The returned array should alternate between each array in turn, one element at a time. So the sequence would look like:

Array 1 first element, Array 2 first element, Array 1 second element, Array 2 second element… and so on.

The elements that make up the arrays are completely random, and the arrays can be of differing lengths.

```
const mix = (firstArray, secondArray) => {
    // your code
}

console.log(mix([1, 2, 3], [4, 5, 6]))
// [1, 4, 2, 5, 3, 6]
console.log(mix(["h", "a", "c"], [7, 4, 17, 10, 48]))
// ["h", 7, "a", 4, "c", 17, 10, 48]
```

3. Order the arrays by sum, ascending or descending
Given an array of arrays (ie. a multidimensional array), each containing only numeric values, order these arrays by the sum of the elements within.

```
const orderBySum = (someArray) => {
    // your code
}

console.log(orderBySum([[1,3], [4,2], [2,1]]))
// [[2,1], [1,3], [4,2]]
```

4. Find the odd one out
Find the odd one out in the provided array. All array elements are equal, except one. Your function must return this odd one out.

```
const findOddOneOut = (someArray) => {
    // your code
}

console.log(findOddOneOut(["a", "a", "b", "a"])) // b
```

5. Find the smallest and largest numbers
Deduce the smallest and the largest number from the supplied numeric array. Return an array containing these numbers, smallest number first.

```
const findSmallestAndLargestNumbers = (someArray) => {
    // your code
}

console.log(findSmallestAndLargestNumbers([14, 28, 3, 8, 2]))
// 2, 28
```

As with the other JavaScript array exercises in this list, it's good to consider how performant or efficient your solution may be. One such way may be to consider the time taken or complexity of your functionality. You can read more on this topic in my post on The Big O Notation.

6. Split the numeric array by odd/even
Write a function to separate a single array in to two separate arrays. The supplied array will only contain numeric values. Your function should output two arrays: one containing odd numbers, the other containing even numbers.

The returned arrays should be ordered appropriately, with distinct values only (no duplicates). The even array should be returned first.

```
const splitByOddAndEven = (someArray) => {
    // your code
}

console.log(splitByOddAndEven([2, 3, 7, 6, 2, 4, 9))
// [[2, 4, 6], [3, 7, 9]]
```

7. Return only unique values
In this JavaScript array exercise, the supplied array may contain duplicate values. Write a function to return only unique values — values that only occur exactly one time within the provided array.

```
const onlyUnique = (someArray) => {
    // your code
```

```
}

console.log(onlyUnique([1, 1, 2, 3, 3, 4, 4, 5)) // [2, 5]
```

8. Spell out the alphabet

The supplied array is a multidimensional array composed entirely of letters, broken down into sets of 3. The letters within each set of 3 are in random order; and the overall array pieces are in random order, also.

The array is composed in such a way that it can be sorted in to an alphabetic order. Your function should reorder all of these pieces to accomplish this.

```
const alphabetize = (someArray) => {
   // your code
}

const jumbledAlphabetically = [
   ["e", "d", "f"],
   ["a", "c", "b"],
   ["m", "o", "n"]
]
console.log(alphabetize(jumbledAlphabet))
// [["a", "b", "c"], ["d", "e", "f"], ["m", "n", "o"]]
```

9. Find the common words

Your function must accept 2 arrays, then return the common words from each. So, a new array containing the words which occur at least once in each of the supplied arrays must be returned back.

Words that are most common throughout both of the arrays should be returned first.

```
const findCommonWords = (firstArray, secondArray) => {
   // your code
}

const firstArray = ["dog", "cat", "parrot"]
const secondArray = ["lizard", "rat", "cat"]
console.log(findCommonWords(firstArray, secondArray)) // ["cat"]
```

To make this JavaScript array exercise more interesting, your function could also accept an array containing n arrays, and compare all of these arrays, instead of just the 2.

10. Does the array contain all of the elements?

Write a function to accept two arrays. Does the first array contain all elements represented in the second array?

```
const containsAllElements = (firstArray, secondArray) => {
```

```
    // your code
}

console.log(containsAllElements(["monday", "tuesday"], ["tuesday"])) // false
```

11. Shift all numeric values to the beginning

The supplied array will contain both numeric and non-numeric characters.

Return a new array where all numeric values are positioned at the beginning of the array, in ascending order. The non-numeric values must come after these, also ordered appropriately.

```
const sortTheArray = (someArray) => {
    // your code
}

console.log(sortTheArray(["b", 6, "a", "q", 7, 2]))
// [2, 6, 7, "a", "b", "q"]
```

12. Move the element

Given an array and two indexes, move the relevant element within the array to its new position.

```
const move = (someArray, firstIndex, secondIndex) => {
    // your code
}

console.log(move([4, 5, 7], 2, 1)) // [4, 7, 5]
```

13. Build a new array by indexes

From the first array, construct a new array based on the indexes supplied.

```
const buildArray = (someArray, indexes) => {
    // your code
}

console.log(buildArray(["mon", "tue", "wed", "thur", "fri"), [1, 3, 4])
// ["tue", "thur", "fri"]
```