```
const arr = [5, 5, 5, 2, 2, 2, 2, 2, 9, 4];
const counts = {};

for (const num of arr) {
  counts[num] = counts[num] ? counts[num] + 1 : 1;
}

console.log(counts);
console.log(counts[5], counts[2], counts[9], counts[4]);



const occurrences = [5, 5, 5, 2, 2, 2, 2, 2, 9, 4].reduce((acc, curr)=> {
  return acc[curr] ? ++acc[curr] : acc[curr] = 1, acc
}, {});

console.log(occurrences)
// program to count down numbers to 1
function countDown(number) {

    // display the number
    console.log(number);

    // decrease the number value
    const newNumber = number - 1;

    // base case
    if (newNumber > 0) {
        countDown(newNumber);
    }
}
```

```
countDown(4);



// program to find the factorial of a number
function factorial(x) {

    // if number is 0
    if (x === 0) {
        return 1;
    }

    // if number is positive
    else {
        return x * factorial(x - 1);
    }
}

const num = 3;

// calling factorial() if num is non-negative
if (num > 0) {
    let result = factorial(num);
    console.log(`The factorial of ${num} is ${result}`);
}




// program to display fibonacci sequence using recursion
function fibonacci(num) {
    if(num < 2) {
```

```
        return num;
    }
    else {
        return fibonacci(num-1) + fibonacci(num - 2);
    }
}

// take nth term input from the user
const nTerms = prompt('Enter the number of terms: ');

if(nTerms <=0) {
    console.log('Enter a positive integer.');
}
else {
    for(let i = 0; i < nTerms; i++) {
        console.log(fibonacci(i));
    }
}
```

```
// take input from the user
const number = parseInt(prompt('Enter a positive integer: '));

// checking if number is negative
```

```
if (number < 0) {
    console.log('Error! Factorial for negative number does not exist.');
}

// if number is 0
else if (number === 0) {
    console.log(`The factorial of ${number} is 1.`);
}

// if number is positive
else {
    let fact = 1;
    for (i = 1; i <= number; i++) {
        fact *= i;
    }
    console.log(`The factorial of ${number} is ${fact}.`);
}
```

```
// program to generate fibonacci series up to n terms

// take input from the user
const number = parseInt(prompt('Enter the number of terms: '));
let n1 = 0, n2 = 1, nextTerm;

console.log('Fibonacci Series:');

for (let i = 1; i <= number; i++) {
    console.log(n1);
```

```
    nextTerm = n1 + n2;
    n1 = n2;
    n2 = nextTerm;
}
```

```
// program to check an Armstrong number of three digits

let sum = 0;
const number = prompt('Enter a three-digit positive integer: ');

// create a temporary variable
let temp = number;
while (temp > 0) {
    // finding the one's digit
    let remainder = temp % 10;

    sum += remainder * remainder * remainder;

    // removing last digit from the number
    temp = parseInt(temp / 10); // convert float into integer
}
// check the condition
if (sum == number) {
    console.log(`${number} is an Armstrong number`);
}
else {
    console.log(`${number} is not an Armstrong number.`);
}
```

```
const number = parseInt(prompt("Enter a positive number: "));
let isPrime = true;

// check if number is equal to 1
if (number === 1) {
   console.log("1 is neither prime nor composite number.");
}

// check if number is greater than 1
else if (number > 1) {

   // looping through 2 to number-1
   for (let i = 2; i < number; i++) {
      if (number % i == 0) {
         isPrime = false;
         break;
      }
   }

   if (isPrime) {
      console.log(`${number} is a prime number`);
   } else {
      console.log(`${number} is a not prime number`);
   }
}
```

```
fahrenheit = celsius * 1.8 + 32
```

```
// program to reverse a string

function reverseString(str) {

    // empty string
    let newString = "";
    for (let i = str.length - 1; i >= 0; i--) {
        newString += str[i];
    }
    return newString;
}

// take input from the user
const string = prompt('Enter a string: ');

const result = reverseString(string);
console.log(result);
```

```
// program to reverse a string

function reverseString(str) {
```

```
    // return a new array of strings
    const arrayStrings = str.split("");

    // reverse the new created array elements
    const reverseArray = arrayStrings.reverse();

    // join all elements of the array into a string
    const joinArray = reverseArray.join("");

    // return the reversed string
    return joinArray;
}

// take input from the user
const string = prompt('Enter a string: ');

const result = reverseString(string);
console.log(result);



// program to check if the string is palindrome or not

function checkPalindrome(string) {

    // find the length of a string
    const len = string.length;

    // loop through half of the string
    for (let i = 0; i < len / 2; i++) {

        // check if first and last string are same
```

```
        if (string[i] !== string[len - 1 - i]) {
            return 'It is not a palindrome';
        }
    }
    return 'It is a palindrome';
}

// take input
const string = prompt('Enter a string: ');

// call the function
const value = checkPalindrome(string);

console.log(value);



// program to reverse a string

function reverseString(str) {

    // empty string
    let newString = "";
    for (let i = str.length - 1; i >= 0; i--) {
        newString += str[i];
    }
    return newString;
}

// take input from the user
const string = prompt('Enter a string: ');
```

```
const result = reverseString(string);
console.log(result);
```

```
<script>
	// Declare and initialize an Array
	var marks = [12, 25, 31, 23, 75, 81, 100];

	// Print Before sorting array
	console.log("Original Array");
	console.log(marks);

	// Sort elements using compare method
	marks.sort(function(a, b){return b - a});

	console.log("After sorting in Ascending order");

	// Print sorted Numeric array
	console.log(marks);
</script>
```

```
<script>
	// Sorting function
	function Numeric_sort(ar) {

		var i = 0, j;
```

```
            while (i < ar.length) {
                j = i + 1;
                while (j < ar.length) {

                    if (ar[j] < ar[i]) {
                        var temp = ar[i];
                        ar[i] = ar[j];
                        ar[j] = temp;
                    }
                    j++;
                }
                i++;
            }
        }

        // Original Array
        var arr = [1, 15, 10, 45, 27, 100];

        // Print Before sorting array
        console.log("Original Array");
        console.log(arr);

        // Function call
        Numeric_sort(arr);

        console.log("Sorted Array");

        // Print sorted Numeric array
        console.log(arr);
</script>
```

Coding challenge #5: Calculate the sum of numbers from 1 to 10

```
var sum = 0;

for(var i = 1; i <= 10; i++)
{
    sum += i;
}

println(sum);
```

```
function fahrenheitToCelsius(n)
{
    return (n - 32) / 1.8;
}

var r = fahrenheitToCelsius(68);
println(r);
```

JavaScript Coding Challenges for Beginners

Mastering these coding challenges may not get you a job at Google... but you'll be one step closer to building your own JavaScript game at codeguppy.com

---

YouTube Channel

After you read this article, please remember to check out also the “Coding Adventures” YouTube channel. You’ll find there a series of fun and engaging coding lessons and cool projects.

Channel main page Coding lessons playlist Coding projects playlist

---

The coding challenges in this article are intended for code newbies, therefore the solutions are implemented using only simple / classical programming elements. Each solution is acompanied by an online link that helps you quickly run it in a code playground.

If you prefer to open all solutions in a single, unified coding playgroud please open the 50 Coding Challenges project.

The code is making use of the codeguppy specific function `println()` to print the results. If you want to run these solutions outside CodeGuppy, just replace `println()` with `console.log()` then run them using your browser console tool or node.js.

Coding challenge #1: Print numbers from 1 to 10

Edit in coding playground

```
for(let i = 1; i <= 10; i++)
{
    println(i);
}
```

Coding challenge #2: Print the odd numbers less than 100

Edit in coding playground

```
for(let i = 1; i <= 100; i += 2)
{
    println(i);
}
```

Coding challenge #3: Print the multiplication table with 7

Edit in coding playground

```
for(let i = 1; i <= 10; i++)
{
    let row = "7 * " + i + " = " + 7 * i;
    println(row);
}
```

Coding challenge #4: Print all the multiplication tables with numbers from 1 to 10

Edit in coding playground

```
for(let i = 1; i <= 10; i++)
{
    printTable(i);
    println("");
}

function printTable(n)
{
```

```
    for(let i = 1; i <= 10; i++)
    {
        let row = n + " * " + i + " = " + n * i;
        println(row);
    }
}
```

Coding challenge #5: Calculate the sum of numbers from 1 to 10

Edit in coding playground

```
let sum = 0;

for(let i = 1; i <= 10; i++)
{
    sum += i;
}

println(sum);
```

Coding challenge #6: Calculate 10!

Edit in coding playground

```
let prod = 1;

for(let i = 1; i <= 10; i++)
{
    prod *= i;
}

println(prod);
```

Coding challenge #7: Calculate the sum of odd numbers greater than 10 and less than 30

Edit in coding playground

```
let sum = 0;

for(let i = 11; i <= 30; i += 2)
{
    sum += i;
}

println(sum);
```

Coding challenge #8: Create a function that will convert from Celsius to Fahrenheit

Edit in coding playground

```
function celsiusToFahrenheit(n)
{
    return n * 1.8 + 32;
}

let r = celsiusToFahrenheit(20);
println(r);
```

Coding challenge #9: Create a function that will convert from Fahrenheit to Celsius

Edit in coding playground

```
function fahrenheitToCelsius(n)
{
    return (n - 32) / 1.8;
}

let r = fahrenheitToCelsius(68);
println(r);
```

Coding challenge #10: Calculate the sum of numbers in an array of numbers

Edit in coding playground

```
function sumArray(ar)
{
    let sum = 0;

    for(let i = 0; i < ar.length; i++)
    {
        sum += ar[i];
    }

    return sum;
}

let ar = [2, 3, -1, 5, 7, 9, 10, 15, 95];
let sum = sumArray(ar);
```

```
println(sum);
```

Coding challenge #11: Calculate the average of the numbers in an array of numbers

Edit in coding playground

```
function averageArray(ar)
{
    let n = ar.length;
    let sum = 0;

    for(let i = 0; i < n; i++)
    {
        sum += ar[i];
    }

    return sum / n;
}

let ar = [1, 3, 9, 15, 90];
let avg = averageArray(ar);

println("Average: ", avg);
```

Coding challenge #12: Create a function that receives an array of numbers and returns an array containing only the positive numbers

Solution 1

Edit in coding playground

```
function getPositives(ar)
{
    let ar2 = [];

    for(let i = 0; i < ar.length; i++)
    {
        let el = ar[i];

        if (el >= 0)
        {
            ar2.push(el);
        }
    }

    return ar2;
}
```

```
let ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
let ar2 = getPositives(ar);

println(ar2);
```

Coding challenge #12: Create a function that receives an array of numbers and returns an array containing only the positive numbers

Solution 2

Edit in coding playground

```
function getPositives(ar)
{
    let ar2 = [];

    for(let el of ar)
    {
        if (el >= 0)
        {
            ar2.push(el);
        }
    }

    return ar2;
}

let ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
let ar2 = getPositives(ar);

println(ar2);
```

Coding challenge #12: Create a function that receives an array of numbers and returns an array containing only the positive numbers

Solution 3

Edit in coding playground

```
function getPositives(ar)
{
    return ar.filter(el => el >= 0);
}

let ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
```

```
let ar2 = getPositives(ar);
println(ar2);
```

Coding challenge #13: Find the maximum number in an array of numbers

Edit in coding playground

```
function findMax(ar)
{
    let max = ar[0];

    for(let i = 0; i < ar.length; i++)
    {
        if (ar[i] > max)
        {
            max = ar[i];
        }
    }

    return max;
}

let ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
let max = findMax(ar);
println("Max: ", max);
```

Coding challenge #14: Print the first 10 Fibonacci numbers without recursion

Edit in coding playground

```
let f0 = 0;
println(f0);

let f1 = 1;
println(f1);

for(let i = 2; i < 10; i++)
{
    let fi = f1 + f0;
    println(fi);

    f0 = f1;
    f1 = fi;
}
```

Coding challenge #15: Create a function that will find the nth Fibonacci number using recursion

Edit in coding playground

```
function findFibonacci(n)
{
    if (n == 0)
        return 0;

    if (n == 1)
        return 1;

    return findFibonacci(n - 1) + findFibonacci(n - 2);
}

let n = findFibonacci(10);
println(n);
```

Coding challenge #16: Create a function that will return a Boolean specifying if a number is prime

Edit in coding playground

```
function isPrime(n)
{
    if (n < 2)
        return false;

    if (n == 2)
        return true;

    let maxDiv = Math.sqrt(n);

    for(let i = 2; i <= maxDiv; i++)
    {
        if (n % i == 0)
        {
            return false;
        }
    }

    return true;
}

println(2, " is prime? ", isPrime(2));
println(3, " is prime? ", isPrime(3));
println(4, " is prime? ", isPrime(4));
println(5, " is prime? ", isPrime(5));
println(9, " is prime? ", isPrime(9));
```

Coding challenge #17: Calculate the sum of digits of a positive integer number

Edit in coding playground

```
function sumDigits(n)
{
    let s = n.toString();
    let sum = 0;

    for(let char of s)
    {
        let digit = parseInt(char);
        sum += digit;
    }

    return sum;
}

let sum = sumDigits(1235231);
println("Sum: ", sum);
```

Coding challenge #18: Print the first 100 prime numbers

Edit in coding playground

```
printPrimes(100);

// Function prints the first nPrimes numbers
function printPrimes(nPrimes)
{
    let n = 0;
    let i = 2;

    while(n < nPrimes)
    {
        if (isPrime(i))
        {
            println(n, " --> ", i);
            n++;
        }

        i++;
    }
}


// Returns true if a number is prime
function isPrime(n)
{
    if (n < 2)
        return false;

    if (n == 2)
```

```
        return true;

    let maxDiv = Math.sqrt(n);

    for(let i = 2; i <= maxDiv; i++)
    {
        if (n % i == 0)
        {
            return false;
        }
    }

    return true;
}
```

Coding challenge #19: Create a function that will return in an array the first "nPrimes" prime numbers greater than a particular number "startAt"

Edit in coding playground

```
println(getPrimes(10, 100));

function getPrimes(nPrimes, startAt)
{
    let ar = [];

    let i = startAt;

    while(ar.length < nPrimes)
    {
        if (isPrime(i))
        {
            ar.push(i);
        }

        i++;
    }

    return ar;
}

// Returns true if a number is prime
function isPrime(n)
{
    if (n < 2)
        return false;

    if (n == 2)
        return true;

    let maxDiv = Math.sqrt(n);

    for(let i = 2; i <= maxDiv; i++)
    {
```

```
        if (n % i == 0)
        {
            return false;
        }
    }

    return true;
}
```

Coding challenge #20: Rotate an array to the left 1 position

Edit in coding playground

```
let ar = [1, 2, 3];
rotateLeft(ar);
println(ar);

function rotateLeft(ar)
{
    let first = ar.shift();
    ar.push(first);
}
```

Coding challenge #21: Rotate an array to the right 1 position

Edit in coding playground

```
let ar = [1, 2, 3];
rotateRight(ar);
println(ar);

function rotateRight(ar)
{
    let last = ar.pop();
    ar.unshift(last);
}
```

Coding challenge #22: Reverse an array

Edit in coding playground

```
let ar = [1, 2, 3];
let ar2 = reverseArray(ar);
println(ar2);
```

```
function reverseArray(ar)
{
    let ar2 = [];

    for(let i = ar.length - 1; i >= 0; i--)
    {
        ar2.push(ar[i]);
    }

    return ar2;
}
```

Coding challenge #23: Reverse a string

Edit in coding playground

```
let s = reverseString("JavaScript");
println(s);

function reverseString(s)
{
    let s2 = "";

    for(let i = s.length - 1; i >= 0; i--)
    {
        let char = s[i];
        s2 += char;
    }

    return s2;
}
```

Coding challenge #24: Create a function that will merge two arrays and return the result as a new array

Edit in coding playground

```
let ar1 = [1, 2, 3];
let ar2 = [4, 5, 6];

let ar = mergeArrays(ar1, ar2);
println(ar);

function mergeArrays(ar1, ar2)
{
   let ar = [];

   for(let el of ar1)
   {
       ar.push(el);
   }
```

```
    for(let el of ar2)
    {
        ar.push(el);
    }

    return ar;
}
```

Coding challenge #25: Create a function that will receive two arrays of numbers as arguments and return an array composed of all the numbers that are either in the first array or second array but not in both

Edit in coding playground

```
let ar1 = [1, 2, 3, 10, 5, 3, 14];
let ar2 = [1, 4, 5, 6, 14];

let ar = mergeExclusive(ar1, ar2);
println(ar);

function mergeExclusive(ar1, ar2)
{
    let ar = [];

    for(let el of ar1)
    {
        if (!ar2.includes(el))
        {
            ar.push(el);
        }
    }

    for(let el of ar2)
    {
        if (!ar1.includes(el))
        {
            ar.push(el);
        }
    }

    return ar;
}
```

Coding challenge #26: Create a function that will receive two arrays and will return an array with elements that are in the first array but not in the second

Edit in coding playground

```
let ar1 = [1, 2, 3, 10, 5, 3, 14];
let ar2 = [-1, 4, 5, 6, 14];

let ar = mergeLeft(ar1, ar2);
println(ar);

function mergeLeft(ar1, ar2)
{
    let ar = [];

    for(let el of ar1)
    {
        if (!ar2.includes(el))
        {
            ar.push(el);
        }
    }

    return ar;
}
```

Coding challenge #27: Create a function that will receive an array of numbers as argument and will return a new array with distinct elements

Solution 1

Edit in coding playground

```
let ar = getDistinctElements([1, 2, 3, 6, -1, 2, 9, 7, 10, -1, 100]);
println(ar);

function getDistinctElements(ar)
{
    let ar2 = [];

    for(let i = 0; i < ar.length; i++)
    {
        if (!isInArray(ar[i], ar2))
        {
            ar2.push(ar[i]);
        }
    }

    return ar2;
}

function isInArray(n, ar)
{
    for(let i = 0; i < ar.length; i++)
    {
        if (ar[i] === n)
            return true;
    }
```

```
    return false;
}
```

Coding challenge #27: Create a function that will receive an array of numbers as argument and will return a new array with distinct elements

Solution 2

Edit in coding playground

```
let ar = getDistinctElements([1, 2, 3, 6, -1, 2, 9, 7, 10, -1, 100]);
println(ar);

function getDistinctElements(ar)
{
    let ar2 = [];

    let lastIndex = ar.length - 1;

    for(let i = 0; i <= lastIndex; i++)
    {
        if (!isInArray(ar[i], ar, i + 1, lastIndex))
        {
            ar2.push(ar[i]);
        }
    }

    return ar2;
}

function isInArray(n, ar, fromIndex, toIndex)
{
    for(let i = fromIndex; i <= toIndex; i++)
    {
        if (ar[i] === n)
            return true;
    }

    return false;
}
```

Coding challenge #28: Calculate the sum of first 100 prime numbers

Edit in coding playground

```
let n = 10;
println("Sum of first ", n, " primes is ", sumPrimes(10));
```

```
function sumPrimes(n)
{
    let foundPrimes = 0;
    let i = 2;
    let sum = 0;

    while(foundPrimes < n)
    {
        if (isPrime(i))
        {
            foundPrimes++;
            sum += i;
        }

        i++;
    }

    return sum;
}

// Returns true if number n is prime
function isPrime(n)
{
    if (n < 2)
        return false;

    if (n == 2)
        return true;

    let maxDiv = Math.sqrt(n);

    for(let i = 2; i <= maxDiv; i++)
    {
        if (n % i === 0)
        {
            return false;
        }
    }

    return true;
}
```

Coding challenge #29: Print the distance between the first 100 prime numbers

Edit in coding playground

```
printDistances(100);

// Print distances between the first n prime numbers
function printDistances(n)
{
    let lastPrime = 2;
    let i = lastPrime + 1;
    let foundPrimes = 1;
```

```
    while(foundPrimes < n)
    {
        if (isPrime(i))
        {
            println(i - lastPrime, "\t", i, " - ", lastPrime);

            foundPrimes++;
            lastPrime = i;
        }

        i++;
    }
}

// Returns true if number n is prime
function isPrime(n)
{
    if (n < 2)
        return false;

    if (n == 2)
        return true;

    let maxDiv = Math.sqrt(n);

    for(let i = 2; i <= maxDiv; i++)
    {
        if (n % i === 0)
        {
            return false;
        }
    }

    return true;
}
```

Coding challenge #30-a: Create a function that will add two positive numbers of indefinite size. The numbers are received as strings and the result should be also provided as string.

Solution 1

Edit in coding playground

```
let n1 = "2909034221912398942349";
let n2 = "1290923909029309499";
let sum = add(n1, n2);

println(n1, "\n", n2, "\n", sum);

function add(sNumber1, sNumber2)
{
```

```
    let s = "";
    let carry = 0;

    let maxSize = Math.max(sNumber1.length, sNumber2.length);

    for(let i = 0; i < maxSize; i++)
    {
        let digit1 = digitFromRight(sNumber1, i);
        let digit2 = digitFromRight(sNumber2, i);

        let sum = digit1 + digit2 + carry;
        let digitSum = sum % 10;
        carry = sum >= 10 ? 1 : 0;

        s = digitSum.toString() + s;
    }

    if (carry > 0)
        s = carry + s;

    return s;
}

function digitFromRight(s, digitNo)
{
    if (digitNo >= s.length)
        return 0;

    let char = s[ s.length - 1 - digitNo ];
    return parseInt(char);
}
```

Coding challenge #30-b: Create a function that will add two positive numbers of indefinite size. The numbers are received as strings and the result should be also provided as string.

Solution 2

Edit in coding playground

```
let n1 = "2909034221912398942349";
let n2 = "1290923909029309499";
let sum = add(n1, n2);

println(n1);
println(n2);
println(sum);

function add(sNumber1, sNumber2)
{
    let maxSize = Math.max(sNumber1.length, sNumber2.length);

    let s1 = sNumber1.padStart(maxSize, "0");
```

```
    let s2 = sNumber2.padStart(maxSize, "0");

    let s = "";
    let carry = 0;

    for(let i = maxSize - 1; i >= 0; i--)
    {
        let digit1 = parseInt(s1[i]);
        let digit2 = parseInt(s2[i]);

        let sum = digit1 + digit2 + carry;
        let digitSum = sum % 10;
        carry = sum >= 10 ? 1 : 0;

        s = digitSum.toString() + s;
    }

    if (carry > 0)
        s = carry + s;

    return s;
}
```

Coding challenge #31a. Create a function that will return the number of words in a text

Edit in coding playground

```
// Solution 1

function countWords(text)
{
    let wasSeparator = true;
    let words = 0;

    for(let c of text)
    {
        // if current character is separator then advance and
        // set that the previous character was separator
        if (isSeparator(c))
        {
            wasSeparator = true;
            continue;
        }

        // if current character is not separator
        // ... but if previous was separator...
        if (wasSeparator)
        {
            words++;
            wasSeparator = false;
        }
    }

    return words;
}
```

```
function isSeparator(c)
{
    let separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?"];
    return separators.includes(c);
}

println(countWords(""));
println(countWords("            "));
println(countWords("JavaScript!!!   "));
println(countWords("     JavaScript"));
println(countWords("    JavaScript is cool      "));
println(countWords("I like to learn JavaScript with codeguppy"));
```

Coding challenge #31b. Create a function that will return the number of words in a text

Edit in coding playground

```
// Solution 2

function countWords(text)
{
    let words = 0;

    if (text.length > 0 && !isSeparator(text[0]))
        words++;

    for(let i = 1; i < text.length; i++)
    {
        let currChr = text[i];
        let prevChr = text[i - 1];

        if (!isSeparator(currChr) && isSeparator(prevChr))
        {
            words++;
        }
    }

    return words;
}

function isSeparator(c)
{
    let separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?"];
    return separators.includes(c);
}

println(countWords(""));
println(countWords("            "));
println(countWords("JavaScript!!!   "));
println(countWords("     JavaScript"));
println(countWords("    JavaScript is cool      "));
println(countWords("I like to learn JavaScript with codeguppy"));
```

Coding challenge #32. Create a function that will capitalize the first letter of each word in a text

Edit in coding playground

```
println(captializeWords("Create a function that will capitalize the first
letter of each word in a text"));

function captializeWords(text)
{
    let text2 = "";

    for(let i = 0; i < text.length; i++)
    {
        let currChr = text[i];
        let prevChr = i > 0 ? text[i - 1] : " ";

        if (!isSeparator(currChr) && isSeparator(prevChr))
        {
            currChr = currChr.toUpperCase();
        }

        text2 += currChr;
    }

    return text2;
}

function isSeparator(c)
{
    let separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?"];
    return separators.includes(c);
}
```

Coding challenge #33. Calculate the sum of numbers received in a comma delimited string

Edit in coding playground

```
println(sumCSV("1.5, 2.3, 3.1, 4, 5.5, 6, 7, 8, 9, 10.9"));

function sumCSV(s)
{
    let ar = s.split(",");

    let sum = 0;

    for(let n of ar)
    {
        sum += parseFloat(n);
    }

    return sum;
}
```

Coding challenge #34. Create a function that will return an array with words inside a text

Edit in coding playground

```
let text = "Create a function, that will return an array (of string), with
the words inside the text";

println(getWords(text));

function getWords(text)
{
    let startWord = -1;
    let ar = [];

    for(let i = 0; i <= text.length; i++)
    {
        let c = i < text.length ? text[i] : " ";

        if (!isSeparator(c) && startWord < 0)
        {
            startWord = i;
        }

        if (isSeparator(c) && startWord >= 0)
        {
            let word = text.substring(startWord, i);
            ar.push(word);

            startWord = -1;
        }
    }

    return ar;
}

function isSeparator(c)
{
    let separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?", "(",
")"];
    return separators.includes(c);
}
```

Coding challenge #35. Create a function to convert a CSV text to a “bi-dimensional” array

Edit in coding playground

```
let data = "John;Smith;954-000-0000\nMich;Tiger;305-000-
0000\nMonique;Vasquez;103-000-0000";
```

```
let ar = csvToArray(data);
println(JSON.stringify(ar));

function csvToArray(data)
{
    let arLines = data.split("\n");

    for(let i = 0; i < arLines.length; i++)
    {
        let arLine = arLines[i].split(";");
        arLines[i] = arLine;
    }

    return arLines;
}
```

Coding challenge #36. Create a function that converts a string to an array of characters

Edit in coding playground

```
println(getChars("I like JavaScript"));

function getChars(s)
{
    return Array.from(s);
}
```

Coding challenge #37. Create a function that will convert a string in an array containing the ASCII codes of each character

Edit in coding playground

```
println(getCharCodes("I like JavaScript"));

function getCharCodes(s)
{
    let ar = [];

    for(let i = 0; i < s.length; i++)
    {
        let code = s.charCodeAt(i);
        ar.push(code);
    }

    return ar;
}
```

Coding challenge #38. Create a function that will convert an array containing ASCII codes in a string

Edit in coding playground

```
println(codesToString([73,32,108,105,107,101,32,74,97,118,97,83,99,114,105,
112,116]));

function codesToString(ar)
{
    return String.fromCharCode(...ar);
}
```

Coding challenge #39. Implement the Caesar cypher

Edit in coding playground

```
let text = "I LOVE JAVASCRIPT";
let textEnc = encrypt(text, 13);
let textDec = decrypt(textEnc, 13);

println(text);
println(textEnc);
println(textDec);

// Decrypt a message by using the same encrypt function
// ... but using the inverse of the key (e.g. rotate in the other
direction)
function decrypt(msg, key)
{
    return encrypt(msg, key * -1);
}

// Function will implement Caesar Cipher to
// encrypt / decrypt the msg by shifting the letters
// of the message acording to the key
function encrypt(msg, key)
{
    let encMsg = "";

    for(let i = 0; i < msg.length; i++)
    {
        let code = msg.charCodeAt(i);

        // Encrypt only letters in 'A' ... 'Z' interval
        if (code >= 65 && code <= 65 + 26 - 1)
        {
            code -= 65;
            code = mod(code + key, 26);
            code += 65;
        }
```

```
        encMsg += String.fromCharCode(code);
    }

    return encMsg;
}


// Modulo function: n mod p
function mod(n, p)
{
    if ( n < 0 )
        n = p - Math.abs(n) % p;

    return n % p;
}
```

Coding challenge #40. Implement the bubble sort algorithm for an array of numbers

Edit in coding playground

```
let ar = [23, 1000, 1, -1, 8, 3];
println(ar);
bubbleSort(ar);
println(ar);

function bubbleSort(ar)
{
    let shouldSort = true;
    let length = ar.length;

    while(shouldSort)
    {
        shouldSort = false;
        length--;

        for(let i = 0; i < length; i++)
        {
            let a = ar[i];
            if ( a > ar[i+1] )
            {
                ar[i] = ar[i+1];
                ar[i+1] = a;
                shouldSort = true;
            }
        }
    }
}
```

Coding challenge #41. Create a function to calculate the distance between two points defined by their x, y coordinates

Edit in coding playground

```
println(getDistance(100, 100, 400, 300));

function getDistance(x1, y1, x2, y2)
{
    let l1 = x2 - x1;
    let l2 = y2 - y1;

    return Math.sqrt(l1 * l1 + l2 * l2);
}
```

Coding challenge #42. Create a function that will return a Boolean value indicating if two circles defined by center coordinates and radius are intersecting

Edit in coding playground

```
println(collisionCircleCircle(200, 200, 100, 300, 300, 50));

function collisionCircleCircle(circle1X, circle1Y, circle1R, circle2X,
circle2Y, circle2R)
{
    return getDistance(circle1X, circle1Y, circle2X, circle2Y) <= circle1R
+ circle2R;
}

// Calculate the distance between the two specified points
function getDistance(x1, y1, x2, y2)
{
    let l1 = x2 - x1;
    let l2 = y2 - y1;

    return Math.sqrt(l1 * l1 + l2 * l2);
}
```

Coding challenge 43. Create a function that will receive a bi-dimensional array as argument and a number and will extract as a unidimensional array the column specified by the number

Edit in coding playground

```
let ar = [ ["John", 120],
           ["Jane", 115],
           ["Thomas", 123],
           ["Mel", 112],
           ["Charley", 122]
         ];

let numbers = extractCol(ar, 1);
println(numbers);

function extractCol(ar, colNo)
{
    let arCol = [];

    for(let i = 0; i < ar.length; i++)
    {
        arCol.push(ar[i][colNo]);
    }

    return arCol;
}
```

Coding challenge #44. Create a function that will convert a string containing a binary number into a number

Edit in coding playground

```
println(binaryToNumber("11111111"));

function binaryToNumber(sBinary)
{
    return parseInt(sBinary, 2);
}
```

Coding challenge #45. Create a function to calculate the sum of all the numbers in a jagged array (array contains numbers or other arrays of numbers on an unlimited number of levels)

Edit in coding playground

```
let ar = [1, 2, [15, [23], [5, 12]], [100]];

println(sumArray(ar));

function sumArray(ar)
{
```

```
    let sum = 0;

    for(let el of ar)
    {
        if (Array.isArray(el))
        {
            el = sumArray(el);
        }

        sum += el;
    }

    return sum;
}
```

Coding challenge #46-a. Find the maximum number in a jagged array of numbers or array of numbers

Edit in coding playground

```
// Solution 1

let ar = [2, 4, 10, [12, 4, [100, 99], 4], [3, 2, 99], 0];

let max = findMax(ar);
println("Max  = ", max);

// Use recursion to find the maximum numeric value in an array of arrays
function findMax(ar)
{
    let max = -Infinity;

    // Cycle through all the elements of the array
    for(let i = 0; i < ar.length; i++)
    {
        let el = ar[i];

        // If an element is of type array then invoke the same function
        // to find out the maximum element of that subarray
        if ( Array.isArray(el) )
        {
            el = findMax( el );
        }

        if ( el > max )
        {
            max = el;
        }
    }

    return max;
}
```

Coding challenge #47. Deep copy a jagged array with numbers or other arrays in a new array

Edit in coding playground

```
let ar1 = [2, 4, 10, [12, 4, [100, 99], 4], [3, 2, 99], 0];
let ar2 = copyArray(ar1);

println(ar2);

function copyArray(ar)
{
    let ar2 = [];

    for(let el of ar)
    {
        if (Array.isArray(el))
        {
            el = copyArray(el);
        }

        ar2.push(el);
    }

    return ar2;
}
```