50 Coding Challenges... in 50 days or less
About these coding challenges

• These coding challenges are intended for code newbies that want to practice JavaScript
• The challenges are designed to be solved only with basic JavaScript language constructs
• Although intended for beginners, advanced users can have fun too
• Try to solve at least 1 coding challenge per day
• Any solution that you find is a good solution… we are not looking for the perfect solution
• For convenience, when you work on these challenges, you can use an online coding playground, such as the one from https://codeguppy.com
• Feel free to share this booklet with your friends and invite them to complete the coding challenges
• This booklet contains also all the solutions to these challenges
• These challenges are great as coding exercises for CS classrooms!
• Have fun!

This booklet is completely free. Feel free to share it with your friends, post it on your website or share it on your social network.
50 coding challenges – Part I

1. Print numbers from 1 to 10
2. Print the odd numbers less than 100
3. Print the multiplication table with 7
4. Print all the multiplication tables with numbers from 1 to 10
5. Calculate the sum of numbers from 1 to 10
6. Calculate 10!
7. Calculate the sum of even numbers greater than 10 and less than 30
8. Create a function that will convert from Celsius to Fahrenheit
9. Create a function that will convert from Fahrenheit to Celsius
10. Calculate the sum of numbers in an array of numbers
11. Calculate the average of the numbers in an array of numbers
12. Create a function that receives an array of numbers as argument and returns an array containing only the positive numbers
13. Find the maximum number in an array of numbers
14. Print the first 10 Fibonacci numbers without recursion
15. Create a function that will find the $n^{th}$ Fibonacci number using recursion
16. Create a function that will return a Boolean specifying if a number is prime
17. Calculate the sum of digits of a positive integer number
18. Print the first 100 prime numbers
19. Create a function that will return in an array the first “p” prime numbers greater than “n”
20. Rotate an array to the left 1 position
21. Rotate an array to the right 1 position
22. Reverse an array
23. Reverse a string
24. Create a function that will merge two arrays and return the result as a new array
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
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
50 coding challenges – Part II

27. Create a function that will receive an array of numbers as argument and will return a new array with distinct elements
28. Calculate the sum of first 100 prime numbers and return them in an array
29. Print the distance between the first 100 prime numbers
30. 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.
31. Create a function that will return the number of words in a text
32. Create a function that will capitalize the first letter of each word in a text
33. Calculate the sum of numbers received in a comma delimited string
34. Create a function that returns an array with words inside a text.
35. Create a function to convert a CSV text to a “bi-dimensional” array
36. Create a function that converts a string to an array of characters
37. Create a function that will convert a string in an array containing the ASCII codes of each character
38. Create a function that will convert an array containing ASCII codes in a string
39. Implement the Caesar cypher
40. Implement the bubble sort algorithm for an array of numbers
41. Create a function to calculate the distance between two points defined by their x, y coordinates
42. Create a function that will return a Boolean value indicating if two circles defined by center coordinates and radius are intersecting
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
44. Create a function that will convert a string containing a binary number into a number
45. Create a function to calculate the sum of all the numbers in a jagged array (contains numbers or other arrays of numbers on an unlimited number of levels)
46. Find the maximum number in a jagged array of numbers or array of numbers
47. Deep copy a jagged array with numbers or other arrays in a new array
48. Create a function to return the longest word in a string
49. Shuffle an array of strings
50. Create a function that will receive n as argument and return an array of n random numbers from 1 to n. The numbers should be unique inside the array.
51. Find the frequency of letters inside a string. Return the result as an array of arrays. Each subarray has 2 elements: letter and number of occurrences.
52. Calculate Fibonacci(500) with high precision (all digits)
53. Calculate 70! with high precision (all digits)
The source code of the solutions presented in this booklet is available online at:

https://codeguppy.com/code.html?t=coding_challenges

You can type-in the full link above, or just go to codeguppy.com and browse to locate the “Coding Challenges” project.
Coding challenge #1: Print numbers from 1 to 10

```javascript
// Coding challenge #1: Print numbers from 1 to 10

for(var i = 1; i <= 10; i++) {
    println(i);
}
```
Coding challenge #2

Print the odd numbers less than 100

```javascript
for(var i = 1; i <= 100; i += 2)
{
    println(i);
}
```
Coding challenge #3

Print the multiplication table with 7

```javascript
for (var i = 1; i <= 10; i++) {
    var row = "7 x " + i + " = " + 7 * i;
    println(row);
}
```
Coding challenge #4

Print the multiplication tables with numbers from 1 to 10

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

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

function printTable(n) {
    for (var i = 1; i <= 10; i++) {
        var row = n + " * " + i + " = " + n * i;
        println(row);
    }
}
```
Coding challenge #5

Calculate the sum of numbers from 1 to 10

```javascript
// 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);
```
Coding challenge #6

Calculate 10!

Reminder \( n! = 1 \times 2 \times \ldots \times n \)
Coding challenge #7

Calculate the sum of odd numbers greater than 10 and less or equal than 30

```javascript
// Coding challenge #7: Calculate the sum of even numbers greater than 10 and less or equal than 30
var sum = 0;
for(var i = 11; i <= 30; i += 2) {
    sum += i;
}
println(sum);
```
Coding challenge #8

Create a function that will convert from Celsius to Fahrenheit

Reminder: \( C = \frac{F - 32}{1.8} \)
Coding challenge #9

Create a function that will convert from Fahrenheit to Celsius

Reminder: $C = \frac{F - 32}{1.8}$
Coding challenge #10

Calculate the sum of numbers in an array of numbers.

Example array: [2, 3, -1, 5, 7, 9, 10, 15, 95]

Expected output: 145

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

function sumArray(arr) {
    var sum = 0;
    for (var i = 0; i < arr.length; i++) {
        sum += arr[i];
    }
    return sum;
}

var ar = [2, 3, -1, 5, 7, 9, 10, 15, 95];
var sum = sumArray(ar);
println(sum);
```
Coding challenge #11

Calculate the average of the numbers in an array of numbers

Example array: [1, 3, 9, 15, 90]

Expected output: 23.6
Coding challenge #12a

Create a function that receives an array of numbers and returns an array containing only the positive numbers.

Requirement: Use a “for” loop
Coding challenge #12b

Create a function that receives an array of numbers and returns an array containing only the positive numbers.

Requirement: Use a “for … of” loop
Coding challenge #12c

Create a function that receives an array of numbers and returns an array containing only the positive numbers.

Requirement: Use .filter() array method
Coding challenge #13

Find the maximum number in an array of numbers

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

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

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

  return max;
}

var ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
var max = findMax(ar);
printin("Max: ", max);
```
Coding challenge #14

Print the first 10 Fibonacci numbers without using recursion.

Reminder:

F(0) = 0
F(1) = 1
F(n) = F(n-1) + F(n-2)
Coding challenge #15

Create a function that will find the nth Fibonacci number using recursion.

Reminder:

\[
F(0) = 0 \\
F(1) = 1 \\
F(n) = F(n-1) + F(n-2)
\]
Coding challenge #16

Create a function that will return a Boolean specifying if a number is prime.
Coding challenge #17

Calculate the sum of digits of a positive integer number

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

function sumDigits(n)
{
    var s = n.toString();
    var sum = 0;
    for(var char of s)
    {
        var digit = parseInt(char);
        sum += digit;
    }
    return sum;
}

var sum = sumDigits(1235231);
println("Sum: ", sum);
```
Coding challenge #18: Print the first 100 prime numbers

```javascript
// Coding challenge #18: Print the first 100 prime numbers

printPrimes(100);

// Function prints the first nPrimes numbers
function printPrimes(nPrimes)
{
    var n = 0;
    var 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;
    var maxDiv = Math.sqrt(n);
    for(var 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 number "startAt"
Coding challenge #20

Rotate an array to the left 1 position

```javascript
// Coding challenge #20: Rotate an array to the left 1 position
var ar = [1, 2, 3];
rotateLeft(ar);
println(ar);

function rotateLeft(ar)
{
    var first = ar.shift();
    ar.push(first);
}
```
Coding challenge #21

Rotate an array to the right 1 position

```javascript
var ar = [1, 2, 3];
rotateRight(ar);
printIn(ar);

function rotateRight(ar)
{
    var last = ar.pop();
ar.unshift(last);
}
```
Coding challenge #22

Reverse an array

```javascript
// Coding challenge #22: Reverse an array

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

function reverseArray(ar) {
    var ar2 = [];
    for(var i = ar.length - 1; i >= 0; i--)
    {
        ar2.push(ar[i]);
    }
    return ar2;
}
```
Coding challenge #23

Reverse a string

```javascript
// Coding challenge #23: Reverse a string

var s = reverseString("JavaScript");
println(s);

function reverseString(s)
{
    var s2 = "";
    for(var i = s.length - 1; i >= 0; i--)
    {
        var 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.

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

var ar1 = [1, 2, 3];
var ar2 = [4, 5, 6];
var ar = mergeArrays(ar1, ar2);
println(ar);

function mergeArrays(ar1, ar2)
{
    var 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.

```javascript
// 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

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

var ar = mergeExclusive(ar1, ar2);
printl(ar);

function mergeExclusive(ar1, ar2) {
    var 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.

```javascript
function mergeLeft(ar1, ar2) {
  var ar = [];
  for (let el of ar1) {
    if (!ar2.includes(el)) {
      ar.push(el);
    }
  }
  return ar;
}
```

Coding challenge #27a

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

Create a function that will receive an array of numbers as argument and will return a new array with distinct elements.

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

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

function getDistinctElements(ar)
{
    var ar2 = [];
    var lastIndexOf = ar.length - 1;
    for(let i = 0; i <= lastIndexOf; i++)
    {
        if (!isInArray(ar[i], ar, i + 1, lastIndex))
        {
            ar2.push(ar[i]);
        }
    }
    return ar2;
}

function isInArray(n, ar, fromIndex, toIndex)
{
    for(var i = fromIndex; i <= toIndex; i++)
    {
        if (ar[i] === n)
        {
            return true;
        }
    }
    return false;
}
```
Coding challenge #28: Calculate the sum of first 100 prime numbers
Coding challenge #29: Print the distance between the first 100 prime numbers

```javascript
// Coding challenge #29: Print the distance between the first 100 prime numbers
printDistances(100);

// Prints distances between the first n prime numbers
function printDistances(n) {
    var lastPrime = 2;
    var i = lastPrime + 1;
    var foundPrimes = 1;
    while (foundPrimes < n) {
        if (isPrime(i)) {
            println(i - lastPrime, \t", i, " \t", 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;
    var maxDiv = Math.sqrt(n);
    for (var i = 2; i <= maxDiv; i++) {
        if (n % i == 0) {
            return false;
        }
    }
    return true;
}
```
Coding challenge #30a: Create a function that will add two indefinite size numbers.

Requirements: Only positive numbers will be used and will be provided to the function as strings. The result should be also returned as a string.
Coding challenge #30b: Create a function that will add two indefinite size numbers. Only positive numbers will be used and will be provided to the function as strings. The result should be also returned as a string.

```
// Coding challenge #30: 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.

var n1 = "39093421912598942349";
var n2 = "12909239050293654999";
var sum = add(n1, n2);
println(n1);
println(n2);
println(sum);

function add(sNumber1, sNumber2)
{
    var maxSize = Math.max(sNumber1.length, sNumber2.length);
    var s1 = sNumber1.padStart(maxSize, "0");
    var s2 = sNumber2.padStart(maxSize, "0");
    var s = "";
    var carry = 0;
    for(var i = maxSize - 1; i >= 0; i--)
    {
        var digit1 = parseInt(s1[i]);
        var digit2 = parseInt(s2[i]);
        var sum = digit1 + digit2 + carry;
        var digitSum = sum % 10;
        carry = sum >= 10 ? 1 : 0;
        s = digitSum.toString() + s;
    }
    if (carry > 0)
    {
        s = carry + s;
    }
    return s;
}
```
Coding challenge #31-a: Create a function that will return the number of words in a text.
Coding challenge #31-b: Create a function that will return the number of words in a text.

```javascript
// Create a function that will return the number of words in a text

function countWords(text)
{
    var words = 0;
    if (text.length > 0 && !isSeparator(text[0]))
        words++;
    for(var i = 1; i < text.length; i++)
    {
        var currChr = text[i];
        var prevChr = text[i - 1];
        if ((isSeparator(currChr) && isSeparator(prevChr))
            { words++; }
    }
    return words;
}

function isSeparator(c)
{
    var separators = [" ", ",", ",", ",", ",", ",", ",", ","];
    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.
Coding challenge #33: Calculate the sum of numbers received in a comma delimited string

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

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

function sumCSV(s)
{
    var ar = s.split(",");
    var sum = 0;
    for(var n of ar)
    {
        var num = parseFloat(n);
        sum += num;
    }
    return sum;
}
```
Coding challenge #34: Create a function that will return an array with words inside a text

```javascript
// Coding challenge #34. Create a function that will return an array of string, with the words

var text = "Create a function, that will return an array (of string), with the word

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)
{
  var separators = [" ", ",", "\n", "\r", ",", ",", ",","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","","];

  return separators.includes(c);
}
```
**Coding challenge #35:** Create a function to convert a CSV text to a “bi-dimensional” array

```javascript
var data = "John;Smith;954-000-0000\n" + "Wich;Tiger;305-000-0000\n" + "Monique;Vasquez;103-000-0000";

var ar = csvToArray(data);
println(JSON.stringify(ar));

function csvToArray(data) {
    var arLines = data.split("\n");
    for(var i = 0; i < arLines.length; i++) {
        var arLine = arLines[i].split(";");
        arLines[i] = arLine;
    }
    return arLines;
}
```
Coding challenge #36: Create a function that converts a string to an array of characters.
Coding challenge #37: Create a function that will convert a string in an array containing the ASCII codes of each character.
Coding challenge #38: Create a function that will convert an array containing ASCII codes in a string.
Coding challenge #39: Implement the Caesar cypher
Coding challenge #40: Implement the bubble sort algorithm for an array of numbers

```javascript
// Coding challenge #40. Implement the bubble sort algorithm
// for an array of numbers
var ar = [23, 1000, 1, -1, 8, 3];
println(ar);
bubbleSort(ar);
println(ar);

function bubbleSort(ar)
{
    var shouldSort = true;
    var length = ar.length;
    while(shouldSort)
    {
        shouldSort = false;
        length--;
        for(var i = 0; i < length; i++)
        {
            var 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.

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

printin(getDistance(100, 100, 400, 300));

function getDistance(x1, y1, x2, y2) {
  var l1 = x2 - x1;
  var 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.
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.
Coding challenge #44: Create a function that will convert a string containing a binary number into a number

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

println(binaryToNumber("111111"));

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)

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

function sumArray(ar)
{
    var sum = 0;
    for(var el of ar)
    {
        if (Array.isArray(el))
        {
            el = sumArray(el);
        }
        sum += el;
    }
    return sum;
}

print(sumArray(ar));
```
Coding challenge #46-a: Find the maximum number in a jagged array of numbers or array of numbers

Requirements: Use recursion
Coding challenge #46-b: Find the maximum number in a jagged array of numbers or array of numbers

Requirements: Do not use recursion
Coding challenge #47: Deep copy a jagged array with numbers or other arrays in a new array

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

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

println(ar2);

function copyArray(ar)
{
  var ar2 = [];
  for(var el of ar)
  {
    if (Array.isArray(el))
      { el = copyArray(el); }
    ar2.push(el);
  }
  return ar2;
}
```
Coding challenge #48: Create a function to return the longest word(s) in a string
coding challenge #49: shuffle an array of strings

```javascript
// Coding challenge #49. Shuffle an array of strings
var ar = "Shuffle", "an", "array", "of", "strings";
println(shuffleArray(ar));

// Shuffle array implemented using Fisher-Yates shuffle algorithm
function shuffleArray(ar)
{
    for(var i = ar.length - 1; i > 0; i--)
    {
        var j = randint(0, i - 1);
        var t = ar[i];
        ar[i] = ar[j];
        ar[j] = t;
    }
    return ar;
}

// Get a random int between min and max (both included)
function randint(min, max)
{
    return Math.floor(Math.random() * (max - min + 1)) + min;
}
```
Coding challenge #50: Create a function that will receive n as argument and return an array of n random numbers from 1 to n.
Coding challenge #51: Find the frequency of characters inside a string. Return the result as an array of objects.

Each object has 2 fields: character and number of occurrences.

```
// Coding challenge #51. Find the frequency of characters inside a string.
// Return the result as an array of objects: character and number of occurrences.
var ar = getCharFrequency("Find the frequency of characters inside a string");
println(JSON.stringify(ar));

function getCharFrequency(text)
{
    var ar = [];
    for(var chr of text)
    {
        updateFrequency(ar, chr);
    }
    return ar;
}

function updateFrequency(ar, chr)
{
    for(var el of ar)
    {
        if (el.chr === chr)
        {
            el.count++;
        }
    }
    ar.push({ chr: chr, count: 1 });
}
```
Coding challenge #52: Calculate Fibonacci(500) with high precision (all digits)
Coding challenge #53: Calculate 70! with high precision (all digits)

function factorial(n) {
    var prod = "1";
    for(var i = 2; i <= n; i++) {
        var digit = parseInt(sNumber[i]);
        var partialResult = multDigit(sNumber, digit);
        partialResult += "0".repeat(partialResults.length);
        partialResults.push(partialResult);
    }
    var sum = "";
    for(var r of partialResults) {
        sum += add(sum, r);
    }
    return sum;
}

function mult(sNumber1, sNumber2) {
    var partialResults = [];
    for(var i = 0; i < sNumber2.length; i++) {
        var digit = parseInt(sNumber2[i]);
        var partialResult = multDigit(sNumber1, digit);
        partialResult += "0".repeat(partialResults.length);
        partialResults.push(partialResult);
    }
    var sum = "";
    for(var r of partialResults) {
        sum += add(sum, r);
    }
    return sum;
}

function multDigit(sNumber, digit) {
    var numberDigit = parseInt(sNumber[digit]);
    var prod = digit * numberDigit + carry;
    var prodDigit = prod % 10;
    carry = Math.floor(prod / 10);
    p = prodDigit.toString() + p;
    return p;
}

function add(sNumber1, sNumber2) {
    var maxSize = Math.max(sNumber1.length, sNumber2.length);
    var s1 = sNumber1.padStart(maxSize, "0");
    var s2 = sNumber2.padStart(maxSize, "0");
    var s = "";
    var carry = 0;
    for(var i = 0; i < maxSize - 1; i++) {
        var digit1 = parseInt(s1[i+1]);
        var digit2 = parseInt(s2[i+1]);
        var sum = digit1 + digit2 + carry;
        var digitSum = sum % 10;
        carry = sum >= 10 ? 1 : 0;
        s += digitSum.toString() + s;
    }
    if (carry > 0) {
        s += carry + s;
    }
    return s;
}
These coding challenges were brought to you by codeguppy.com - the fun coding site for kids, teens and creative adults

Don’t forget to visit https://codeguppy.com for more fun projects!

For news and updates follow @codeguppy on Twitter!