

50 coding challenges - Part I

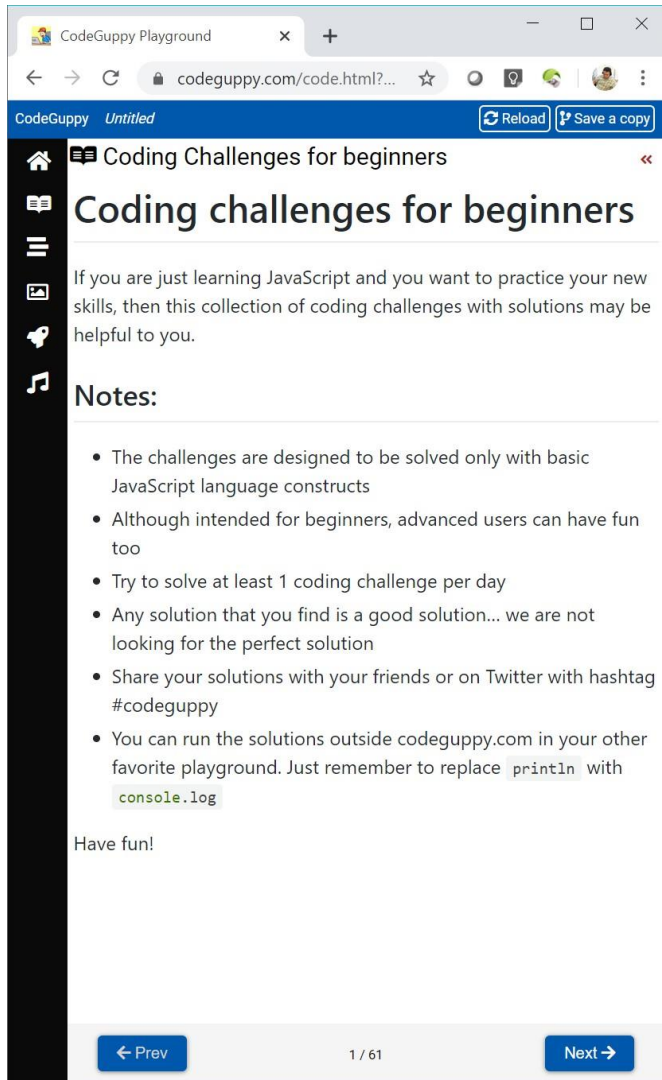
1. Print numbers from 1 to 10
2. Print the odd numbers less than 100
3. Print the multiplication table with 7
4. Print all the multiplication tables with numbers from 1 to 10
5. Calculate the sum of numbers from 1 to 10
6. Calculate 10!
7. Calculate the sum of even numbers greater than 10 and less than 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)



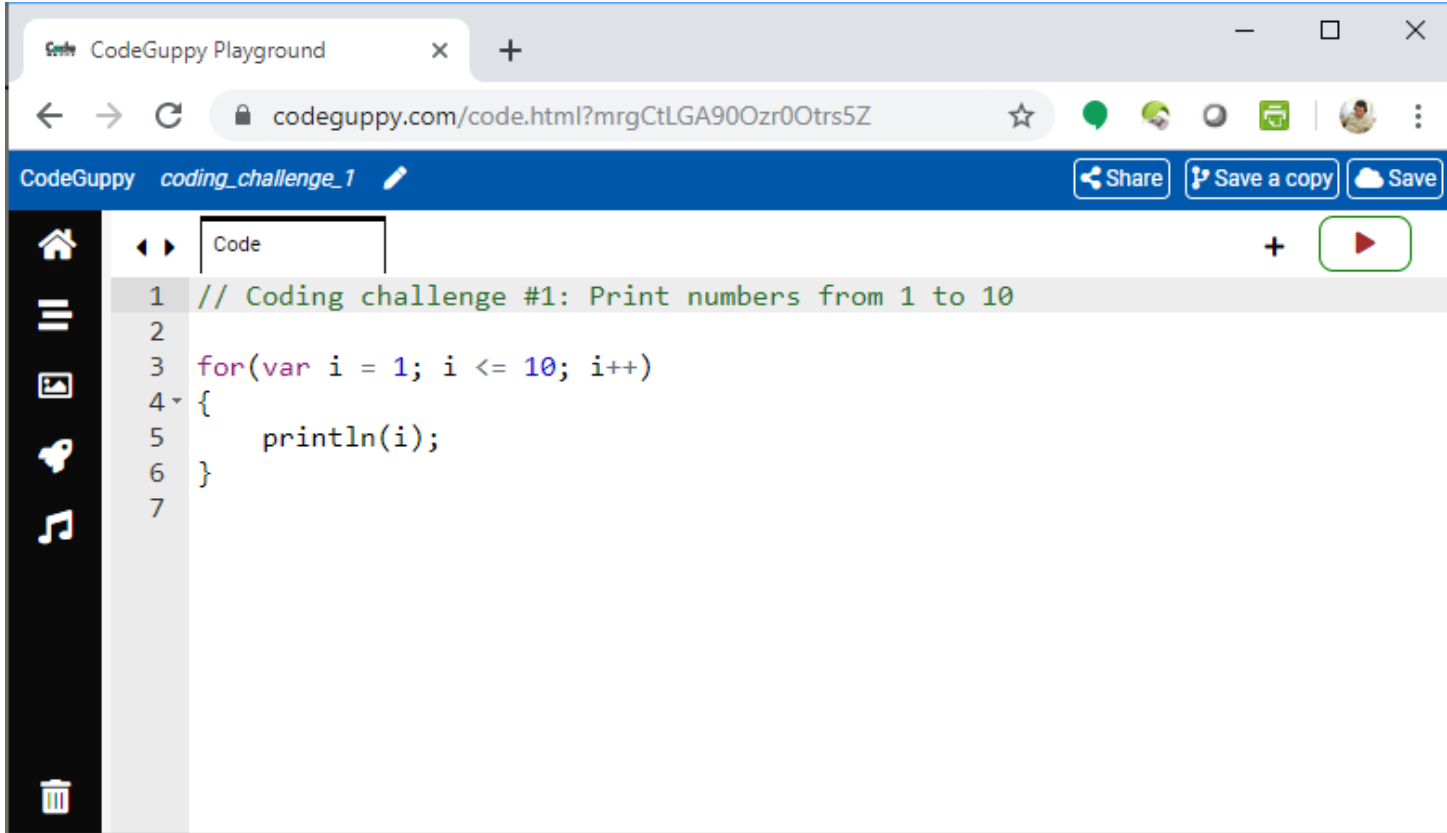


The source code of the solutions presented in this booklet is available online at:

You can type-in the full link above, or just go to and browse to locate the “Coding Challenges” project.



Coding challenge #1: Print numbers from 1 to 10



The screenshot shows a web browser window with the CodeGuppy Playground interface. The address bar displays the URL `codeguppy.com/code.html?mrgCtLGA90Ozr0Otrs5Z`. The page title is "CodeGuppy coding_challenge_1". The code editor contains the following JavaScript code:

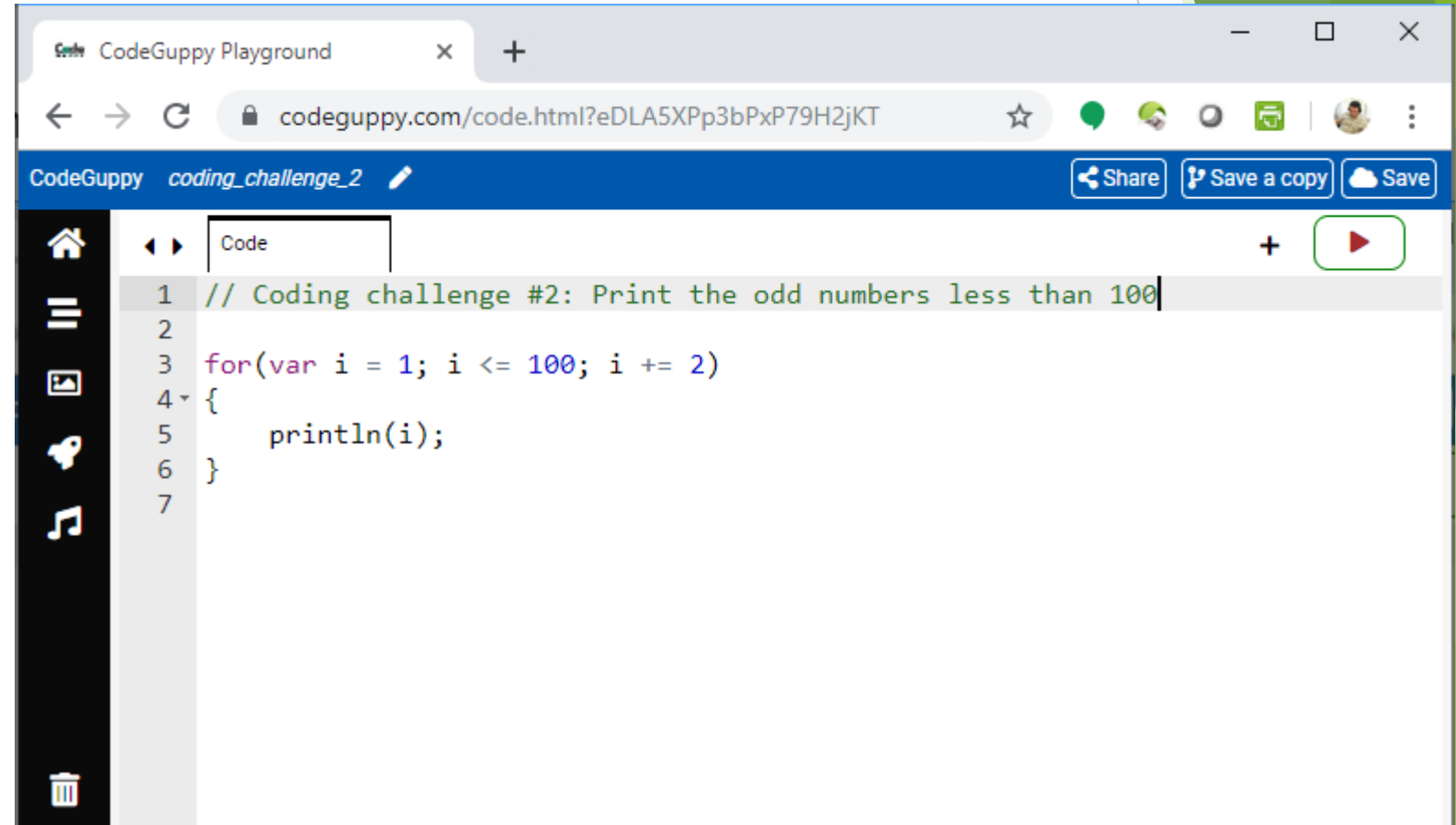
```
1 // Coding challenge #1: Print numbers from 1 to 10
2
3 for(var i = 1; i <= 10; i++)
4 {
5     println(i);
6 }
7
```

The interface includes a left sidebar with icons for home, list, document, chat, and music. The top right of the editor has buttons for "Share", "Save a copy", and "Save". A play button is visible on the right side of the code editor.



Coding challenge #2

Print the odd numbers
less than 100



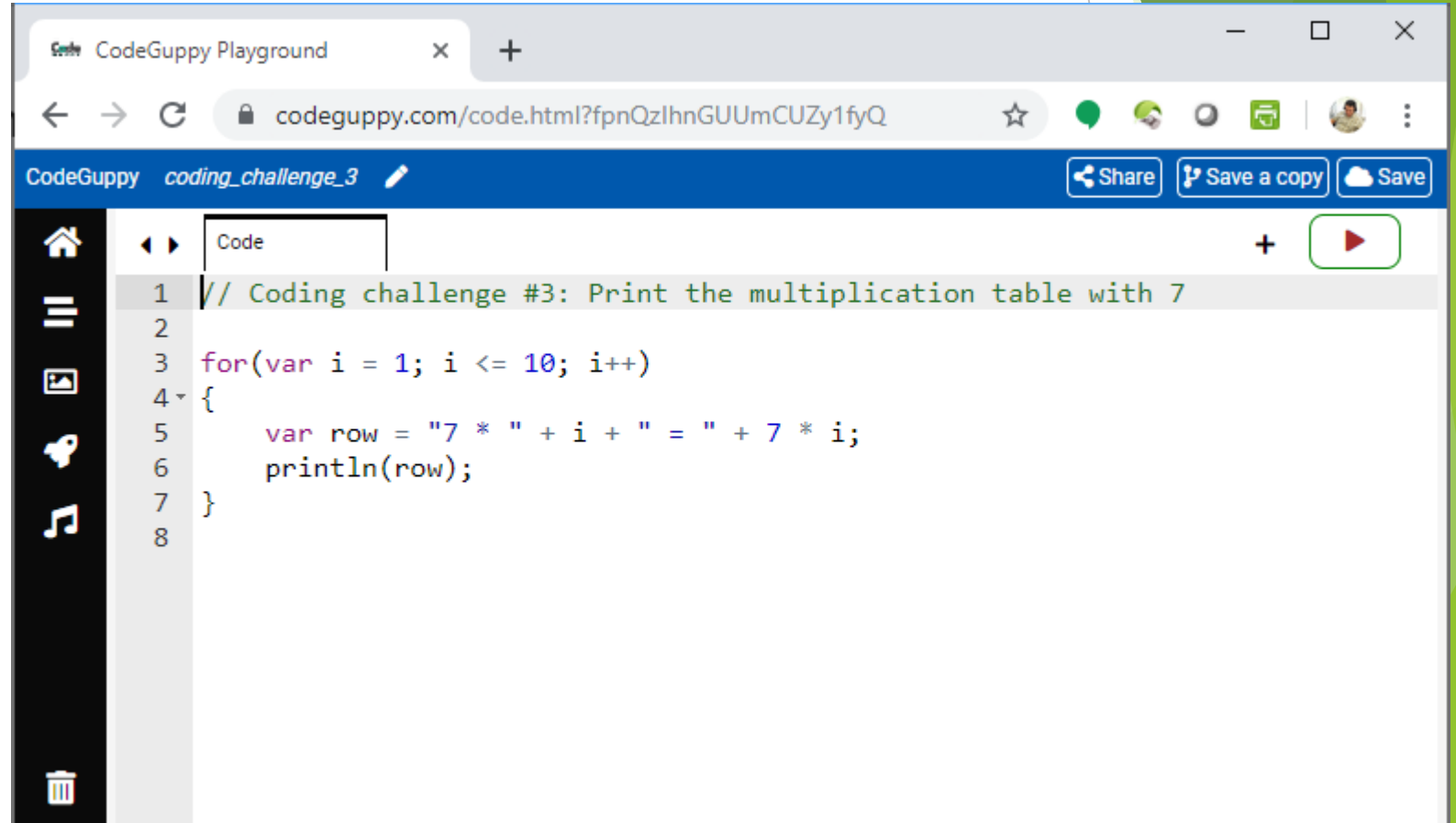
The screenshot shows a web browser window with the title "CodeGuppy Playground". The address bar displays the URL "codeguppy.com/code.html?eDLA5XPp3bPxP79H2jKT". The browser's toolbar includes navigation buttons (back, forward, refresh), a star icon for bookmarks, and several extension icons. Below the address bar, a blue header bar contains the text "CodeGuppy coding_challenge_2" and three buttons: "Share", "Save a copy", and "Save". The main content area is a code editor with a dark sidebar on the left containing icons for home, list, image, notification, music, and trash. The code editor has a tab labeled "Code" and a line number column on the left. The code being edited is as follows:

```
1 // Coding challenge #2: Print the odd numbers less than 100
2
3 for(var i = 1; i <= 100; i += 2)
4 {
5     println(i);
6 }
7
```



Coding challenge #3

Print the multiplication table with 7



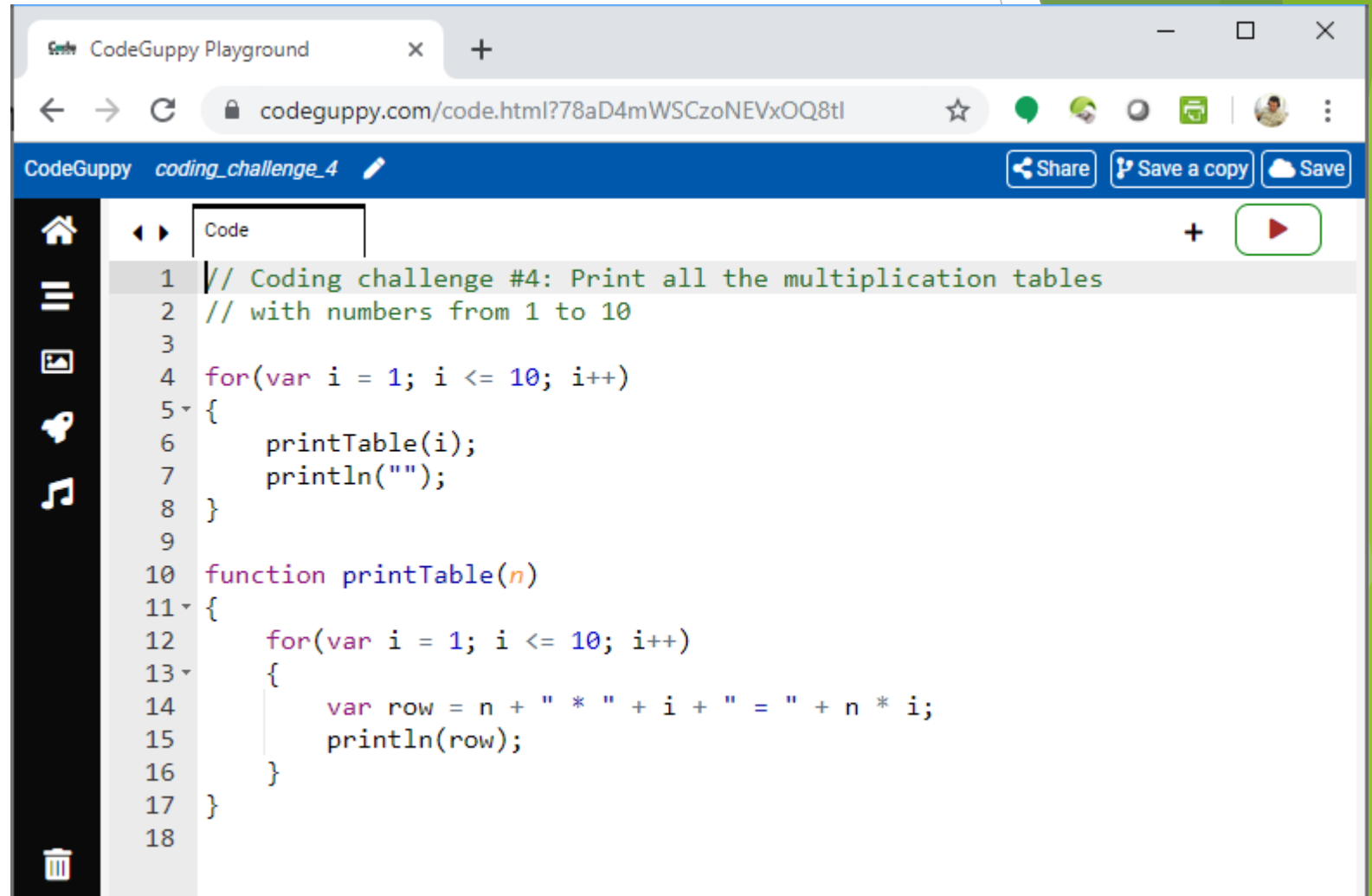
The screenshot shows a web browser window titled "CodeGuppy Playground". The address bar displays the URL "codeguppy.com/code.html?fpnQzIhnGUUmCUZy1fyQ". The page header includes the "CodeGuppy" logo, the filename "coding_challenge_3", and buttons for "Share", "Save a copy", and "Save". The main area is a code editor with a dark sidebar on the left containing icons for home, list, file, share, and music. The code editor has a tab labeled "Code" and a line number column on the left. The code is as follows:

```
1 // Coding challenge #3: Print the multiplication table with 7
2
3 for(var i = 1; i <= 10; i++)
4 {
5     var row = "7 * " + i + " = " + 7 * i;
6     println(row);
7 }
8
```



Coding challenge #4

Print the multiplication tables with numbers from 1 to 10



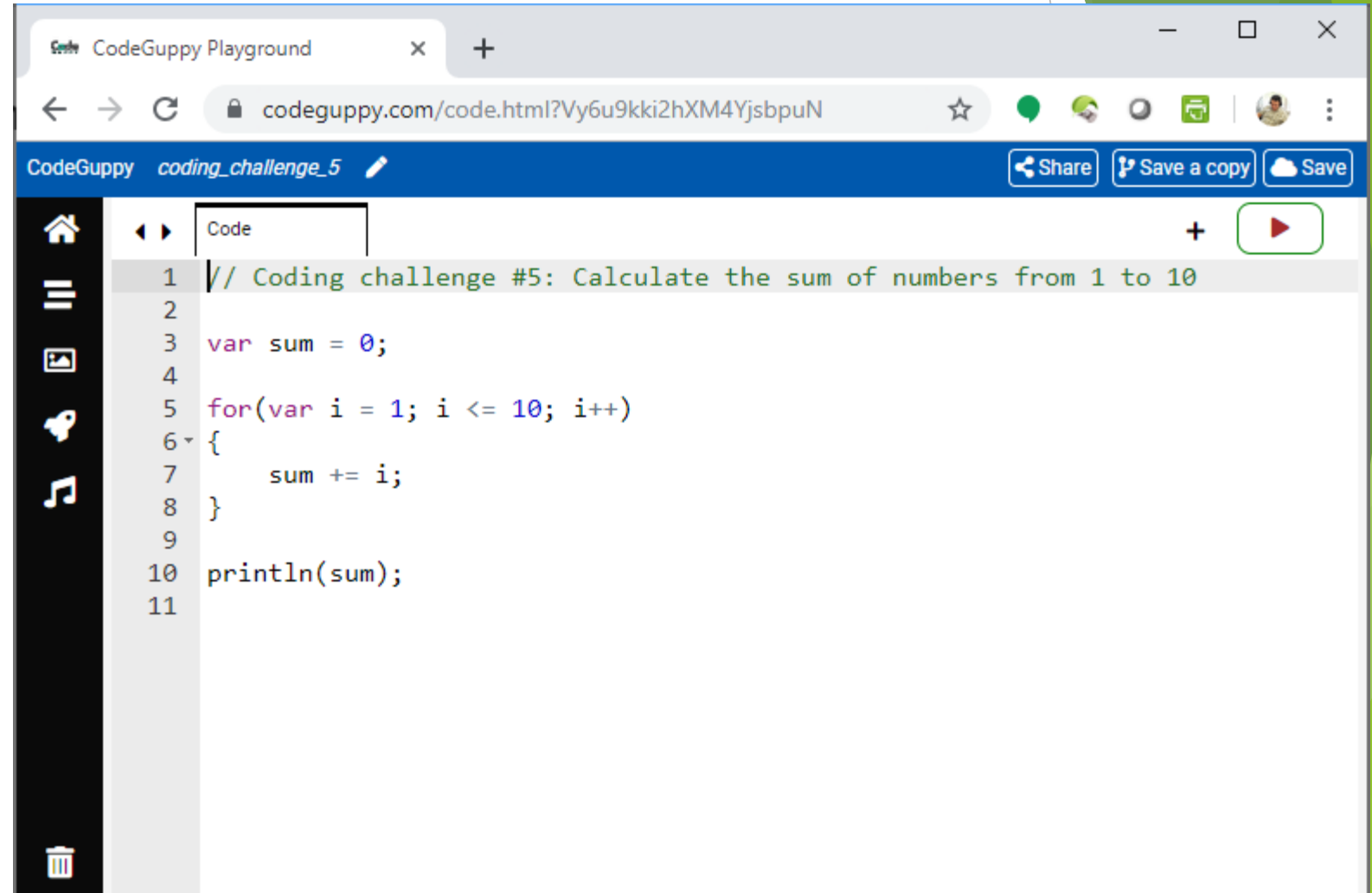
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?78aD4mWSCzoNEVxOQ8tl`. The browser's address bar and navigation buttons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for home, list, image, and music, and a main code editor area. The code editor displays the following JavaScript code:

```
1 // Coding challenge #4: Print all the multiplication tables
2 // with numbers from 1 to 10
3
4 for(var i = 1; i <= 10; i++)
5 {
6     printTable(i);
7     println("");
8 }
9
10 function printTable(n)
11 {
12     for(var i = 1; i <= 10; i++)
13     {
14         var row = n + " * " + i + " = " + n * i;
15         println(row);
16     }
17 }
18
```



Coding challenge #5

Calculate the sum of numbers from 1 to 10



The screenshot shows a web browser window with the title "CodeGuppy Playground". The address bar displays the URL "codeguppy.com/code.html?Vy6u9kki2hXM4YjsbpuN". The page header includes the text "CodeGuppy coding_challenge_5" and buttons for "Share", "Save a copy", and "Save". The main content area is a code editor with a dark sidebar on the left containing icons for home, list, file, search, and trash. The code editor has a tab labeled "Code" and a play button in the top right corner. The code is as follows:

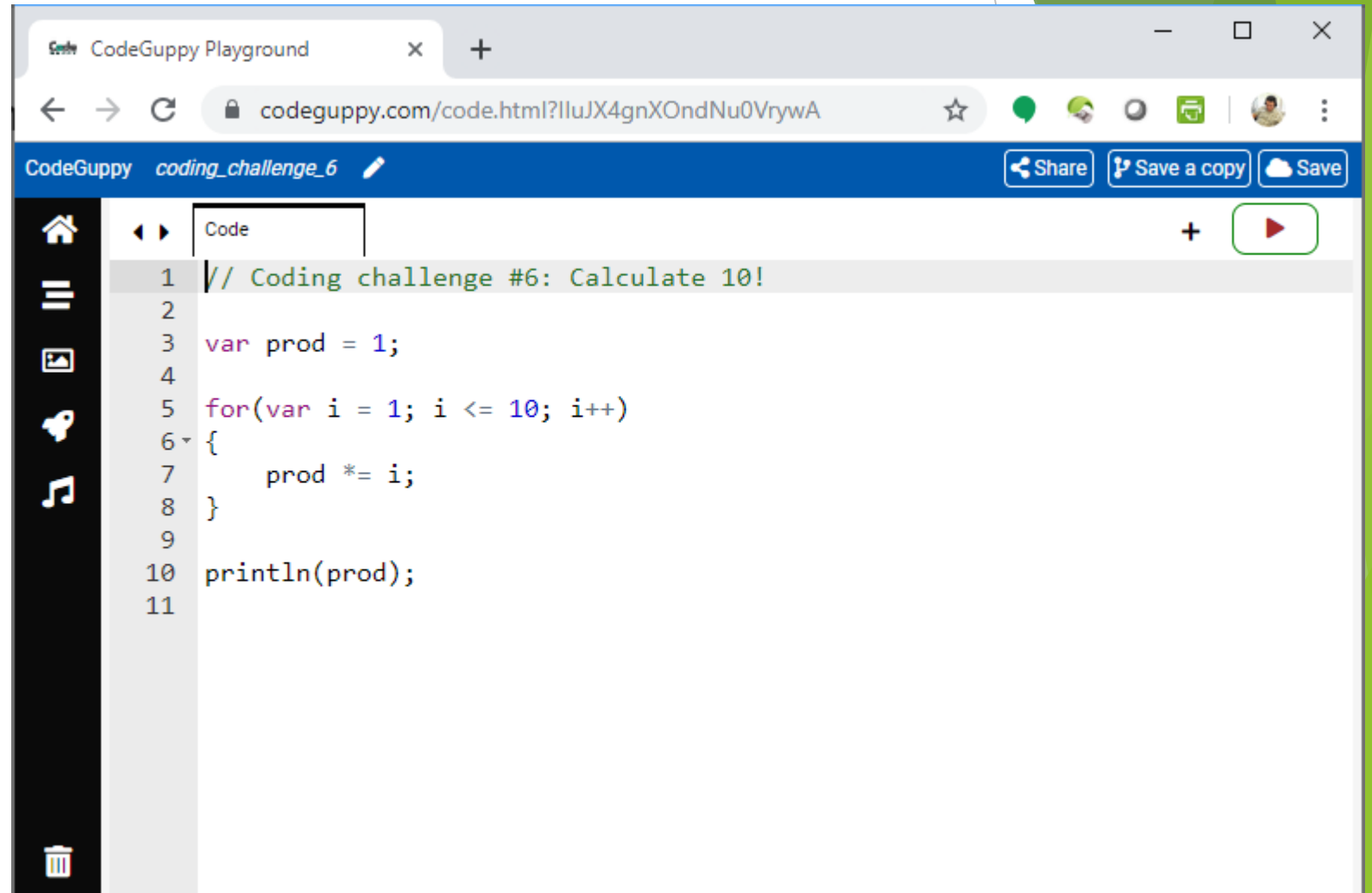
```
1 // Coding challenge #5: Calculate the sum of numbers from 1 to 10
2
3 var sum = 0;
4
5 for(var i = 1; i <= 10; i++)
6 {
7     sum += i;
8 }
9
10 println(sum);
11
```



Coding challenge #6

Calculate 10!

Reminder $n! = 1 * 2 * \dots * n$



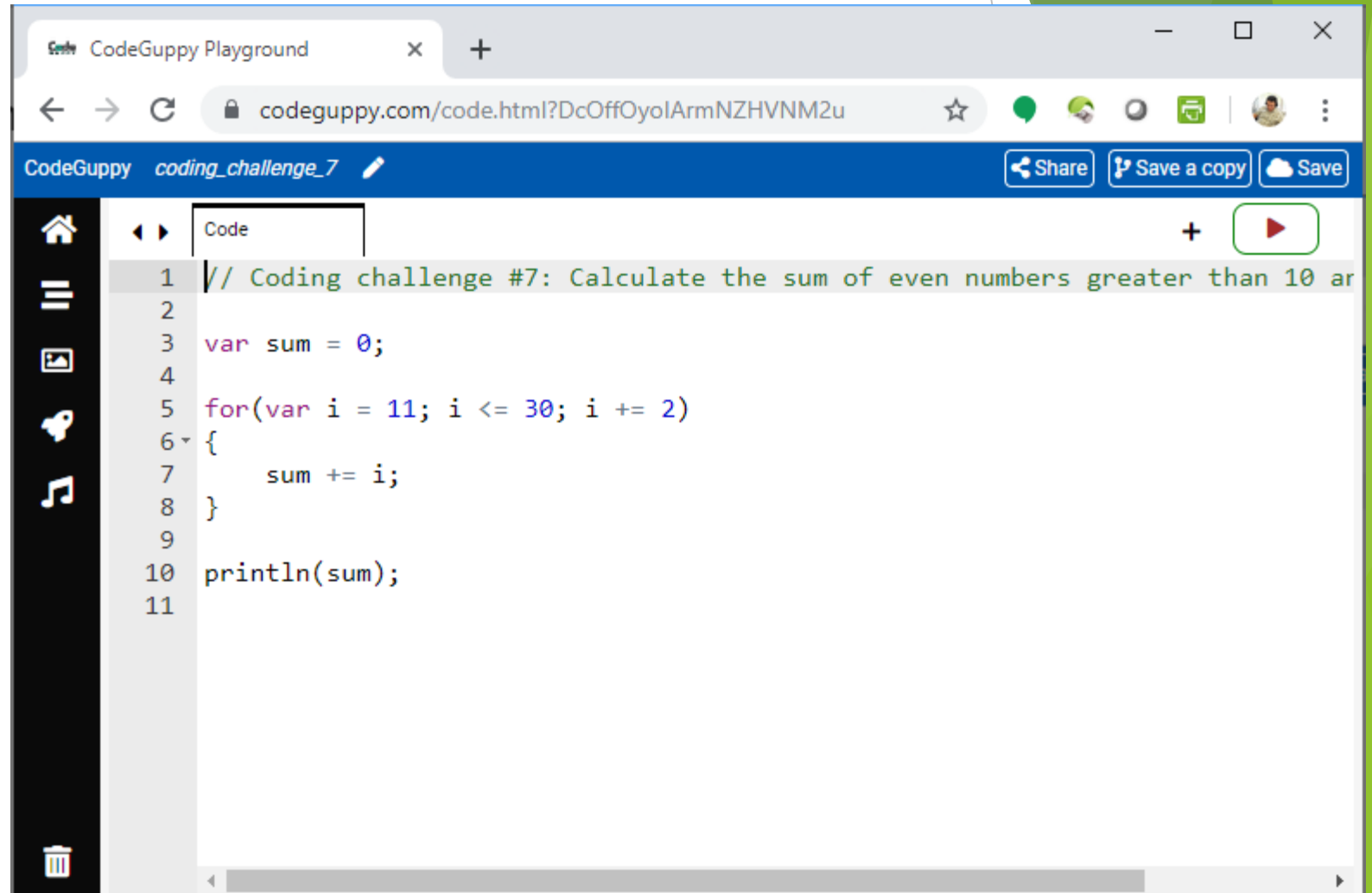
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?lluJX4gnXOndNu0VrywA". The browser's address bar and navigation buttons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for home, list, image, cloud, and music, and a main code editor. The code editor contains the following JavaScript code:

```
1 // Coding challenge #6: Calculate 10!
2
3 var prod = 1;
4
5 for(var i = 1; i <= 10; i++)
6 {
7     prod *= i;
8 }
9
10 println(prod);
11
```



Coding challenge #7

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



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?DcOffOyoIArmNZHVNM2u". The browser's address bar and navigation buttons are visible. Below the browser window, there is a blue header bar with the text "CodeGuppy coding_challenge_7" and buttons for "Share", "Save a copy", and "Save". The main area of the browser displays a code editor with a dark sidebar on the left containing icons for home, list, image, share, and music. The code editor has a tab labeled "Code" and a red play button in the top right corner. The code is written in C# and is as follows:

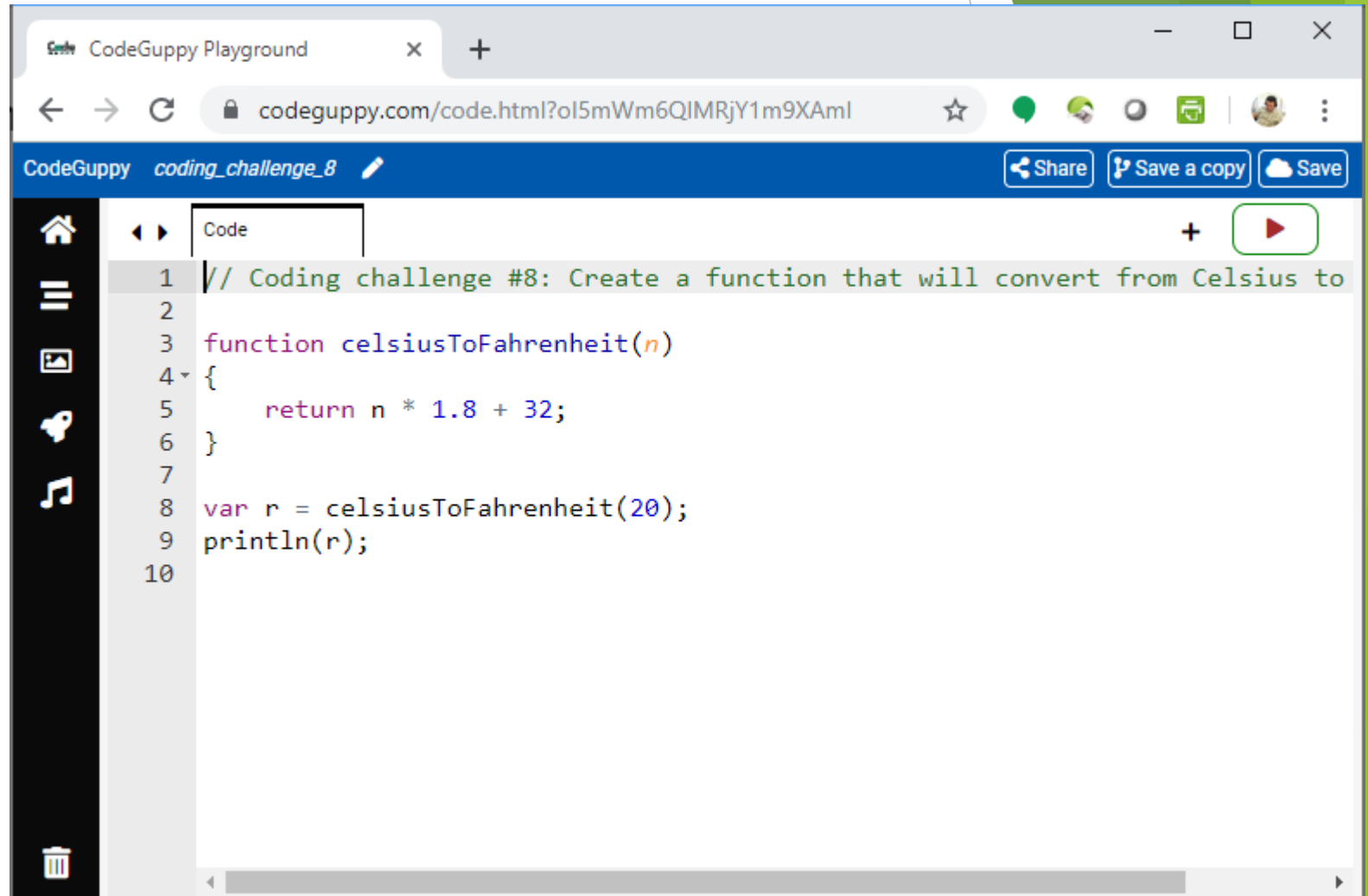
```
1 // Coding challenge #7: Calculate the sum of even numbers greater than 10 ar
2
3 var sum = 0;
4
5 for(var i = 11; i <= 30; i += 2)
6 {
7     sum += i;
8 }
9
10 println(sum);
11
```



Coding challenge #8

Create a function that will convert from Celsius to Fahrenheit

Reminder: $C = (F - 32) / 1.8$



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?ol5mWm6QIMRjY1m9XAml`. The browser's address bar and navigation buttons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for home, list, file, chat, and music. The main area is a code editor with a tab labeled "Code". The code in the editor is as follows:

```
1 // Coding challenge #8: Create a function that will convert from Celsius to
2
3 function celsiusToFahrenheit(n)
4 {
5     return n * 1.8 + 32;
6 }
7
8 var r = celsiusToFahrenheit(20);
9 println(r);
10
```

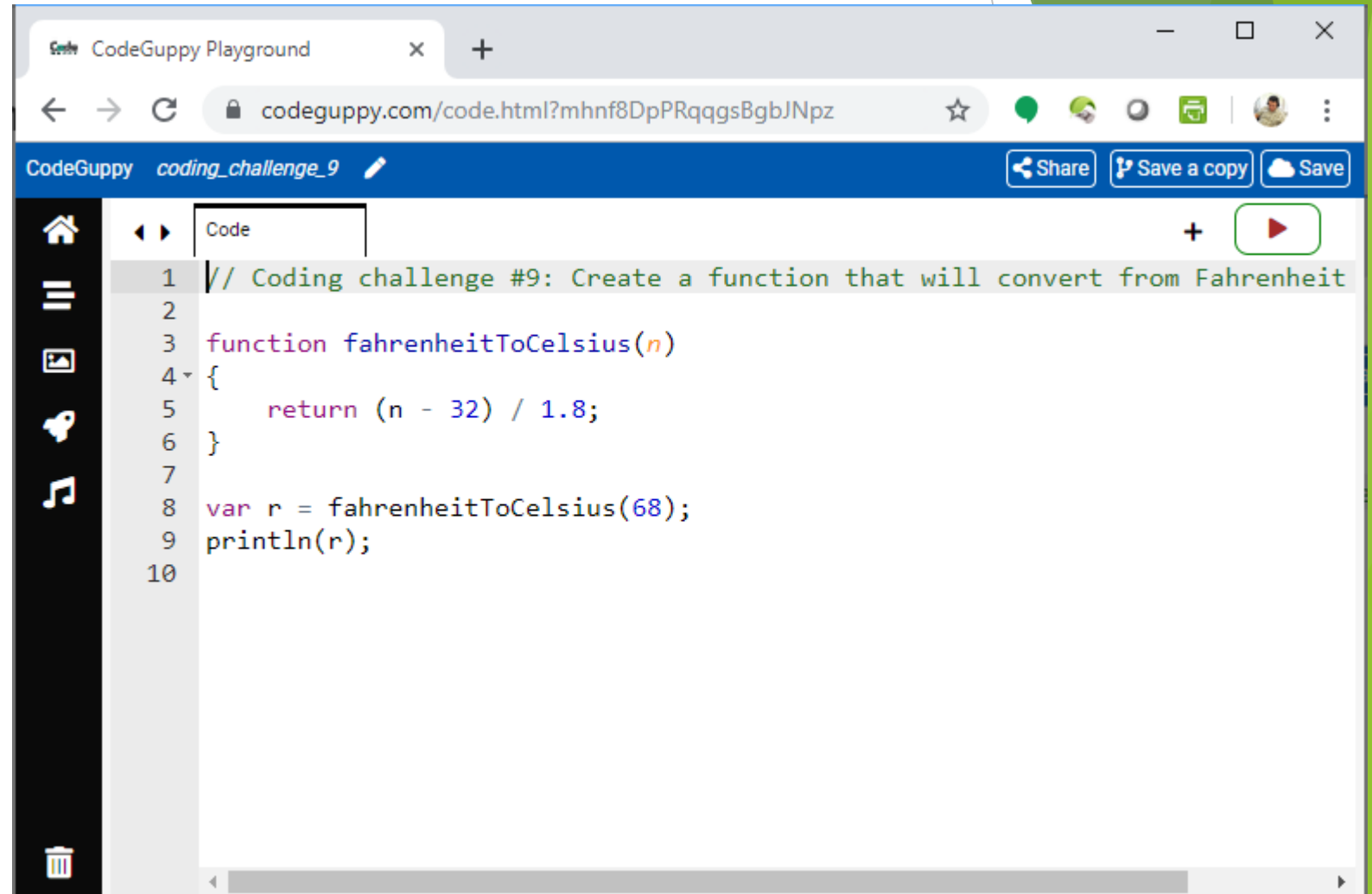
At the top right of the code editor, there are buttons for "Share", "Save a copy", and "Save". A red play button icon is also visible in the top right corner of the editor area.



Coding challenge #9

Create a function that will convert from Fahrenheit to Celsius

Reminder: $C = (F - 32) / 1.8$



The screenshot shows a web browser window with the CodeGuppy Playground interface. The address bar shows the URL `codeguppy.com/code.html?mhnf8DpPRqqgsBgbJNpz`. The page title is "CodeGuppy coding_challenge_9". The code editor contains the following JavaScript code:

```
1 // Coding challenge #9: Create a function that will convert from Fahrenheit
2
3 function fahrenheitToCelsius(n)
4 {
5     return (n - 32) / 1.8;
6 }
7
8 var r = fahrenheitToCelsius(68);
9 println(r);
10
```

The interface includes a left sidebar with icons for home, list, file, share, and trash. The top right of the editor has buttons for "Share", "Save a copy", and "Save". A "Code" tab is active at the top of the editor.



Coding challenge #10

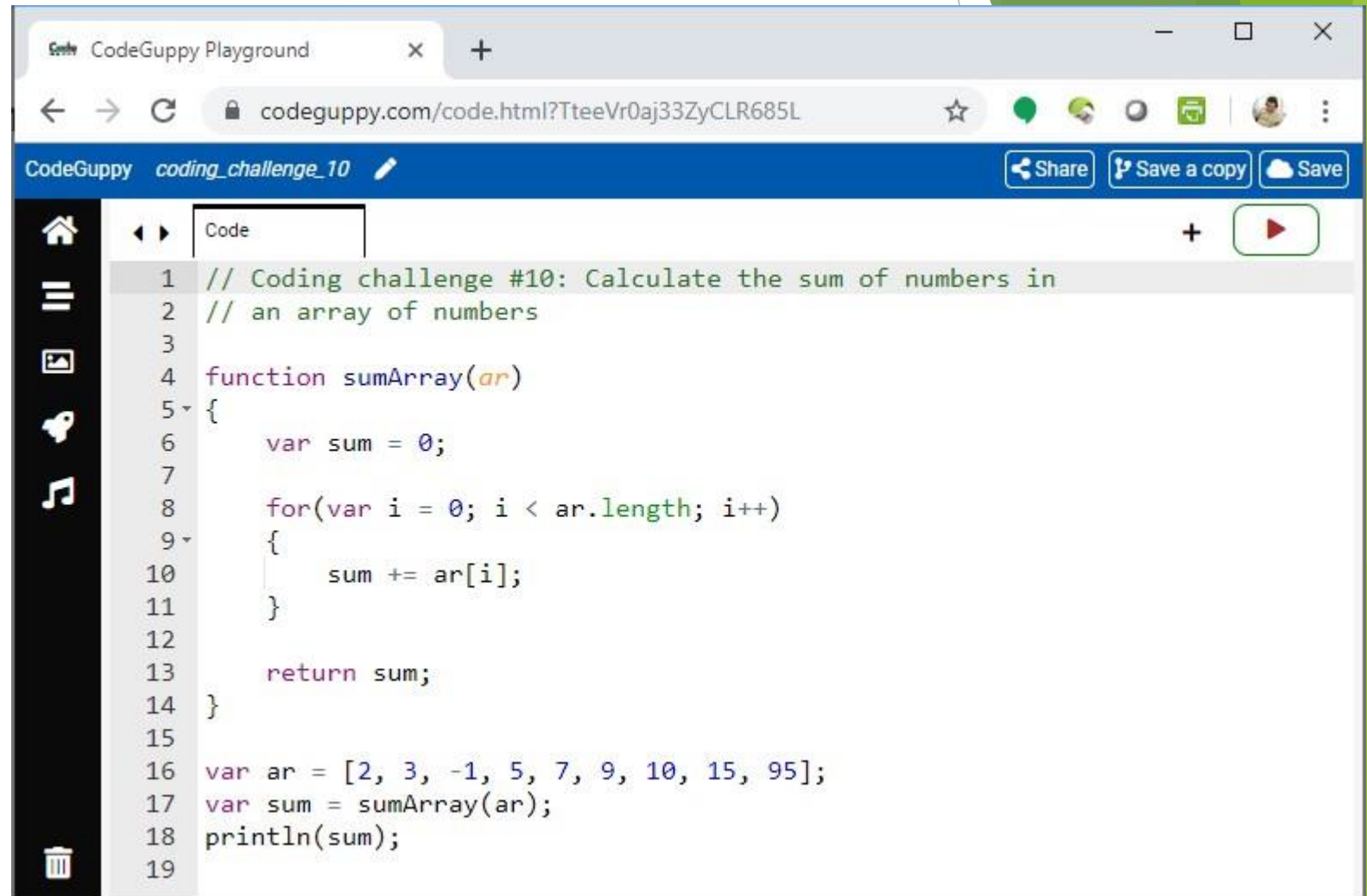
Calculate the sum of numbers
in an array of numbers.

Example array:

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

Expected output:

145



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?TteeVr0aj33ZyCLR685L". The page has a blue header with the text "CodeGuppy coding_challenge_10" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, file, search, and trash. The main area is a code editor with a tab labeled "Code" and a red play button. The code is as follows:

```
1 // Coding challenge #10: Calculate the sum of numbers in
2 // an array of numbers
3
4 function sumArray(ar)
5 {
6     var sum = 0;
7
8     for(var i = 0; i < ar.length; i++)
9     {
10         sum += ar[i];
11     }
12
13     return sum;
14 }
15
16 var ar = [2, 3, -1, 5, 7, 9, 10, 15, 95];
17 var sum = sumArray(ar);
18 println(sum);
19
```

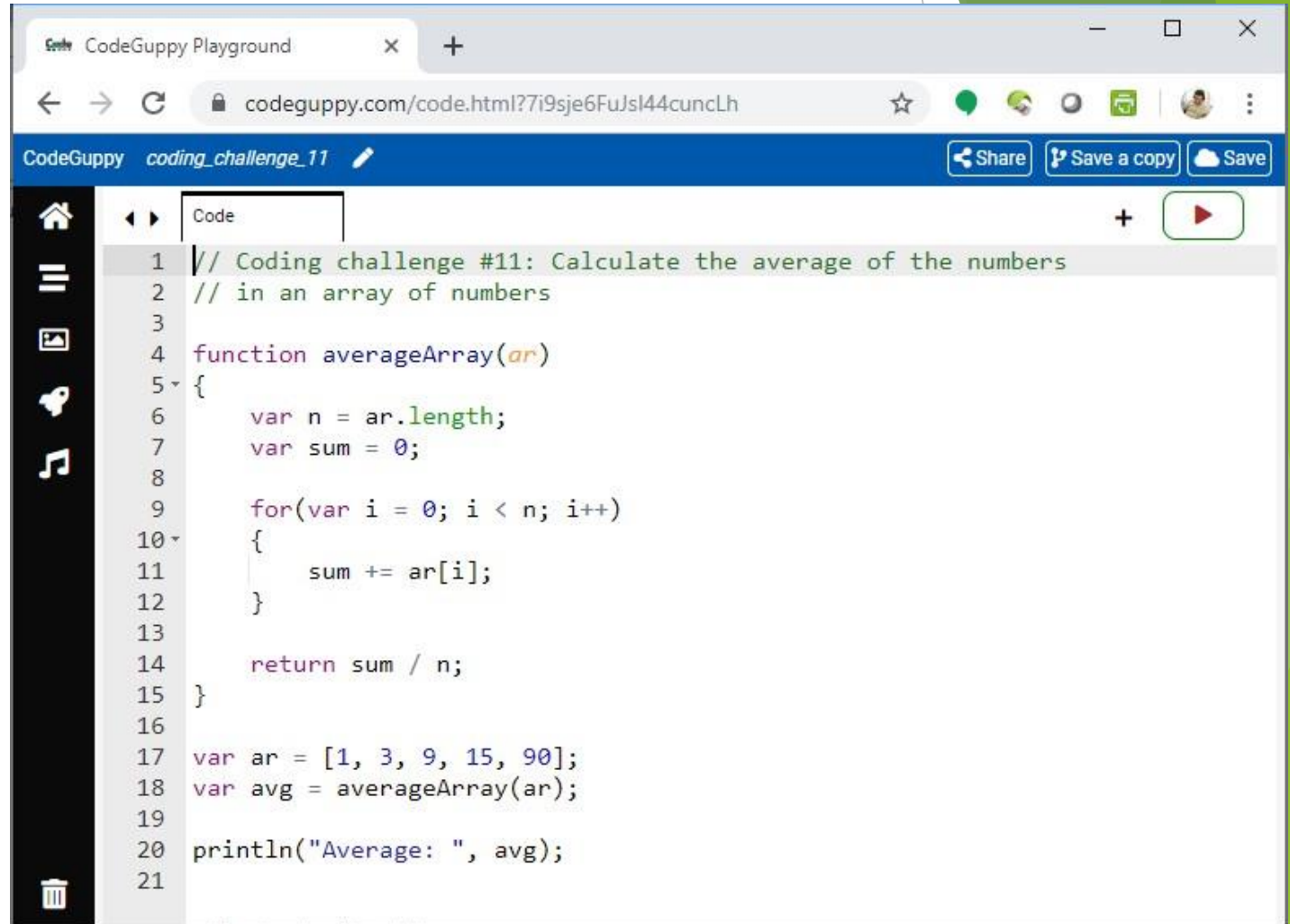


Coding challenge #11

Calculate the average of the numbers in an array of numbers

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

Expected output:
23.6



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?7i9sje6FuJsI44cuncLh". The browser's address bar and navigation buttons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for home, code, and other features. The main area displays a code editor with the following JavaScript code:

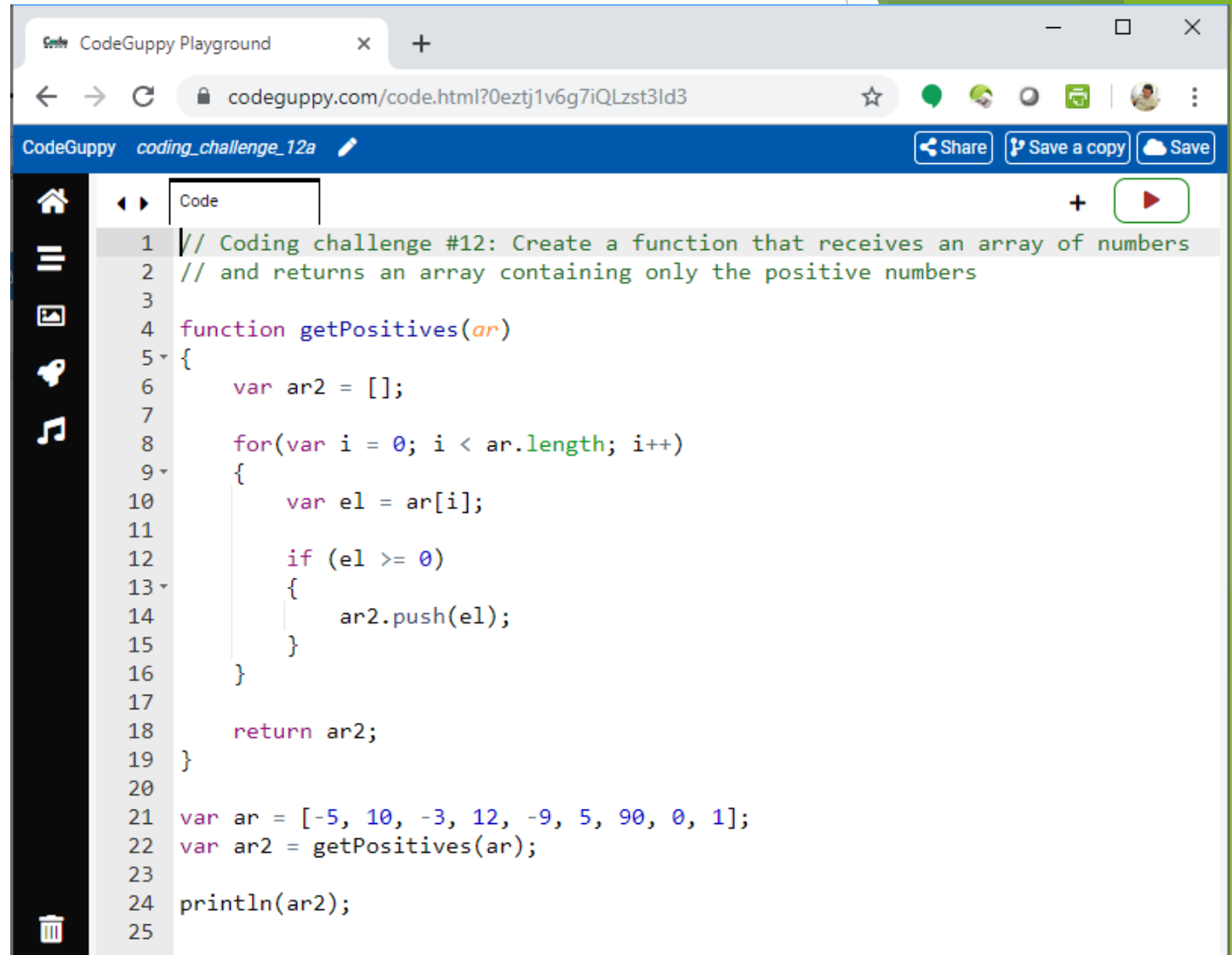
```
1 // Coding challenge #11: Calculate the average of the numbers
2 // in an array of numbers
3
4 function averageArray(ar)
5 {
6     var n = ar.length;
7     var sum = 0;
8
9     for(var i = 0; i < n; i++)
10    {
11        sum += ar[i];
12    }
13
14    return sum / n;
15 }
16
17 var ar = [1, 3, 9, 15, 90];
18 var avg = averageArray(ar);
19
20 println("Average: ", avg);
21
```



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



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?0eztj1v6g7iQLzst3ld3". The page has a blue header with the CodeGuppy logo, the filename "coding_challenge_12a", and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, run, code, console, and a trash can. The main area is a code editor with a "Code" tab and a "Run" button (a red play icon). The code is as follows:

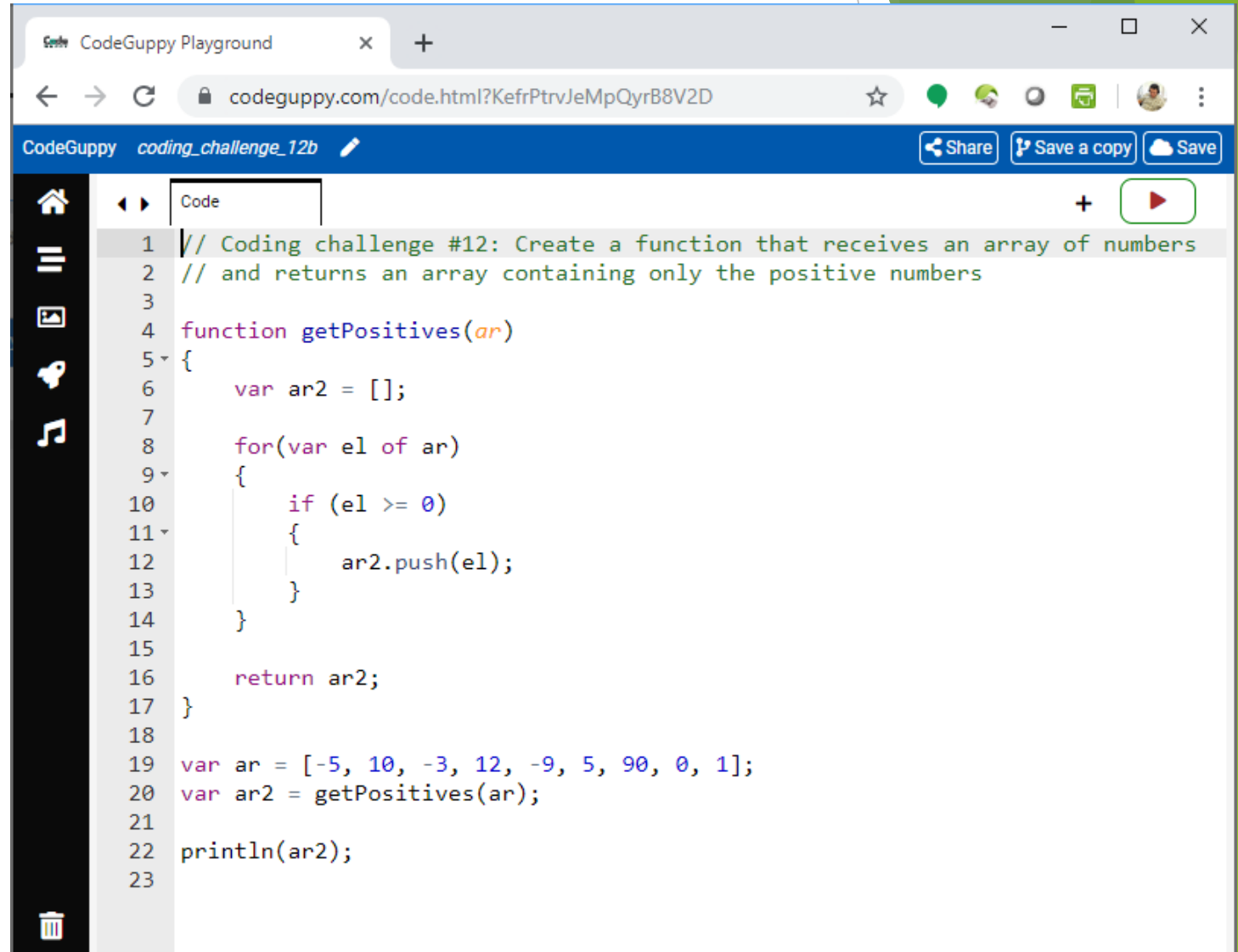
```
1 // Coding challenge #12: Create a function that receives an array of numbers
2 // and returns an array containing only the positive numbers
3
4 function getPositives(ar)
5 {
6     var ar2 = [];
7
8     for(var i = 0; i < ar.length; i++)
9     {
10         var el = ar[i];
11
12         if (el >= 0)
13         {
14             ar2.push(el);
15         }
16     }
17
18     return ar2;
19 }
20
21 var ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
22 var ar2 = getPositives(ar);
23
24 println(ar2);
25
```



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



The screenshot shows a web browser window with the address bar displaying `codeguppy.com/code.html?KefrPtrvJeMpQyrB8V2D`. The page title is "CodeGuppy coding_challenge_12b". The main content area shows a code editor with the following JavaScript code:

```
1 // Coding challenge #12: Create a function that receives an array of numbers
2 // and returns an array containing only the positive numbers
3
4 function getPositives(ar)
5 {
6     var ar2 = [];
7
8     for(var el of ar)
9     {
10         if (el >= 0)
11         {
12             ar2.push(el);
13         }
14     }
15
16     return ar2;
17 }
18
19 var ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
20 var ar2 = getPositives(ar);
21
22 println(ar2);
23
```

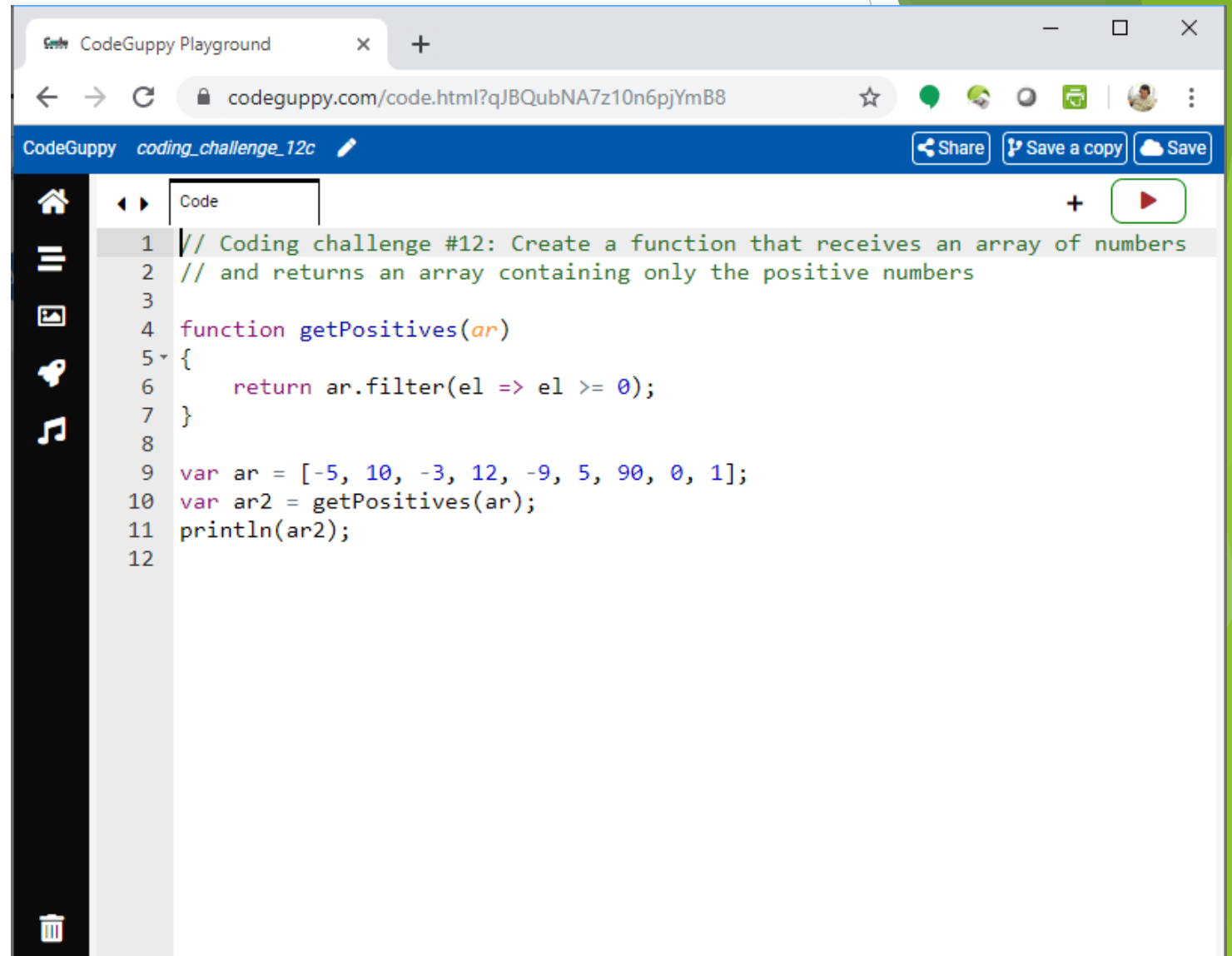
The code defines a function `getPositives` that takes an array `ar` and returns a new array `ar2` containing only the positive numbers from `ar`. It uses a `for...of` loop to iterate over the elements of `ar` and an `if` statement to check if each element is greater than or equal to zero. The function is then called with the array `[-5, 10, -3, 12, -9, 5, 90, 0, 1]`, and the result is printed to the console.



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



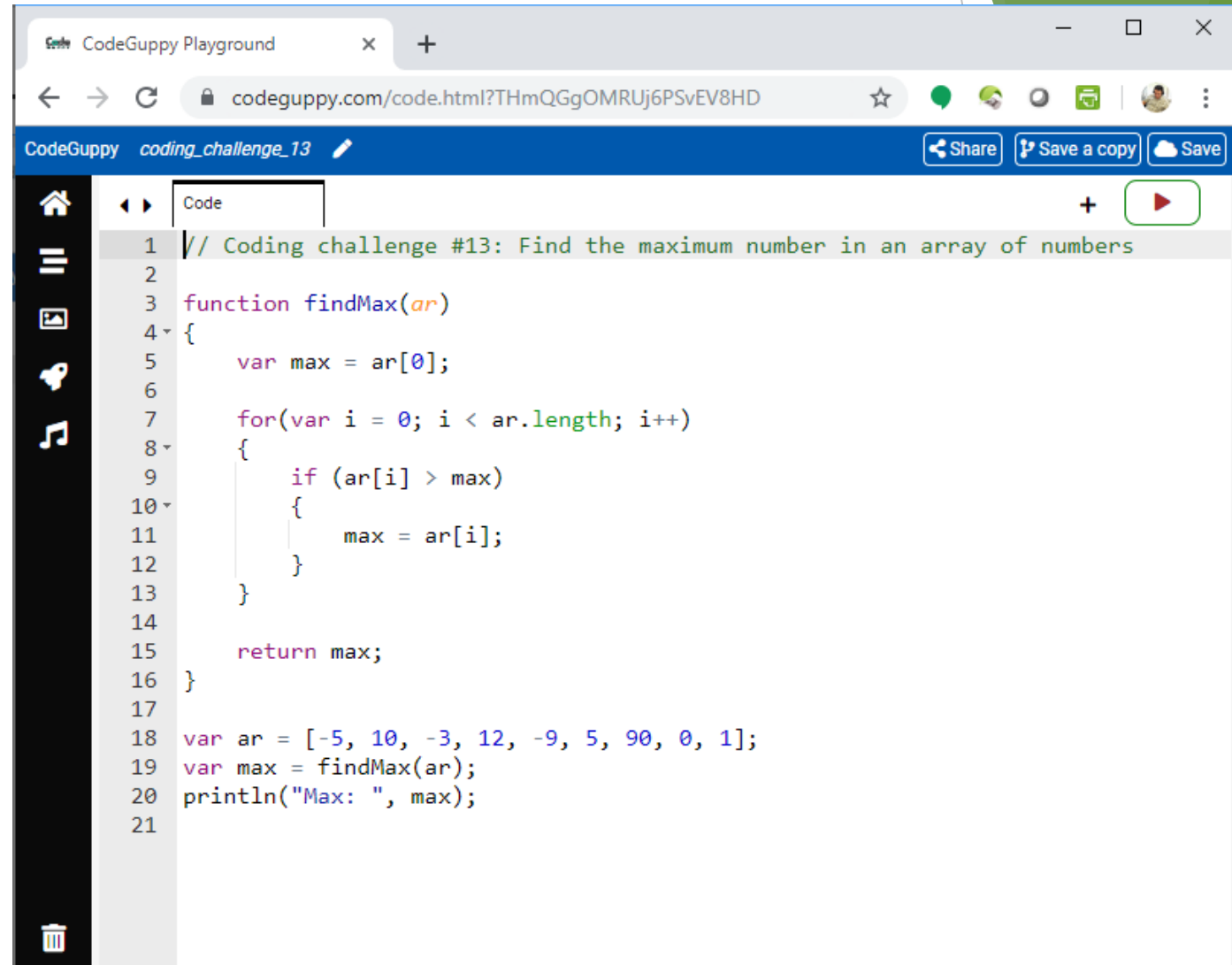
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?qJBQubNA7z10n6pjYmB8`. The browser's address bar and navigation buttons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for home, list, image, notification, and music. The main area is a code editor with a tab labeled "Code". The code in the editor is as follows:

```
1 // Coding challenge #12: Create a function that receives an array of numbers
2 // and returns an array containing only the positive numbers
3
4 function getPositives(ar)
5 {
6     return ar.filter(el => el >= 0);
7 }
8
9 var ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
10 var ar2 = getPositives(ar);
11 println(ar2);
12
```



Coding challenge #13

Find the maximum number
in an array of numbers



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?THmQGgOMRUj6PSvEV8HD". The page has a blue header with the text "CodeGuppy coding_challenge_13" and buttons for "Share", "Save a copy", and "Save". On the left side, there is a vertical toolbar with icons for home, list, image, speech bubble, music note, and trash. The main area is a code editor with a tab labeled "Code" and a play button. The code is as follows:

```
1 // Coding challenge #13: Find the maximum number in an array of numbers
2
3 function findMax(ar)
4 {
5     var max = ar[0];
6
7     for(var i = 0; i < ar.length; i++)
8     {
9         if (ar[i] > max)
10        {
11            max = ar[i];
12        }
13    }
14
15    return max;
16 }
17
18 var ar = [-5, 10, -3, 12, -9, 5, 90, 0, 1];
19 var max = findMax(ar);
20 println("Max: ", max);
21
```



Coding challenge #14

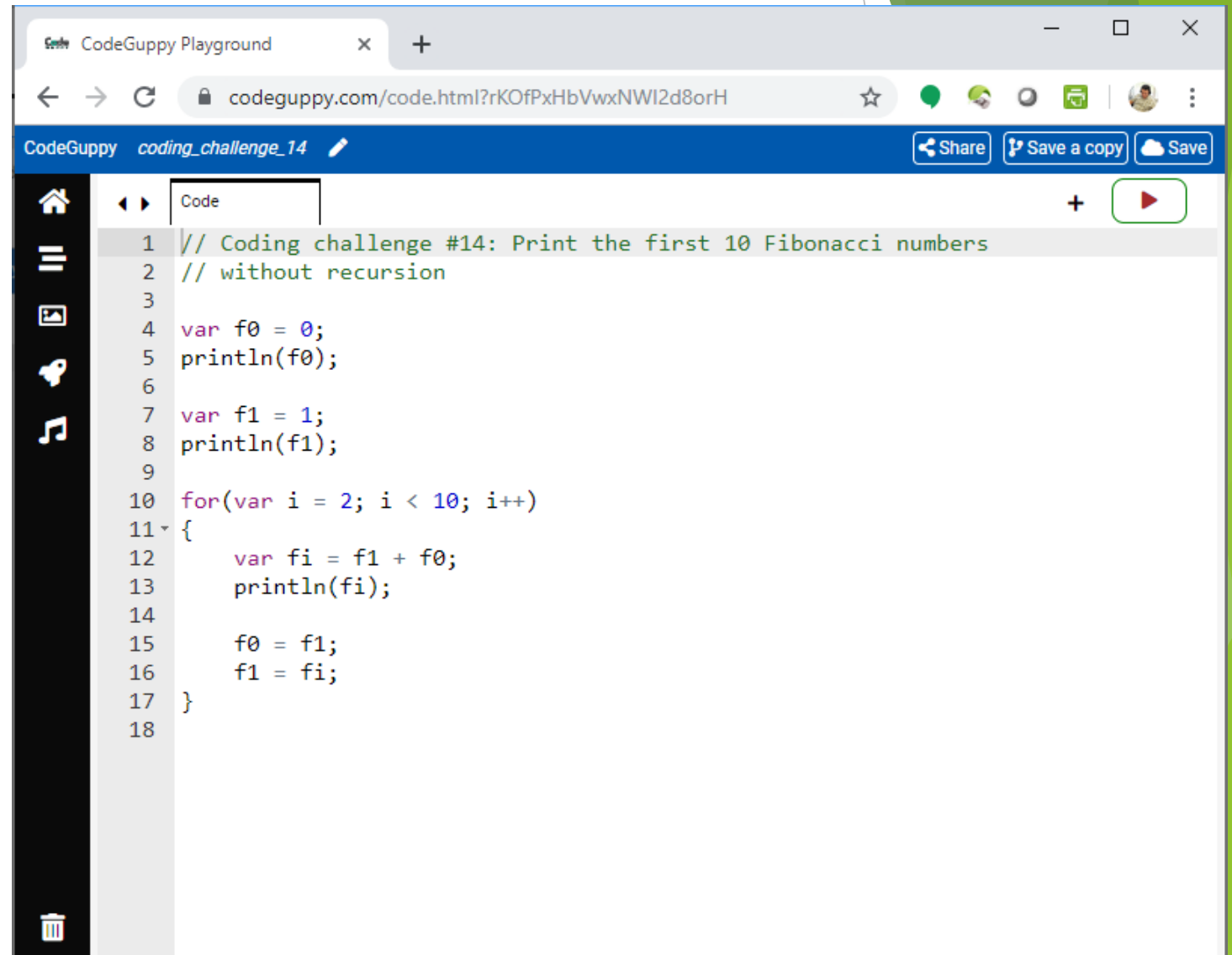
Print the first 10 Fibonacci numbers without using recursion.

Reminder:

$$F(0) = 0$$

$$F(1) = 1$$

$$F(n) = F(n-1) + F(n-2)$$



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?rKOFPxHbVwxNWI2d8orH". The page has a blue header with the text "CodeGuppy coding_challenge_14" and buttons for "Share", "Save a copy", and "Save". On the left side, there is a dark sidebar with icons for home, list, image, key, music, and trash. The main area is a code editor with a tab labeled "Code" and a red play button. The code in the editor is as follows:

```
1 // Coding challenge #14: Print the first 10 Fibonacci numbers
2 // without recursion
3
4 var f0 = 0;
5 println(f0);
6
7 var f1 = 1;
8 println(f1);
9
10 for(var i = 2; i < 10; i++)
11 {
12     var fi = f1 + f0;
13     println(fi);
14
15     f0 = f1;
16     f1 = fi;
17 }
18
```



Coding challenge #15

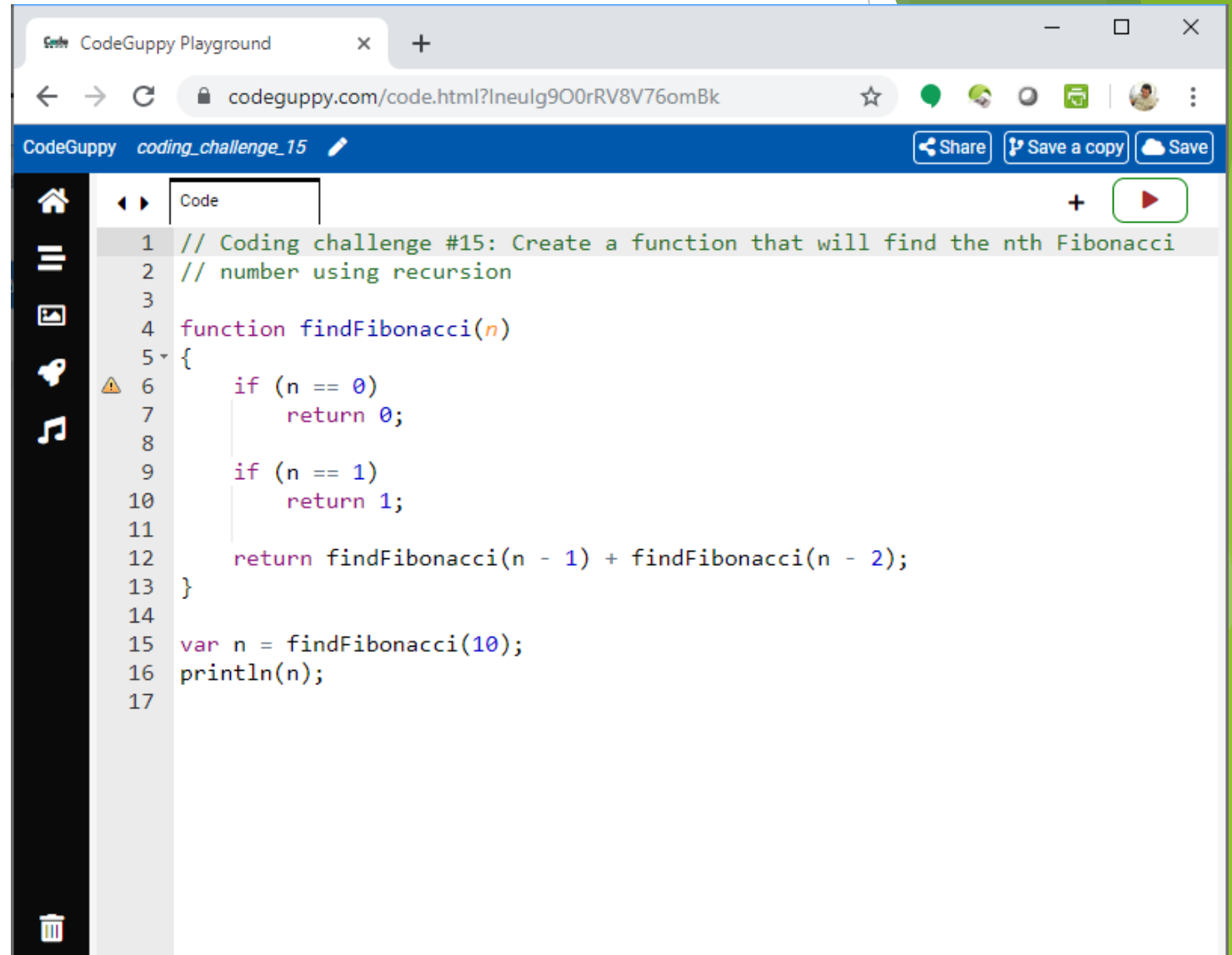
Create a function that will find the nth Fibonacci number using recursion.

Reminder:

$$F(0) = 0$$

$$F(1) = 1$$

$$F(n) = F(n-1) + F(n-2)$$



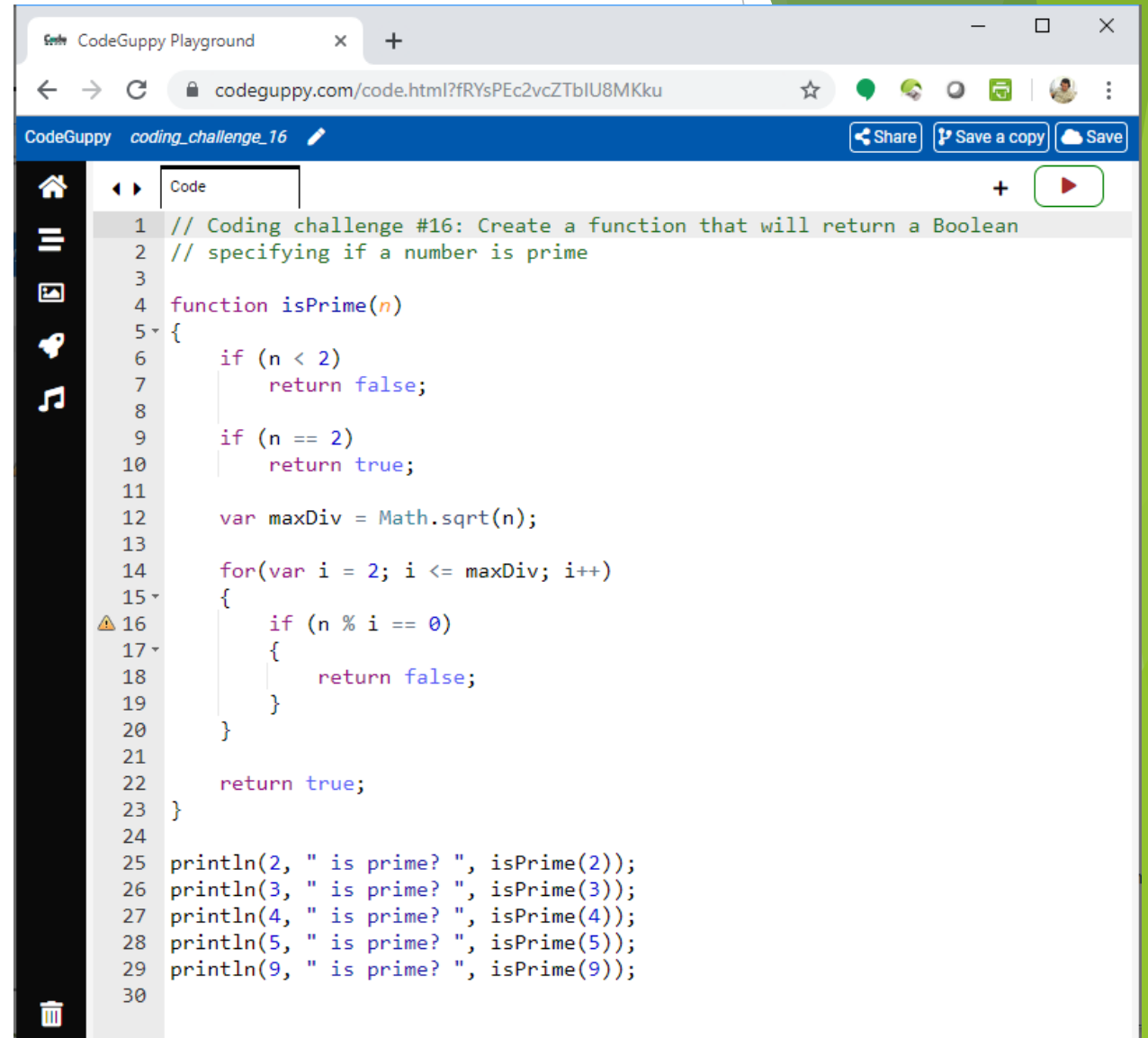
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?Ineulg9O0rRV8V76omBk". The page has a blue header with the text "CodeGuppy coding_challenge_15" and buttons for "Share", "Save a copy", and "Save". On the left side, there is a dark sidebar with icons for home, list, document, bug, and music. The main area displays a code editor with the following JavaScript code:

```
1 // Coding challenge #15: Create a function that will find the nth Fibonacci
2 // number using recursion
3
4 function findFibonacci(n)
5 {
6     if (n == 0)
7         return 0;
8
9     if (n == 1)
10        return 1;
11
12    return findFibonacci(n - 1) + findFibonacci(n - 2);
13 }
14
15 var n = findFibonacci(10);
16 println(n);
17
```



Coding challenge #16

Create a function that will return a Boolean specifying if a number is prime



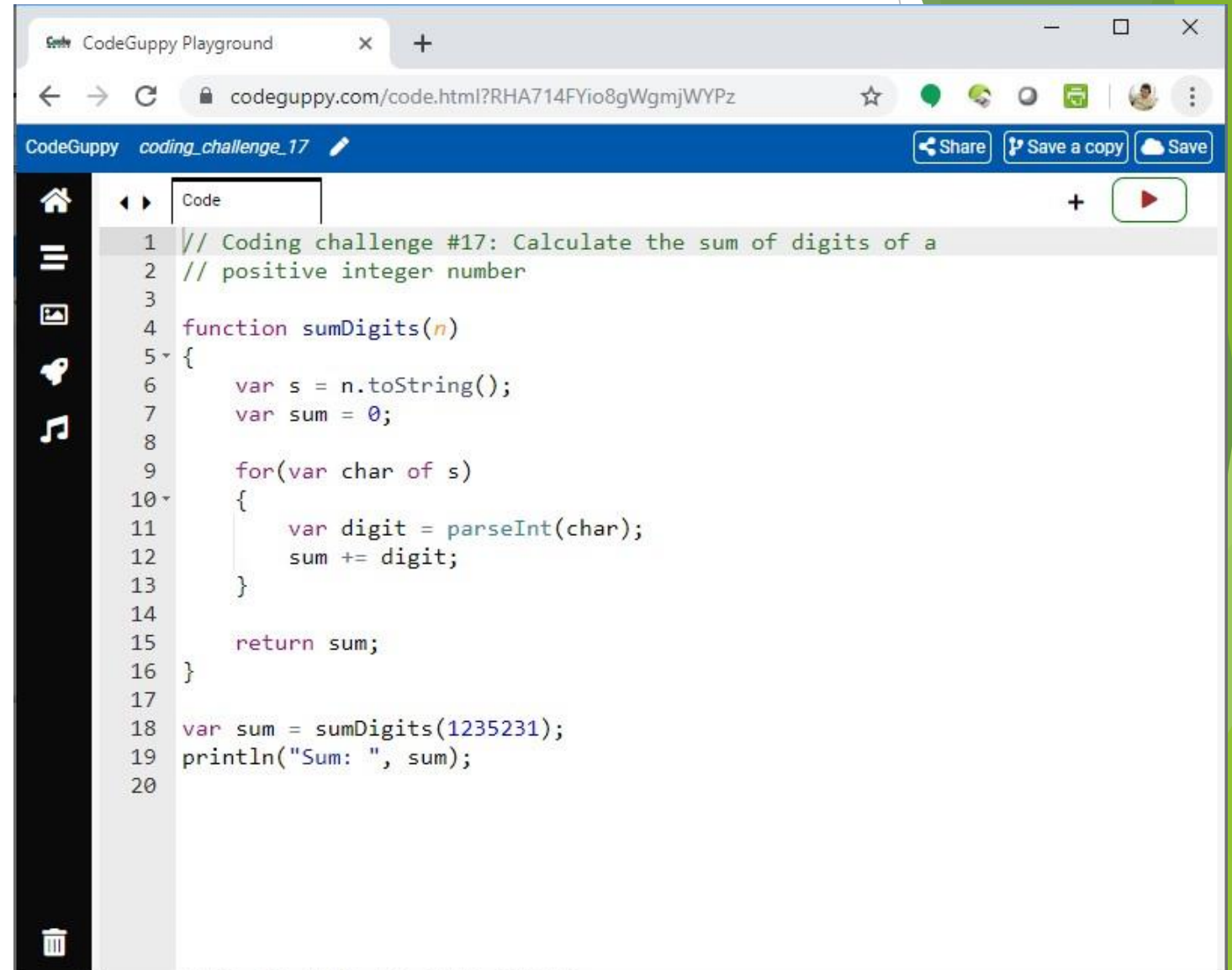
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?fRYsPEc2vcZTbIU8MKku". The page has a blue header with "CodeGuppy coding_challenge_16" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, image, speech, and music. The main area is a code editor with a "Code" tab and a "Run" button (a red play icon). The code is as follows:

```
1 // Coding challenge #16: Create a function that will return a Boolean
2 // specifying if a number is prime
3
4 function isPrime(n)
5 {
6     if (n < 2)
7         return false;
8
9     if (n == 2)
10        return true;
11
12    var maxDiv = Math.sqrt(n);
13
14    for(var i = 2; i <= maxDiv; i++)
15    {
16        if (n % i == 0)
17        {
18            return false;
19        }
20    }
21
22    return true;
23 }
24
25 println(2, " is prime? ", isPrime(2));
26 println(3, " is prime? ", isPrime(3));
27 println(4, " is prime? ", isPrime(4));
28 println(5, " is prime? ", isPrime(5));
29 println(9, " is prime? ", isPrime(9));
30
```



Coding challenge #17

Calculate the sum of digits
of a positive integer number

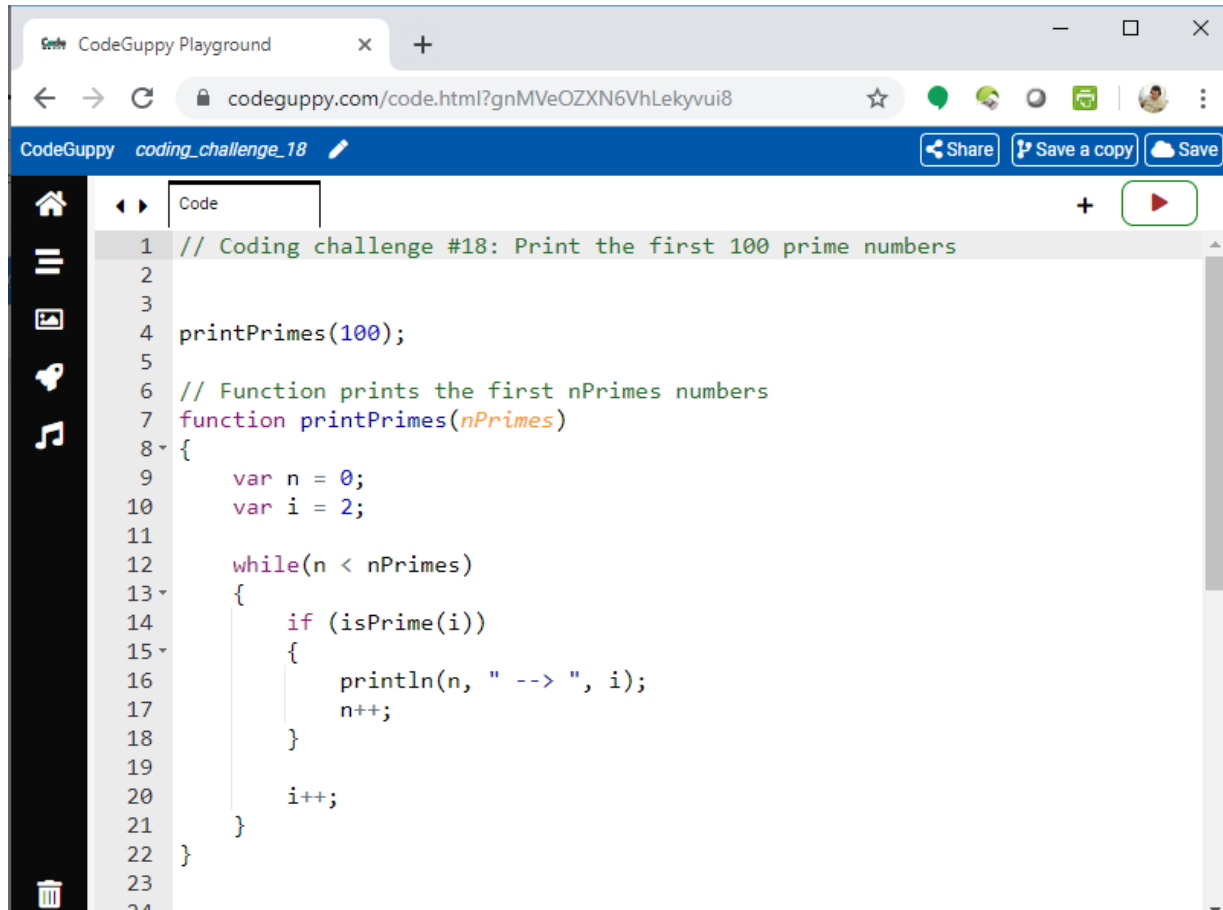


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?RHA714FYio8gWgmjWYPz". The page has a blue header with the text "CodeGuppy coding_challenge_17" and buttons for "Share", "Save a copy", and "Save". On the left side, there is a dark sidebar with icons for home, run, save, and other functions. The main area displays a code editor with the following JavaScript code:

```
1 // Coding challenge #17: Calculate the sum of digits of a
2 // positive integer number
3
4 function sumDigits(n)
5 {
6     var s = n.toString();
7     var sum = 0;
8
9     for(var char of s)
10    {
11        var digit = parseInt(char);
12        sum += digit;
13    }
14
15    return sum;
16 }
17
18 var sum = sumDigits(1235231);
19 println("Sum: ", sum);
20
```



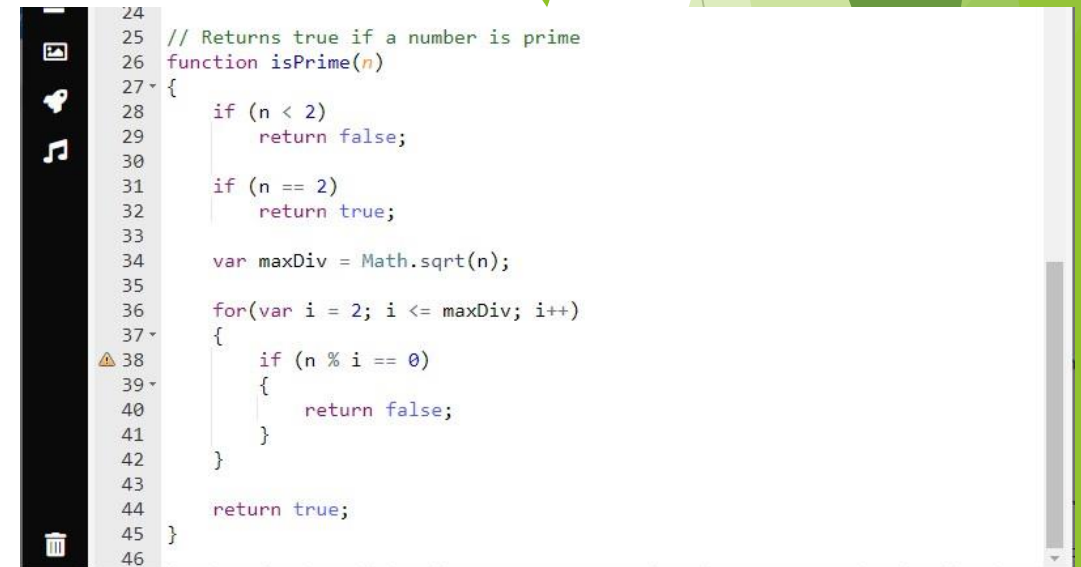
Coding challenge #18: Print the first 100 prime numbers



The screenshot shows a web browser window with the CodeGuppy Playground interface. The code editor contains the following JavaScript code:

```
1 // Coding challenge #18: Print the first 100 prime numbers
2
3
4 printPrimes(100);
5
6 // Function prints the first nPrimes numbers
7 function printPrimes(nPrimes)
8 {
9     var n = 0;
10    var i = 2;
11
12    while(n < nPrimes)
13    {
14        if (isPrime(i))
15        {
16            println(n, " --> ", i);
17            n++;
18        }
19        i++;
20    }
21 }
22
23
24
```

A dashed green arrow originates from the `isPrime(i)` call on line 14 and points towards the right, indicating a call to the `isPrime` function defined in the adjacent code block.



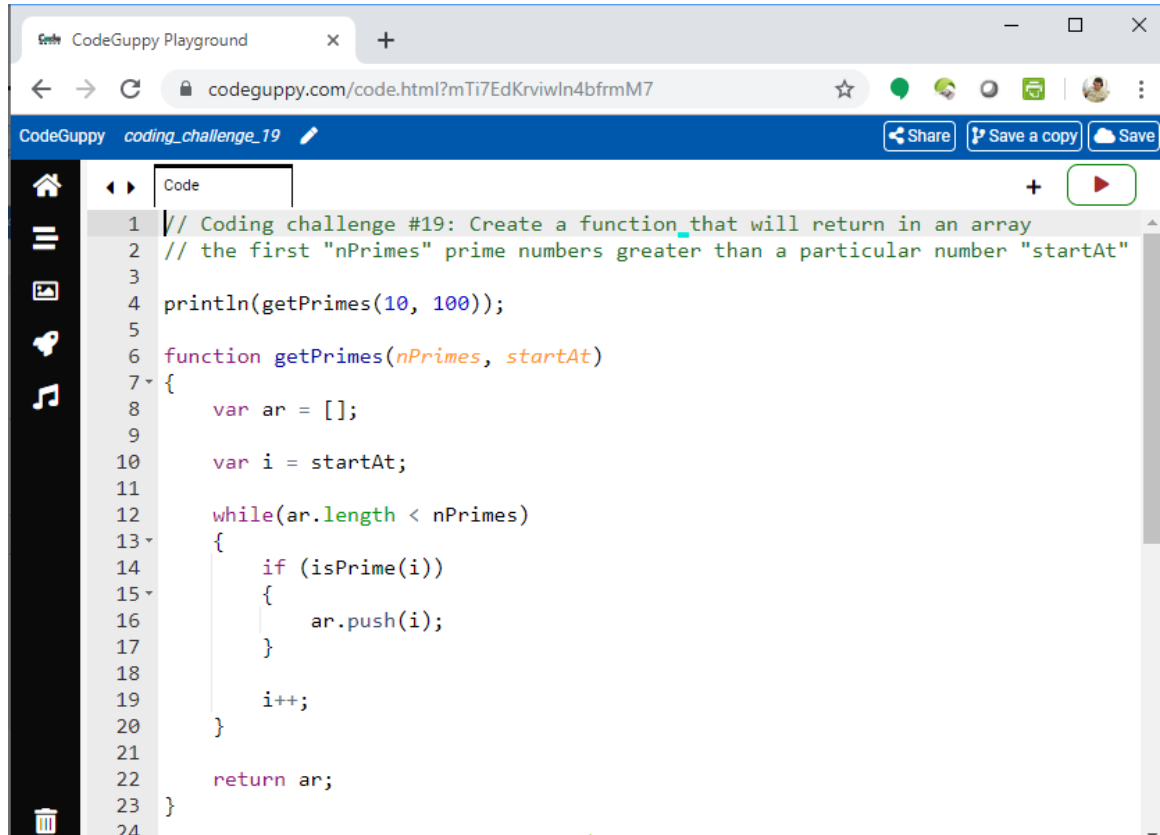
The screenshot shows a second code editor window containing the `isPrime` function implementation:

```
24
25 // Returns true if a number is prime
26 function isPrime(n)
27 {
28     if (n < 2)
29         return false;
30
31     if (n == 2)
32         return true;
33
34     var maxDiv = Math.sqrt(n);
35
36     for(var i = 2; i <= maxDiv; i++)
37     {
38         if (n % i == 0)
39         {
40             return false;
41         }
42     }
43
44     return true;
45 }
46
```

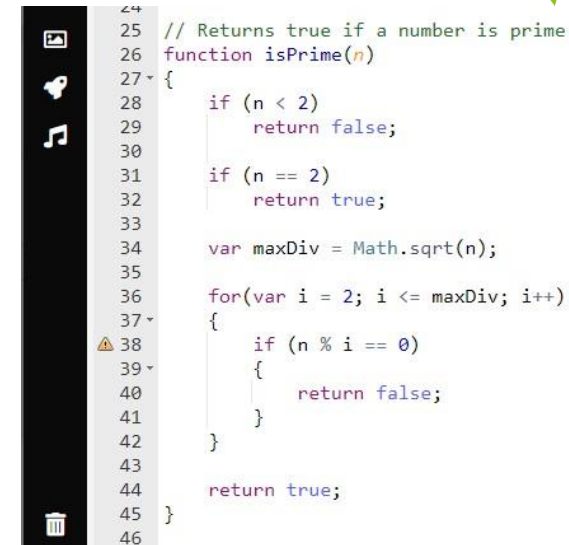
A dashed green arrow points from the `isPrime` function definition in this block back to the `isPrime(i)` call in the first code block, completing the call chain.



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



```
1 // Coding challenge #19: Create a function that will return in an array
2 // the first "nPrimes" prime numbers greater than a particular number "startAt"
3
4 println(getPrimes(10, 100));
5
6 function getPrimes(nPrimes, startAt)
7 {
8     var ar = [];
9
10    var i = startAt;
11
12    while(ar.length < nPrimes)
13    {
14        if (isPrime(i))
15        {
16            ar.push(i);
17        }
18
19        i++;
20    }
21
22    return ar;
23 }
24
```

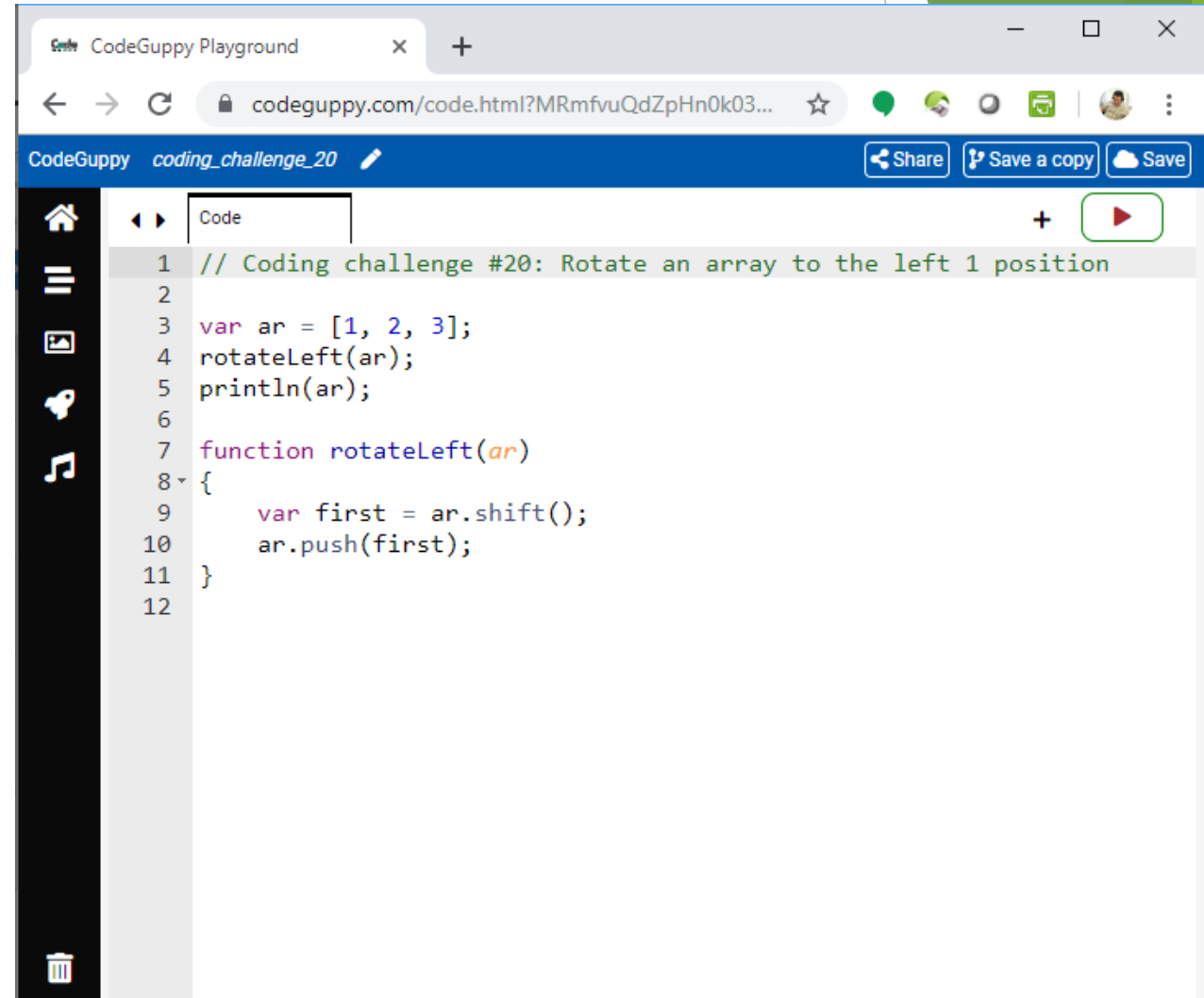


```
24
25 // Returns true if a number is prime
26 function isPrime(n)
27 {
28     if (n < 2)
29         return false;
30
31     if (n == 2)
32         return true;
33
34     var maxDiv = Math.sqrt(n);
35
36     for(var i = 2; i <= maxDiv; i++)
37     {
38         if (n % i == 0)
39         {
40             return false;
41         }
42     }
43
44     return true;
45 }
46
```



Coding challenge #20

Rotate an array to the left 1 position



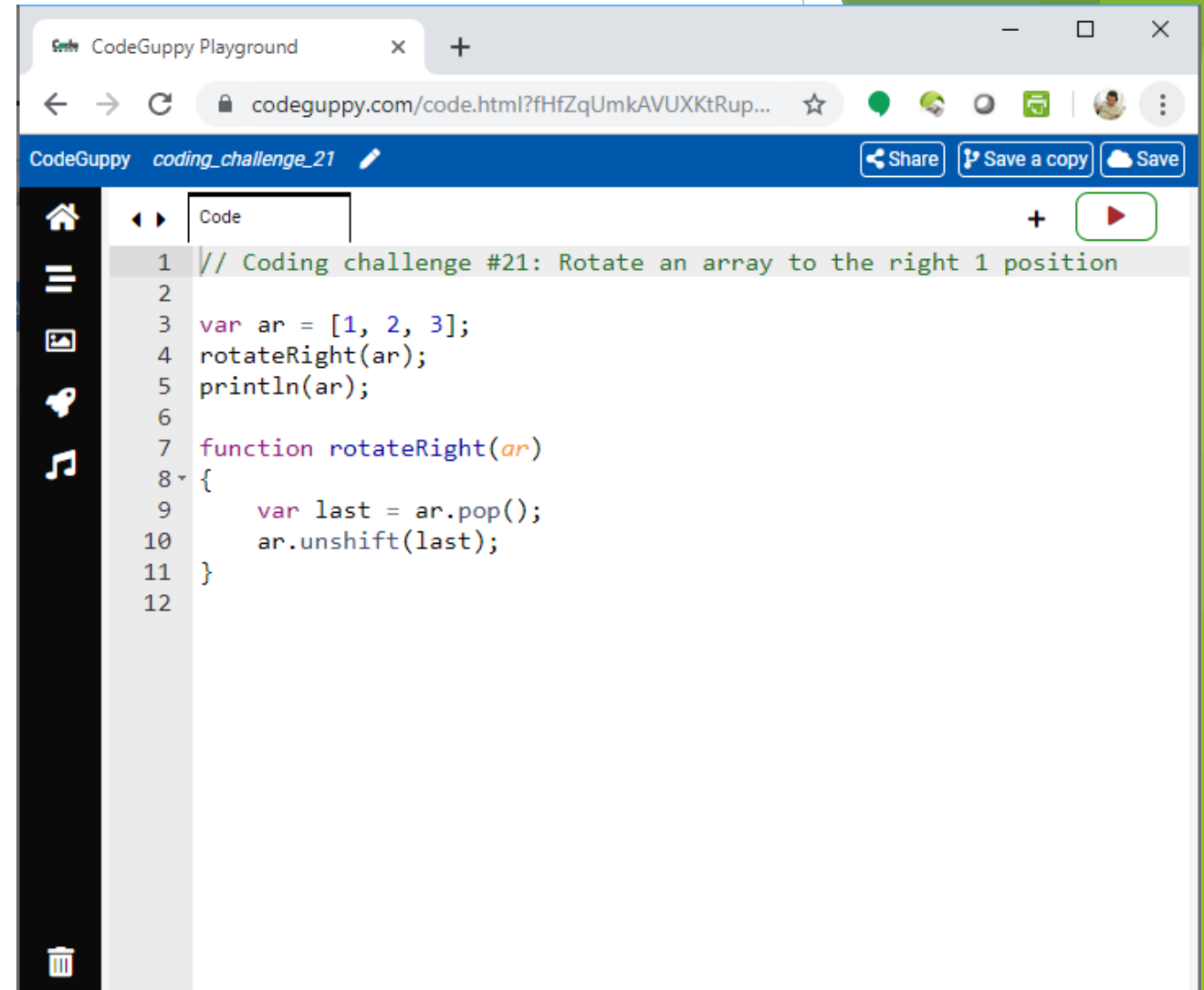
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?MRmfvuQdZpHn0k03...". The browser's address bar and navigation icons are visible. Below the browser window, there is a blue header bar with the text "CodeGuppy coding_challenge_20" and buttons for "Share", "Save a copy", and "Save". The main area of the browser displays a code editor with a dark sidebar on the left containing icons for home, list, file, share, and music. The code editor has a tab labeled "Code" and a red play button in the top right corner. The code is as follows:

```
1 // Coding challenge #20: Rotate an array to the left 1 position
2
3 var ar = [1, 2, 3];
4 rotateLeft(ar);
5 println(ar);
6
7 function rotateLeft(ar)
8 {
9     var first = ar.shift();
10    ar.push(first);
11 }
12
```



Coding challenge #21

Rotate an array to the right 1 position



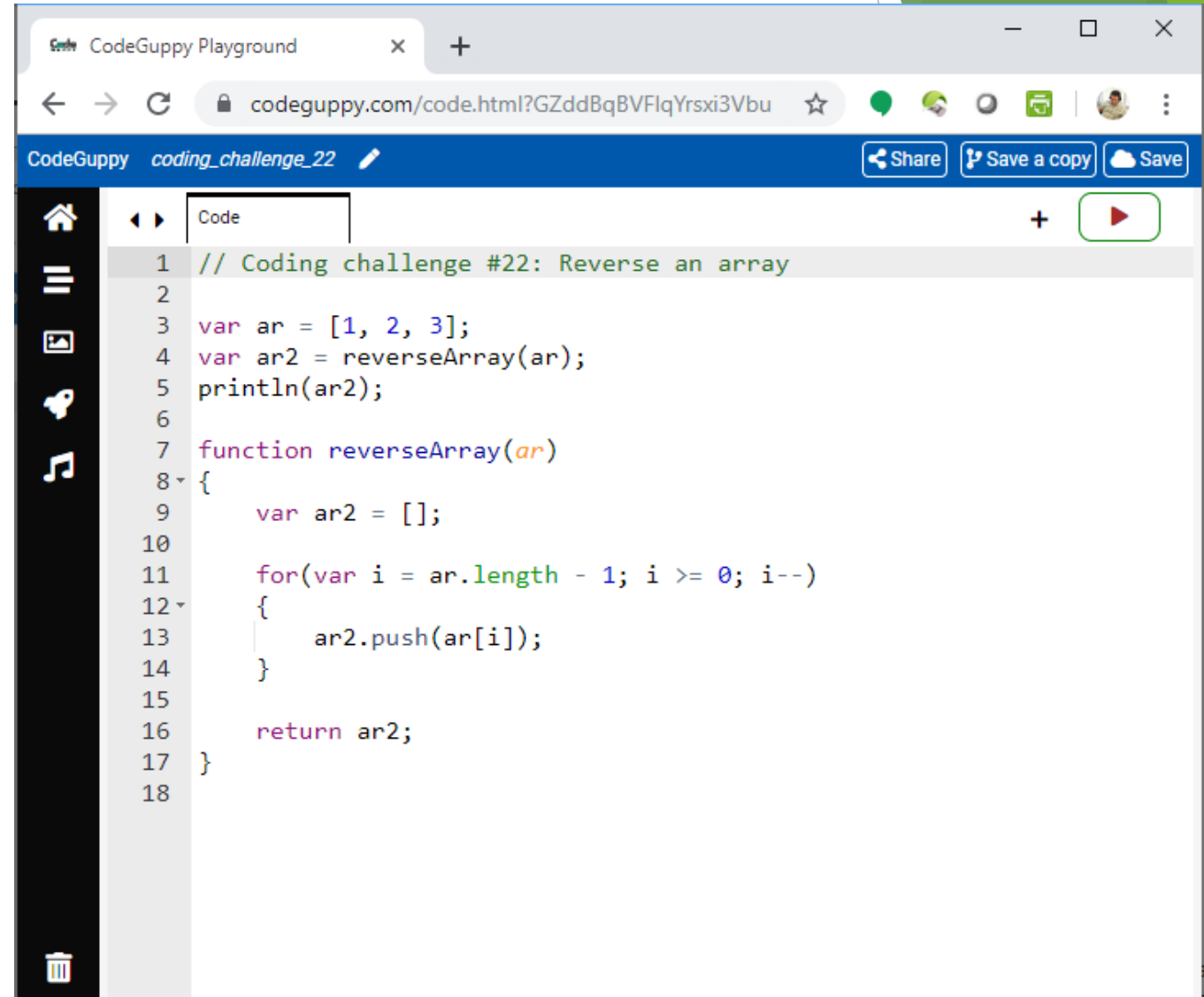
The screenshot shows a web browser window with the address bar displaying 'codeguppy.com/code.html?fHfZqUmKAVUXKtRup...'. The page title is 'CodeGuppy coding_challenge_21'. The main content area is a code editor with a dark sidebar on the left containing icons for home, list, image, chat, and music. The code editor has a 'Code' tab and a 'Run' button (a red triangle in a green circle). The code is as follows:

```
1 // Coding challenge #21: Rotate an array to the right 1 position
2
3 var ar = [1, 2, 3];
4 rotateRight(ar);
5 println(ar);
6
7 function rotateRight(ar)
8 {
9     var last = ar.pop();
10    ar.unshift(last);
11 }
12
```



Coding challenge #22

Reverse an array



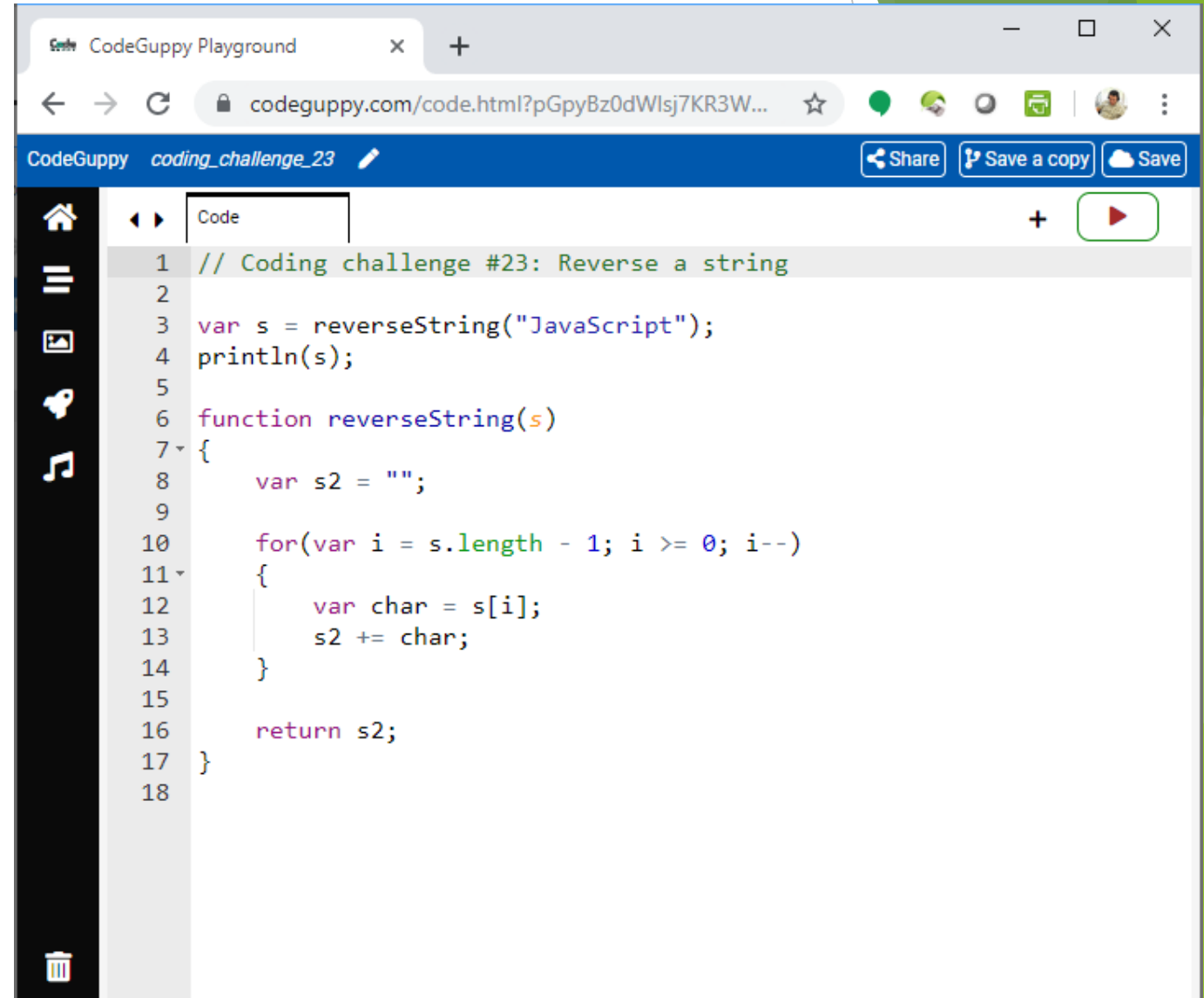
The screenshot shows a web browser window with the address bar displaying `codeguppy.com/code.html?GZddBqBVFlqYrsxi3Vbu`. The page title is "CodeGuppy coding_challenge_22". The main content area is a code editor with a dark sidebar on the left containing icons for home, list, image, chat, and music. The code editor has a tab labeled "Code" and a "Run" button (a green circle with a red play icon) in the top right corner. The code is as follows:

```
1 // Coding challenge #22: Reverse an array
2
3 var ar = [1, 2, 3];
4 var ar2 = reverseArray(ar);
5 println(ar2);
6
7 function reverseArray(ar)
8 {
9     var ar2 = [];
10
11     for(var i = ar.length - 1; i >= 0; i--)
12     {
13         ar2.push(ar[i]);
14     }
15
16     return ar2;
17 }
18
```



Coding challenge #23

Reverse a string



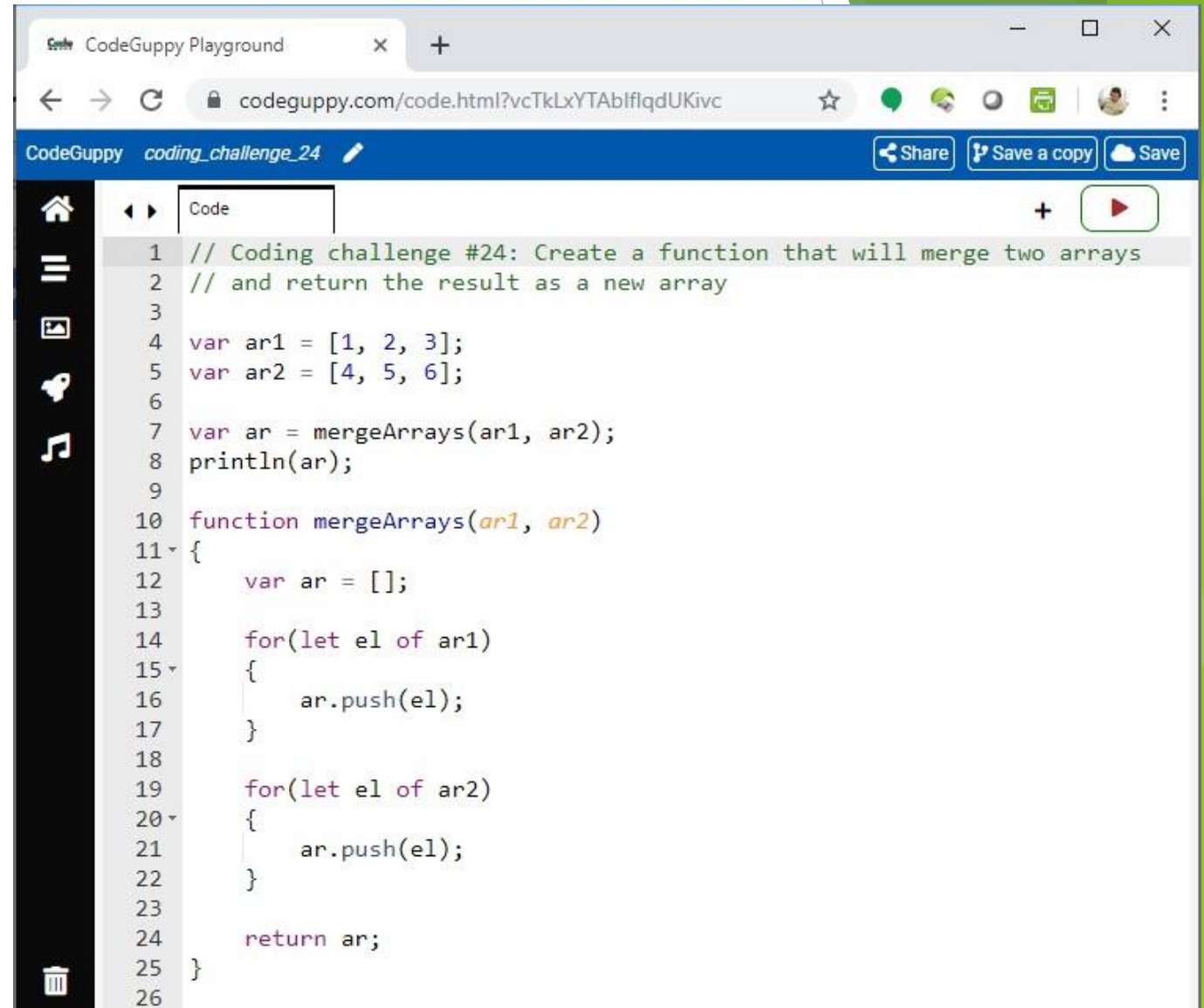
The screenshot shows a web browser window with the CodeGuppy Playground interface. The address bar shows the URL `codeguppy.com/code.html?pGpyBz0dWlsj7KR3W...`. The page title is "CodeGuppy coding_challenge_23". The code editor contains the following JavaScript code:

```
1 // Coding challenge #23: Reverse a string
2
3 var s = reverseString("JavaScript");
4 println(s);
5
6 function reverseString(s)
7 {
8     var s2 = "";
9
10    for(var i = s.length - 1; i >= 0; i--)
11    {
12        var char = s[i];
13        s2 += char;
14    }
15
16    return s2;
17 }
18
```



Coding challenge #24

Create a function that will merge two arrays and return the result as a new array



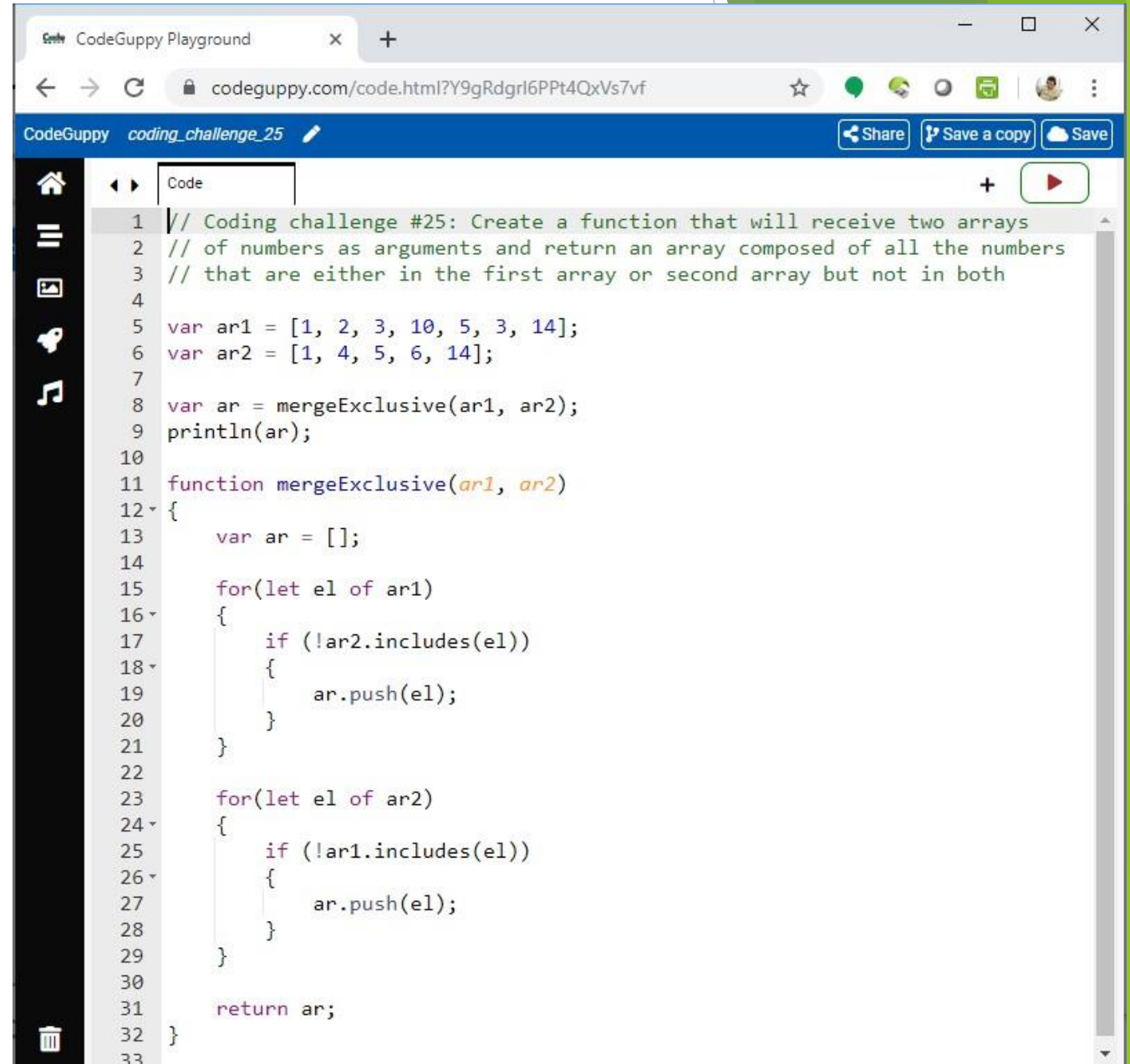
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?vcTkLxYTABlflqdUKivc". The page has a blue header with the text "CodeGuppy coding_challenge_24" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, file, search, and trash. The main area is a code editor with a "Code" tab and a "Run" button (a red play icon). The code is as follows:

```
1 // Coding challenge #24: Create a function that will merge two arrays
2 // and return the result as a new array
3
4 var ar1 = [1, 2, 3];
5 var ar2 = [4, 5, 6];
6
7 var ar = mergeArrays(ar1, ar2);
8 println(ar);
9
10 function mergeArrays(ar1, ar2)
11 {
12     var ar = [];
13
14     for(let el of ar1)
15     {
16         ar.push(el);
17     }
18
19     for(let el of ar2)
20     {
21         ar.push(el);
22     }
23
24     return ar;
25 }
26
```



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



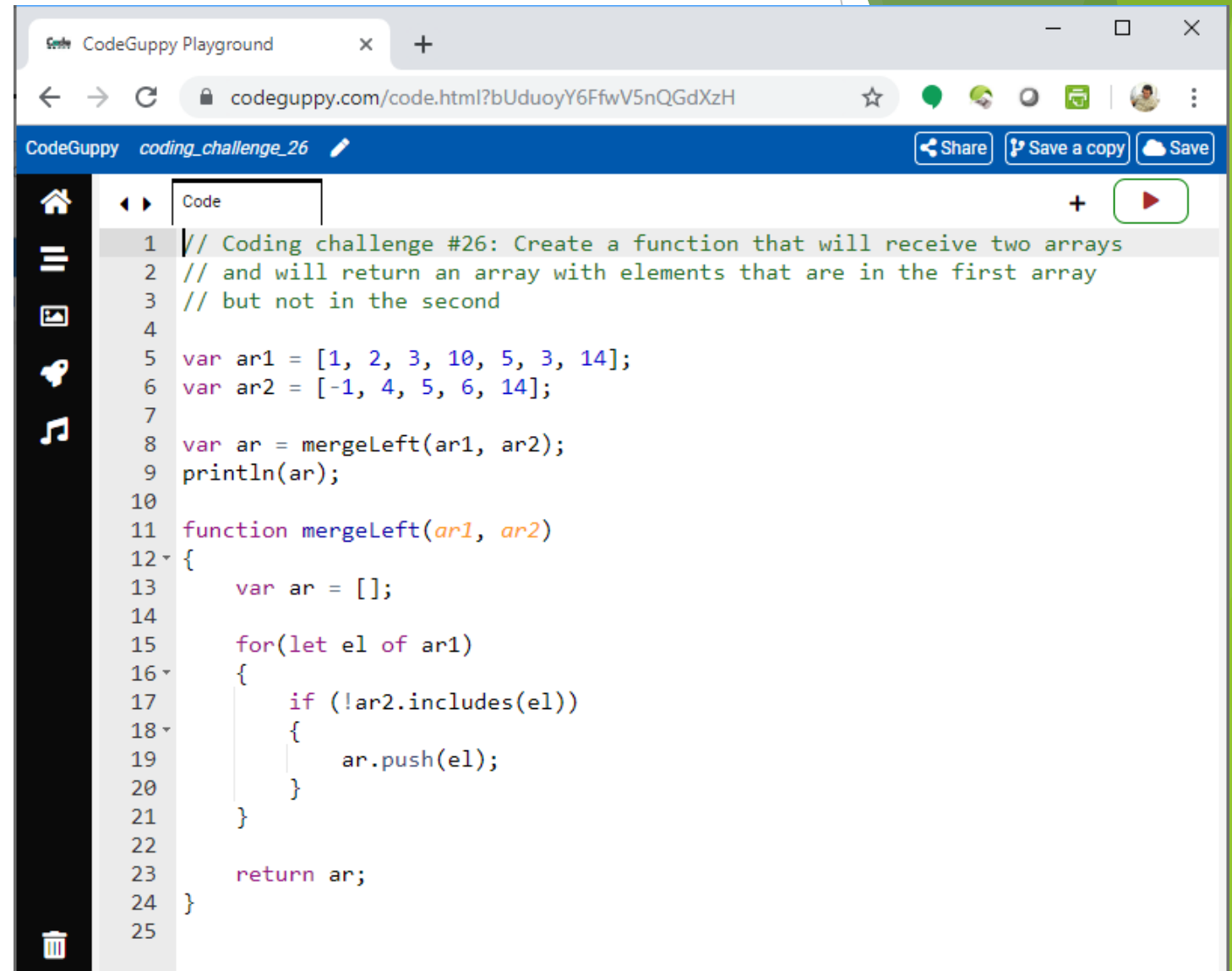
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?Y9gRdgrl6PPt4QxVs7vf". The page has a blue header with the text "CodeGuppy coding_challenge_25" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, search, and other functions. The main area is a code editor with a light blue background and a dark left margin showing line numbers from 1 to 33. The code is as follows:

```
1 // Coding challenge #25: Create a function that will receive two arrays
2 // of numbers as arguments and return an array composed of all the numbers
3 // that are either in the first array or second array but not in both
4
5 var ar1 = [1, 2, 3, 10, 5, 3, 14];
6 var ar2 = [1, 4, 5, 6, 14];
7
8 var ar = mergeExclusive(ar1, ar2);
9 println(ar);
10
11 function mergeExclusive(ar1, ar2)
12 {
13     var ar = [];
14
15     for(let el of ar1)
16     {
17         if (!ar2.includes(el))
18         {
19             ar.push(el);
20         }
21     }
22
23     for(let el of ar2)
24     {
25         if (!ar1.includes(el))
26         {
27             ar.push(el);
28         }
29     }
30
31     return ar;
32 }
33
```



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



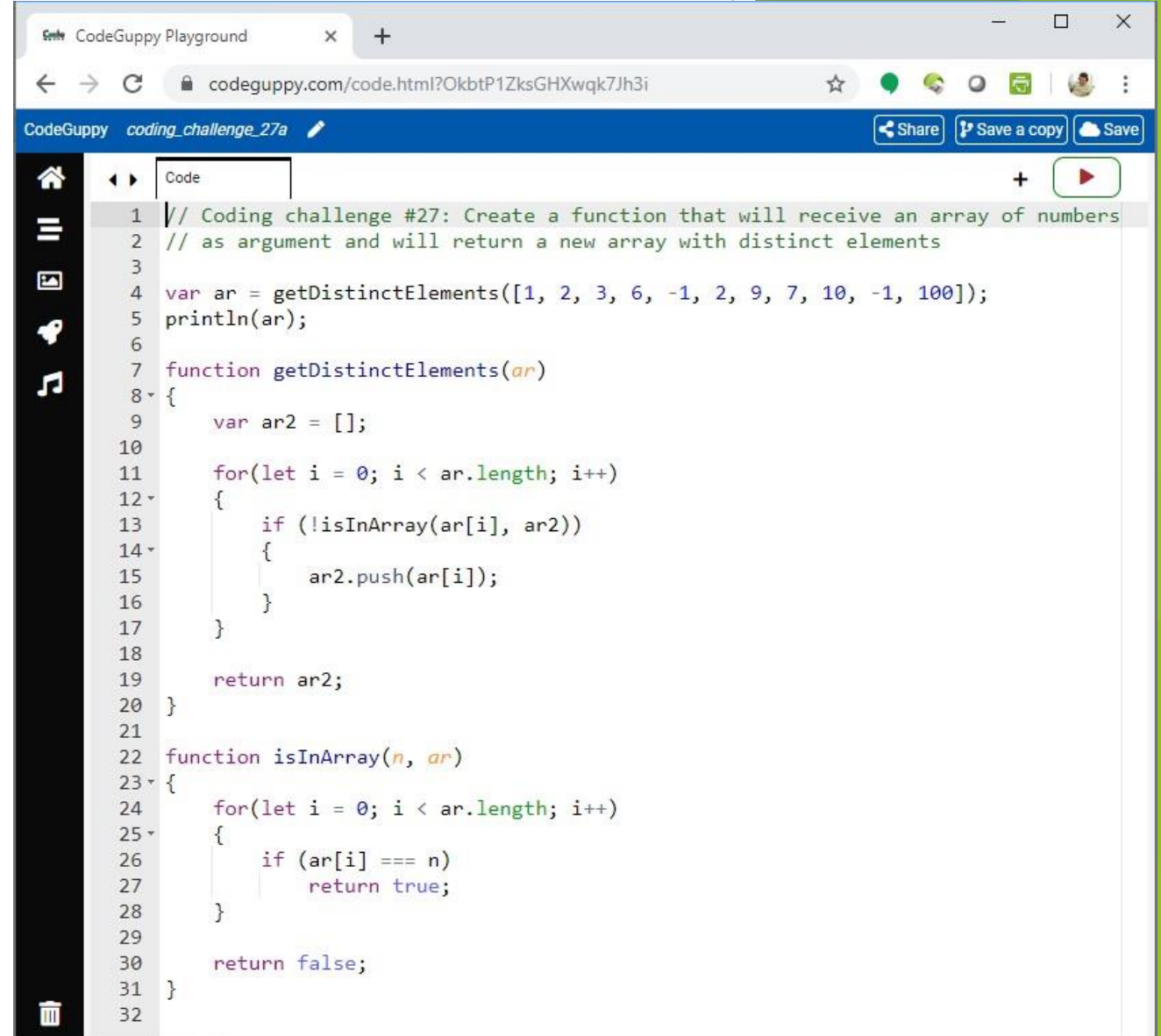
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?bUduoyY6FfwV5nQGdXzH`. The page has a blue header with the text "CodeGuppy coding_challenge_26" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, image, bell, music, and trash. The main area is a code editor with a tab labeled "Code" and a play button. The code is as follows:

```
1 // Coding challenge #26: Create a function that will receive two arrays
2 // and will return an array with elements that are in the first array
3 // but not in the second
4
5 var ar1 = [1, 2, 3, 10, 5, 3, 14];
6 var ar2 = [-1, 4, 5, 6, 14];
7
8 var ar = mergeLeft(ar1, ar2);
9 println(ar);
10
11 function mergeLeft(ar1, ar2)
12 {
13     var ar = [];
14
15     for(let el of ar1)
16     {
17         if (!ar2.includes(el))
18         {
19             ar.push(el);
20         }
21     }
22
23     return ar;
24 }
25
```



Coding challenge #27a

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



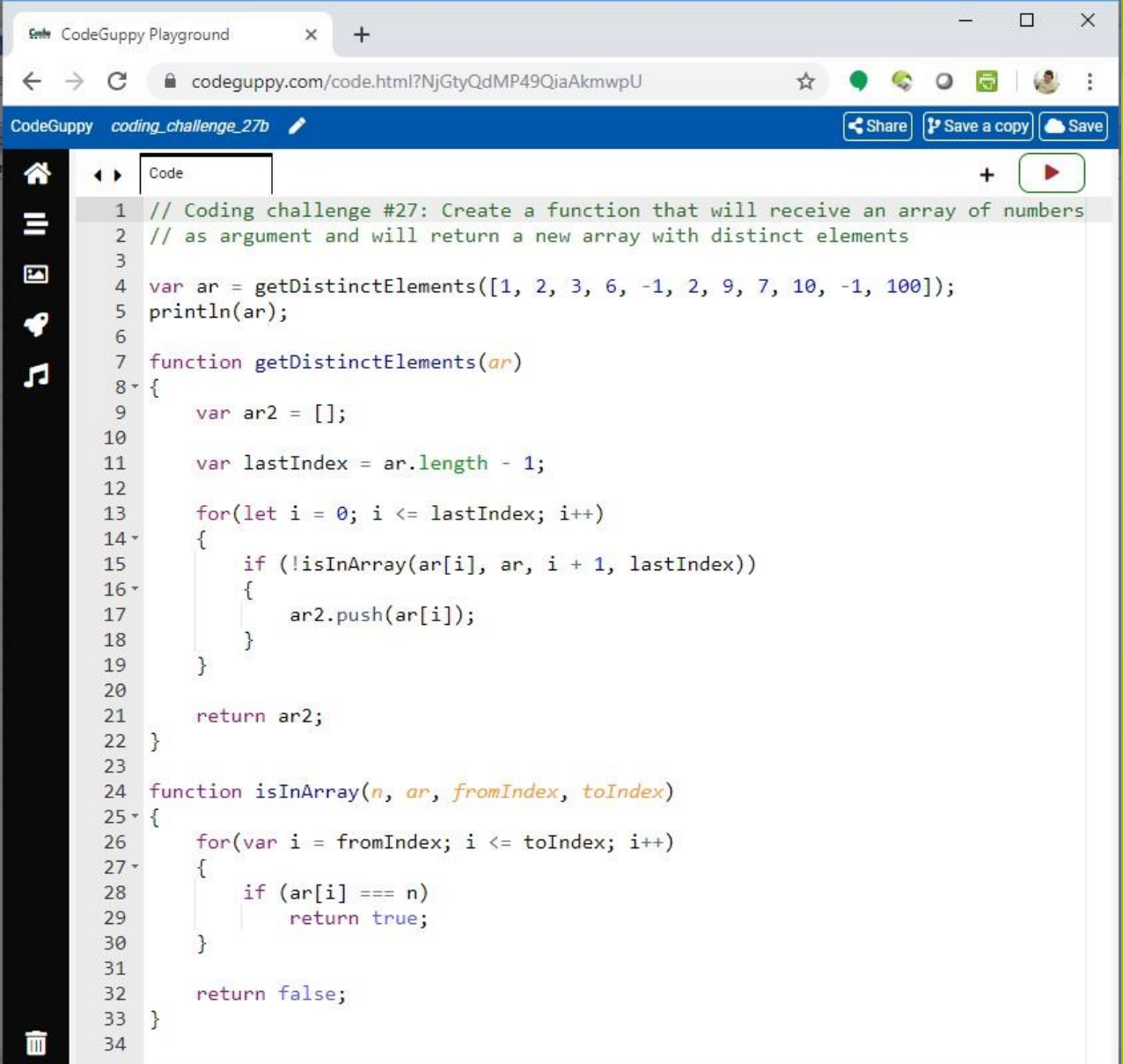
The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?OkbtP1ZksGHXwqk7Jh3i". The page has a blue header with the text "CodeGuppy coding_challenge_27a" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, image, speech, and music. The main area is a code editor with a "Code" tab and a "Run" button (a red play icon). The code is as follows:

```
1 // Coding challenge #27: Create a function that will receive an array of numbers
2 // as argument and will return a new array with distinct elements
3
4 var ar = getDistinctElements([1, 2, 3, 6, -1, 2, 9, 7, 10, -1, 100]);
5 println(ar);
6
7 function getDistinctElements(ar)
8 {
9     var ar2 = [];
10
11     for(let i = 0; i < ar.length; i++)
12     {
13         if (!isArray(ar[i], ar2))
14         {
15             ar2.push(ar[i]);
16         }
17     }
18
19     return ar2;
20 }
21
22 function isArray(n, ar)
23 {
24     for(let i = 0; i < ar.length; i++)
25     {
26         if (ar[i] === n)
27             return true;
28     }
29
30     return false;
31 }
32
```



Coding challenge #27b

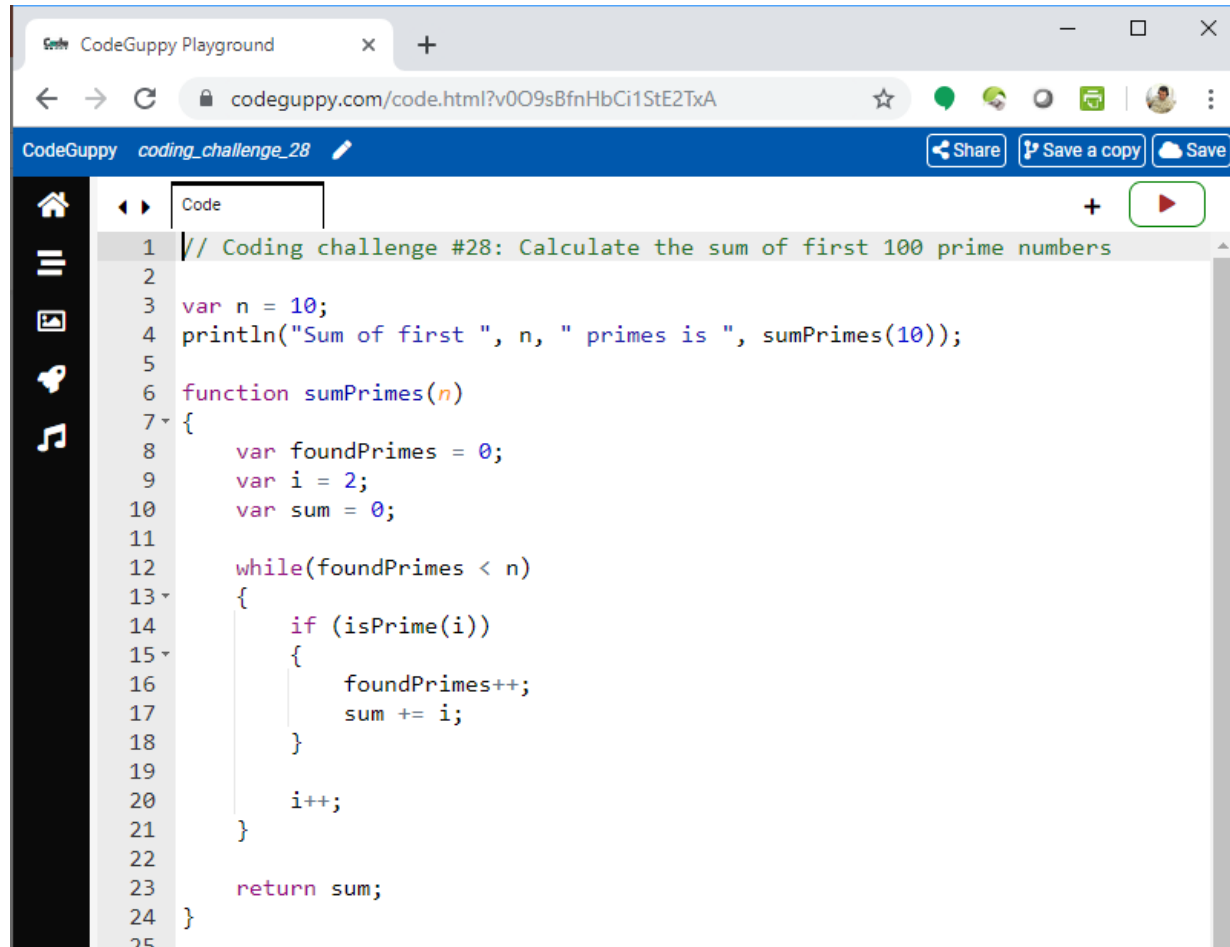
Create a function that will receive an array of numbers as argument and will return a new array with distinct elements



```
1 // Coding challenge #27: Create a function that will receive an array of numbers
2 // as argument and will return a new array with distinct elements
3
4 var ar = getDistinctElements([1, 2, 3, 6, -1, 2, 9, 7, 10, -1, 100]);
5 println(ar);
6
7 function getDistinctElements(ar)
8 {
9     var ar2 = [];
10
11     var lastIndex = ar.length - 1;
12
13     for(let i = 0; i <= lastIndex; i++)
14     {
15         if (!isInArray(ar[i], ar, i + 1, lastIndex))
16         {
17             ar2.push(ar[i]);
18         }
19     }
20
21     return ar2;
22 }
23
24 function isInArray(n, ar, fromIndex, toIndex)
25 {
26     for(var i = fromIndex; i <= toIndex; i++)
27     {
28         if (ar[i] === n)
29             return true;
30     }
31
32     return false;
33 }
34
```

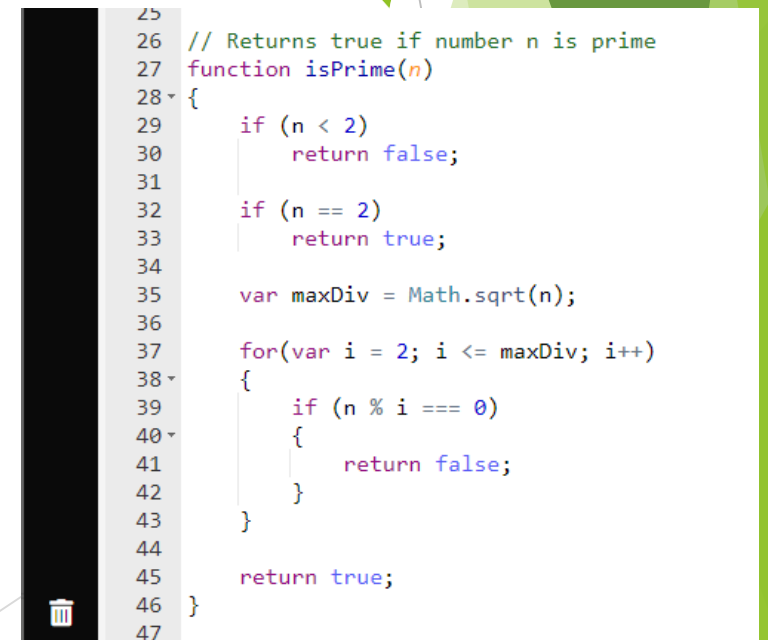


Coding challenge #28: Calculate the sum of first 100 prime numbers



The screenshot shows a web browser window with the CodeGuppy Playground interface. The address bar shows the URL `codeguppy.com/code.html?v009sBfnHbCi1StE2TxA`. The page title is "CodeGuppy coding_challenge_28". The code editor contains the following JavaScript code:

```
1 // Coding challenge #28: Calculate the sum of first 100 prime numbers
2
3 var n = 10;
4 println("Sum of first ", n, " primes is ", sumPrimes(10));
5
6 function sumPrimes(n)
7 {
8     var foundPrimes = 0;
9     var i = 2;
10    var sum = 0;
11
12    while(foundPrimes < n)
13    {
14        if (isPrime(i))
15        {
16            foundPrimes++;
17            sum += i;
18        }
19
20        i++;
21    }
22
23    return sum;
24 }
25
```

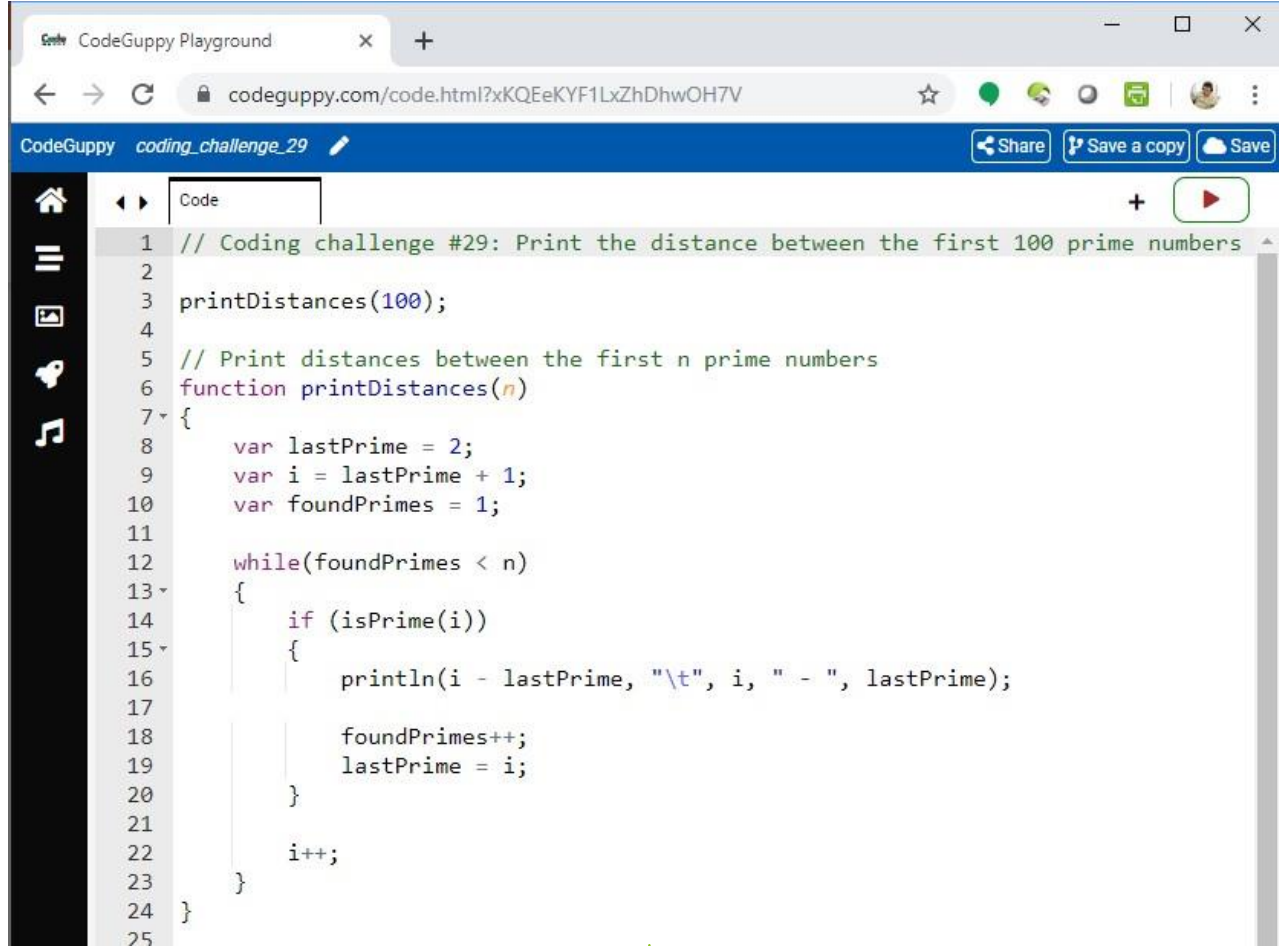


The screenshot shows a code editor with the `isPrime` function implementation. A dashed green arrow points from the `isPrime(i)` call in the main code to this function definition.

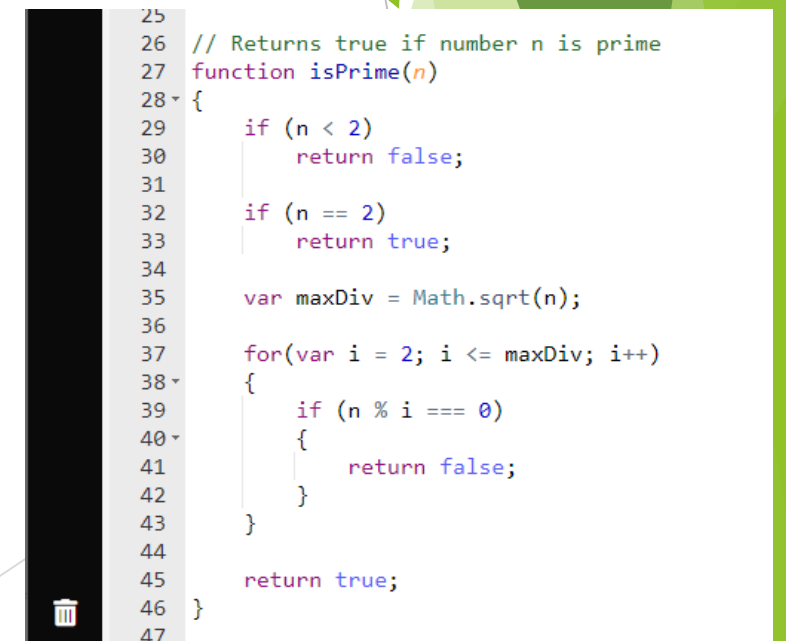
```
25
26 // Returns true if number n is prime
27 function isPrime(n)
28 {
29     if (n < 2)
30         return false;
31
32     if (n == 2)
33         return true;
34
35     var maxDiv = Math.sqrt(n);
36
37     for(var i = 2; i <= maxDiv; i++)
38     {
39         if (n % i === 0)
40         {
41             return false;
42         }
43     }
44
45     return true;
46 }
47
```



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



```
1 // Coding challenge #29: Print the distance between the first 100 prime numbers
2
3 printDistances(100);
4
5 // Print distances between the first n prime numbers
6 function printDistances(n)
7 {
8     var lastPrime = 2;
9     var i = lastPrime + 1;
10    var foundPrimes = 1;
11
12    while(foundPrimes < n)
13    {
14        if (isPrime(i))
15        {
16            println(i - lastPrime, "\t", i, " - ", lastPrime);
17
18            foundPrimes++;
19            lastPrime = i;
20        }
21
22        i++;
23    }
24 }
25
```



```
25
26 // Returns true if number n is prime
27 function isPrime(n)
28 {
29     if (n < 2)
30         return false;
31
32     if (n == 2)
33         return true;
34
35     var maxDiv = Math.sqrt(n);
36
37     for(var i = 2; i <= maxDiv; i++)
38     {
39         if (n % i === 0)
40         {
41             return false;
42         }
43     }
44
45     return true;
46 }
47
```



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.

```
CodeGuppy Playground
codeguppy.com/code.html?v5A0QBsdHaiAVA2CPN5y
coding_challenge_30a

1
2 // Coding challenge #30: Create a function that will add two
3 // positive numbers of indefinite size. The numbers are received as strings
4 // and the result should be also provided as string.
5
6 var n1 = "2909034221912398942349";
7 var n2 = "1290923909029309499";
8 var sum = add(n1, n2);
9
10 println(n1, "\n", n2, "\n", sum);
11
12 function add(sNumber1, sNumber2)
13 {
14     var s = "";
15     var carry = 0;
16
17     var maxSize = Math.max(sNumber1.length, sNumber2.length);
18
19     for(var i = 0; i < maxSize; i++)
20     {
21         var digit1 = digitFromRight(sNumber1, i);
22         var digit2 = digitFromRight(sNumber2, i);
23
24         var sum = digit1 + digit2 + carry;
25         var digitSum = sum % 10;
26         carry = sum >= 10 ? 1 : 0;
27
28         s = digitSum.toString() + s;
29     }
30
31     if (carry > 0)
32         s = carry + s;
33
34     return s;
35 }
36
```

```
36
37 function digitFromRight(s, digitNo)
38 {
39     if (digitNo >= s.length)
40         return 0;
41
42     var char = s[ s.length - 1 - digitNo ];
43     return parseInt(char);
44 }
45
```



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.

A screenshot of the CodeGuppy Playground web application. The browser address bar shows the URL 'codeguppy.com/code.html?yMQXcPgfrYxualxqQmZc'. The page title is 'CodeGuppy coding_challenge_30b'. There are buttons for 'Share', 'Save a copy', and 'Save'. The code editor shows a JavaScript function 'add' that takes two strings 'sNumber1' and 'sNumber2' and returns their sum as a string. The code includes comments, variable declarations, and a detailed loop for digit-by-digit addition with carry handling.

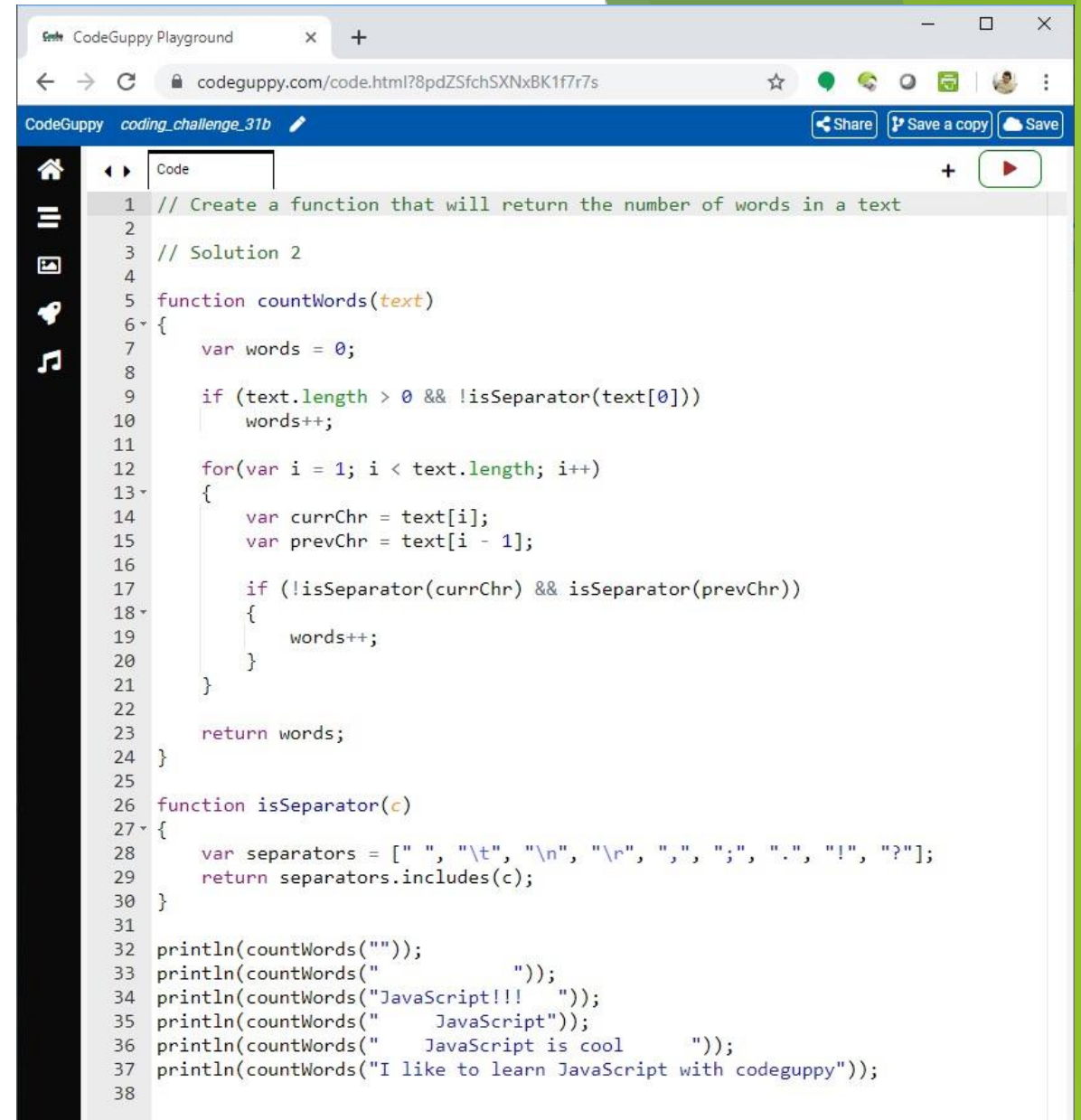
```
1
2 // Coding challenge #30: Create a function that will add two
3 // positive numbers of indefinite size. The numbers are received as strings
4 // and the result should be also provided as string.
5
6 var n1 = "2909034221912398942349";
7 var n2 = "1290923909029309499";
8 var sum = add(n1, n2);
9
10 println(n1);
11 println(n2);
12 println(sum);
13
14 function add(sNumber1, sNumber2)
15 {
16     var maxSize = Math.max(sNumber1.length, sNumber2.length);
17
18     var s1 = sNumber1.padStart(maxSize, "0");
19     var s2 = sNumber2.padStart(maxSize, "0");
20
21     var s = "";
22     var carry = 0;
23
24     for(var i = maxSize - 1; i >= 0; i--)
25     {
26         var digit1 = parseInt(s1[i]);
27         var digit2 = parseInt(s2[i]);
28
29         var sum = digit1 + digit2 + carry;
30         var digitSum = sum % 10;
31         carry = sum >= 10 ? 1 : 0;
32
33         s = digitSum.toString() + s;
34     }
35
36     if (carry > 0)
37         s = carry + s;
38
39     return s;
40 }
41
```


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

A screenshot of the CodeGuppy Playground web application. The browser address bar shows the URL 'codeguppy.com/code.html?r4kwkcWiHfzQZkM1qrX4'. The page title is 'CodeGuppy coding_challenge_31a'. The code editor shows a JavaScript function 'countWords' that iterates through a string and counts words based on separators. The function 'isSeparator' is also defined. The code is as follows:

```
1 // Create a function that will return the number of words in a text
2
3 // Solution 1
4
5 function countWords(text)
6 {
7     var wasSeparator = true;
8     var words = 0;
9
10    for(var c of text)
11    {
12        // if current character is separator then advance and
13        // set that the previous character was separator
14        if (isSeparator(c))
15        {
16            wasSeparator = true;
17            continue;
18        }
19
20        // if current character is not separator
21        // ... but if previous was separator...
22        if (wasSeparator)
23        {
24            words++;
25            wasSeparator = false;
26        }
27    }
28
29    return words;
30 }
31
32 function isSeparator(c)
33 {
34     var separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?"];
35     return separators.includes(c);
36 }
37
38 println(countWords(""));
39 println(countWords(" "));
40 println(countWords("JavaScript!!! "));
41 println(countWords("    JavaScript"));
42 println(countWords("    JavaScript is cool    "));
43 println(countWords("I like to learn JavaScript with codeguppy"));
44
```

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

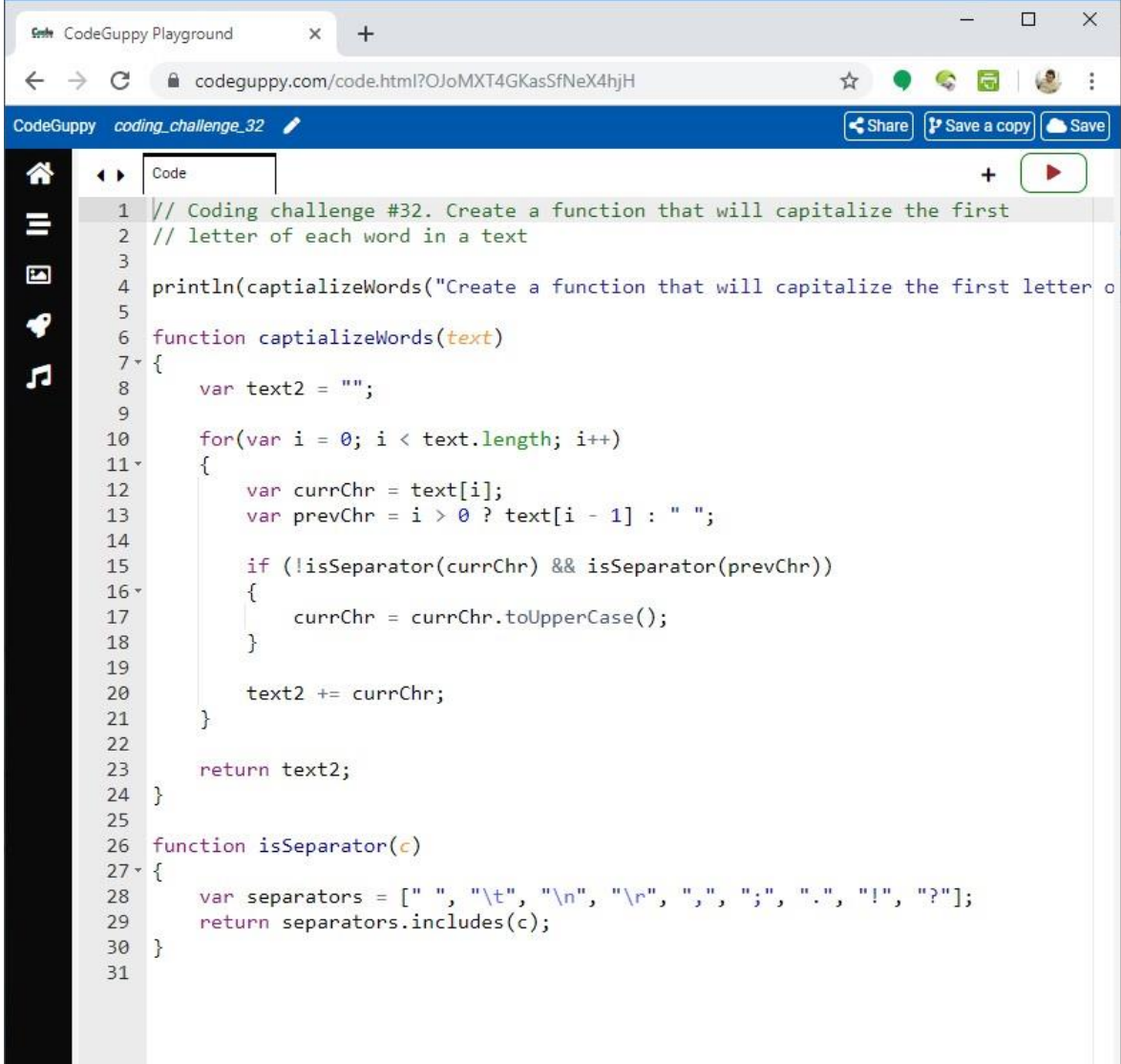


The screenshot shows the CodeGuppy Playground interface. The browser address bar displays the URL `codeguppy.com/code.html?8pdZSfchSXNxBK1f7r7s`. The playground title is `coding_challenge_31b`. The code editor contains the following JavaScript code:

```
1 // Create a function that will return the number of words in a text
2
3 // Solution 2
4
5 function countWords(text)
6 {
7     var words = 0;
8
9     if (text.length > 0 && !isSeparator(text[0]))
10         words++;
11
12     for(var i = 1; i < text.length; i++)
13     {
14         var currChr = text[i];
15         var prevChr = text[i - 1];
16
17         if (!isSeparator(currChr) && isSeparator(prevChr))
18         {
19             words++;
20         }
21     }
22
23     return words;
24 }
25
26 function isSeparator(c)
27 {
28     var separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?"];
29     return separators.includes(c);
30 }
31
32 println(countWords(""));
33 println(countWords(""));
34 println(countWords("JavaScript!!! "));
35 println(countWords("    JavaScript"));
36 println(countWords("    JavaScript is cool    "));
37 println(countWords("I like to learn JavaScript with codeguppy"));
38
```



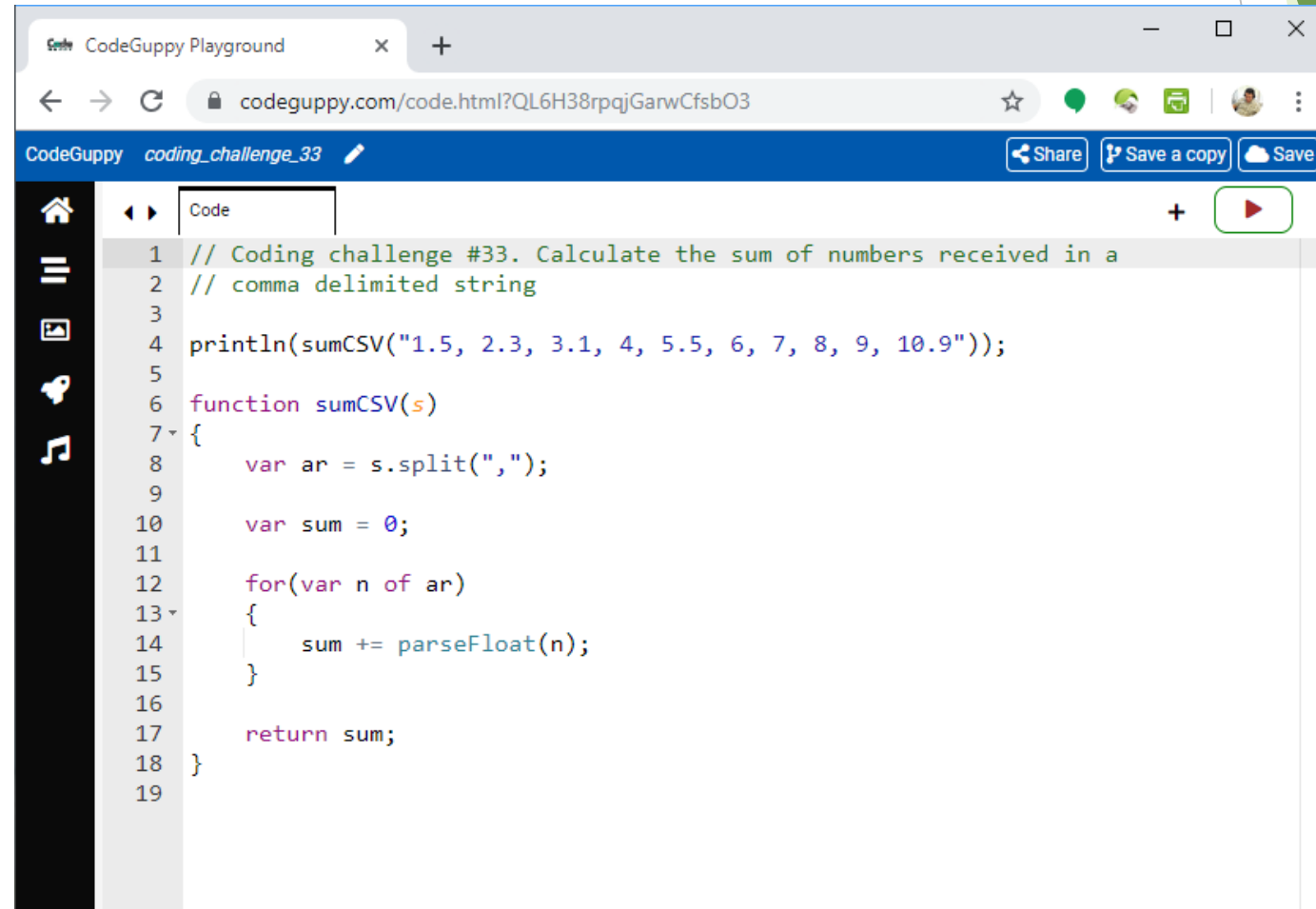
Coding challenge #32: Create a function that will capitalize the first letter of each word in a text



```
1 // Coding challenge #32. Create a function that will capitalize the first
2 // letter of each word in a text
3
4 println(captializeWords("Create a function that will capitalize the first letter o
5
6 function captializeWords(text)
7 {
8     var text2 = "";
9
10    for(var i = 0; i < text.length; i++)
11    {
12        var currChr = text[i];
13        var prevChr = i > 0 ? text[i - 1] : " ";
14
15        if (!isSeparator(currChr) && isSeparator(prevChr))
16        {
17            currChr = currChr.toUpperCase();
18        }
19
20        text2 += currChr;
21    }
22
23    return text2;
24 }
25
26 function isSeparator(c)
27 {
28     var separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?"];
29     return separators.includes(c);
30 }
31
```



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



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?QL6H38rpqjGarwCfsbO3`. The browser's address bar and navigation buttons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for home, list, image, chat, and music. The main area is labeled "Code" and contains the following JavaScript code:

```
1 // Coding challenge #33. Calculate the sum of numbers received in a
2 // comma delimited string
3
4 println(sumCSV("1.5, 2.3, 3.1, 4, 5.5, 6, 7, 8, 9, 10.9"));
5
6 function sumCSV(s)
7 {
8     var ar = s.split(",");
9
10    var sum = 0;
11
12    for(var n of ar)
13    {
14        sum += parseFloat(n);
15    }
16
17    return sum;
18 }
19
```

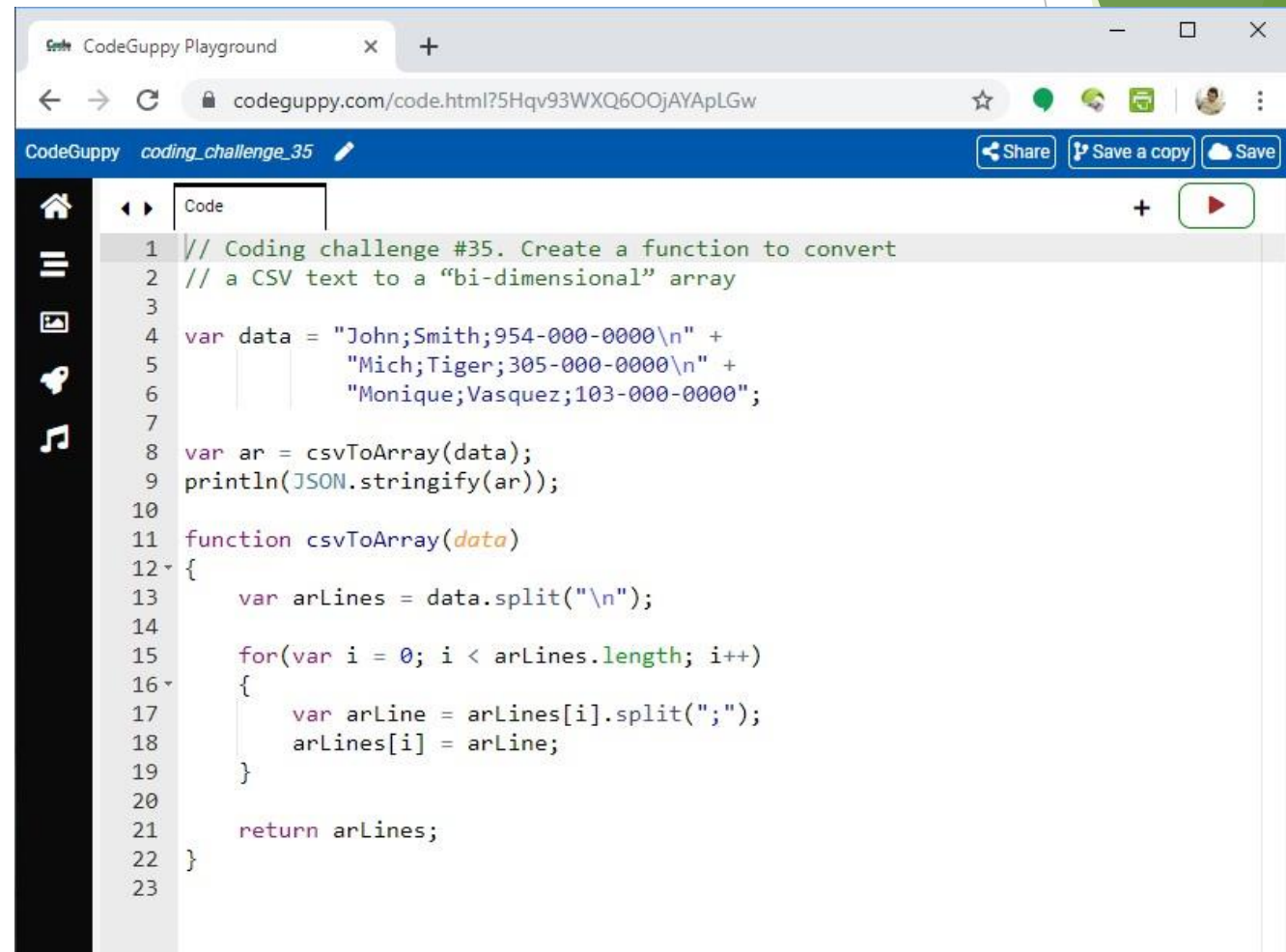


Coding challenge #34: Create a function that will return an array with words inside a text

A screenshot of the CodeGuppy Playground web application. The browser address bar shows the URL 'codeguppy.com/code.html?UI0E4OGnkyTZnoszAzf'. The page title is 'CodeGuppy coding_challenge_34'. The code editor contains a JavaScript solution for the challenge. The code defines a 'getWords' function that takes a text string and returns an array of words. It uses a loop to iterate through the text, identifying word boundaries based on a list of separators. The 'isSeparator' function checks if a character is in the separator list. The 'getWords' function initializes a start index and an array, then iterates through the text, pushing words into the array when a separator is found. The final output is the array of words.

```
1 // Coding challenge #34. Create a function that will return
2 // an array with words inside a text
3
4 var text = "Create a function, that will return an array (of string), with the wor
5
6 println(getWords(text));
7
8 function getWords(text)
9 {
10     let startWord = -1;
11     let ar = [];
12
13     for(let i = 0; i <= text.length; i++)
14     {
15         let c = i < text.length ? text[i] : " ";
16
17         if (!isSeparator(c) && startWord < 0)
18         {
19             startWord = i;
20         }
21
22         if (isSeparator(c) && startWord >= 0)
23         {
24             let word = text.substring(startWord, i);
25             ar.push(word);
26
27             startWord = -1;
28         }
29     }
30
31     return ar;
32 }
33
34 function isSeparator(c)
35 {
36     var separators = [" ", "\t", "\n", "\r", ",", ";", ".", "!", "?", "(", ")"];
37     return separators.includes(c);
38 }
39
```

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

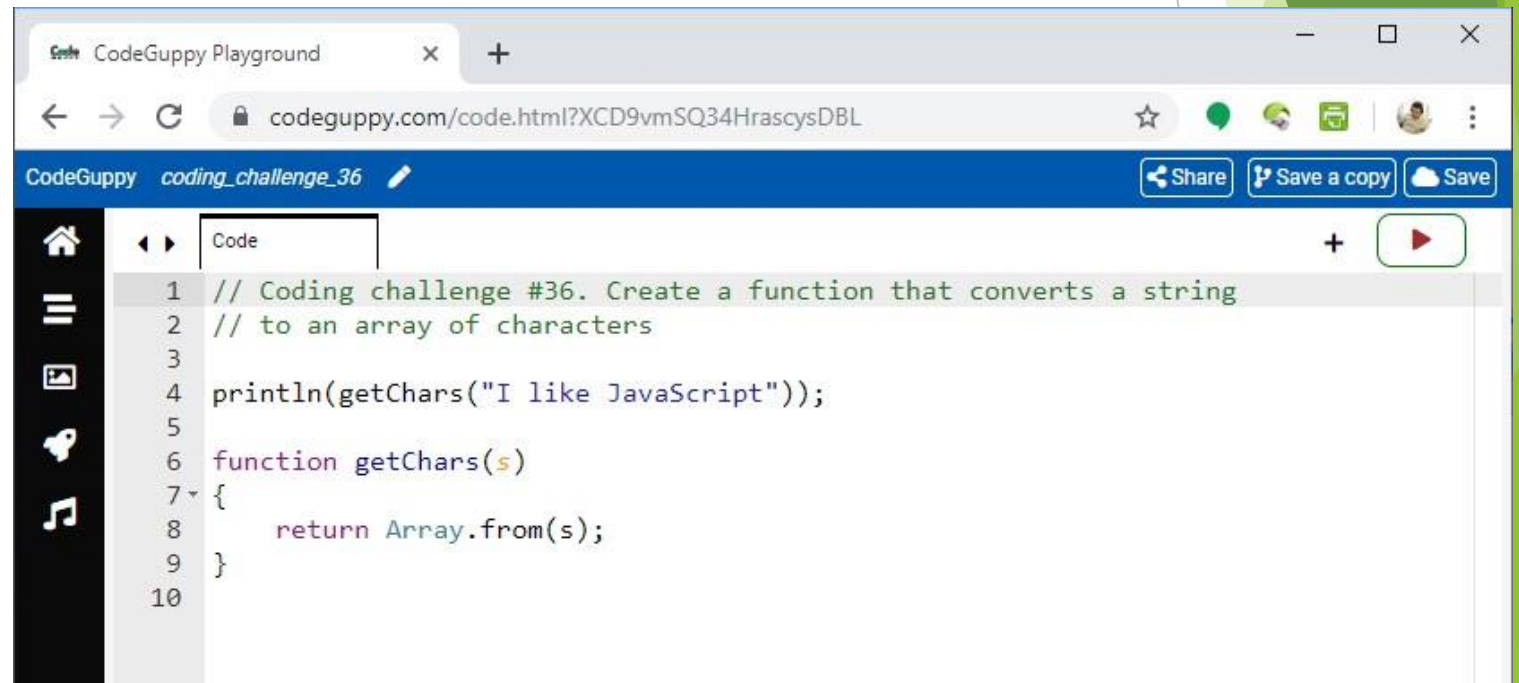


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?5Hqv93WXQ6OOjAYApLGw`. The browser's address bar and navigation icons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for file management and a main code editor. The code editor contains the following JavaScript code:

```
1 // Coding challenge #35. Create a function to convert
2 // a CSV text to a “bi-dimensional” array
3
4 var data = "John;Smith;954-000-0000\n" +
5           "Mich;Tiger;305-000-0000\n" +
6           "Monique;Vasquez;103-000-0000";
7
8 var ar = csvToArray(data);
9 println(JSON.stringify(ar));
10
11 function csvToArray(data)
12 {
13     var arLines = data.split("\n");
14
15     for(var i = 0; i < arLines.length; i++)
16     {
17         var arLine = arLines[i].split(";");
18         arLines[i] = arLine;
19     }
20
21     return arLines;
22 }
23
```



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

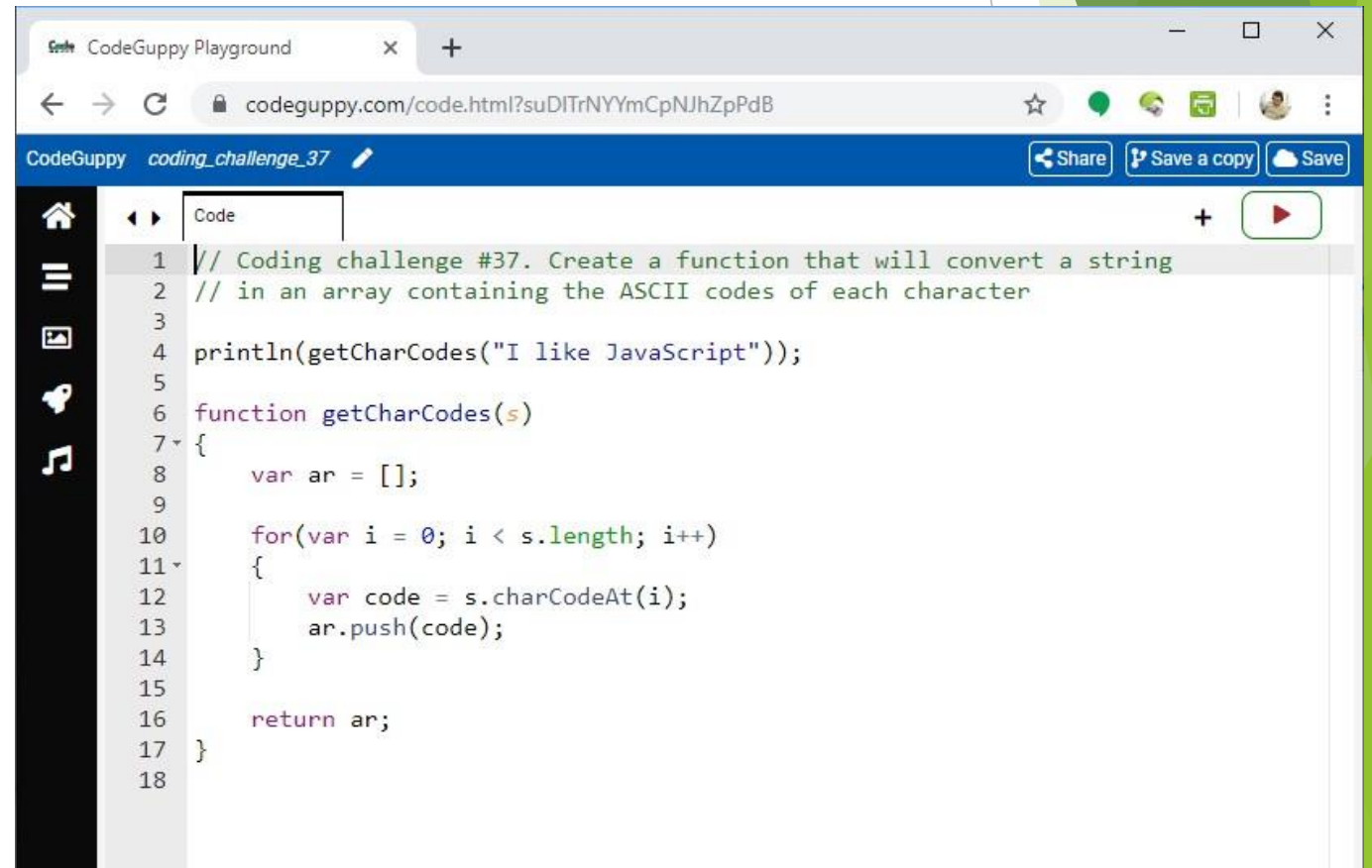


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?XCD9vmSQ34HrascysDBL". The browser's address bar and navigation buttons are visible. Below the browser window, there is a blue header bar with the text "CodeGuppy coding_challenge_36" and buttons for "Share", "Save a copy", and "Save". The main area of the browser displays a code editor with a dark sidebar on the left containing icons for home, list, file, and other functions. The code editor has a tab labeled "Code" and contains the following JavaScript code:

```
1 // Coding challenge #36. Create a function that converts a string
2 // to an array of characters
3
4 println(getChars("I like JavaScript"));
5
6 function getChars(s)
7 {
8     return Array.from(s);
9 }
10
```



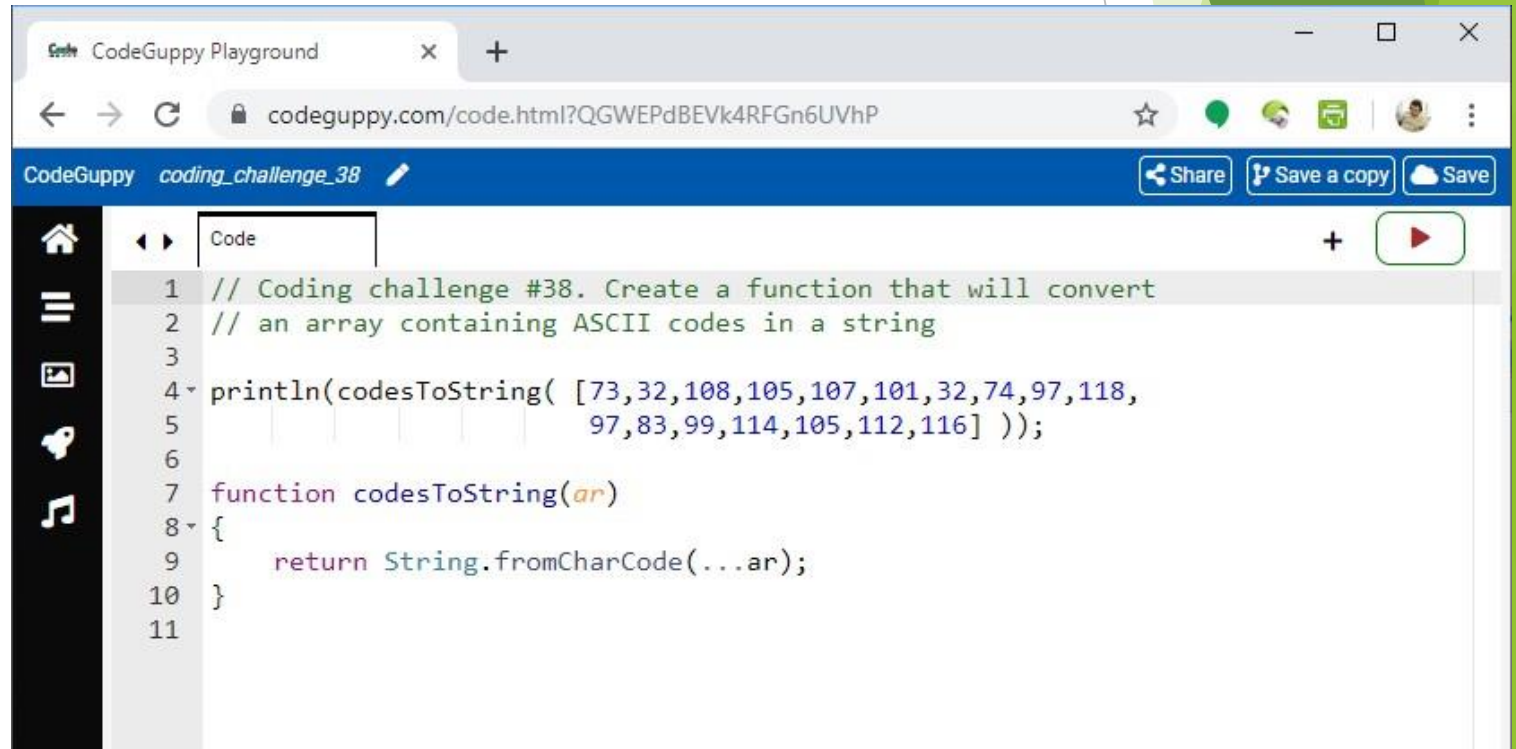
Coding challenge #37: Create a function that will convert a string in an array containing the ASCII codes of each character



```
1 // Coding challenge #37. Create a function that will convert a string
2 // in an array containing the ASCII codes of each character
3
4 println(getCharCodes("I like JavaScript"));
5
6 function getCharCodes(s)
7 {
8   var ar = [];
9
10  for(var i = 0; i < s.length; i++)
11  {
12    var code = s.charCodeAt(i);
13    ar.push(code);
14  }
15
16  return ar;
17 }
18
```



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

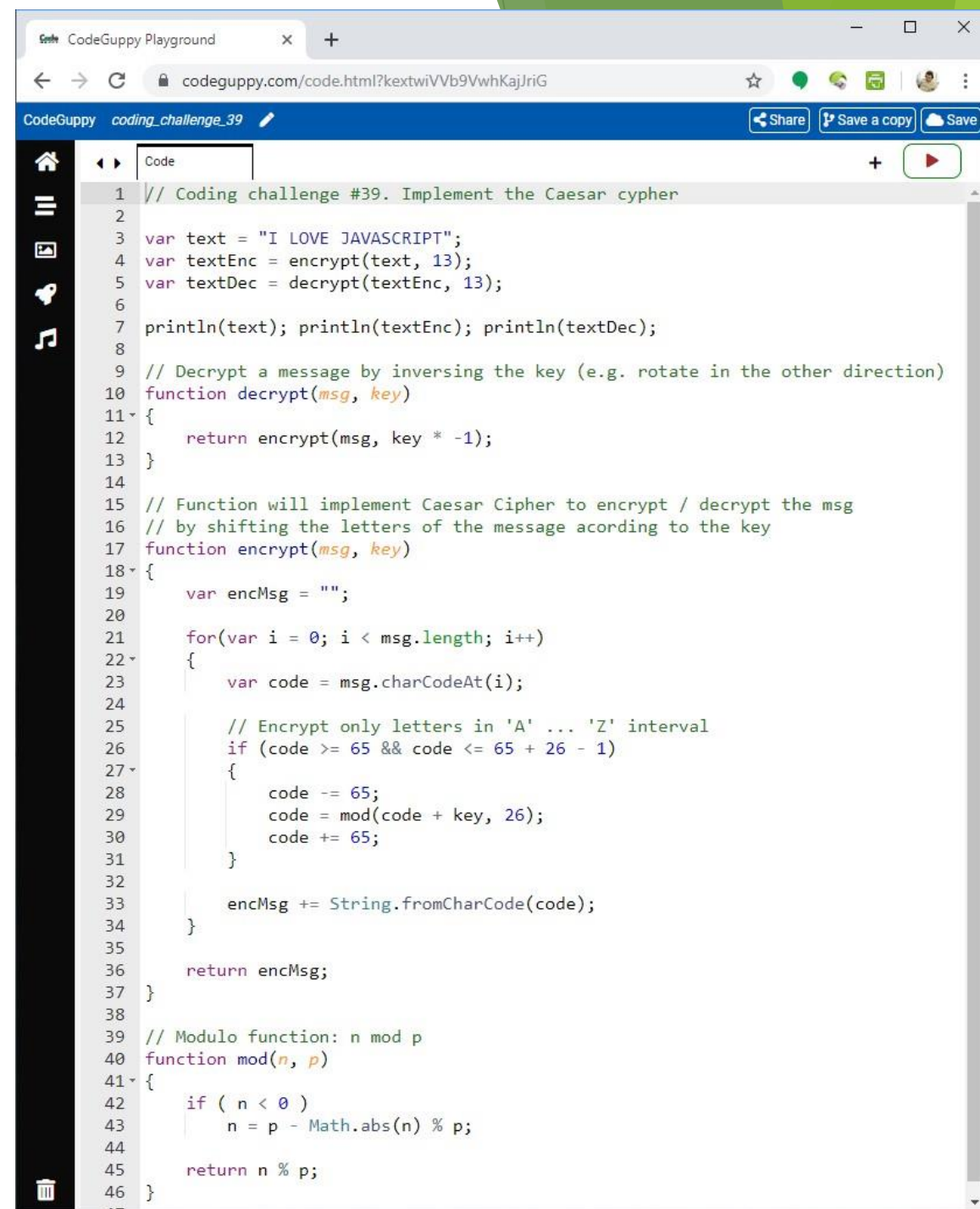


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?QGWEpdBEV4k4RFGn6UVhP`. The page has a blue header with the text "CodeGuppy coding_challenge_38" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, image, and code. The main area is a code editor with a tab labeled "Code" and a "Run" button (a red play icon). The code is as follows:

```
1 // Coding challenge #38. Create a function that will convert
2 // an array containing ASCII codes in a string
3
4 println(codesToString( [73,32,108,105,107,101,32,74,97,118,
5                        97,83,99,114,105,112,116] ));
6
7 function codesToString(ar)
8 {
9     return String.fromCharCode(...ar);
10 }
11
```



Coding challenge #39: Implement the Caesar cypher

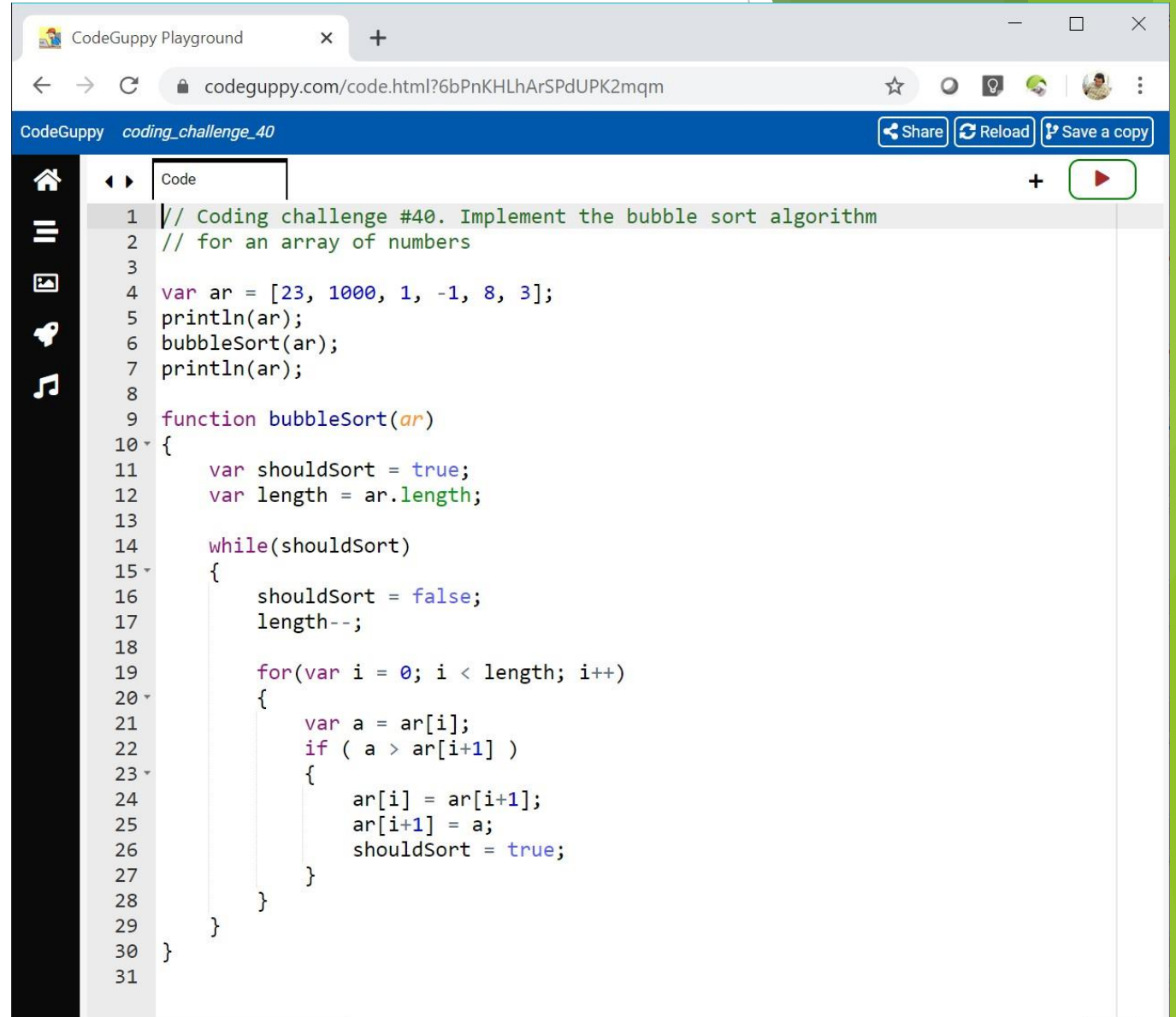


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?textwVb9VwhKajJriG". The page has a blue header with the "CodeGuppy" logo, the filename "coding_challenge_39", and buttons for "Share", "Save a copy", and "Save". A left sidebar contains icons for home, list, image, speech, and music. The main area is a code editor with a "Code" tab and a line number gutter on the left. The code is as follows:

```
1 // Coding challenge #39. Implement the Caesar cypher
2
3 var text = "I LOVE JAVASCRIPT";
4 var textEnc = encrypt(text, 13);
5 var textDec = decrypt(textEnc, 13);
6
7 println(text); println(textEnc); println(textDec);
8
9 // Decrypt a message by inversing the key (e.g. rotate in the other direction)
10 function decrypt(msg, key)
11 {
12     return encrypt(msg, key * -1);
13 }
14
15 // Function will implement Caesar Cipher to encrypt / decrypt the msg
16 // by shifting the letters of the message according to the key
17 function encrypt(msg, key)
18 {
19     var encMsg = "";
20
21     for(var i = 0; i < msg.length; i++)
22     {
23         var code = msg.charCodeAt(i);
24
25         // Encrypt only letters in 'A' ... 'Z' interval
26         if (code >= 65 && code <= 65 + 26 - 1)
27         {
28             code += key;
29             code = mod(code, 26);
30             code += 65;
31         }
32
33         encMsg += String.fromCharCode(code);
34     }
35
36     return encMsg;
37 }
38
39 // Modulo function: n mod p
40 function mod(n, p)
41 {
42     if ( n < 0 )
43         n = p - Math.abs(n) % p;
44
45     return n % p;
46 }
```



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

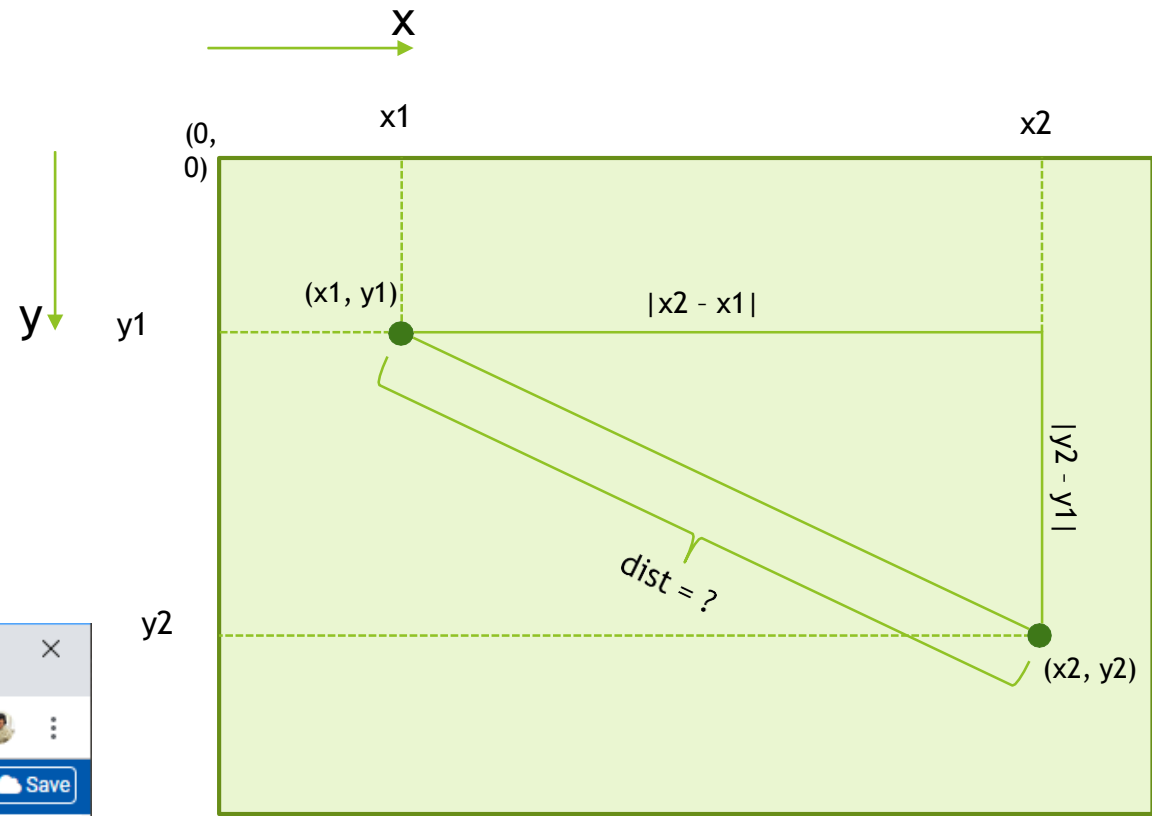


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?6bPnKHLhArSPdUPK2mqm". The browser's address bar and navigation buttons are visible. Below the browser window, there is a blue header bar with the text "CodeGuppy coding_challenge_40" and buttons for "Share", "Reload", and "Save a copy". On the left side of the code editor, there is a vertical toolbar with icons for home, list, image, lock, and music. The code editor itself has a tab labeled "Code" and a play button in the top right corner. The code is written in JavaScript and implements the bubble sort algorithm. It starts with a comment, initializes an array, prints it, calls the bubbleSort function, and prints it again. The bubbleSort function uses a while loop and a for loop to sort the array.

```
1 // Coding challenge #40. Implement the bubble sort algorithm
2 // for an array of numbers
3
4 var ar = [23, 1000, 1, -1, 8, 3];
5 println(ar);
6 bubbleSort(ar);
7 println(ar);
8
9 function bubbleSort(ar)
10 {
11     var shouldSort = true;
12     var length = ar.length;
13
14     while(shouldSort)
15     {
16         shouldSort = false;
17         length--;
18
19         for(var i = 0; i < length; i++)
20         {
21             var a = ar[i];
22             if ( a > ar[i+1] )
23             {
24                 ar[i] = ar[i+1];
25                 ar[i+1] = a;
26                 shouldSort = true;
27             }
28         }
29     }
30 }
31
```



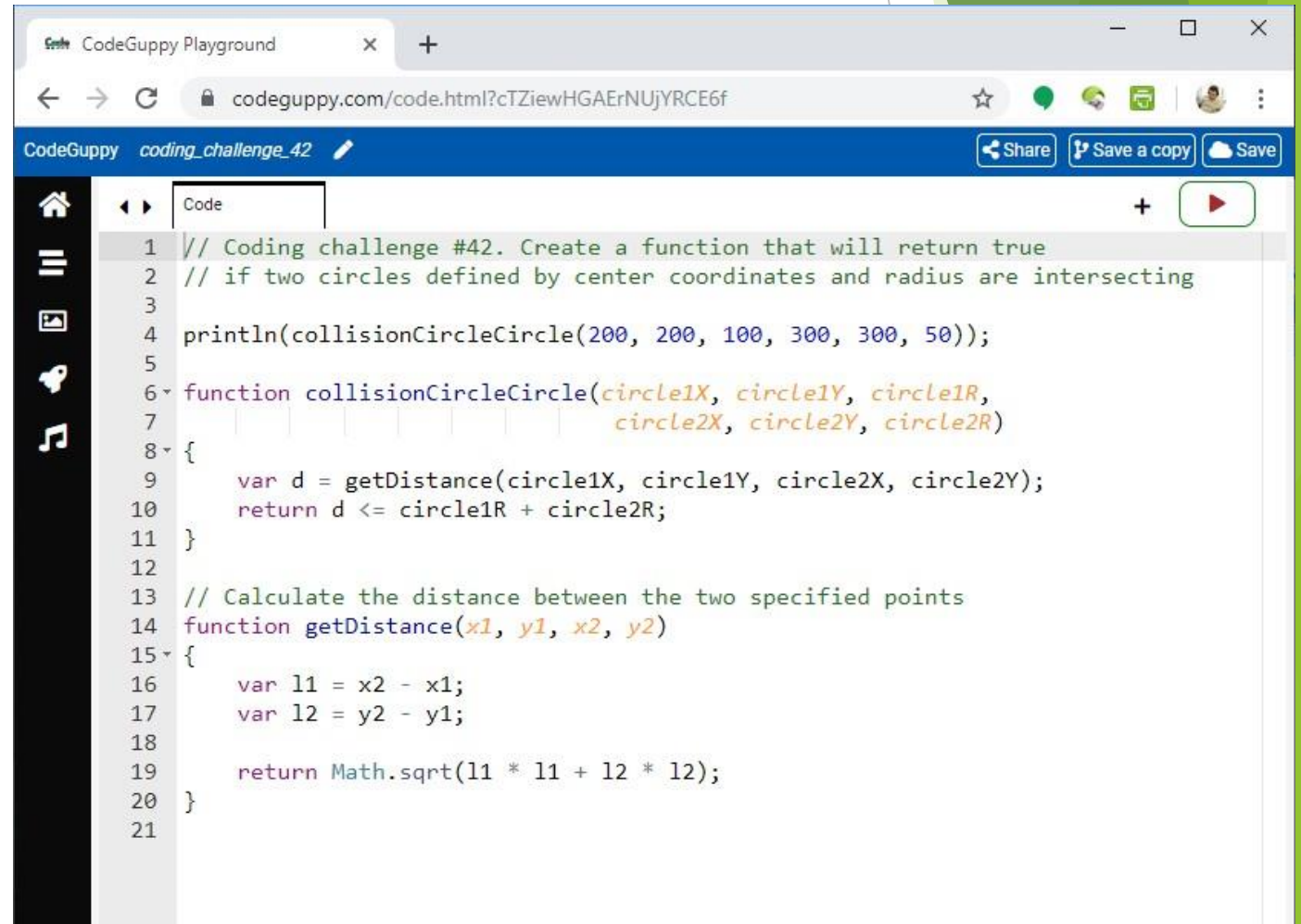
Coding challenge #41: Create a function to calculate the distance between two points defined by their x, y coordinates



```
CodeGuppy Playground
codeguppy.com/code.html?mnAuF3BjhDaFwBtDUnI4
CodeGuppy coding_challenge_41
Share Save a copy Save

Code
1 // Coding challenge #41. Create a function to calculate the
2 // distance between two points defined by their x, y coordinates
3
4 println(getDistance(100, 100, 400, 300));
5
6 function getDistance(x1, y1, x2, y2)
7 {
8     var l1 = x2 - x1;
9     var l2 = y2 - y1;
10
11     return Math.sqrt(l1 * l1 + l2 * l2);
12 }
13
```

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

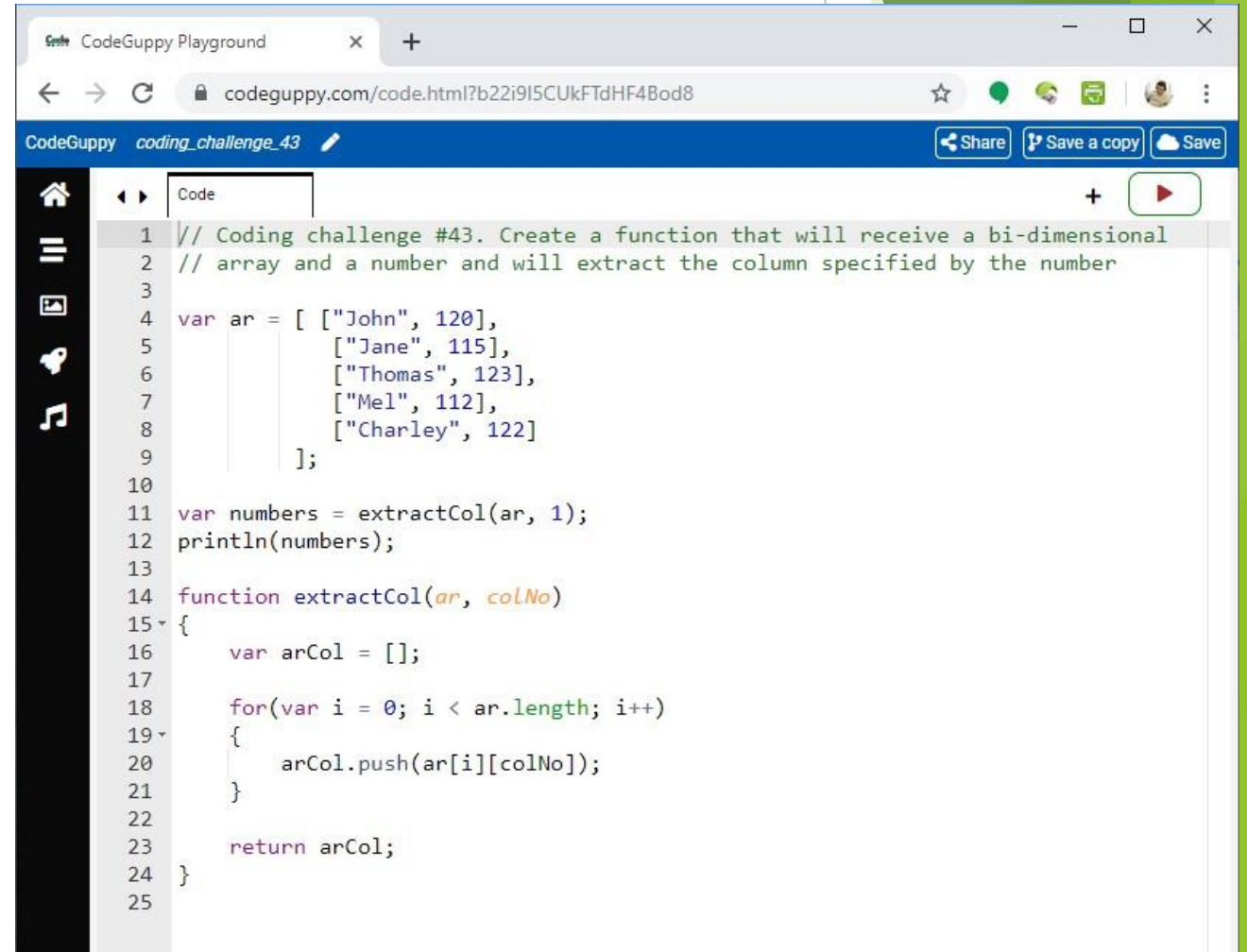


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?cTZiewHGAErNUjYRCE6f". The browser's address bar and navigation buttons are visible. Below the browser window, there is a blue header bar with the text "CodeGuppy coding_challenge_42" and buttons for "Share", "Save a copy", and "Save". On the left side of the code editor, there is a vertical toolbar with icons for home, run, save, and other functions. The main area of the browser displays a JavaScript code editor with the following code:

```
1 // Coding challenge #42. Create a function that will return true
2 // if two circles defined by center coordinates and radius are intersecting
3
4 println(collisionCircleCircle(200, 200, 100, 300, 300, 50));
5
6 function collisionCircleCircle(circle1X, circle1Y, circle1R,
7                               circle2X, circle2Y, circle2R)
8 {
9     var d = getDistance(circle1X, circle1Y, circle2X, circle2Y);
10    return d <= circle1R + circle2R;
11 }
12
13 // Calculate the distance between the two specified points
14 function getDistance(x1, y1, x2, y2)
15 {
16     var l1 = x2 - x1;
17     var l2 = y2 - y1;
18
19     return Math.sqrt(l1 * l1 + l2 * l2);
20 }
21
```



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

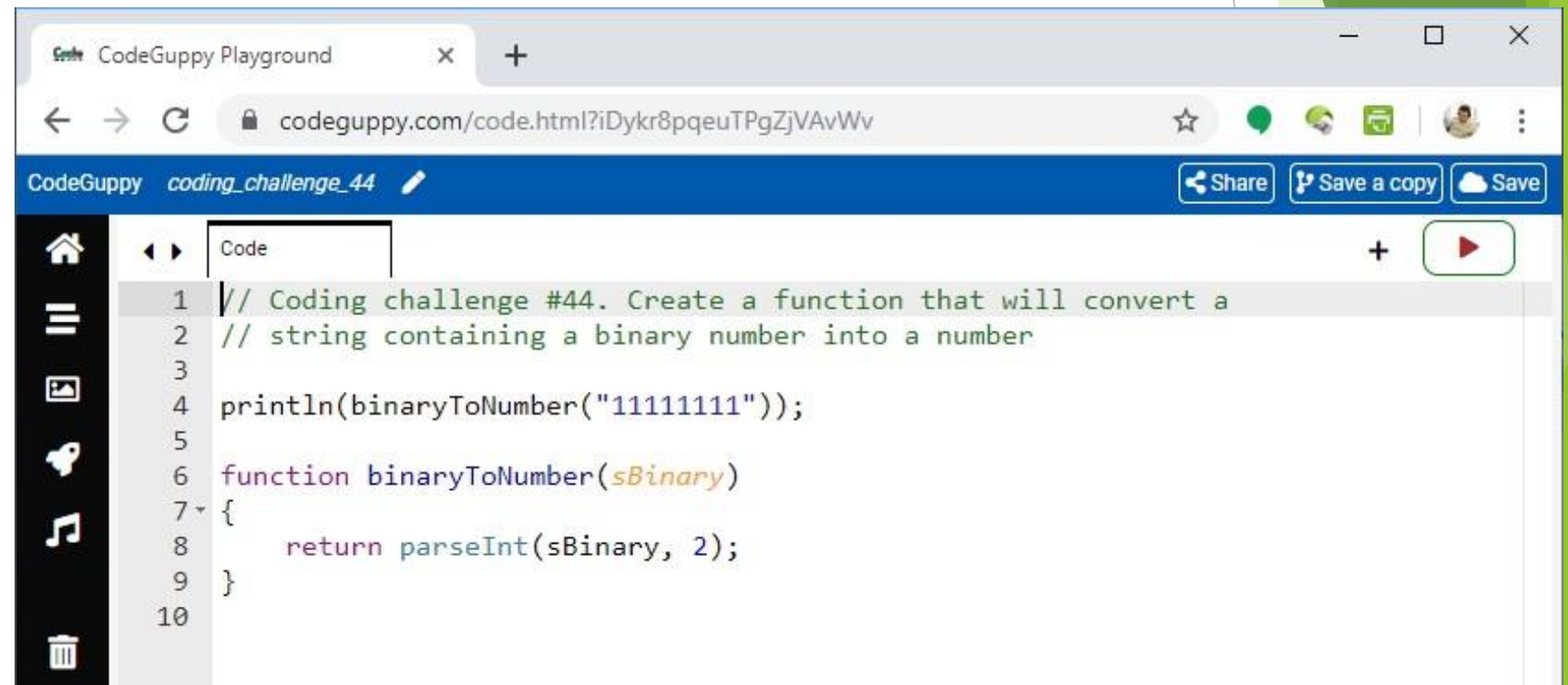


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?b22i9I5CUkFTdHF4Bod8". The page has a blue header with the text "CodeGuppy coding_challenge_43" and buttons for "Share", "Save a copy", and "Save". On the left side, there is a dark sidebar with icons for home, list, image, and music. The main area is a code editor with a tab labeled "Code". The code is as follows:

```
1 // Coding challenge #43. Create a function that will receive a bi-dimensional
2 // array and a number and will extract the column specified by the number
3
4 var ar = [ ["John", 120],
5             ["Jane", 115],
6             ["Thomas", 123],
7             ["Mel", 112],
8             ["Charley", 122]
9           ];
10
11 var numbers = extractCol(ar, 1);
12 println(numbers);
13
14 function extractCol(ar, colNo)
15 {
16     var arCol = [];
17
18     for(var i = 0; i < ar.length; i++)
19     {
20         arCol.push(ar[i][colNo]);
21     }
22
23     return arCol;
24 }
25
```



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

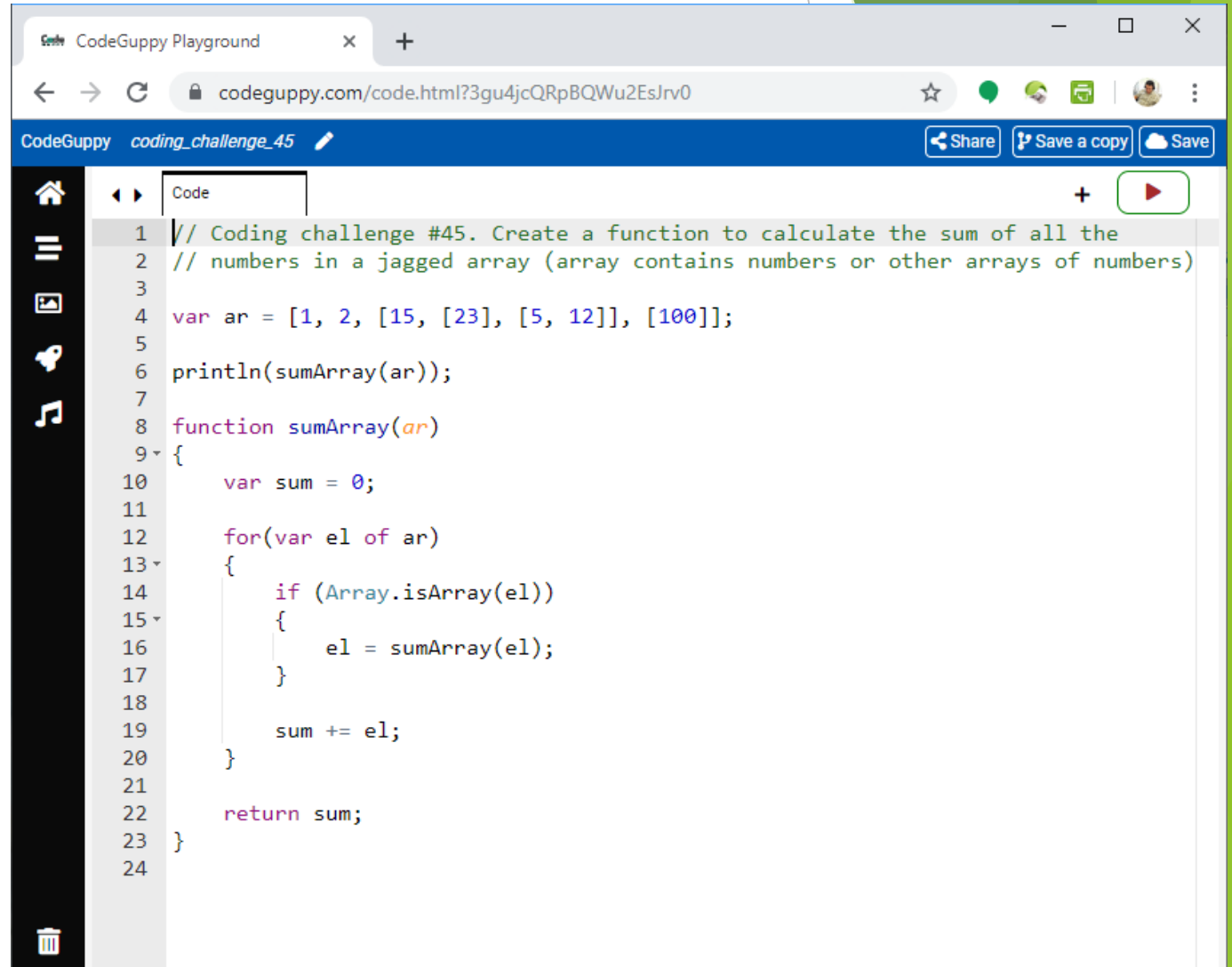


The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?iDykr8pqueuTPgZjVAvWv`. The browser's address bar and navigation icons are visible. Below the browser window, the CodeGuppy interface is shown, including a sidebar with icons for home, list, image, chat, music, and trash. The main area is titled "Code" and contains the following code:

```
1 // Coding challenge #44. Create a function that will convert a
2 // string containing a binary number into a number
3
4 println(binaryToNumber("11111111"));
5
6 function binaryToNumber(sBinary)
7 {
8     return parseInt(sBinary, 2);
9 }
10
```

At the bottom left of the slide, there is a small icon of a globe inside a square frame.

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)



```
1 // Coding challenge #45. Create a function to calculate the sum of all the
2 // numbers in a jagged array (array contains numbers or other arrays of numbers)
3
4 var ar = [1, 2, [15, [23], [5, 12]], [100]];
5
6 println(sumArray(ar));
7
8 function sumArray(ar)
9 {
10     var sum = 0;
11
12     for(var el of ar)
13     {
14         if (Array.isArray(el))
15         {
16             el = sumArray(el);
17         }
18
19         sum += el;
20     }
21
22     return sum;
23 }
24
```



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

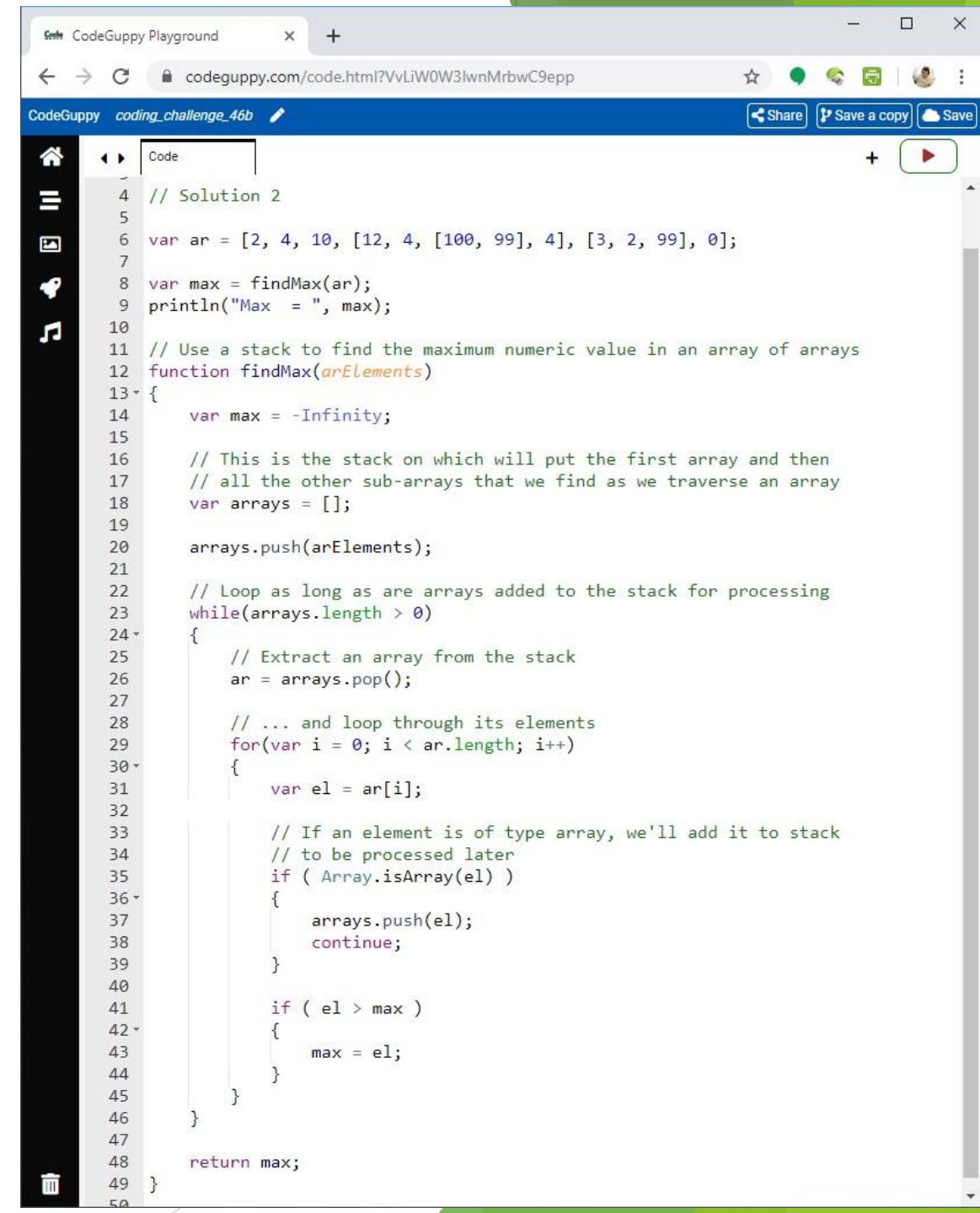
Requirements: Use recursion

A screenshot of the CodeGuppy Playground web application. The browser address bar shows 'codeguppy.com/code.html?oT5nrux2yAgIRsuXBSK'. The page title is 'CodeGuppy coding_challenge_46a'. The code editor shows a JavaScript solution for finding the maximum value in a jagged array using recursion. The code includes comments and a test array: [2, 4, 10, [12, 4, [100, 99], 4], [3, 2, 99], 0]. The solution uses a recursive function 'findMax' that iterates through the array and recursively calls itself on any sub-arrays to find the maximum value. The interface includes a left sidebar with icons for file explorer, console, and other tools, and a top bar with 'Share', 'Save a copy', and 'Save' buttons.

```
1 // Coding challenge #46. Find the maximum number in a jagged array
2 // (array contains numbers or other arrays of numbers)
3
4 // Solution 1
5
6 var ar = [2, 4, 10, [12, 4, [100, 99], 4], [3, 2, 99], 0];
7
8 var max = findMax(ar);
9 println("Max = ", max);
10
11 // Use recursion to find the maximum numeric value in an array of arrays
12 function findMax(ar)
13 {
14     var max = -Infinity;
15
16     // Cycle through all the elements of the array
17     for(var i = 0; i < ar.length; i++)
18     {
19         var el = ar[i];
20
21         // If an element is of type array then invoke the same function
22         // to find out the maximum element of that subarray
23         if ( Array.isArray(el) )
24         {
25             el = findMax( el );
26         }
27
28         if ( el > max )
29         {
30             max = el;
31         }
32     }
33
34     return max;
35 }
36
```


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

Requirements: Do not use recursion

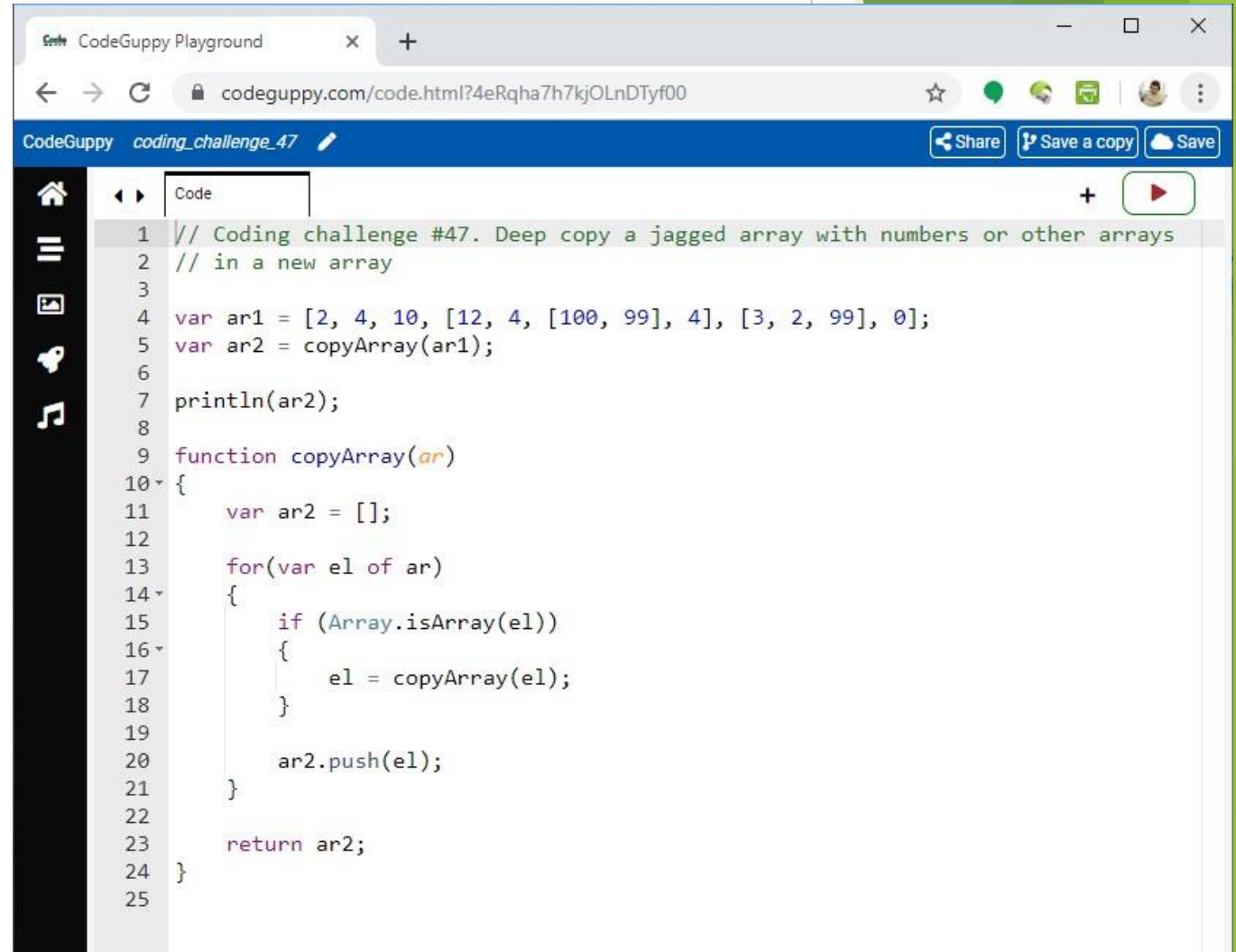


The screenshot shows a web browser window with the address bar displaying 'codeguppy.com/code.html?VvLiW0W3lwnMrbwC9epp'. The page title is 'CodeGuppy coding_challenge_46b'. The code editor shows the following JavaScript code:

```
4 // Solution 2
5
6 var ar = [2, 4, 10, [12, 4, [100, 99], 4], [3, 2, 99], 0];
7
8 var max = findMax(ar);
9 println("Max = ", max);
10
11 // Use a stack to find the maximum numeric value in an array of arrays
12 function findMax(arElements)
13 {
14     var max = -Infinity;
15
16     // This is the stack on which will put the first array and then
17     // all the other sub-arrays that we find as we traverse an array
18     var arrays = [];
19
20     arrays.push(arElements);
21
22     // Loop as long as are arrays added to the stack for processing
23     while(arrays.length > 0)
24     {
25         // Extract an array from the stack
26         ar = arrays.pop();
27
28         // ... and loop through its elements
29         for(var i = 0; i < ar.length; i++)
30         {
31             var el = ar[i];
32
33             // If an element is of type array, we'll add it to stack
34             // to be processed later
35             if ( Array.isArray(el) )
36             {
37                 arrays.push(el);
38                 continue;
39             }
40
41             if ( el > max )
42             {
43                 max = el;
44             }
45         }
46     }
47
48     return max;
49 }
```



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



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL `codeguppy.com/code.html?4eRqha7h7kjOLnDTyf00`. The browser's address bar and navigation buttons are visible. Below the browser window, there is a dark sidebar with icons for home, list, document, bell, and music. The main area displays a code editor with the following JavaScript code:

```
1 // Coding challenge #47. Deep copy a jagged array with numbers or other arrays
2 // in a new array
3
4 var ar1 = [2, 4, 10, [12, 4, [100, 99], 4], [3, 2, 99], 0];
5 var ar2 = copyArray(ar1);
6
7 println(ar2);
8
9 function copyArray(ar)
10 {
11     var ar2 = [];
12
13     for(var el of ar)
14     {
15         if (Array.isArray(el))
16         {
17             el = copyArray(el);
18         }
19
20         ar2.push(el);
21     }
22
23     return ar2;
24 }
25
```



CodeGuppy Playground

codeguppy.com/code.html?6O219iv12e5UaC30fcbG

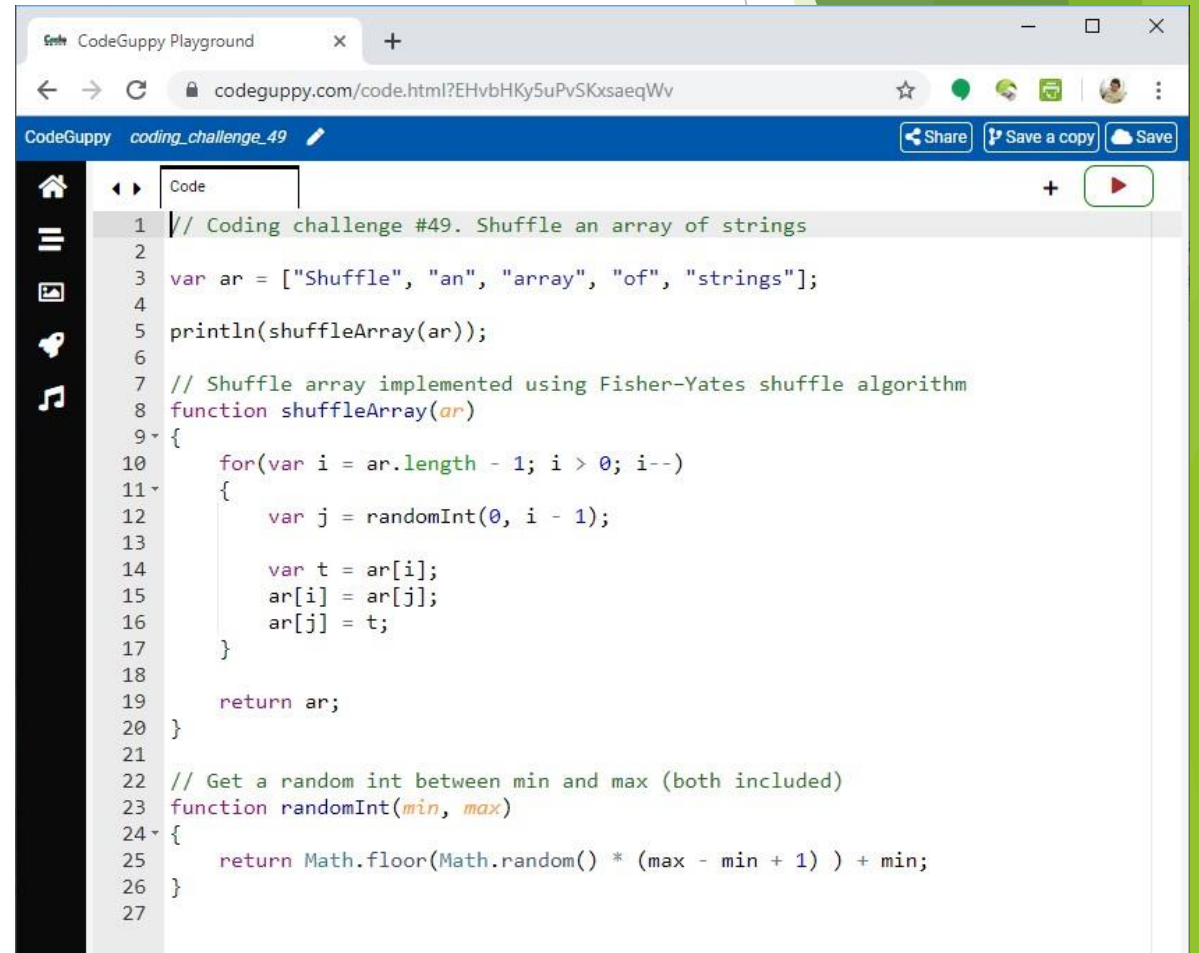
coding_challenge_48

```
1 // Coding challenge #48. Create a function to return
2 // the longest word(s) in a string
3
4 var text = "Create a function to return the longest word(s) in a sentence.";
5
6 println(getLongestWords(text));
7
8 function getLongestWords(text)
9 {
10     var words = getWords(text);
11
12     var maxSize = 0;
13     var maxPositions = [];
14
15     for(var i = 0; i < words.length; i++)
16     {
17         var currWordSize = words[i].length;
18
19         if (currWordSize > maxSize)
20         {
21             maxSize = currWordSize;
22             maxPositions = [ i ];
23         }
24         else if (currWordSize === maxSize)
25         {
26             maxPositions.push(i);
27         }
28     }
29
30     return getElements(words, maxPositions);
31 }
32
33 // Get only the elements from specified positions from the array
34 function getElements(ar, arPositions)
35 {
36     var arNew = [];
37
38     for(var pos of arPositions)
39     {
40         arNew.push(ar[pos]);
41     }
42
43     return arNew;
44 }
45
46
```

Coding challenge #48: Create a function to return the longest word(s) in a string

```
46 // Returns an array with the words from specified text
47 function getWords(text)
48 {
49     let startWord = -1;
50     let ar = [];
51
52     for(let i = 0; i <= text.length; i++)
53     {
54         let c = i < text.length ? text[i] : " ";
55
56         if (!isSeparator(c) && startWord < 0)
57         {
58             startWord = i;
59         }
60
61         if (isSeparator(c) && startWord >= 0)
62         {
63             let word = text.substring(startWord, i);
64             ar.push(word);
65
66             startWord = -1;
67         }
68     }
69
70     return ar;
71 }
72
73 function isSeparator(c)
74 {
75     var separators = [ " ", "\t", "\n", "\r", ",", ";", ".", "!", "?", "(", ")" ];
76     return separators.includes(c);
77 }
78
```

Coding challenge #49: Shuffle an array of strings



```
1 // Coding challenge #49. Shuffle an array of strings
2
3 var ar = ["Shuffle", "an", "array", "of", "strings"];
4
5 println(shuffleArray(ar));
6
7 // Shuffle array implemented using Fisher-Yates shuffle algorithm
8 function shuffleArray(ar)
9 {
10     for(var i = ar.length - 1; i > 0; i--)
11     {
12         var j = randomInt(0, i - 1);
13
14         var t = ar[i];
15         ar[i] = ar[j];
16         ar[j] = t;
17     }
18
19     return ar;
20 }
21
22 // Get a random int between min and max (both included)
23 function randomInt(min, max)
24 {
25     return Math.floor(Math.random() * (max - min + 1) ) + min;
26 }
27
```



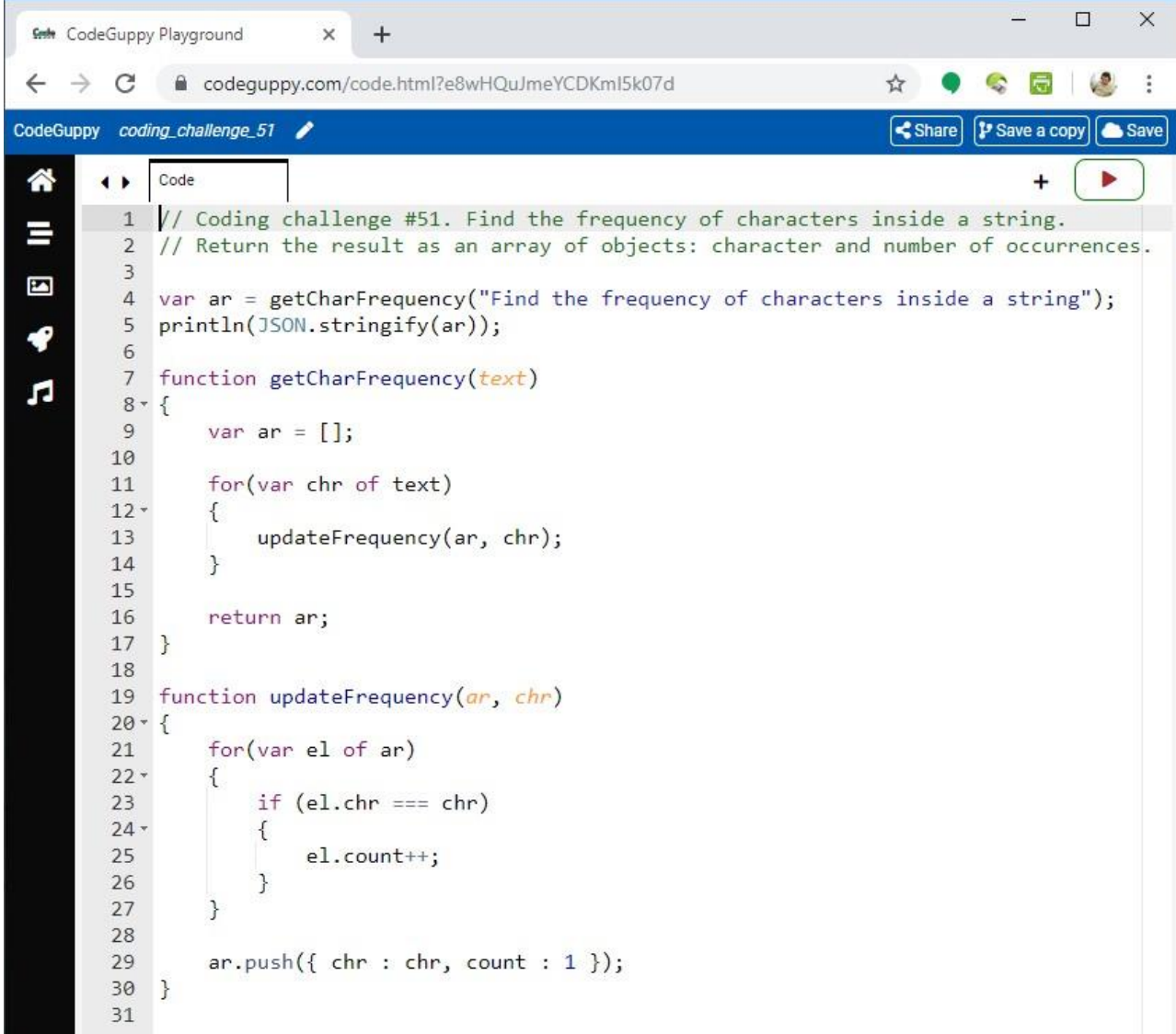
Coding challenge #50: Create a function that will receive *n* as argument and return an array of *n* random numbers from 1 to *n*

A screenshot of the CodeGuppy Playground web application. The browser address bar shows the URL 'codeguppy.com/code.html?GfCrzPkGSPKbvLuf6KyA'. The page title is 'CodeGuppy coding_challenge_50'. The code editor displays a JavaScript solution for the challenge. The code includes a function 'getRandomNumbers(n)' that generates an array of random numbers, a 'shuffleArray(ar)' function using the Fisher-Yates algorithm, and a 'randomInt(min, max)' function. The code is as follows:

```
1 // Coding challenge #50. Create a function that will receive n as argument
2 // and return an array of n unique random numbers from 1 to n.
3
4 println(getRandomNumbers(10));
5
6 function getRandomNumbers(n)
7 {
8     var ar = [];
9
10    for(var i = 1; i <= n; i++)
11    {
12        ar.push(i);
13    }
14
15    shuffleArray(ar);
16
17    return ar;
18 }
19
20 // Shuffle array implemented using Fisher-Yates shuffle algorithm
21 function shuffleArray(ar)
22 {
23     for(var i = ar.length - 1; i > 0; i--)
24     {
25         var j = randomInt(0, i - 1);
26
27         var t = ar[i];
28         ar[i] = ar[j];
29         ar[j] = t;
30     }
31
32     return ar;
33 }
34
35 // Get a random int between min and max (both included)
36 function randomInt(min, max)
37 {
38     return Math.floor(Math.random() * (max - min + 1) ) + min;
39 }
40
```

Coding challenge #51: Find the frequency of characters inside a string. Return the result as an array of objects.

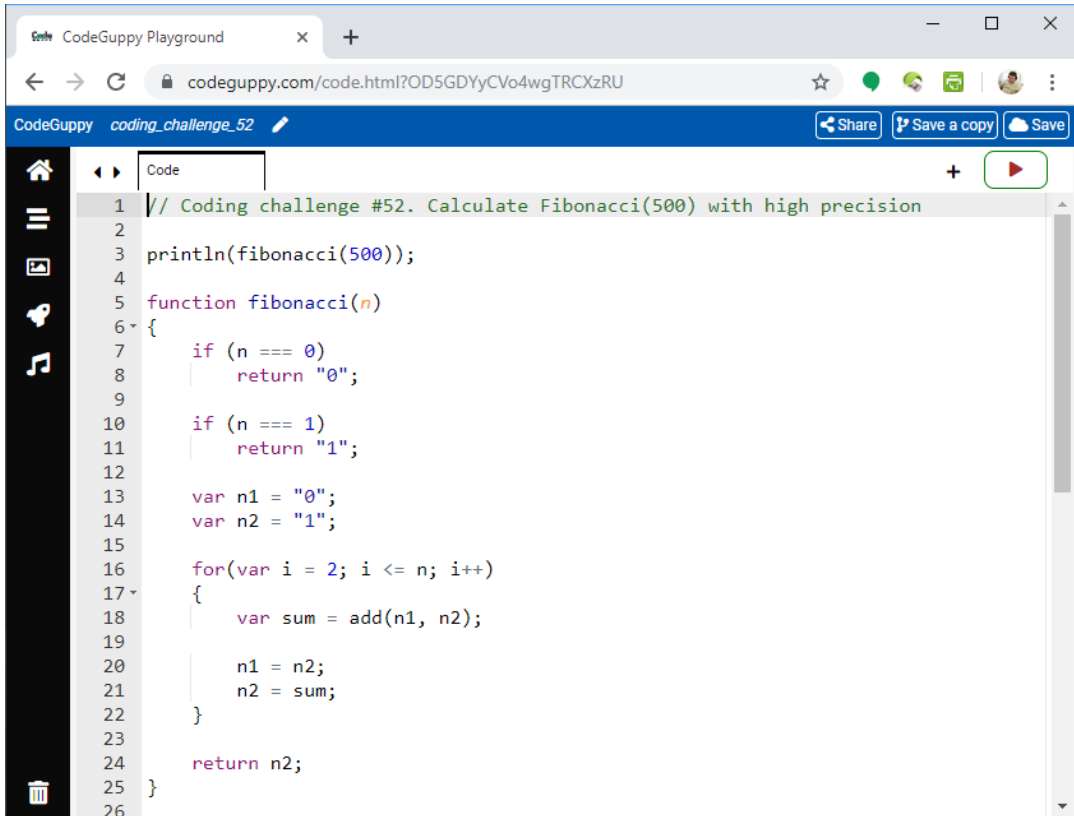
Each object has 2 fields: character and number of occurrences.



The screenshot shows a web browser window titled "CodeGuppy Playground" with the URL "codeguppy.com/code.html?e8wHQuJmeYCDKml5k07d". The page has a blue header with "CodeGuppy coding_challenge_51" and buttons for "Share", "Save a copy", and "Save". On the left is a dark sidebar with icons for home, list, code, console, and settings. The main area displays a code editor with the following JavaScript code:

```
1 // Coding challenge #51. Find the frequency of characters inside a string.
2 // Return the result as an array of objects: character and number of occurrences.
3
4 var ar = getCharFrequency("Find the frequency of characters inside a string");
5 println(JSON.stringify(ar));
6
7 function getCharFrequency(text)
8 {
9     var ar = [];
10
11     for(var chr of text)
12     {
13         updateFrequency(ar, chr);
14     }
15
16     return ar;
17 }
18
19 function updateFrequency(ar, chr)
20 {
21     for(var el of ar)
22     {
23         if (el.chr === chr)
24         {
25             el.count++;
26         }
27     }
28
29     ar.push({ chr : chr, count : 1 });
30 }
31
```





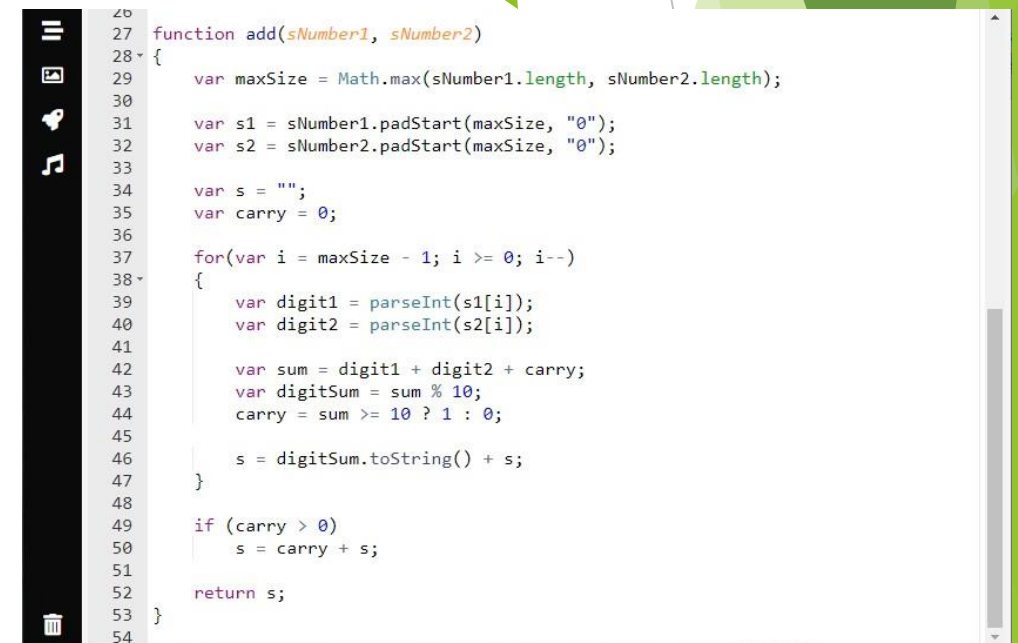
CodeGuppy Playground

codeguppy.com/code.html?OD5GDYyCVo4wgTRCXzRU

coding_challenge_52

```
1 // Coding challenge #52. Calculate Fibonacci(500) with high precision
2
3 println(fibonacci(500));
4
5 function fibonacci(n)
6 {
7     if (n === 0)
8         return "0";
9
10    if (n === 1)
11        return "1";
12
13    var n1 = "0";
14    var n2 = "1";
15
16    for(var i = 2; i <= n; i++)
17    {
18        var sum = add(n1, n2);
19
20        n1 = n2;
21        n2 = sum;
22    }
23
24    return n2;
25 }
26
```

Coding challenge #52: Calculate Fibonacci(500) with high precision (all digits)



```
27 function add(sNumber1, sNumber2)
28 {
29     var maxSize = Math.max(sNumber1.length, sNumber2.length);
30
31     var s1 = sNumber1.padStart(maxSize, "0");
32     var s2 = sNumber2.padStart(maxSize, "0");
33
34     var s = "";
35     var carry = 0;
36
37     for(var i = maxSize - 1; i >= 0; i--)
38     {
39         var digit1 = parseInt(s1[i]);
40         var digit2 = parseInt(s2[i]);
41
42         var sum = digit1 + digit2 + carry;
43         var digitSum = sum % 10;
44         carry = sum >= 10 ? 1 : 0;
45
46         s = digitSum.toString() + s;
47     }
48
49     if (carry > 0)
50         s = carry + s;
51
52     return s;
53 }
54
```



Coding challenge #53: Calculate 70! with high precision (all digits)

```
CodeGuppy Playground
codeguppy.com/code.html?m4AfgJmCABGNEvKIUNtM
CodeGuppy coding_challenge_53
Code
1 // Coding challenge #53. Calculate 70! with high precision (all decimals)
2
3 println(factorial(70));
4
5 // Calculate factorial(n) ... using big number calculations
6 function factorial(n)
7 {
8     var prod = "1";
9
10    for(var i = 2; i <= n; i++)
11    {
12        prod = mult(prod, i.toString());
13    }
14
15    return prod;
16 }
17
18 // Multiplies sNumber1 * sNumber2
19 // Each number is provided as string
20 function mult(sNumber1, sNumber2)
21 {
22     // Calculate partial results according to multiplication algorithm
23     var partialResults = [];
24
25     for(var i = sNumber2.length - 1; i >= 0; i--)
26     {
27         var digit = parseInt(sNumber2[i]);
28
29         var partialResult = multDigit(sNumber1, digit);
30         partialResult += "0".repeat(partialResults.length);
31
32         partialResults.push(partialResult);
33     }
34
35     // Sum partial results to obtain the product
36     var sum = "";
37
38     for(var r of partialResults)
39     {
40         sum = add(sum, r);
41     }
42
43     return sum;
44 }
45
```

```
46 // Multiplies number sNumber (as string) with a single digit number
47 function multDigit(sNumber, digit)
48 {
49     var p = "";
50     var carry = 0;
51
52     for(var i = sNumber.length - 1; i >= 0; i--)
53     {
54         var numberDigit = parseInt(sNumber[i]);
55
56         var prod = digit * numberDigit + carry;
57         var prodDigit = prod % 10;
58         carry = Math.floor(prod / 10);
59
60         p = prodDigit.toString() + p;
61     }
62
63     if (carry > 0)
64         p = carry + p;
65
66     return p;
67 }
```

```
70 function add(sNumber1, sNumber2)
71 {
72     var maxSize = Math.max(sNumber1.length, sNumber2.length);
73
74     var s1 = sNumber1.padStart(maxSize, "0");
75     var s2 = sNumber2.padStart(maxSize, "0");
76
77     var s = "";
78     var carry = 0;
79
80     for(var i = maxSize - 1; i >= 0; i--)
81     {
82         var digit1 = parseInt(s1[i]);
83         var digit2 = parseInt(s2[i]);
84
85         var sum = digit1 + digit2 + carry;
86         var digitSum = sum % 10;
87         carry = sum >= 10 ? 1 : 0;
88
89         s = digitSum.toString() + s;
90     }
91
92     if (carry > 0)
93         s = carry + s;
94
95     return s;
96 }
97
```