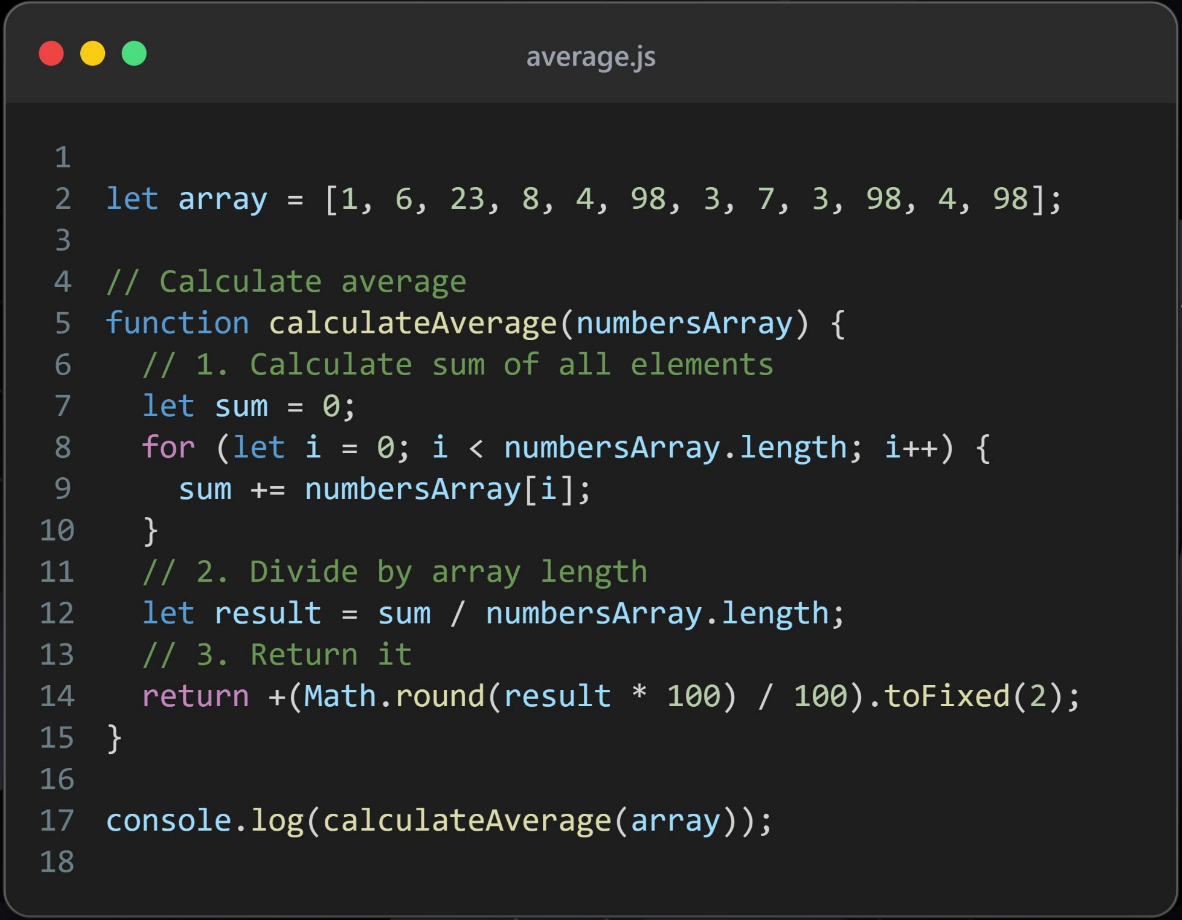
JavaScript

STUDIA PODYPLOMOWE
POLITECHNIKA BIAŁOSTOCKA

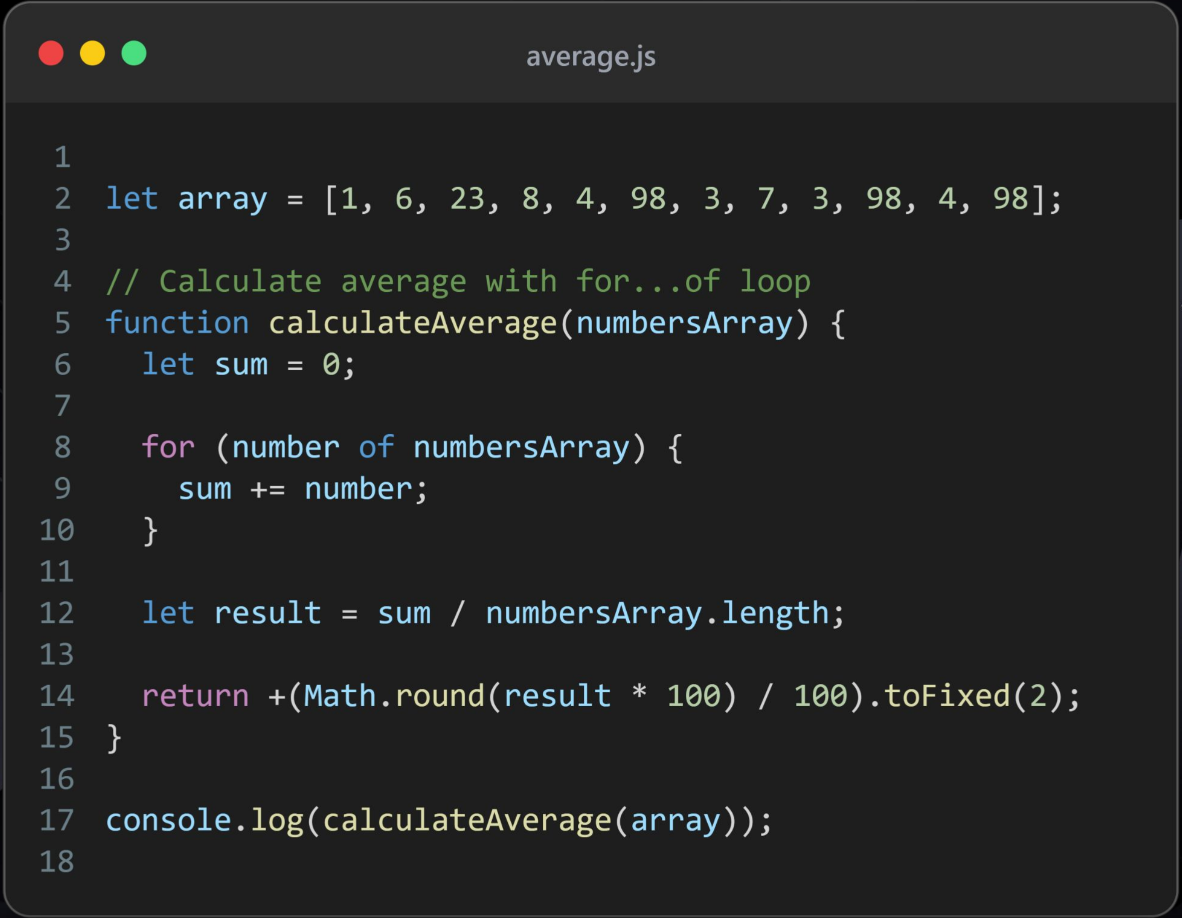
homework

Calculate average

- Write a function that takes an array of numbers as an input and returns it’s average
- Round to 2 decimal places


average.js

```javascript

let array = [1, 6, 23, 8, 4, 98, 3, 7, 3, 98, 4, 98];

// Calculate average
function calculateAverage(numbersArray) {
  // 1. Calculate sum of all elements
  let sum = 0;
  for (let i = 0; i < numbersArray.length; i++) {
    sum += numbersArray[i];
  }
  // 2. Divide by array length
  let result = sum / numbersArray.length;
  // 3. Return it
  return +(Math.round(result * 100) / 100).toFixed(2);
}

console.log(calculateAverage(array));

```


average.js

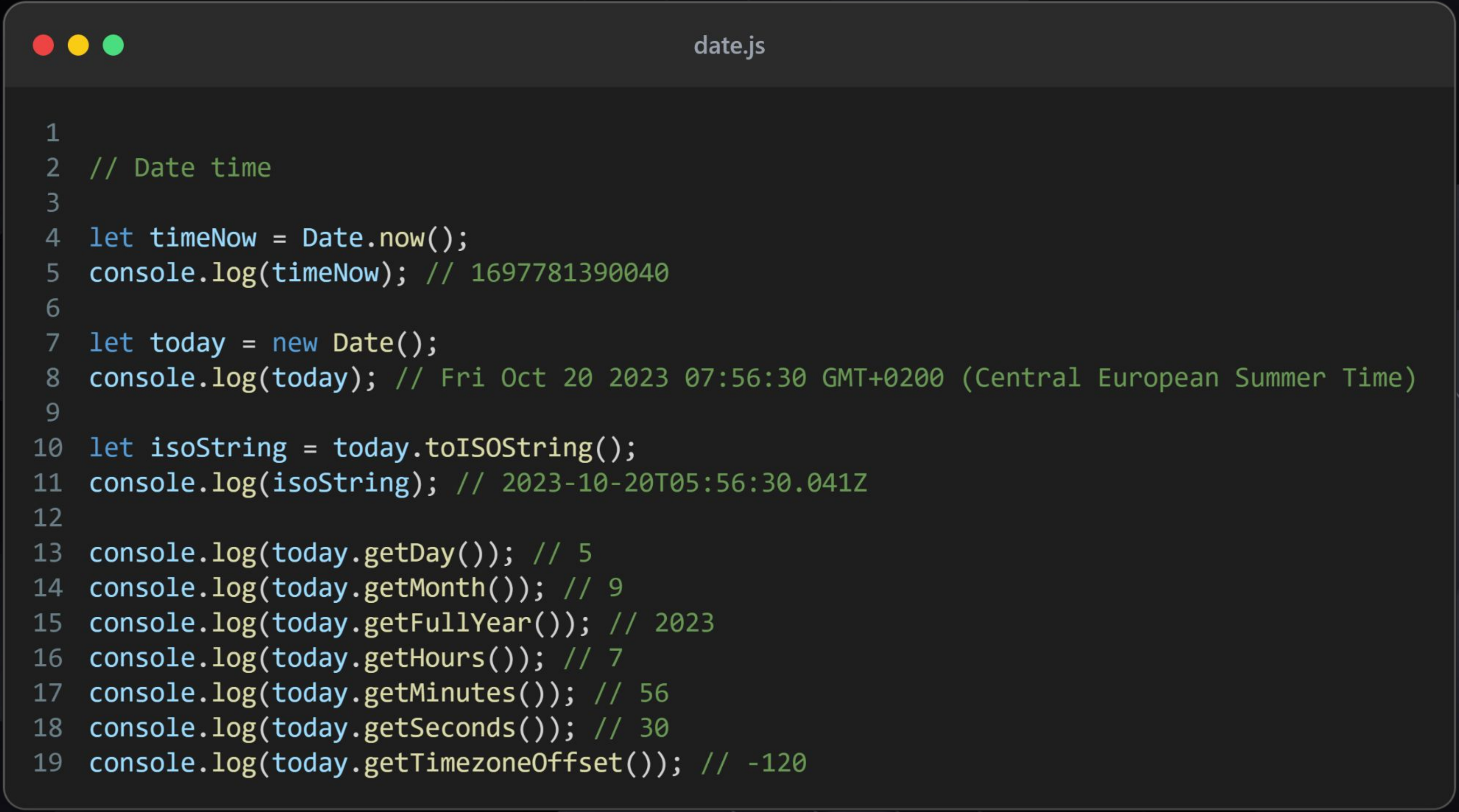
```javascript

let array = [1, 6, 23, 8, 4, 98, 3, 7, 3, 98, 4, 98];

// Calculate average with for...of loop
function calculateAverage(numbersArray) {
  let sum = 0;

  for (number of numbersArray) {
    sum += number;
  }

  let result = sum / numbersArray.length;

  return +(Math.round(result * 100) / 100).toFixed(2);
}

console.log(calculateAverage(array));

```

Date in JS

Mozilla MDN Date JS

Date in JS

- Time in milliseconds that has elapsed since the epoch (01.01.1970 UTC)
- Date and time works in the local time zone and offset!
- YYYY-MM-DDTHH:mm:ss.sssZ


date.js

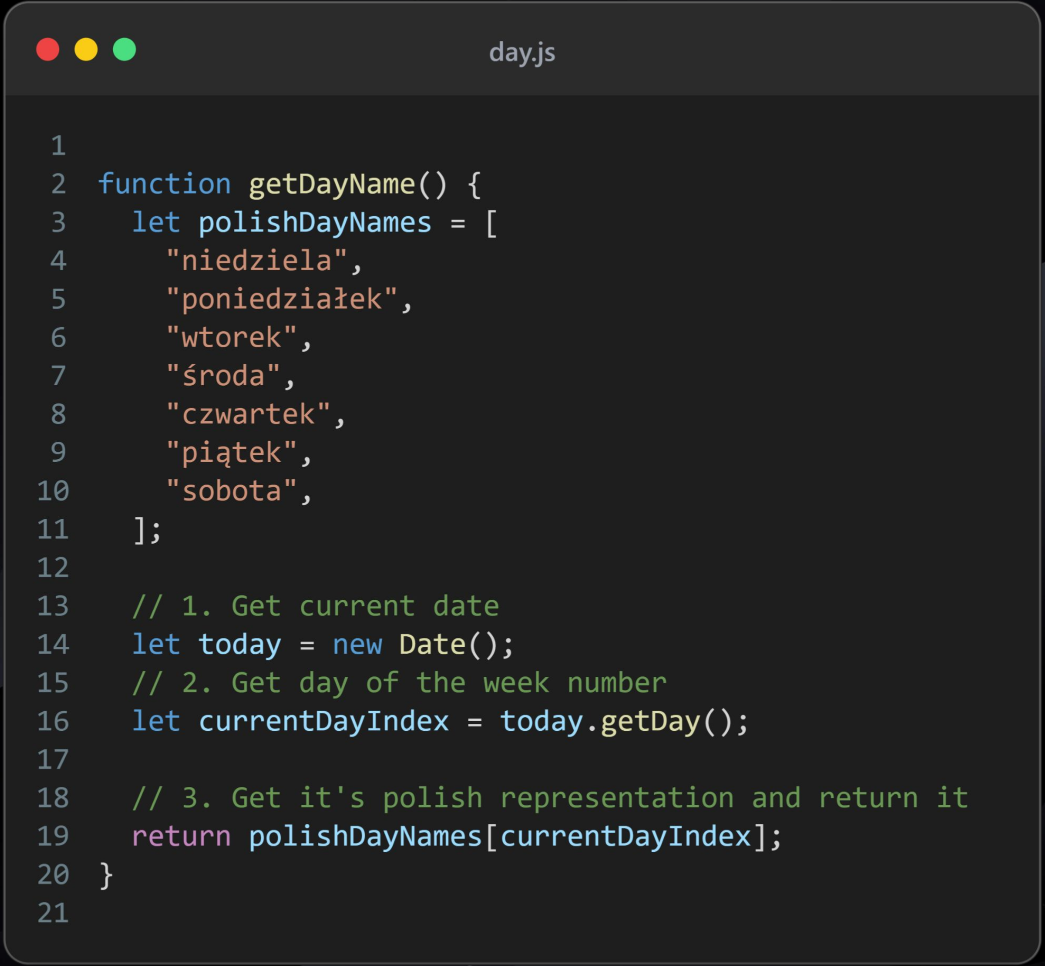
```javascript

// Date time

let timeNow = Date.now();
console.log(timeNow); // 1697781390040

let today = new Date();
console.log(today); // Fri Oct 20 2023 07:56:30 GMT+0200 (Central European Summer Time)

let isoString = today.toISOString();
console.log(isoString); // 2023-10-20T05:56:30.041Z

console.log(today.getDay()); // 5
console.log(today.getMonth()); // 9
console.log(today.getFullYear()); // 2023
console.log(today.getHours()); // 7
console.log(today.getMinutes()); // 56
console.log(today.getSeconds()); // 30
console.log(today.getTimezoneOffset()); // -120
```

Date - alternatives

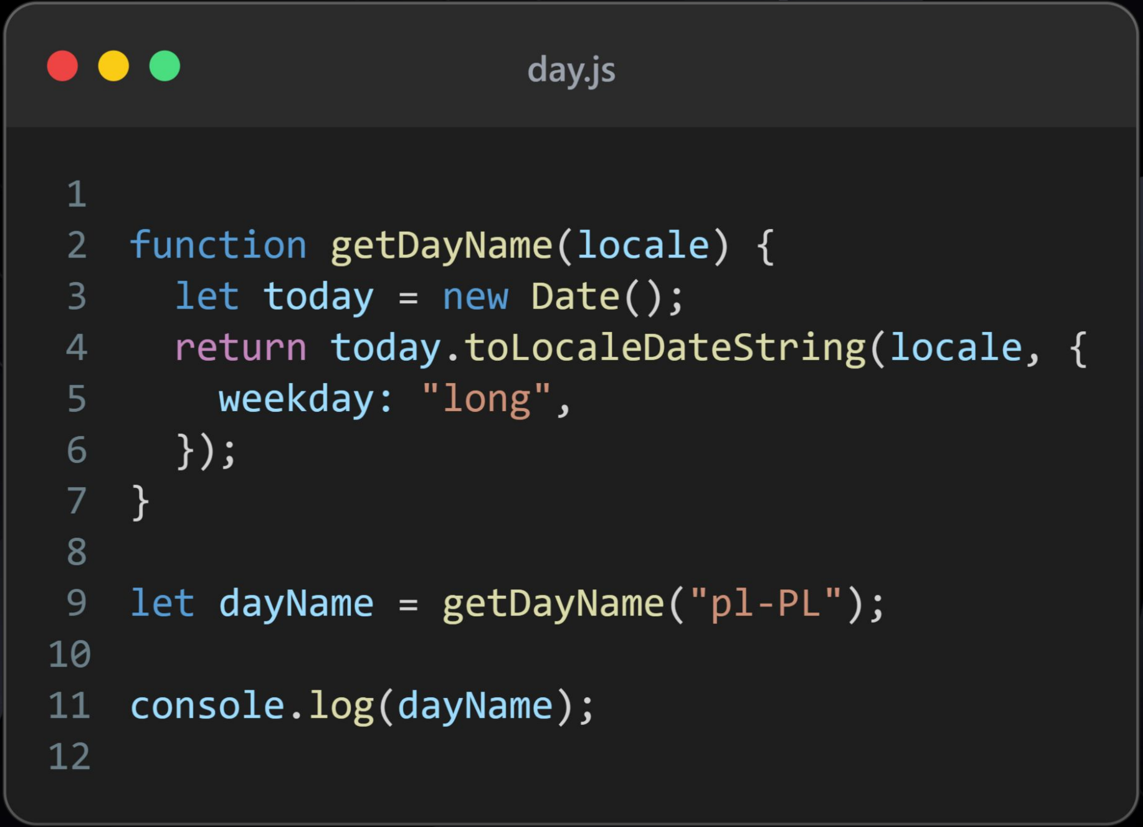
- Day.js
- date-fns
- temporal - tc39 stage 3 proposal

What's the day today?

- Write a function that will tell us what the day is today in Polish


day.js

```javascript

function getDayName() {
  let polishDayNames = [
    "niedziela",
    "poniedziałek",
    "wtorek",
    "środa",
    "czwartek",
    "piątek",
    "sobota",
  ];

  // 1. Get current date
  let today = new Date();
  // 2. Get day of the week number
  let currentDayIndex = today.getDay();

  // 3. Get it's polish representation and return it
  return polishDayNames[currentDayIndex];
}

```

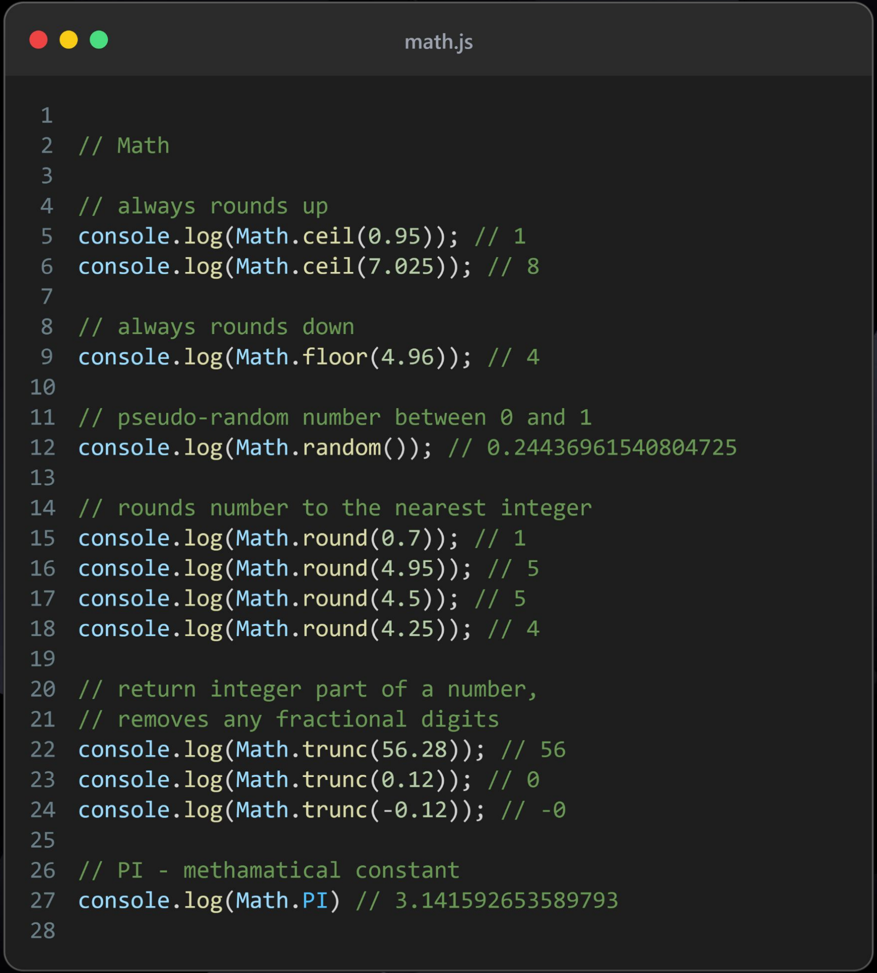


day.js

```
function getDayName(locale) {
  let today = new Date();
  return today.toLocaleDateString(locale, {
    weekday: "long",
  });
}

let dayName = getDayName("pl-PL");

console.log(dayName);
```

Math – JS

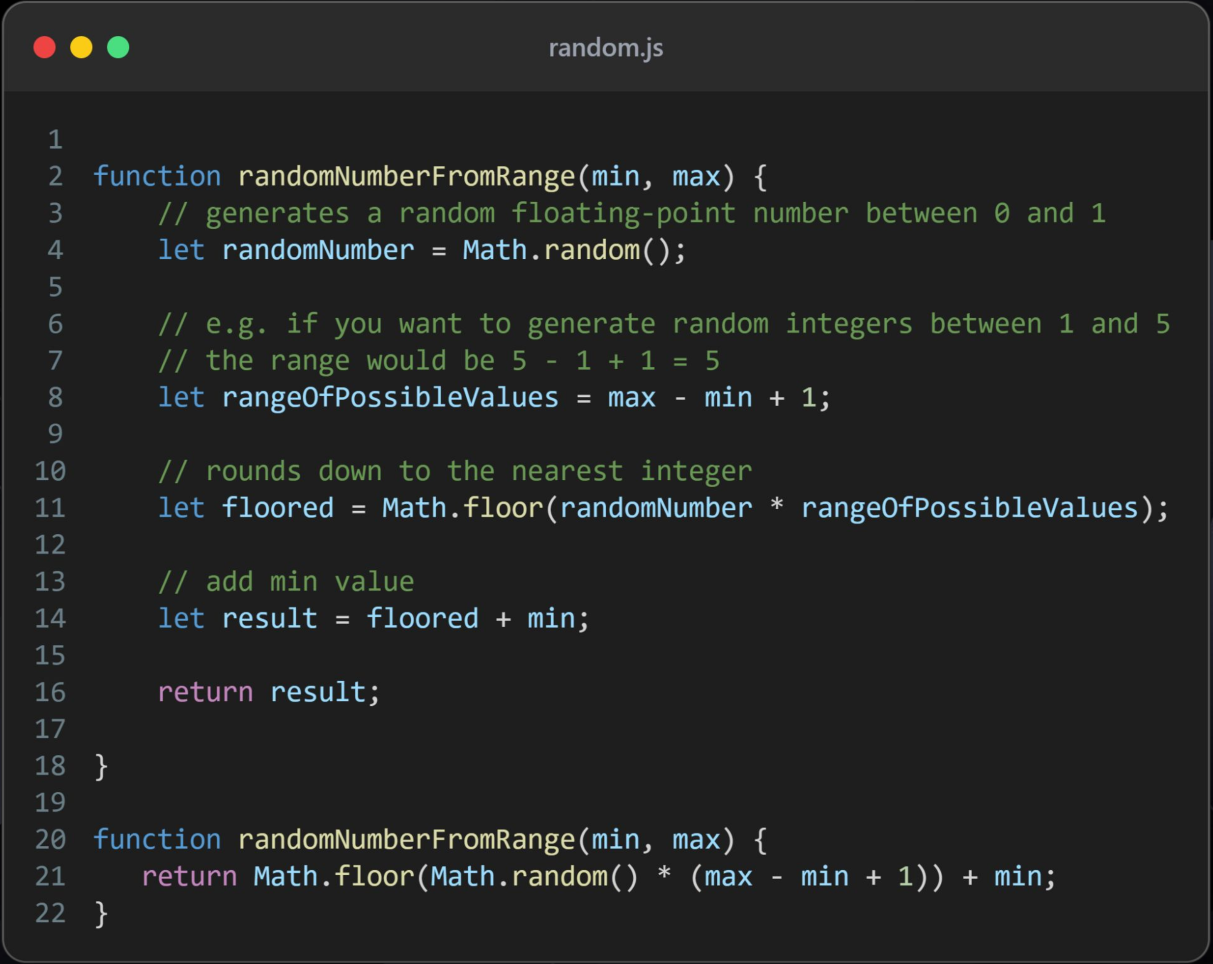
- Static object containing methods
- Contains properties and methods for mathematical constants and functions
- Works with the Number type. Won’t work with BigInt


math.js

```

// Math

// always rounds up
console.log(Math.ceil(0.95)); // 1
console.log(Math.ceil(7.025)); // 8

// always rounds down
console.log(Math.floor(4.96)); // 4

// pseudo-random number between 0 and 1
console.log(Math.random()); // 0.24436961540804725

// rounds number to the nearest integer
console.log(Math.round(0.7)); // 1
console.log(Math.round(4.95)); // 5
console.log(Math.round(4.5)); // 5
console.log(Math.round(4.25)); // 4

// return integer part of a number,
// removes any fractional digits
console.log(Math.trunc(56.28)); // 56
console.log(Math.trunc(0.12)); // 0
console.log(Math.trunc(-0.12)); // -0

// PI - methamatical constant
console.log(Math.PI) // 3.141592653589793

```

Random number from range

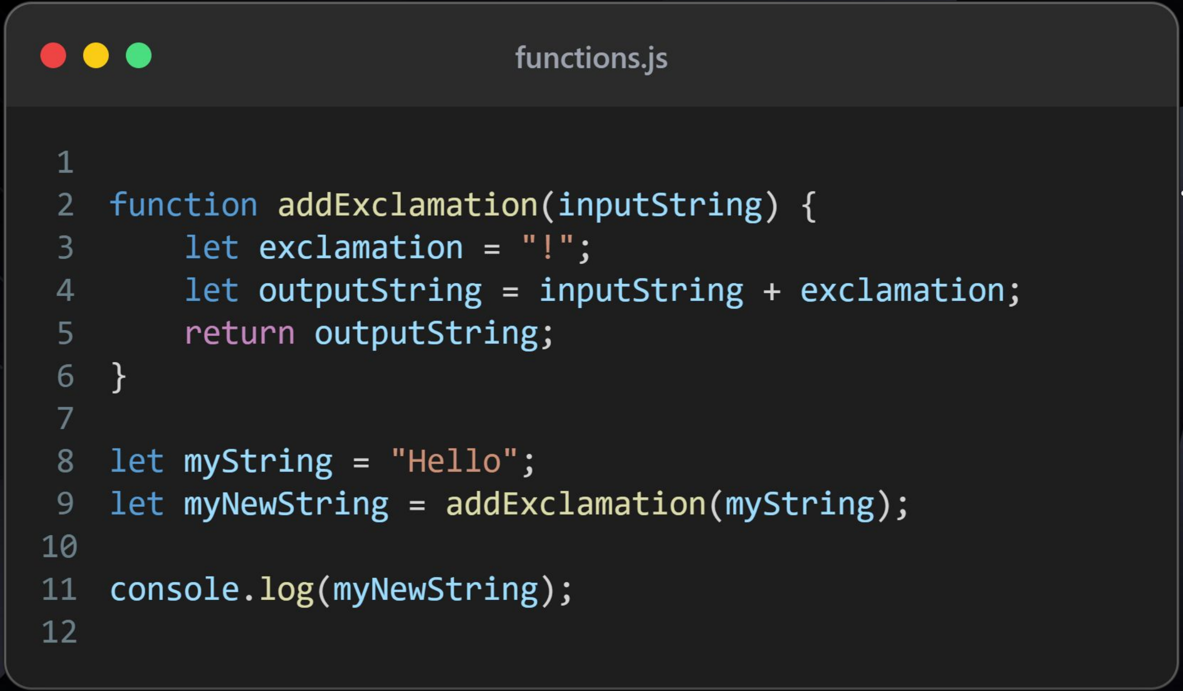
- Write a function that will return random number in the given range


random.js

```javascript

function randomNumberFromRange(min, max) {
    // generates a random floating-point number between 0 and 1
    let randomNumber = Math.random();

    // e.g. if you want to generate random integers between 1 and 5
    // the range would be 5 - 1 + 1 = 5
    let rangeOfPossibleValues = max - min + 1;

    // rounds down to the nearest integer
    let floored = Math.floor(randomNumber * rangeOfPossibleValues);

    // add min value
    let result = floored + min;

    return result;

}

function randomNumberFromRange(min, max) {
   return Math.floor(Math.random() * (max - min + 1)) + min;
}
```

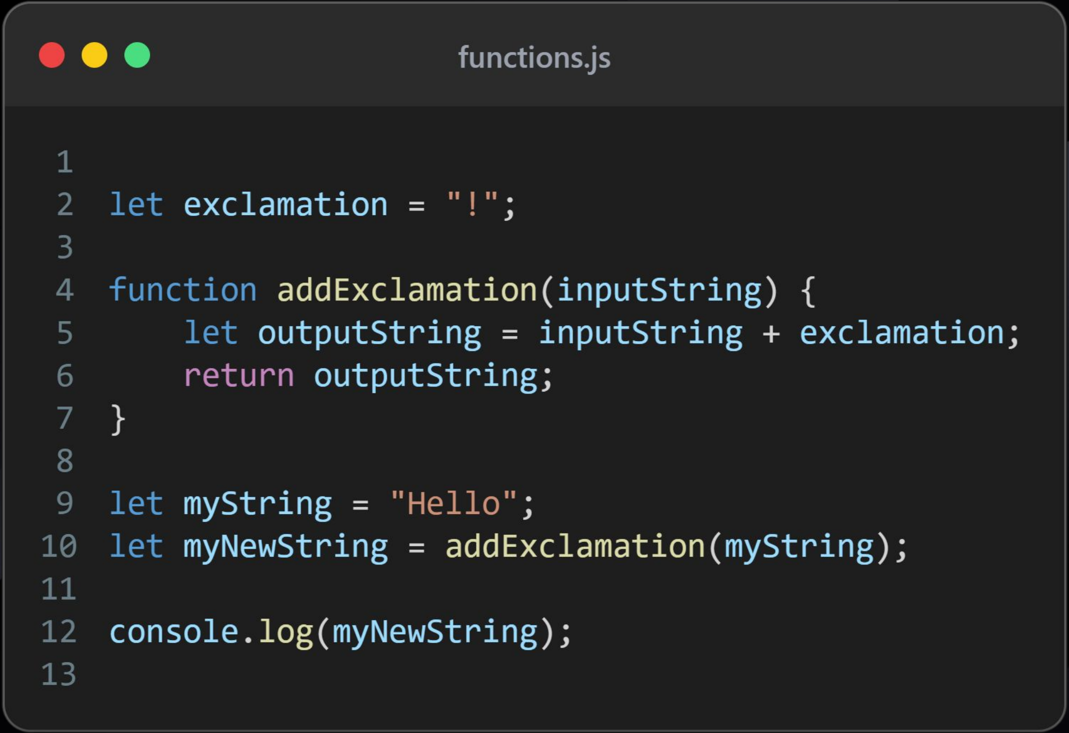
Functions – hard parts

- Execution context
- Execution thread
- Call stack
- Memory



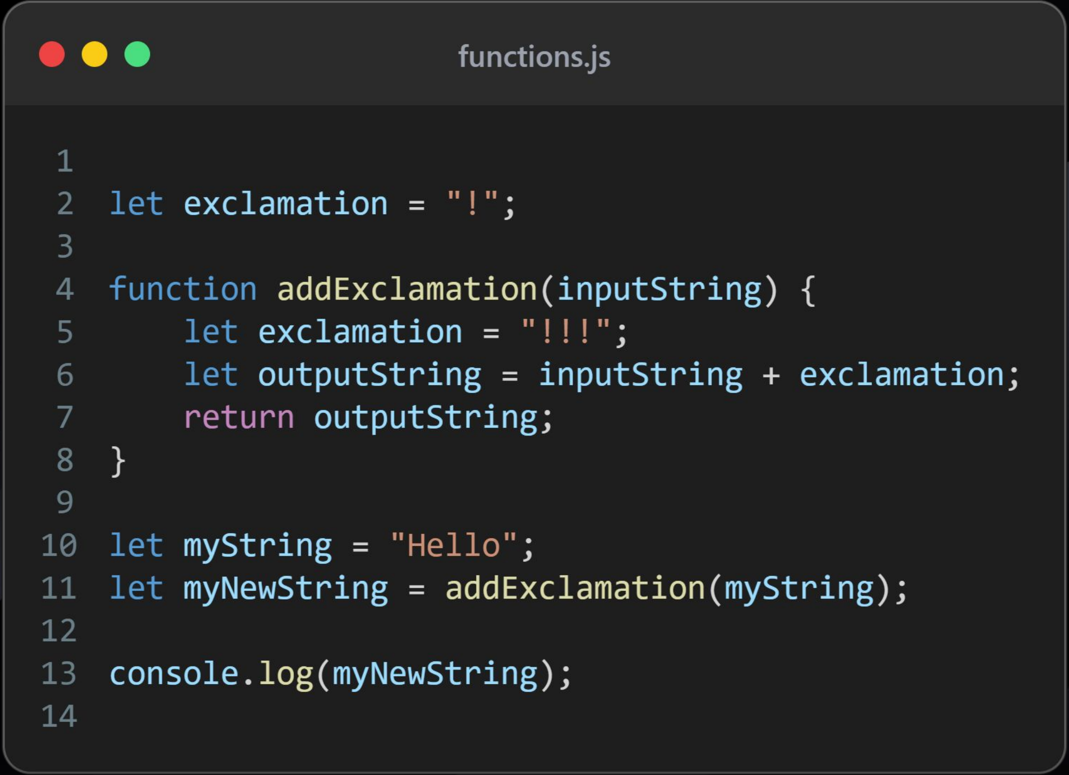
functions.js

```javascript

function addExclamation(inputString) {
    let exclamation = "!";
    let outputString = inputString + exclamation;
    return outputString;
}

let myString = "Hello";
let myNewString = addExclamation(myString);

console.log(myNewString);

```

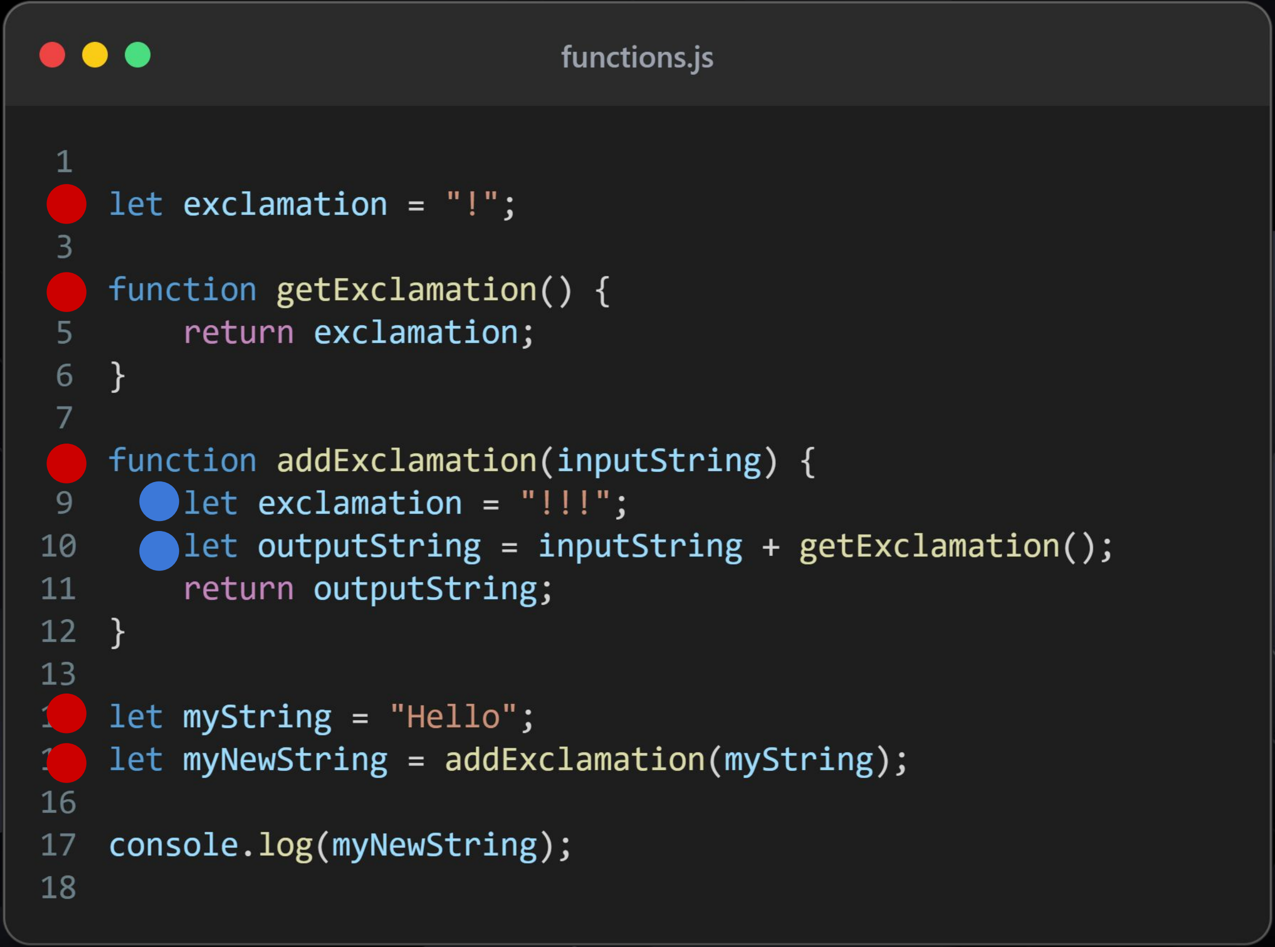


functions.js

```javascript

let exclamation = "!";

function addExclamation(inputString) {
    let outputString = inputString + exclamation;
    return outputString;
}

let myString = "Hello";
let myNewString = addExclamation(myString);

console.log(myNewString);

```


functions.js

```

let exclamation = "!";

function addExclamation(inputString) {
    let exclamation = "!!!";
    let outputString = inputString + exclamation;
    return outputString;
}

let myString = "Hello";
let myNewString = addExclamation(myString);

console.log(myNewString);

```


functions.js

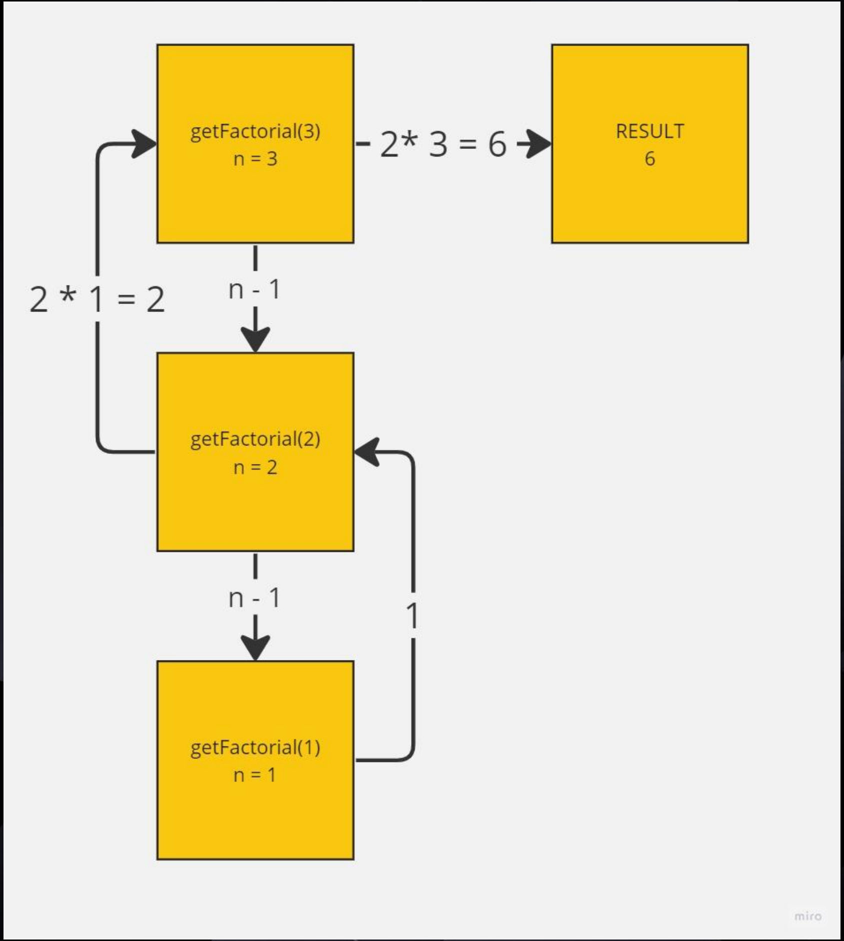
```

let exclamation = "!";

function getExclamation() {
    return exclamation;
}

function addExclamation(inputString) {
    let exclamation = "!!!";
    let outputString = inputString + getExclamation();
    return outputString;
}

let myString = "Hello";
let myNewString = addExclamation(myString);

console.log(myNewString);

```

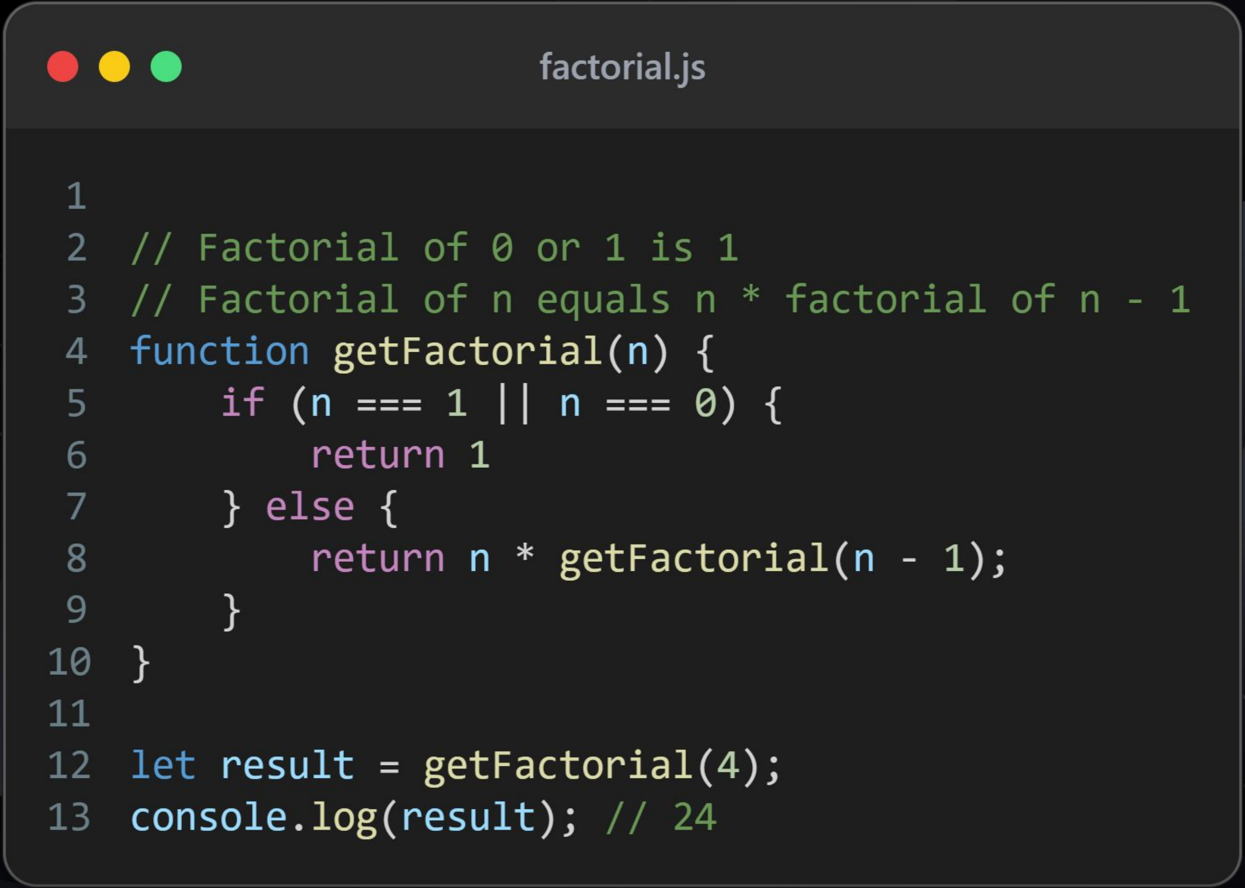
Scope

- Visibility of variables in any given point in JS
- Policy that manages the accessibility of variables
- The inner scope can access the variables of its outer scope

Factorial

- Calculate factorial of given number
- Use recursion
- 4! = 4 * 3! = 4 * 3 * 2! = 4 * 3 * 2 * 1

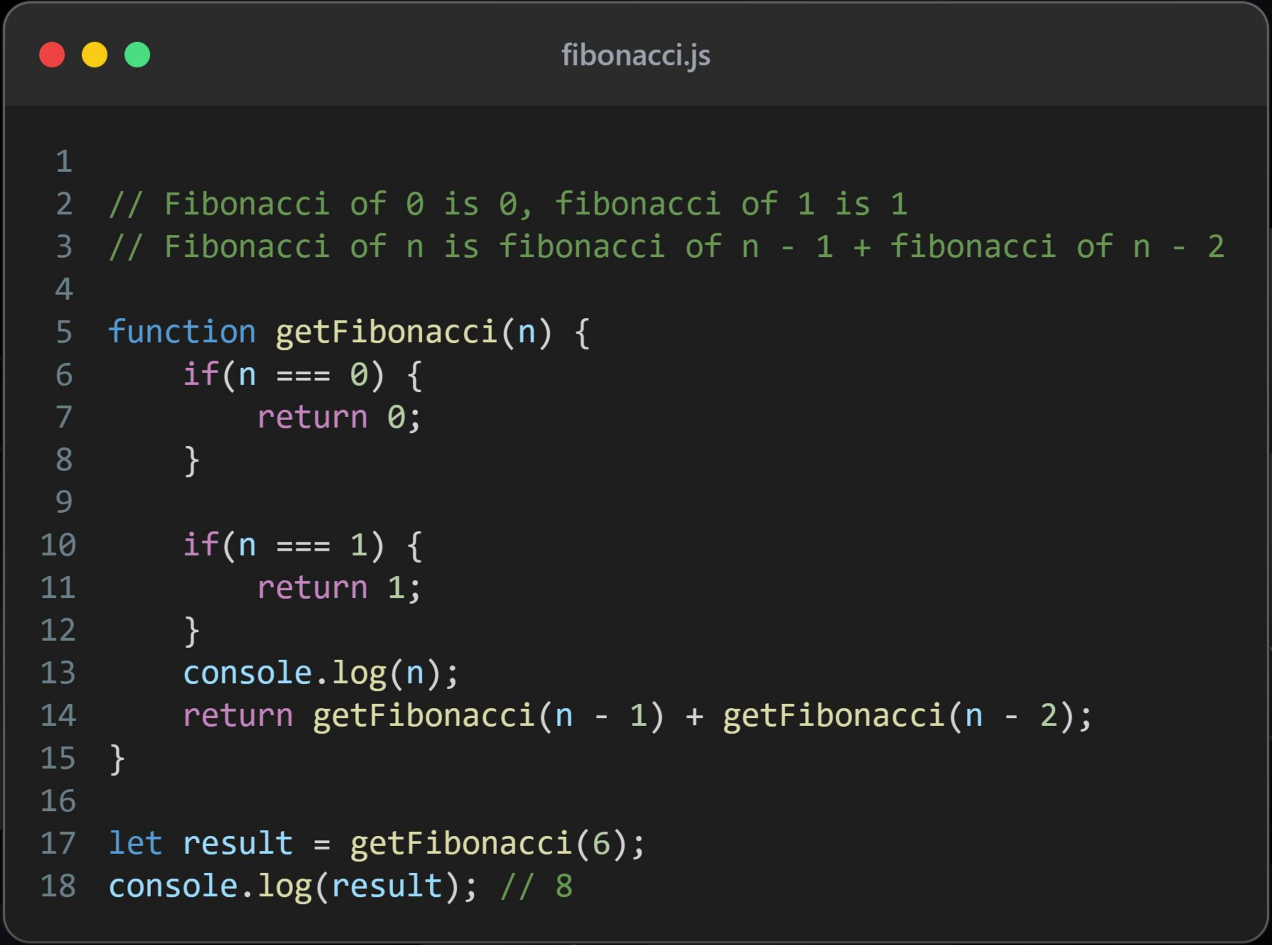

getFactorial(3)
n = 3
2* 3 = 6
RESULT
6
n - 1
2 * 1 = 2
getFactorial(2)
n = 2
n - 1
1
getFactorial(1)
n = 1
miro


factorial.js

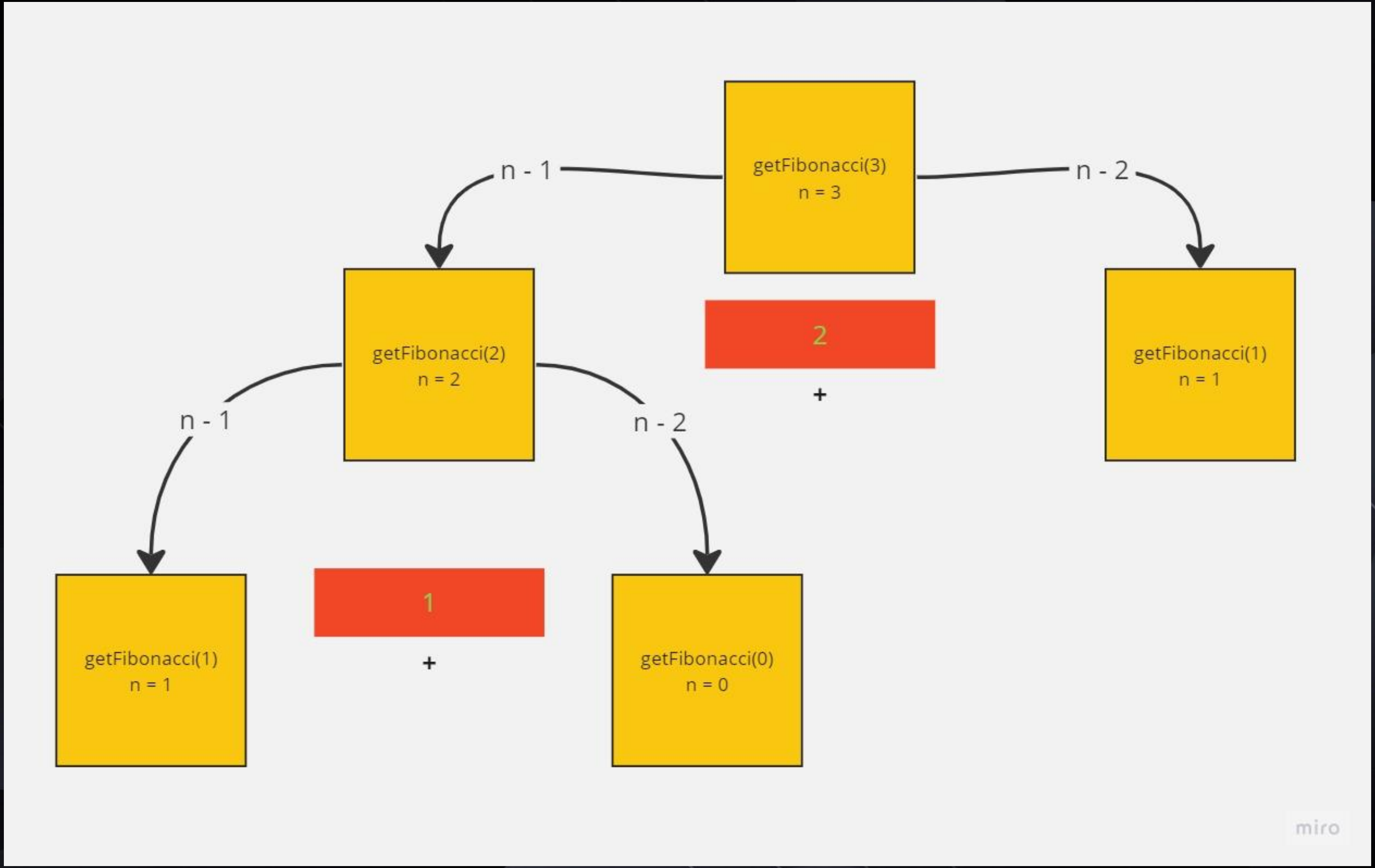
```javascript

// Factorial of 0 or 1 is 1
// Factorial of n equals n * factorial of n - 1
function getFactorial(n) {
    if (n === 1 || n === 0) {
        return 1
    } else {
        return n * getFactorial(n - 1);
    }
}

let result = getFactorial(4);
console.log(result); // 24
```

Fibonacci numbers

- Calculate given number of Fibonacci numbers
- Use recursion
- 0, 1, 1, 2, 3, 5, 8, 13, ...



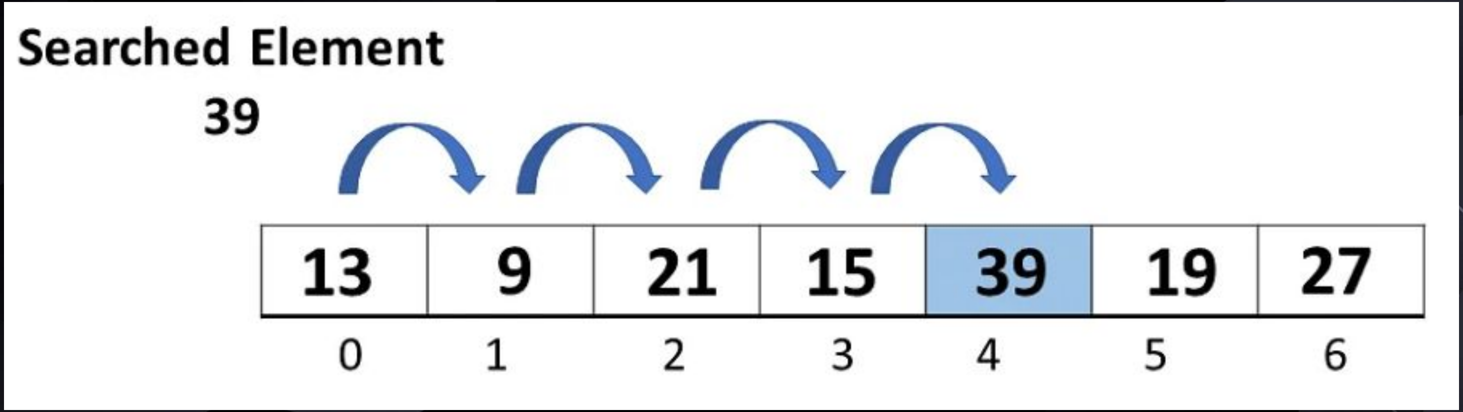
```javascript

// Fibonacci of 0 is 0, fibonacci of 1 is 1
// Fibonacci of n is fibonacci of n - 1 + fibonacci of n - 2

function getFibonacci(n) {
    if(n === 0) {
        return 0;
    }

    if(n === 1) {
        return 1;
    }
    console.log(n);
    return getFibonacci(n - 1) + getFibonacci(n - 2);
}

let result = getFibonacci(6);
console.log(result); // 8
```

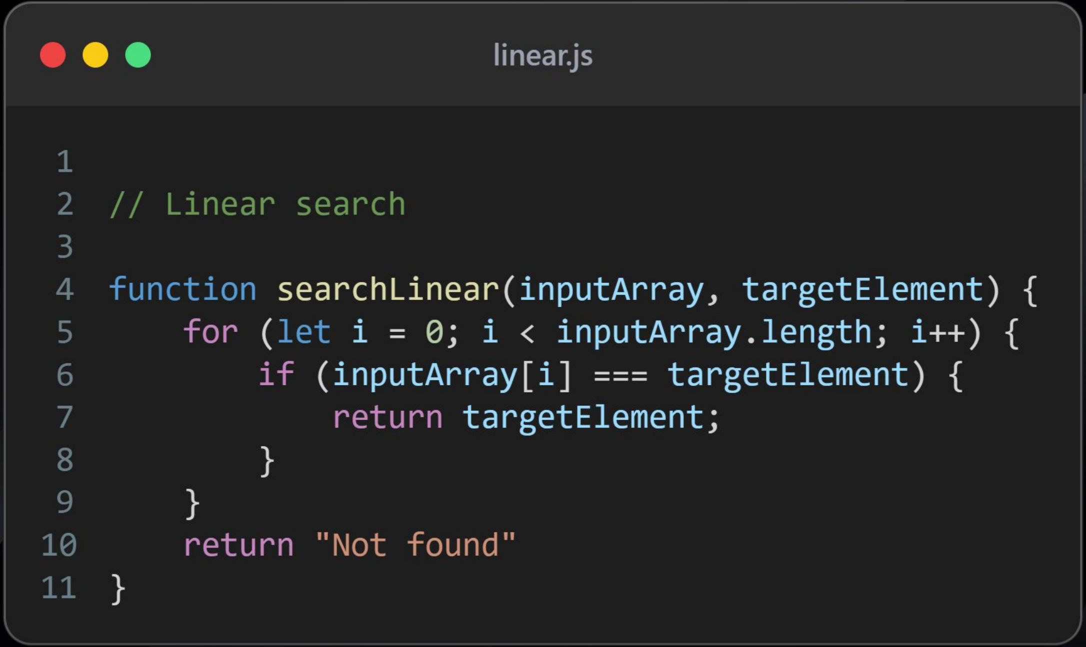

getFibonacci(3)
n = 3
n - 1
n - 2
getFibonacci(2)
n = 2
2
+
getFibonacci(1)
n = 1
n - 1
n - 2
getFibonacci(1)
n = 1
1
+
getFibonacci(0)
n = 0
miro

Search algorithms

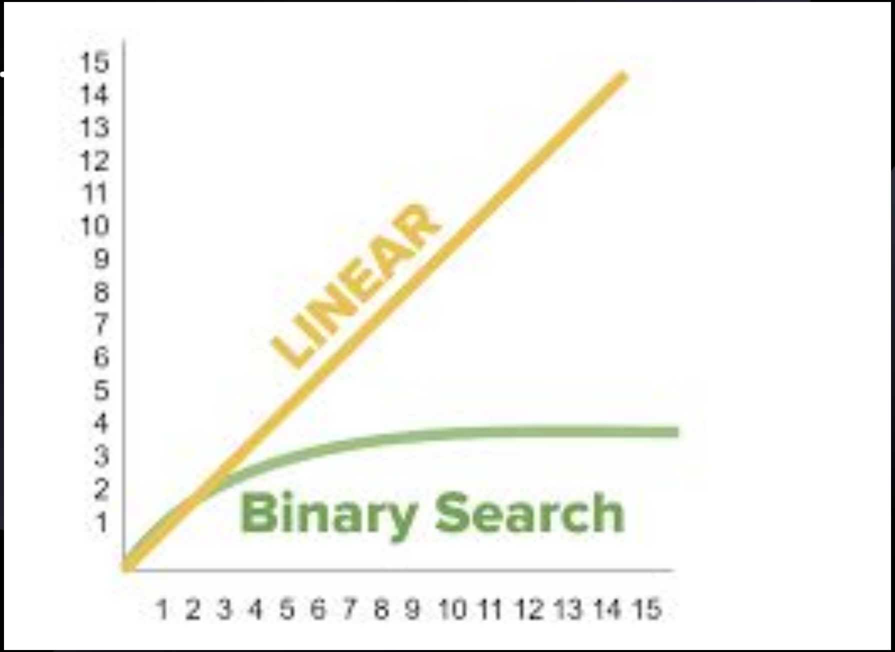
- Linear search
- Binary search

Linear search

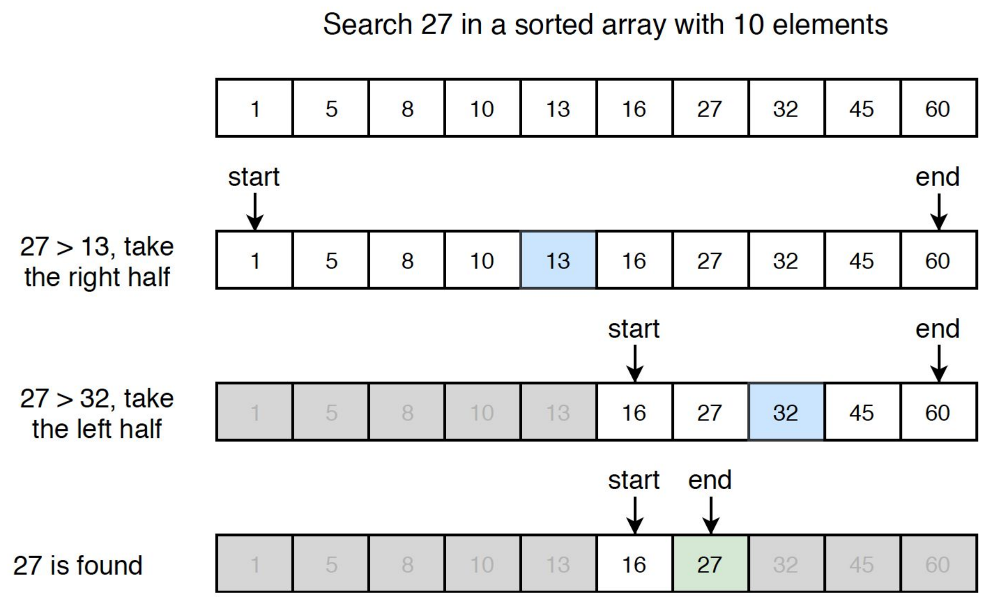




```

// Linear search

function searchLinear(inputArray, targetElement) {
    for (let i = 0; i < inputArray.length; i++) {
        if (inputArray[i] === targetElement) {
            return targetElement;
        }
    }
    return "Not found"
}
```


LINEAR
Binary Search
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Binary Search



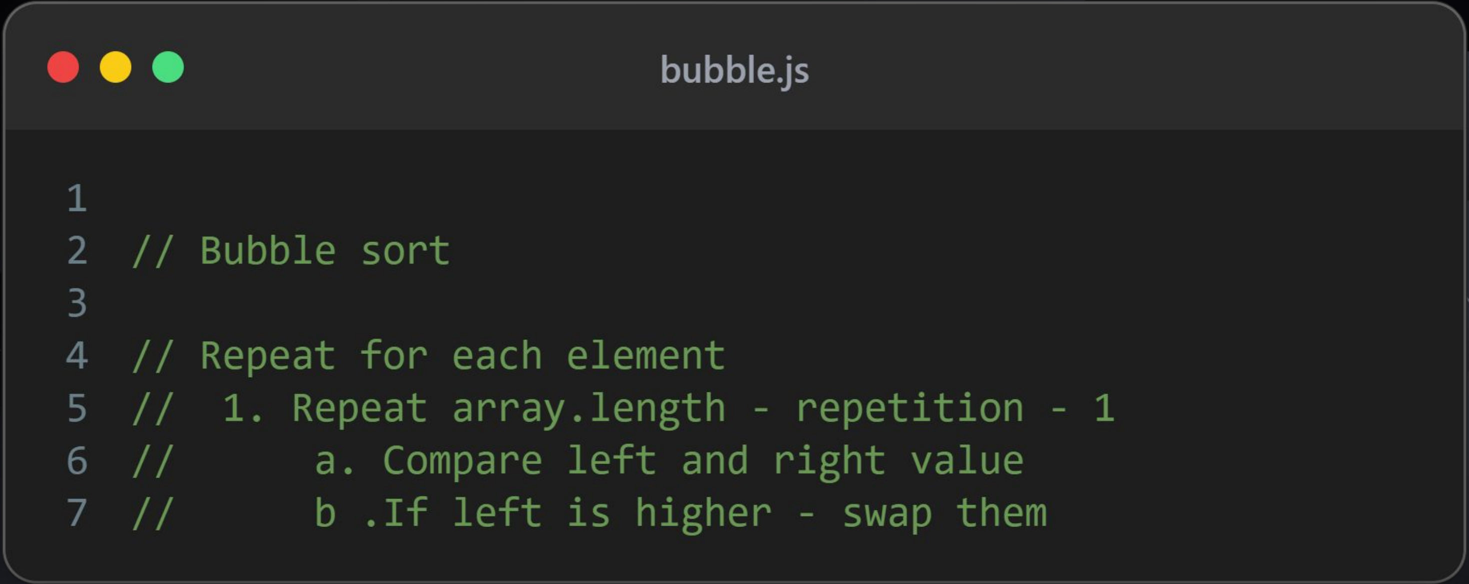
Binary search

- If no elements
  - Return -1
- If sought number in middle
  - Return middle index
- Else if number < middle number
  - Search left half
- Else if number > middle number
  - Search right half

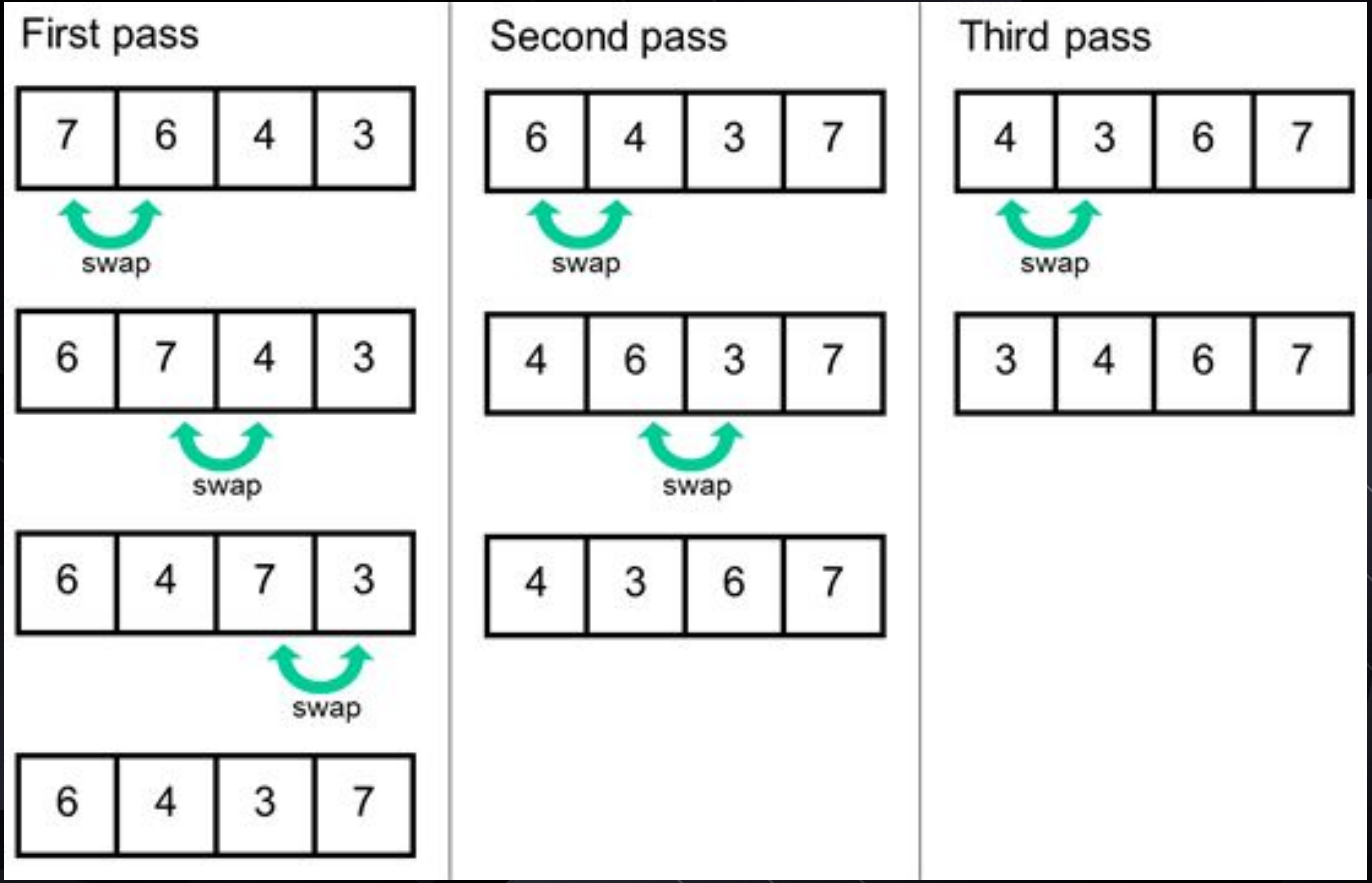
Sort algorithms

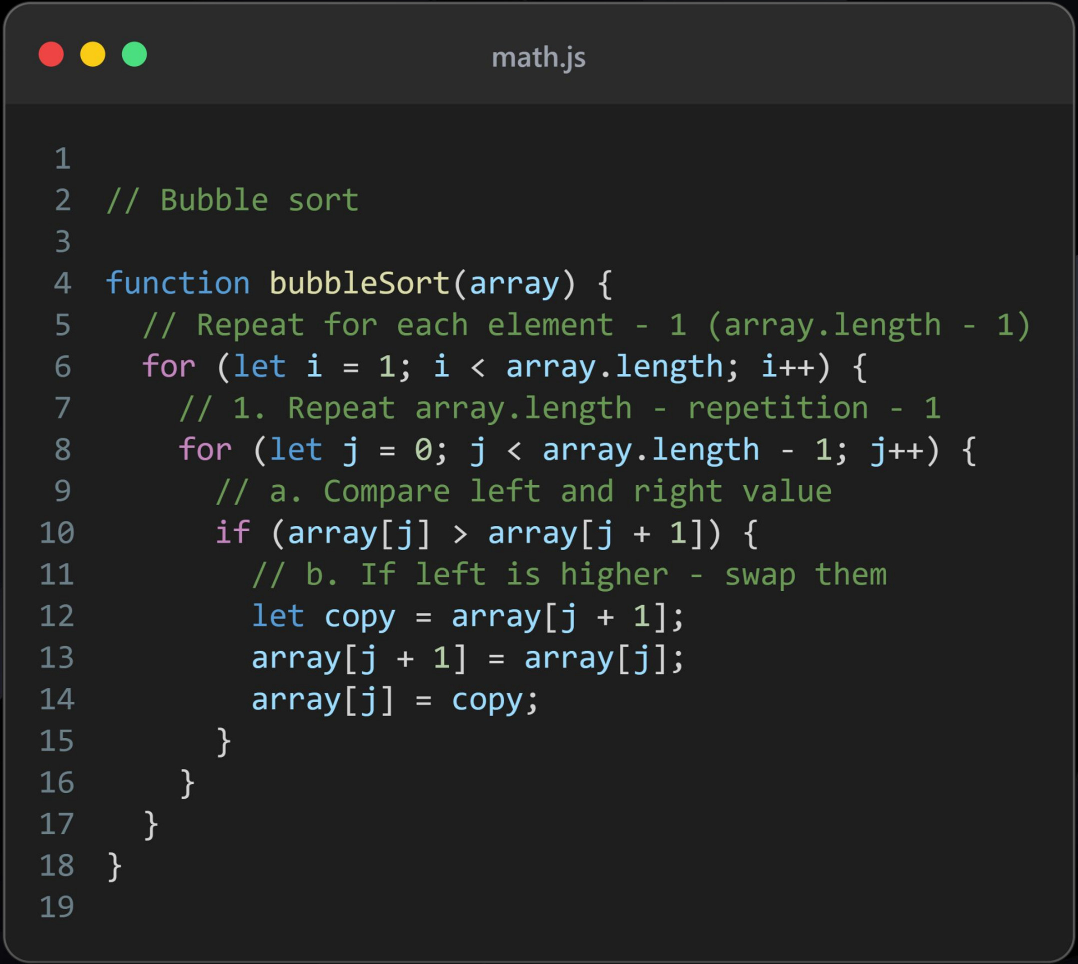
- Selection sort
- Bubble sort
- Merge sort

Sound of sorting


bubble.js

```

// Bubble sort

// Repeat for each element
//  1. Repeat array.length - repetition - 1
//      a. Compare left and right value
//      b .If left is higher - swap them
```

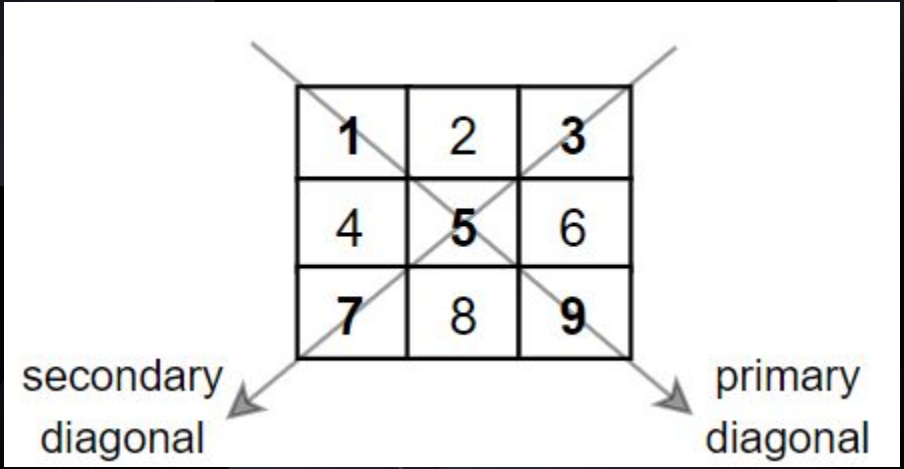

First pass
7 6 4 3
swap
6 7 4 3
swap
6 4 7 3
swap
6 4 3 7
Second pass
6 4 3 7
swap
4 6 3 7
swap
4 3 6 7
Third pass
4 3 6 7
swap
3 4 6 7

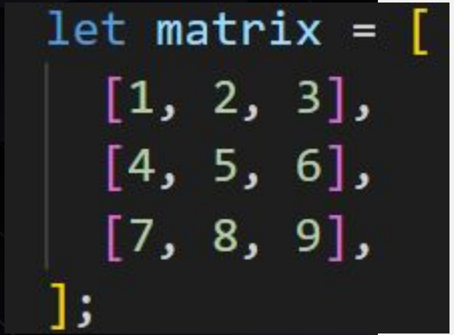


```javascript

// Bubble sort

function bubbleSort(array) {
  // Repeat for each element - 1 (array.length - 1)
  for (let i = 1; i < array.length; i++) {
    // 1. Repeat array.length - repetition - 1
    for (let j = 0; j < array.length - 1; j++) {
      // a. Compare left and right value
      if (array[j] > array[j + 1]) {
        // b. If left is higher - swap them
        let copy = array[j + 1];
        array[j + 1] = array[j];
        array[j] = copy;
      }
    }
  }
}

```

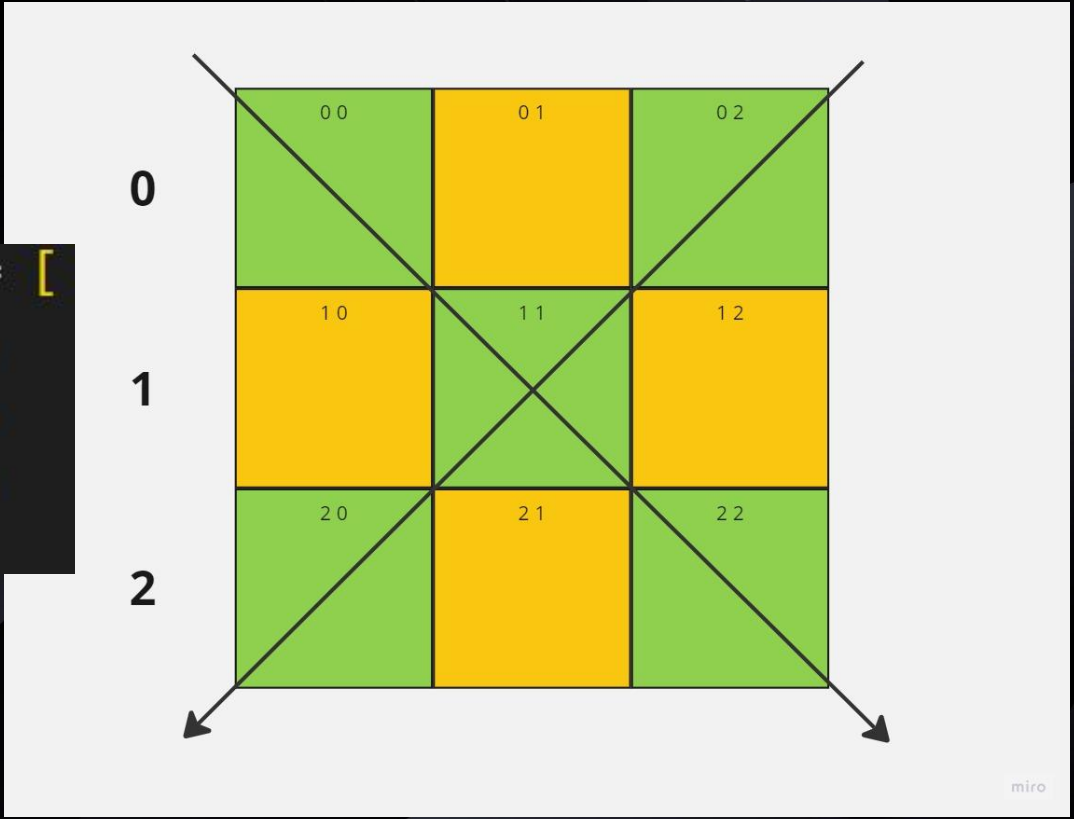
Diagonal sum

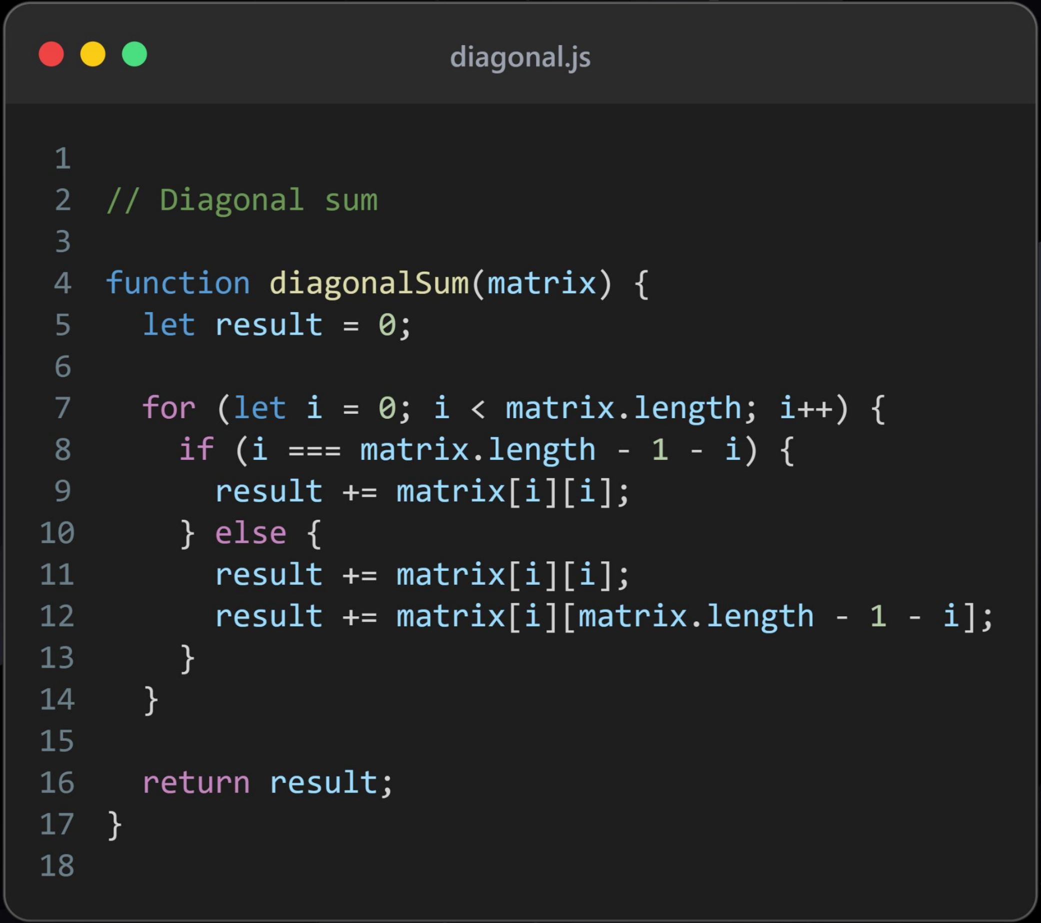
- Calculate the sum of all the elements on the primary diagonal and all the elements on the secondary diagonal that are not part of the primary diagonal





```
let matrix = [
  [1, 2, 3],
  [4, 5, 6],
  [7, 8, 9],
];
```




diagonal.js

```javascript

// Diagonal sum

function diagonalSum(matrix) {
  let result = 0;

  for (let i = 0; i < matrix.length; i++) {
    if (i === matrix.length - 1 - i) {
      result += matrix[i][i];
    } else {
      result += matrix[i][i];
      result += matrix[i][matrix.length - 1 - i];
    }
  }

  return result;
}

```

HOMEWORK

- Array exercises
- Days till friday
- Caesar Cipher - encrypt and decrypt
- Prime numbers
- Selection sort and merge sort
- Fibonnacci numbers
- Binary search