



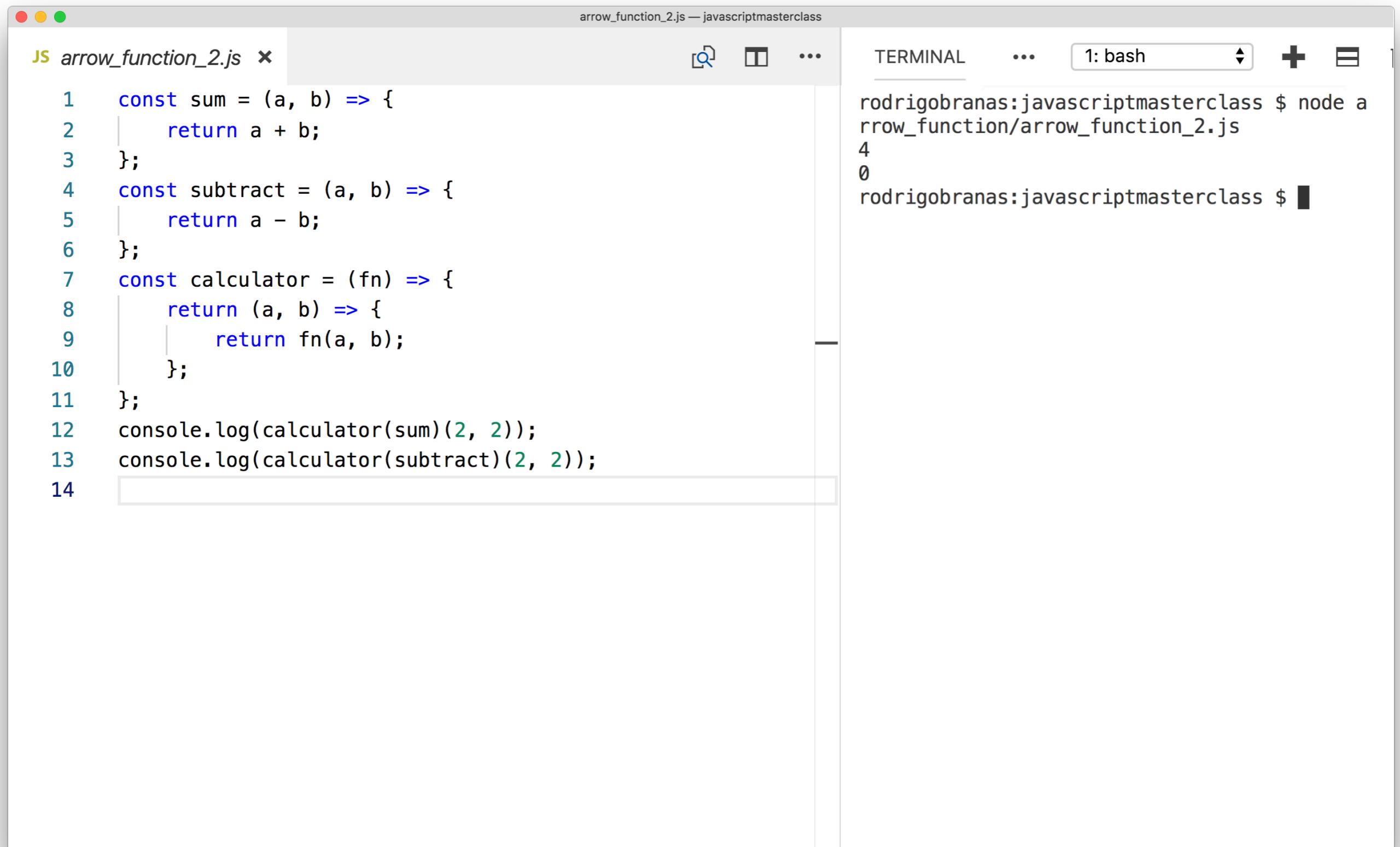
Arrow Function

As **arrow functions** tem uma abordagem mais simples e direta para escrever uma função e podem melhorar a legibilidade do código em diversas situações

The screenshot shows a Mac OS X desktop environment with a terminal window open. The title bar of the terminal window reads "arrow_function_1.js — javascriptmasterclass". The main pane of the terminal displays the command "node arrow_function/arrow_function_1.js" followed by the output "4" and "0". The terminal has a standard OS X interface with red, yellow, and green close buttons at the top left.

```
JS arrow_function_1.js x arrow_function_1.js — javascriptmasterclass TERMINAL ... 1: bash rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_1.js 4 0 rodrigobranas:javascriptmasterclass $
```

```
1  const sum = function(a, b) {
2     return a + b;
3 }
4  const subtract = function(a, b) {
5     return a - b;
6 }
7  const calculator = function(fn) {
8     return function(a, b) {
9         return fn(a, b);
10    };
11 }
12 console.log(calculator(sum)(2, 2));
13 console.log(calculator(subtract)(2, 2));
14
```



A screenshot of a Mac OS X terminal window titled "arrow_function_2.js — javascriptmasterclass". The window is divided into two main sections: a code editor on the left and a terminal on the right.

The code editor contains the following JavaScript code:

```
JS arrow_function_2.js ×
1  const sum = (a, b) => {
2    return a + b;
3  };
4  const subtract = (a, b) => {
5    return a - b;
6  };
7  const calculator = (fn) => {
8    return (a, b) => {
9      return fn(a, b);
10   };
11 };
12 console.log(calculator(sum)(2, 2));
13 console.log(calculator(subtract)(2, 2));
14
```

The terminal section shows the output of running the script:

```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_2.js
4
0
rodrigobranas:javascriptmasterclass $ █
```

arrow_function_3.js — javascriptmasterclass

JS arrow_function_3.js x

1 const sum = (a, b) => a + b;
2 const subtract = (a, b) => a - b;
3 const calculator = (fn) => (a, b) => fn(a, b);
4 console.log(calculator(sum)(2, 2));
5 console.log(calculator(subtract)(2, 2));
6

TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_3.js  
4  
0  
rodrigobranas:javascriptmasterclass $
```

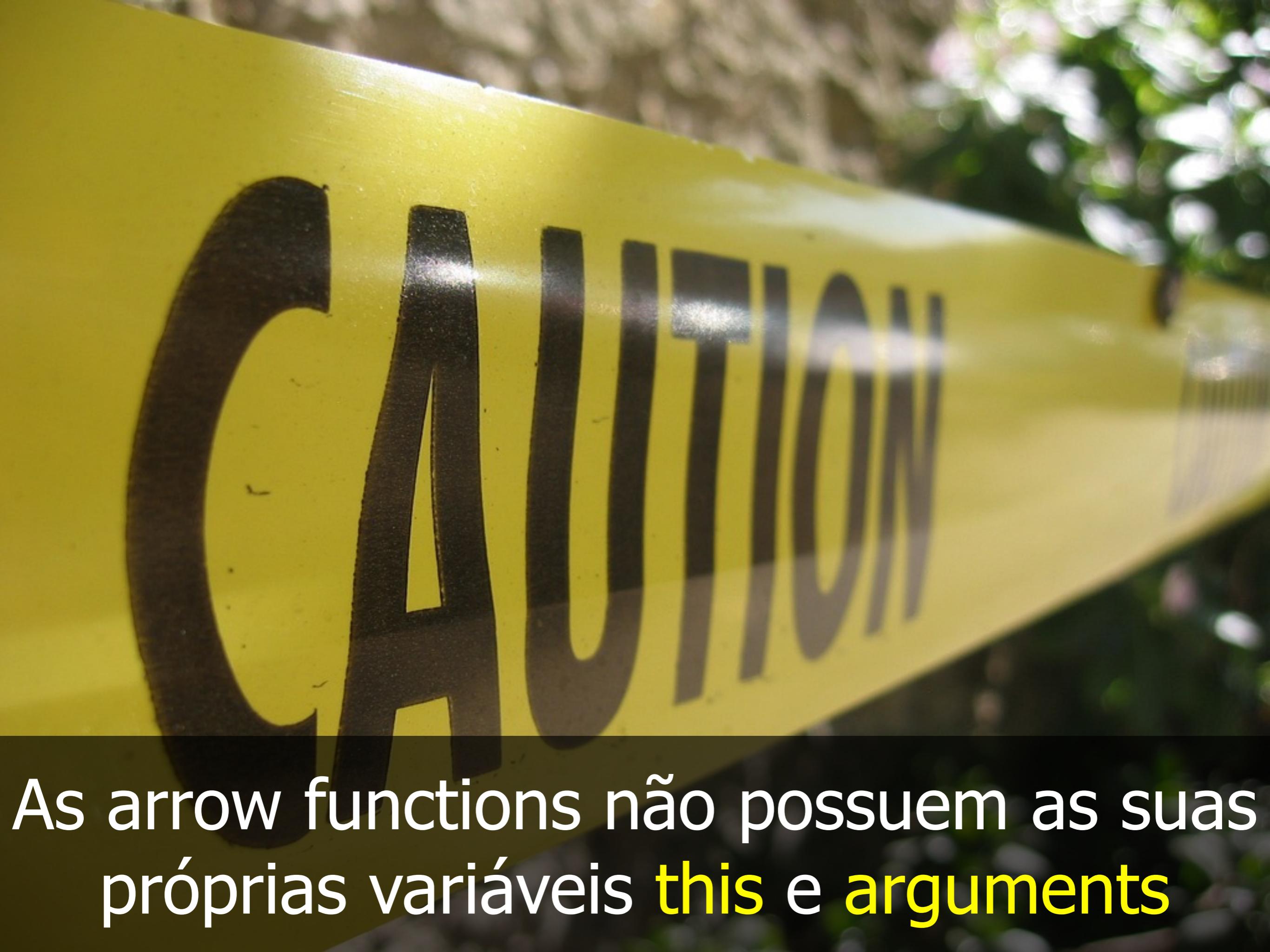
arrow_function_4.js — javascriptmasterclass

JS arrow_function_4.js x

1 const sum = (a, b) => a + b;
2 const subtract = (a, b) => a - b;
3 const calculator = fn => (a, b) => fn(a, b);
4 console.log(calculator(sum)(2, 2));
5 console.log(calculator(subtract)(2, 2));
6

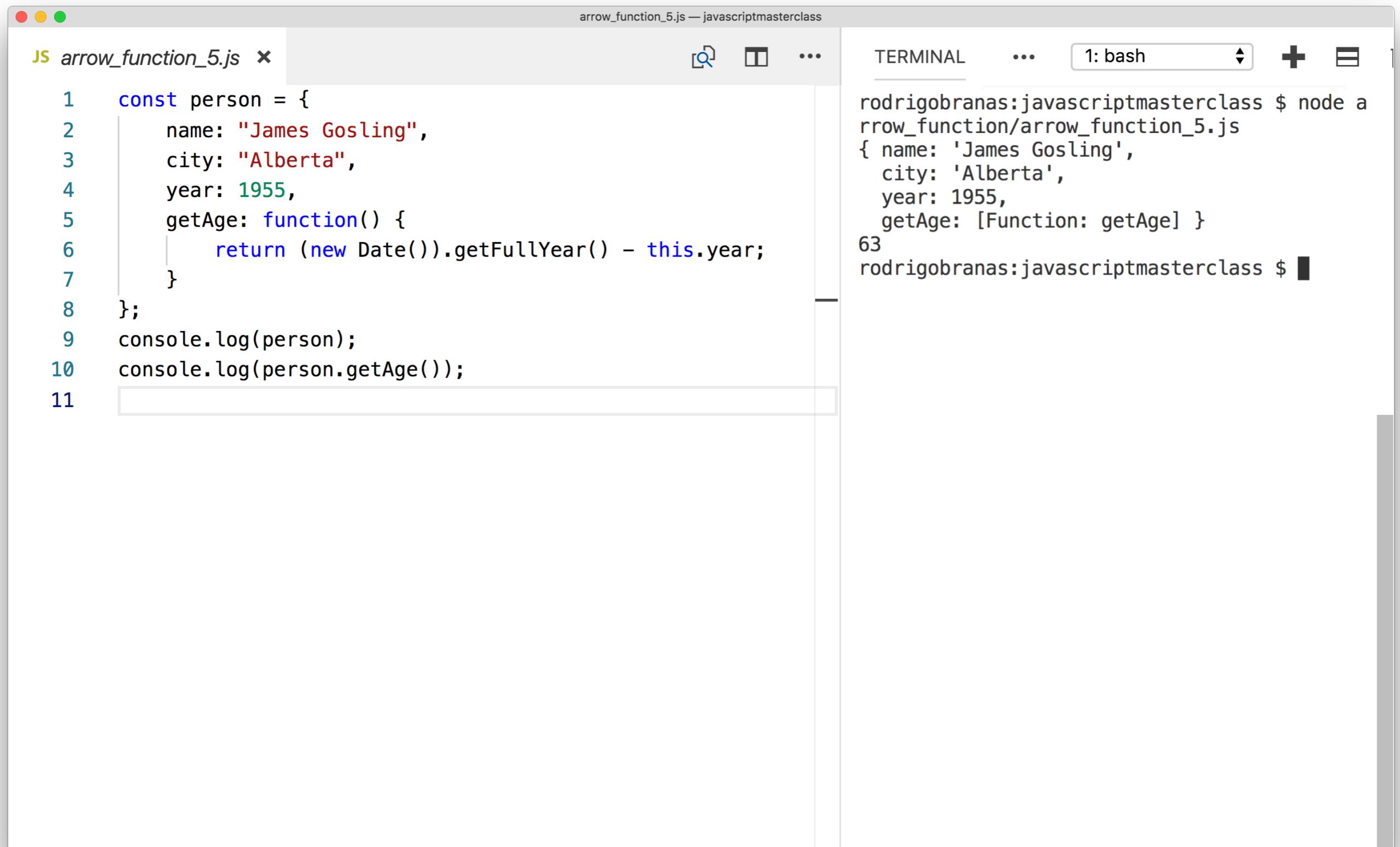
TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_4.js  
4  
0  
rodrigobranas:javascriptmasterclass $
```



CAUTION

As arrow functions não possuem as suas
próprias variáveis **this** e **arguments**



A screenshot of a Mac OS X terminal window titled "arrow_function_5.js — javascriptmasterclass". The window is divided into two main sections: a code editor on the left and a terminal on the right.

The code editor contains the following JavaScript code:

```
JS arrow_function_5.js ×
1 const person = {
2   name: "James Gosling",
3   city: "Alberta",
4   year: 1955,
5   getAge: function() {
6     return (new Date()).getFullYear() - this.year;
7   }
8 };
9 console.log(person);
10 console.log(person.getAge());
11
```

The terminal section shows the output of running the script with "node".

```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_5.js
{ name: 'James Gosling',
  city: 'Alberta',
  year: 1955,
  getAge: [Function: getAge] }
63
rodrigobranas:javascriptmasterclass $
```

The screenshot shows a Mac OS X desktop environment with a terminal window open. The terminal window has a title bar "arrow_function_6.js — javascriptmasterclass". The main pane of the terminal displays the output of a Node.js command:

```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_6.js
{ name: 'James Gosling',
  city: 'Alberta',
  year: 1955,
  getAge: [Function: getAge] }
NaN
```

The terminal window also includes standard OS X interface elements like a magnifying glass icon, a trash can icon, and a close button.

The code editor on the left contains the following JavaScript code:

```
JS arrow_function_6.js ×
1 const person = {
2   name: "James Gosling",
3   city: "Alberta",
4   year: 1955,
5   getAge: () => {
6     return (new Date()).getFullYear() - this.year;
7   }
8 };
9 console.log(person);
10 console.log(person.getAge());
11
```

A screenshot of a macOS desktop environment. On the left is a code editor window titled "arrow_function_7.js — javascriptmasterclass". The file contains the following JavaScript code:

```
JS arrow_function_7.js ×
1 const sum = function() {
2     let total = 0;
3     for(let argument in arguments) {
4         total += arguments[argument];
5     }
6     return total;
7 };
8 console.log(sum(1,2,3,4,5,6,7,8,9));
9
```

The code defines a function named "sum" that takes multiple arguments and returns their sum. It uses a for loop to iterate over the arguments object. The code is run in the terminal window on the right, which shows the output of the script being executed.

TERMINAL 1: bash

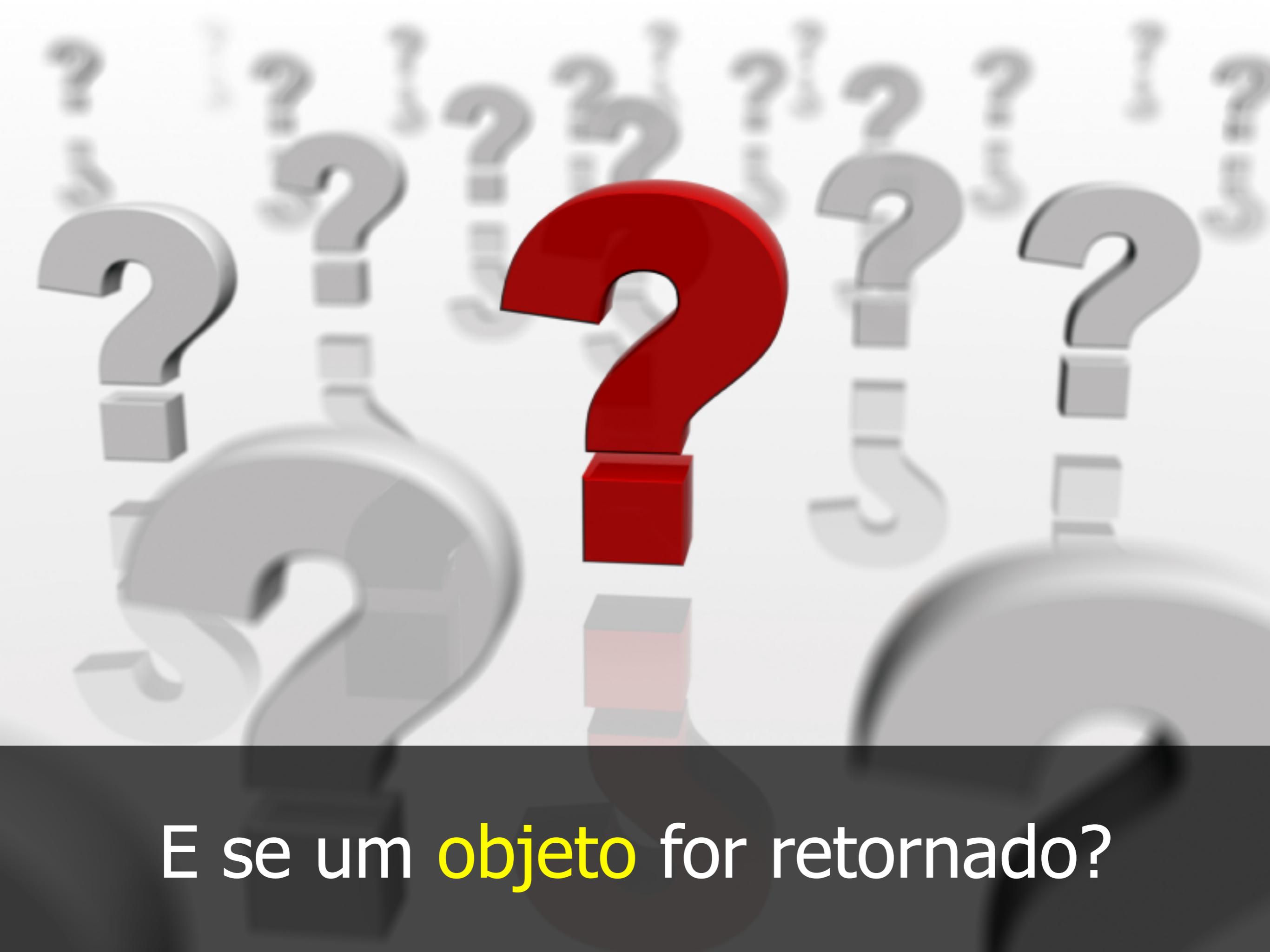
```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_7.js
45
rodrigobranas:javascriptmasterclass $
```

The screenshot shows a macOS desktop environment with a terminal window open. The title bar of the terminal window reads "arrow_function_8.js — javascriptmasterclass". The main pane of the terminal displays the output of a Node.js command:

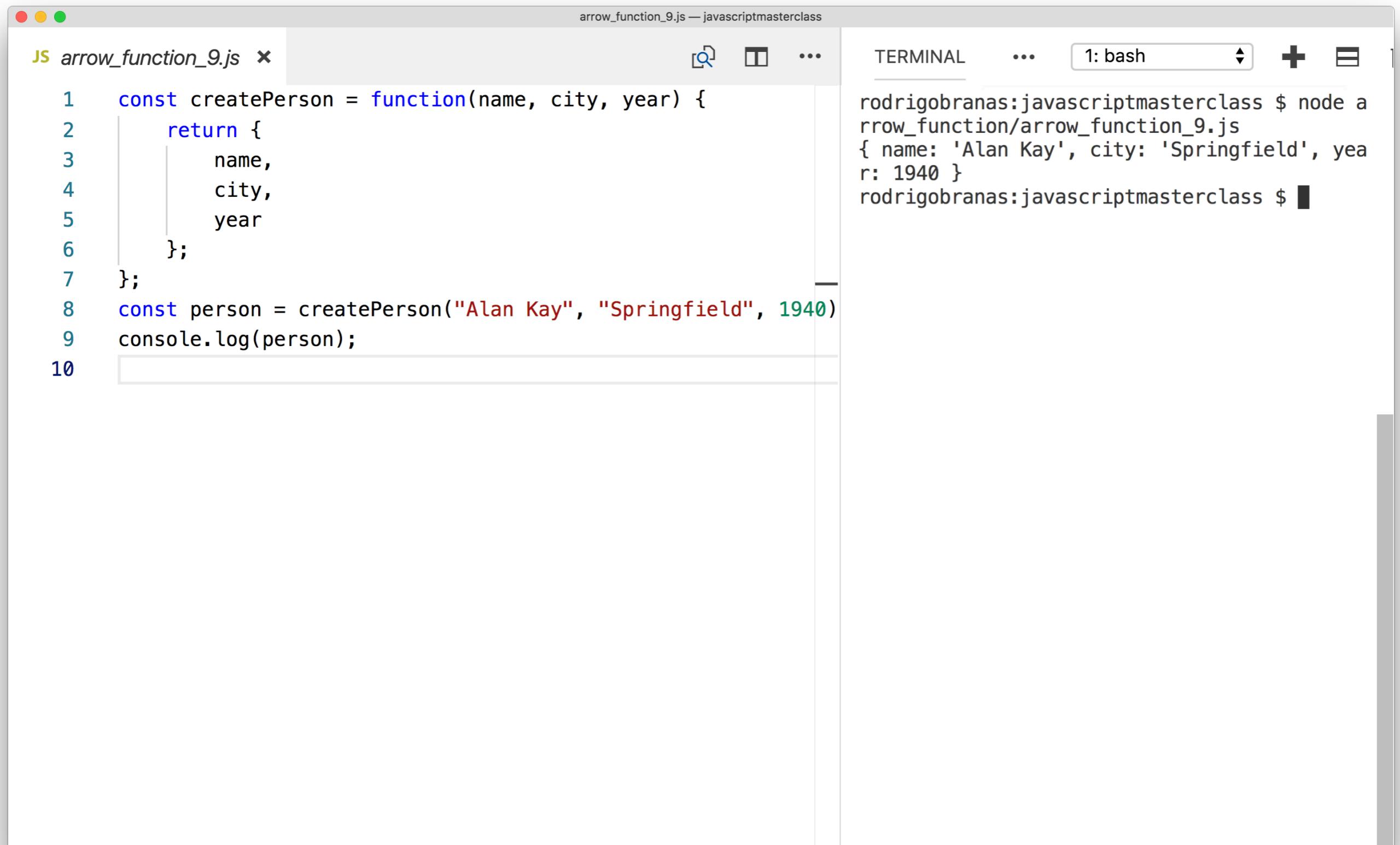
```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_8.js
0[object Object]function require(path) {
  try {
    exports.requireDepth += 1;
    return mod.require(path);
  } finally {
    exports.requireDepth -= 1;
  }
} [object Object]/Users/rodrigobranas/development/workspace/javascriptmasterclass/arrow_function/arrow_function_8.js/Users/rodrigobranas/development/workspace/javascriptmasterclass/arrow_function
rodrigobranas:javascriptmasterclass $
```

The code in the file "arrow_function_8.js" is as follows:

```
JS arrow_function_8.js x
1 const sum = () => {
2   let total = 0;
3   for(let argument in arguments) {
4     total += arguments[argument];
5   }
6   return total;
7 };
8 console.log(sum(1,2,3,4,5,6,7,8,9));
9
```



E se um objeto for retornado?



A screenshot of a Mac OS X terminal window titled "arrow_function_9.js — javascriptmasterclass". The window is divided into two main sections: a code editor on the left and a terminal on the right.

The code editor contains the following JavaScript code:

```
JS arrow_function_9.js ×
1 const createPerson = function(name, city, year) {
2     return {
3         name,
4         city,
5         year
6     };
7 }
8 const person = createPerson("Alan Kay", "Springfield", 1940)
9 console.log(person);
10
```

The terminal section shows the output of running the script with the command "node arrow_function_9.js". The output is:

```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_9.js
{ name: 'Alan Kay', city: 'Springfield', year: 1940 }
rodrigobranas:javascriptmasterclass $
```

The screenshot shows a macOS desktop environment with a terminal window open. The window title is "arrow_function_10.js — javascriptmasterclass". The main pane displays the contents of the file "arrow_function_10.js", which contains the following code:

```
JS arrow_function_10.js x
1 const createPerson = (name, city, year) => {name, city, year}
2 const person = createPerson("Alan Kay", "Springfield", 1940)
3 console.log(person);
4
```

The code defines an arrow function named `createPerson` that takes three parameters: `name`, `city`, and `year`. It returns an object with these three properties. A variable `person` is assigned the result of calling `createPerson` with the arguments "Alan Kay", "Springfield", and 1940. Finally, `console.log` is called on `person`. The file has four lines of content.

Below the code editor is a terminal window titled "1: bash". The terminal output shows the command `node arrow_function/arrow_function_10.js` being run, followed by the output `undefined` and the prompt "rodrigobranas:javascriptmasterclass \$".

arrow_function_11.js — javascriptmasterclass

JS arrow_function_11.js ×

1 const createPerson = (name, city, year) => ({name, city, year})
2 const person = createPerson("Alan Kay", "Springfield", 1940)
3 console.log(person);
4

TERMINAL ... 1: bash

```
rodrigobranas:javascriptmasterclass $ node arrow_function/arrow_function_11.js
{ name: 'Alan Kay', city: 'Springfield', year: 1940 }
rodrigobranas:javascriptmasterclass $
```