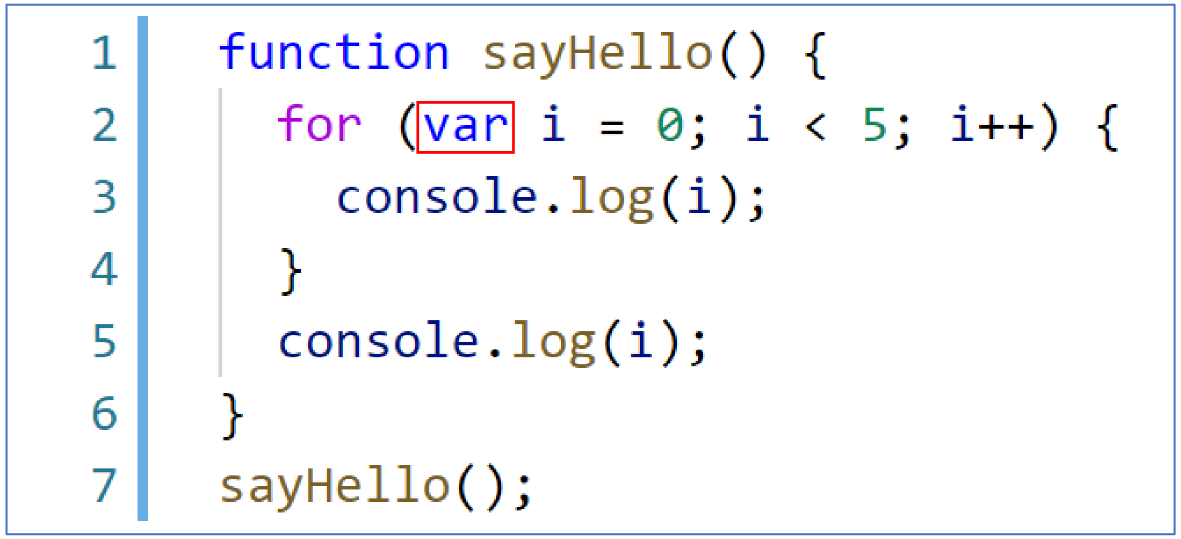
ES6 Refresher

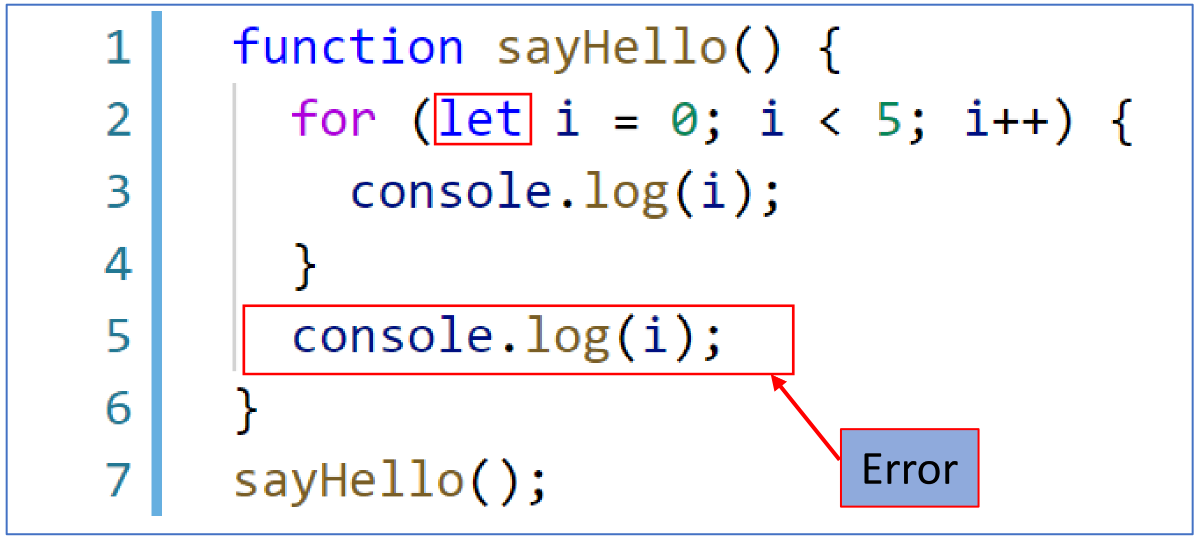
Content

1. Var vs Let vs Const

- var → function variable
- let → block variable
- const → block constant

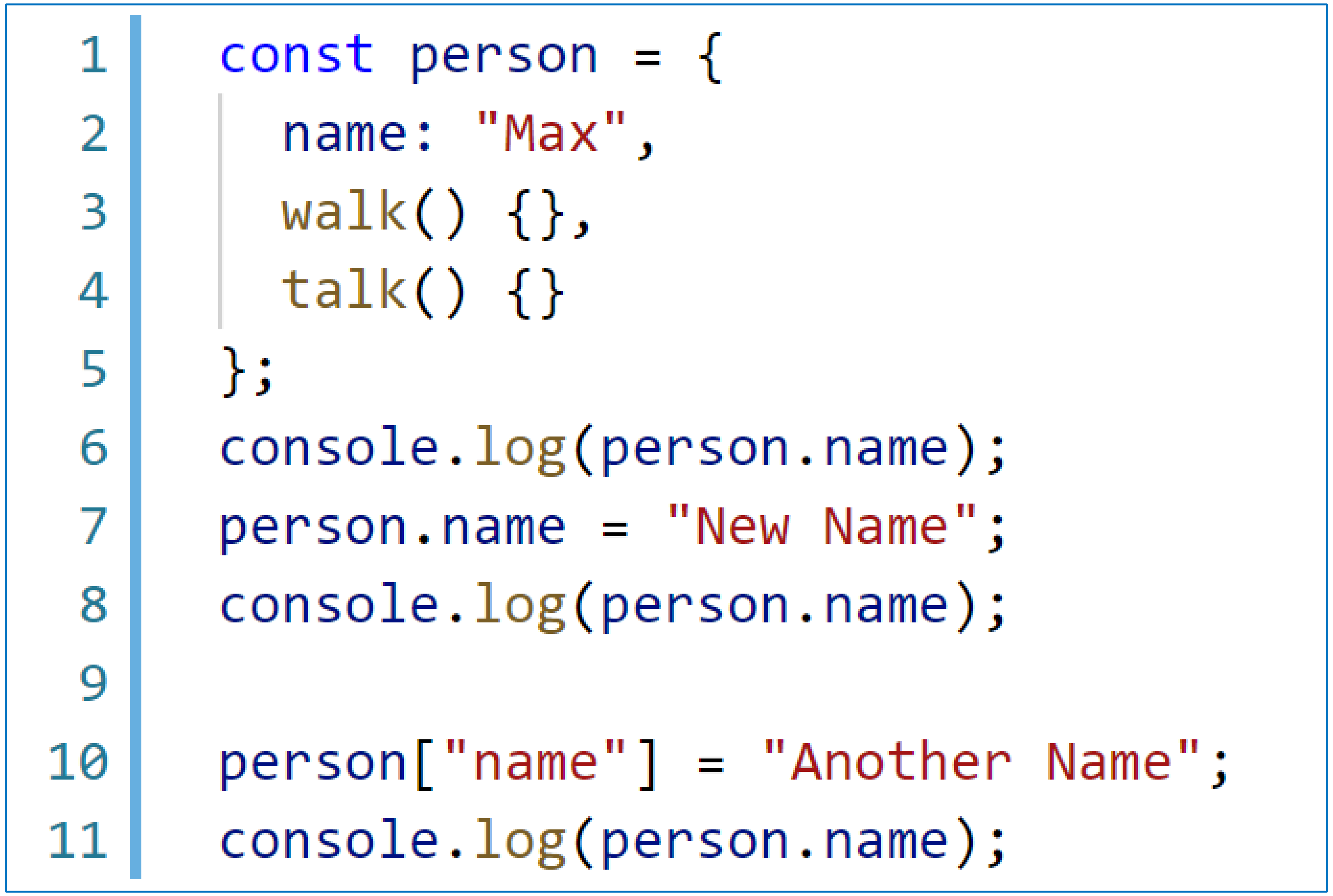


```
function sayHello() {
  for (var i = 0; i < 5; i++) {
    console.log(i);
  }
  console.log(i);
}
sayHello();
```



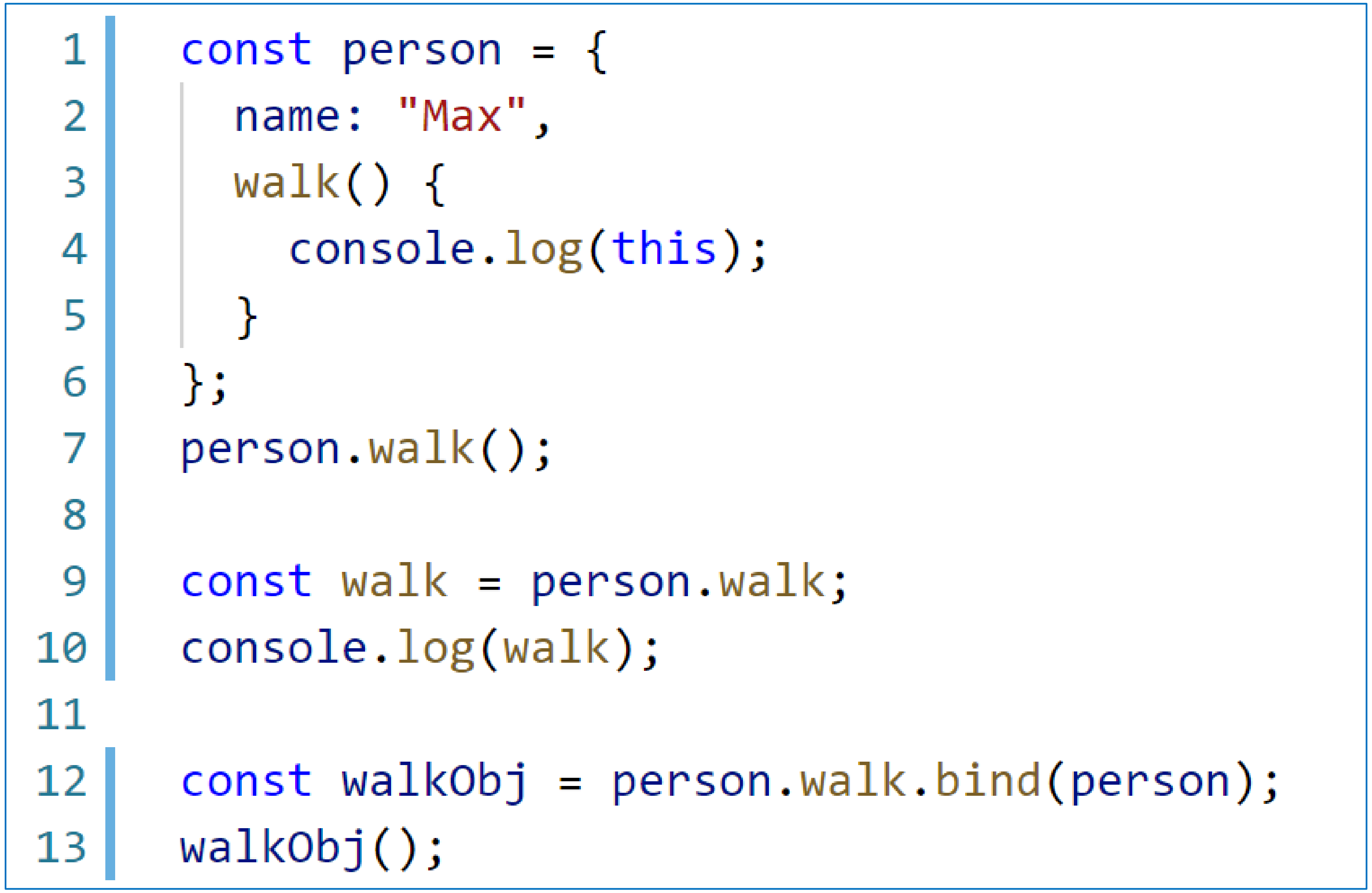
```
function sayHello() {
  for (let i = 0; i < 5; i++) {
    console.log(i);
  }
  console.log(i);
}
sayHello();
```

2. Objects



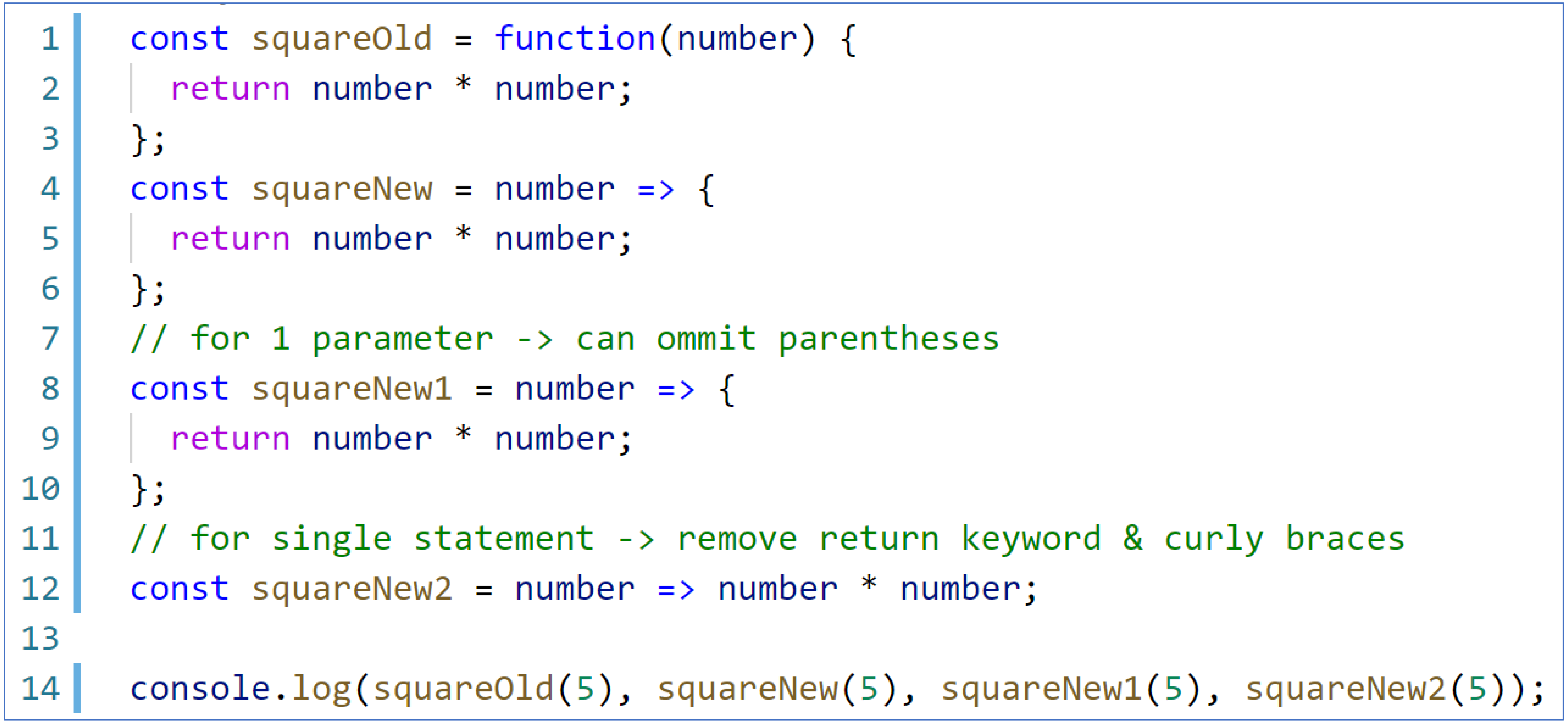
```javascript
const person = {
  name: "Max",
  walk() {},
  talk() {}
};
console.log(person.name);
person.name = "New Name";
console.log(person.name);

person["name"] = "Another Name";
console.log(person.name);
```

3. The this Keyword & Binding this

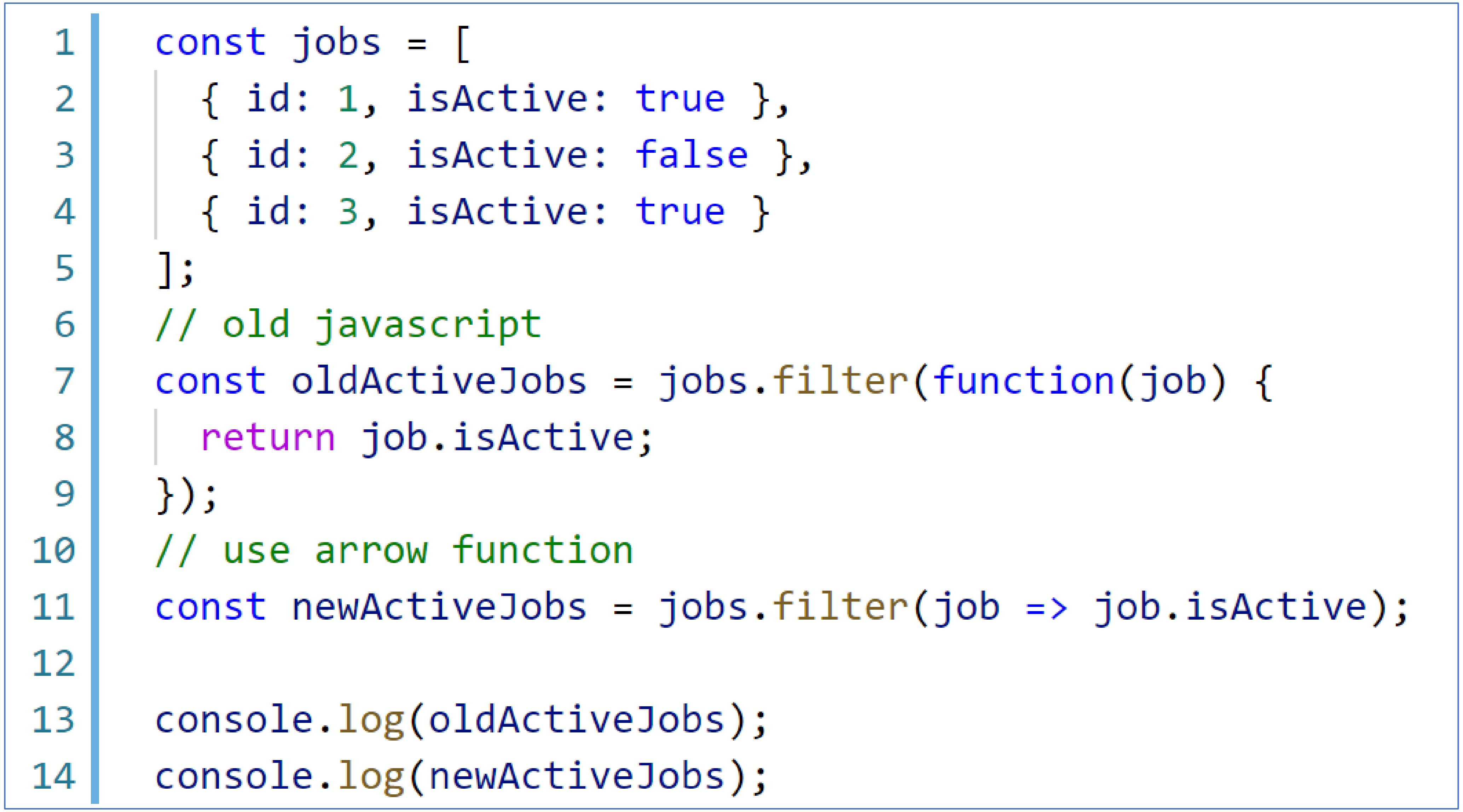


```javascript
const person = {
  name: "Max",
  walk() {
    console.log(this);
  }
};
person.walk();

const walk = person.walk;
console.log(walk);

const walkObj = person.walk.bind(person);
walkObj();
```

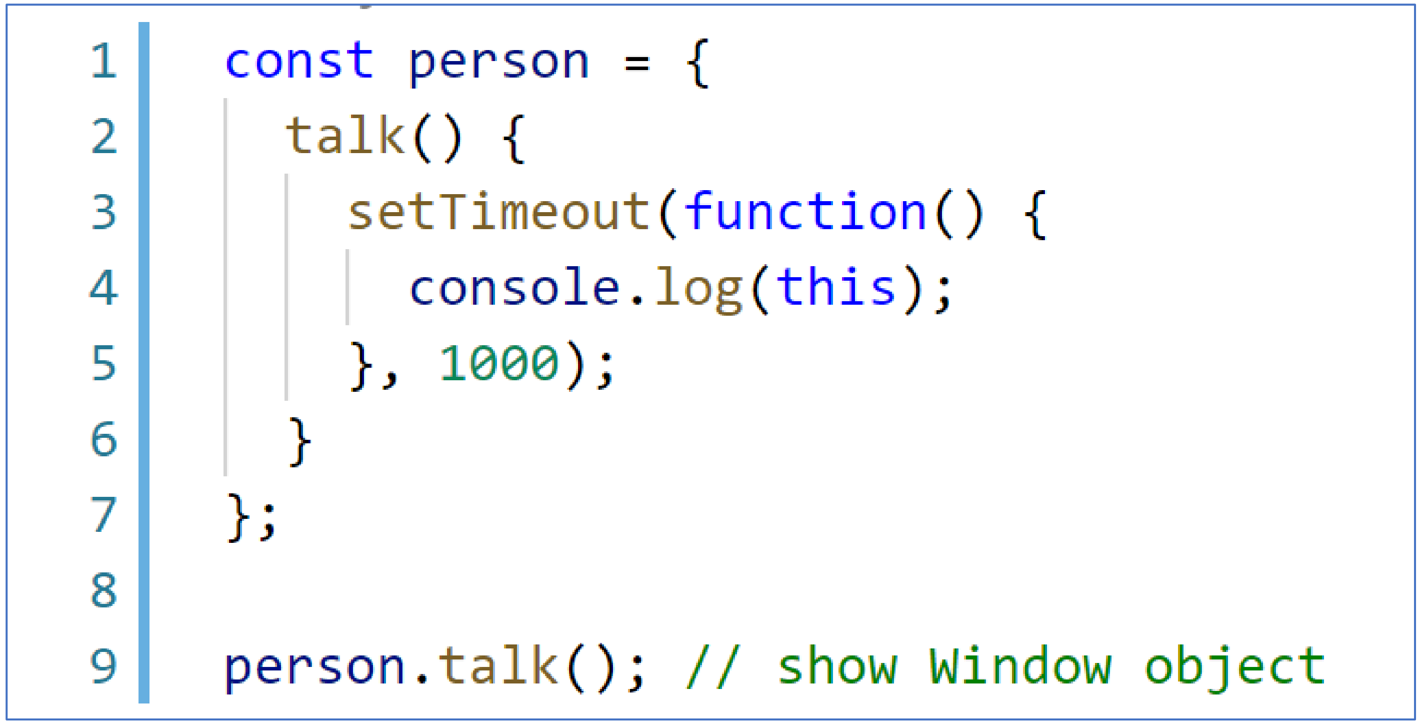
4. Arrow Functions & this



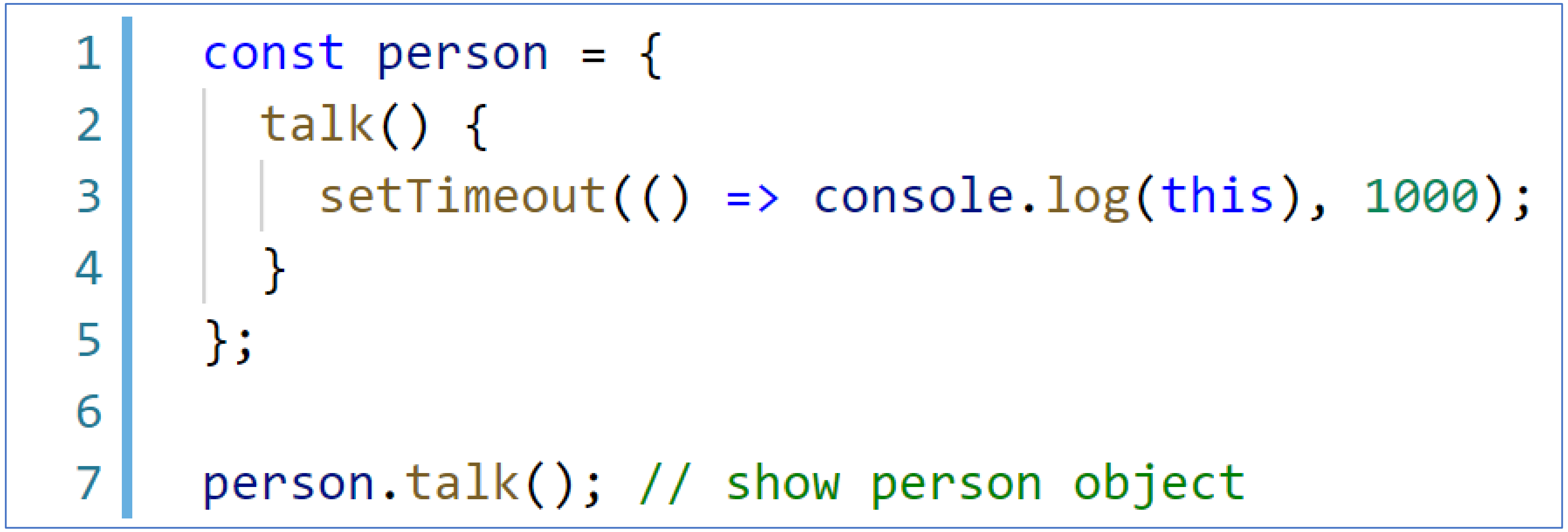
```
const squareOld = function(number) {
  return number * number;
};
const squareNew = number => {
  return number * number;
};
// for 1 parameter -> can ommit parentheses
const squareNew1 = number => {
  return number * number;
};
// for single statement -> remove return keyword & curly braces
const squareNew2 = number => number * number;

console.log(squareOld(5), squareNew(5), squareNew1(5), squareNew2(5));
```



```javascript
const jobs = [
  { id: 1, isActive: true },
  { id: 2, isActive: false },
  { id: 3, isActive: true }
];
// old javascript
const oldActiveJobs = jobs.filter(function(job) {
  return job.isActive;
});
// use arrow function
const newActiveJobs = jobs.filter(job => job.isActive);

console.log(oldActiveJobs);
console.log(newActiveJobs);
```

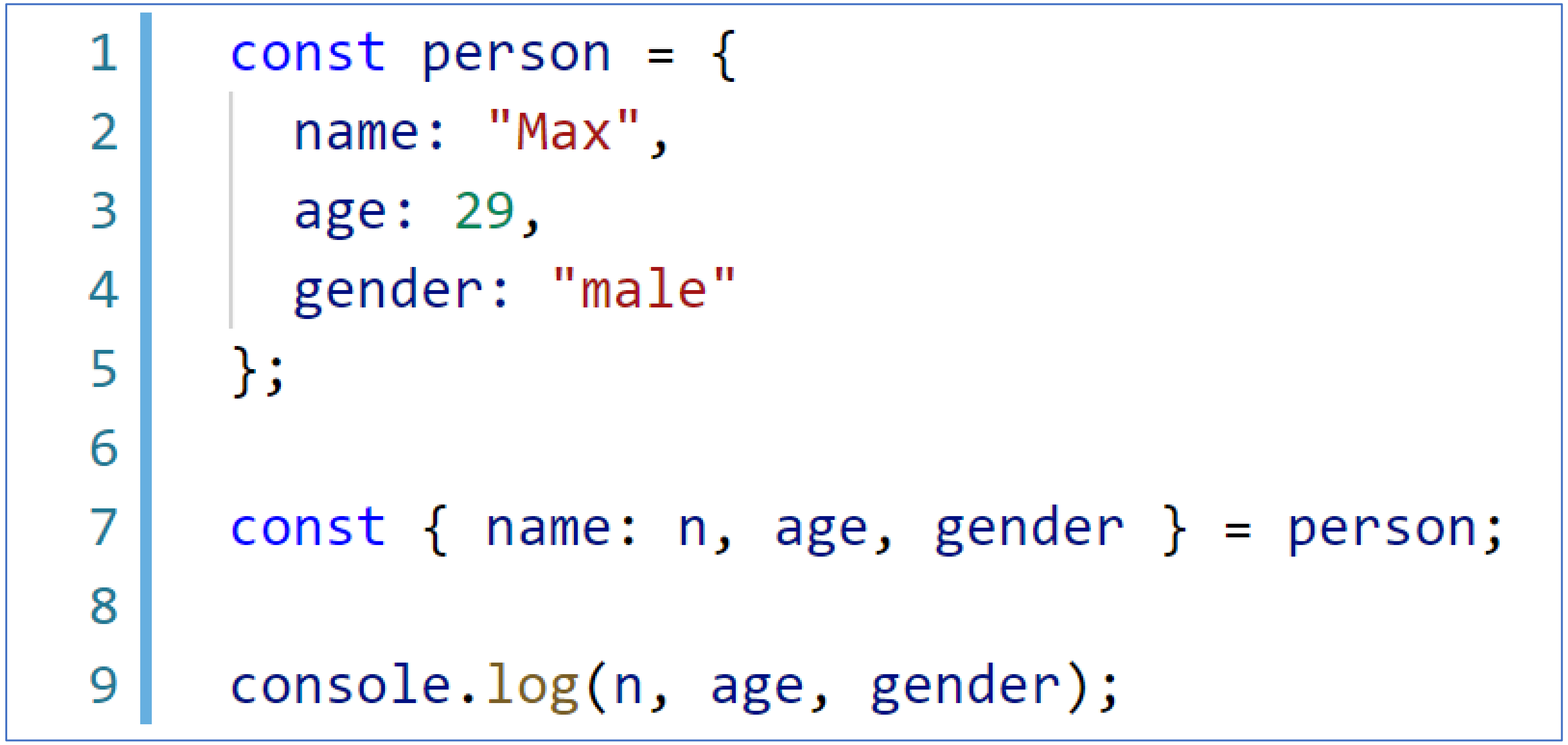


```javascript
const person = {
  talk() {
    setTimeout(function() {
      console.log(this);
    }, 1000);
  }
};

person.talk(); // show Window object
```

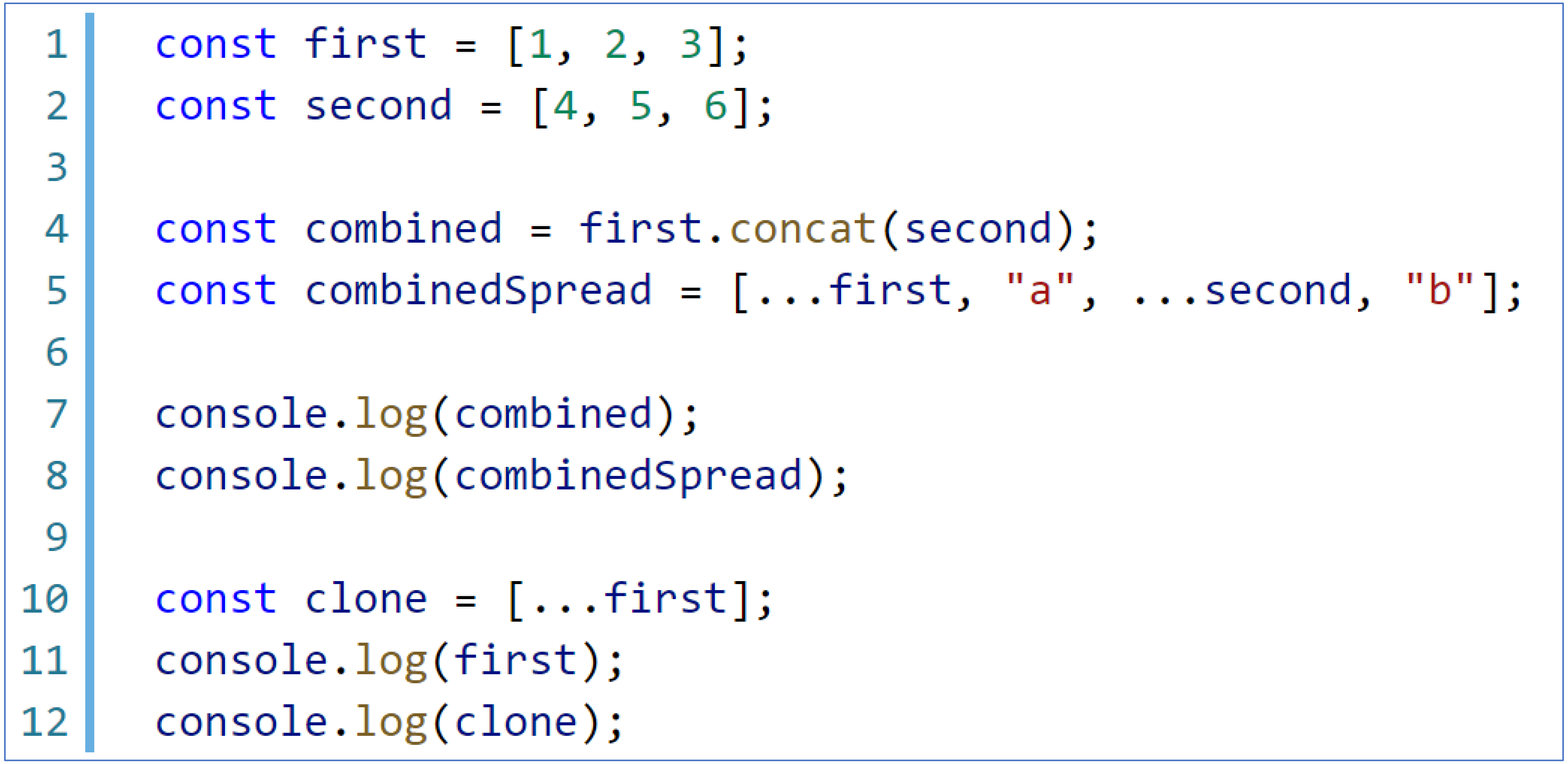


```javascript
const person = {
  talk() {
    setTimeout(() => console.log(this), 1000);
  }
};

person.talk(); // show person object
```

5. Object Destructuring


```javascript
const person = {
  name: "Max",
  age: 29,
  gender: "male"
};

const { name: n, age, gender } = person;

console.log(n, age, gender);
```

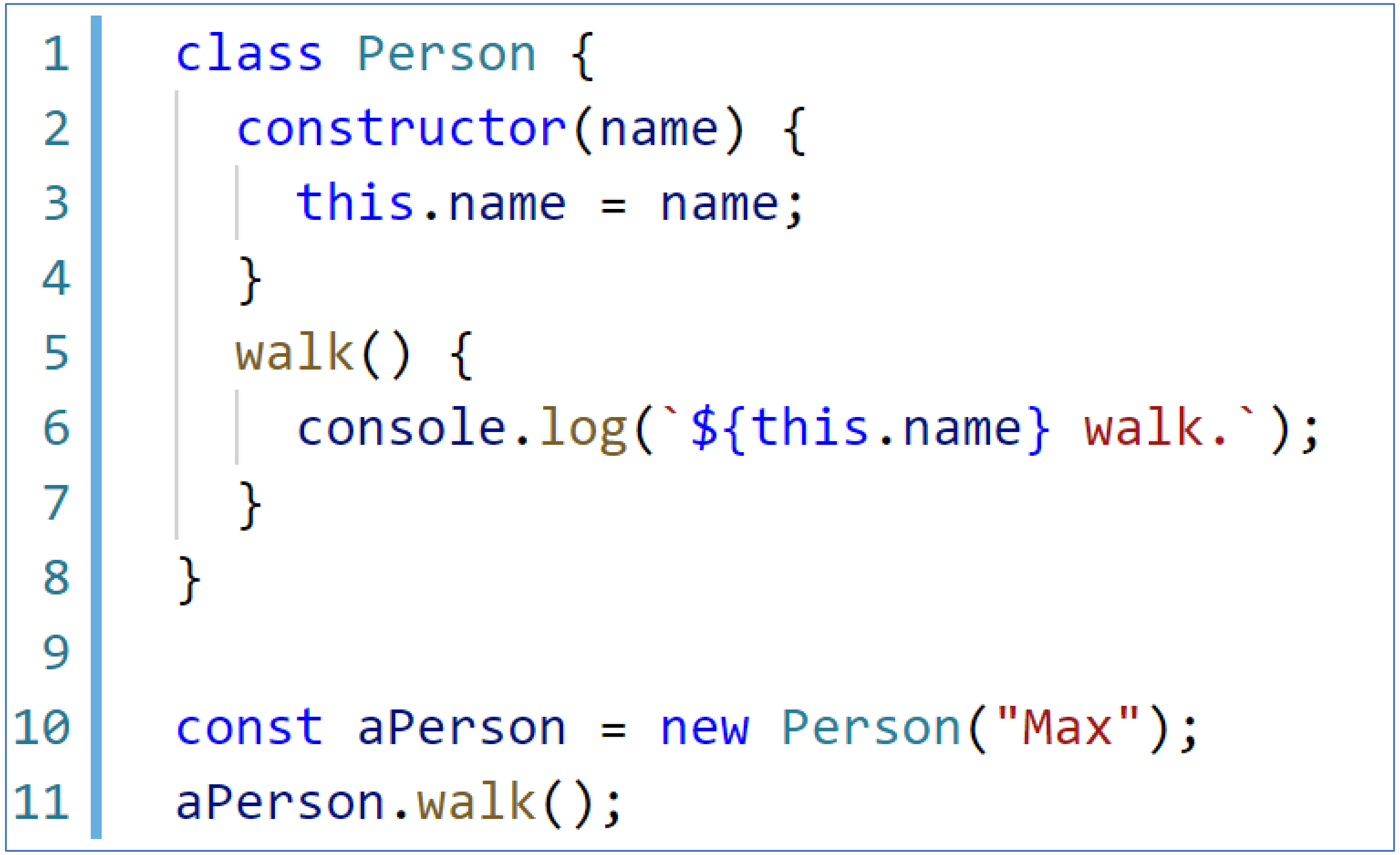
6. Spread Operator



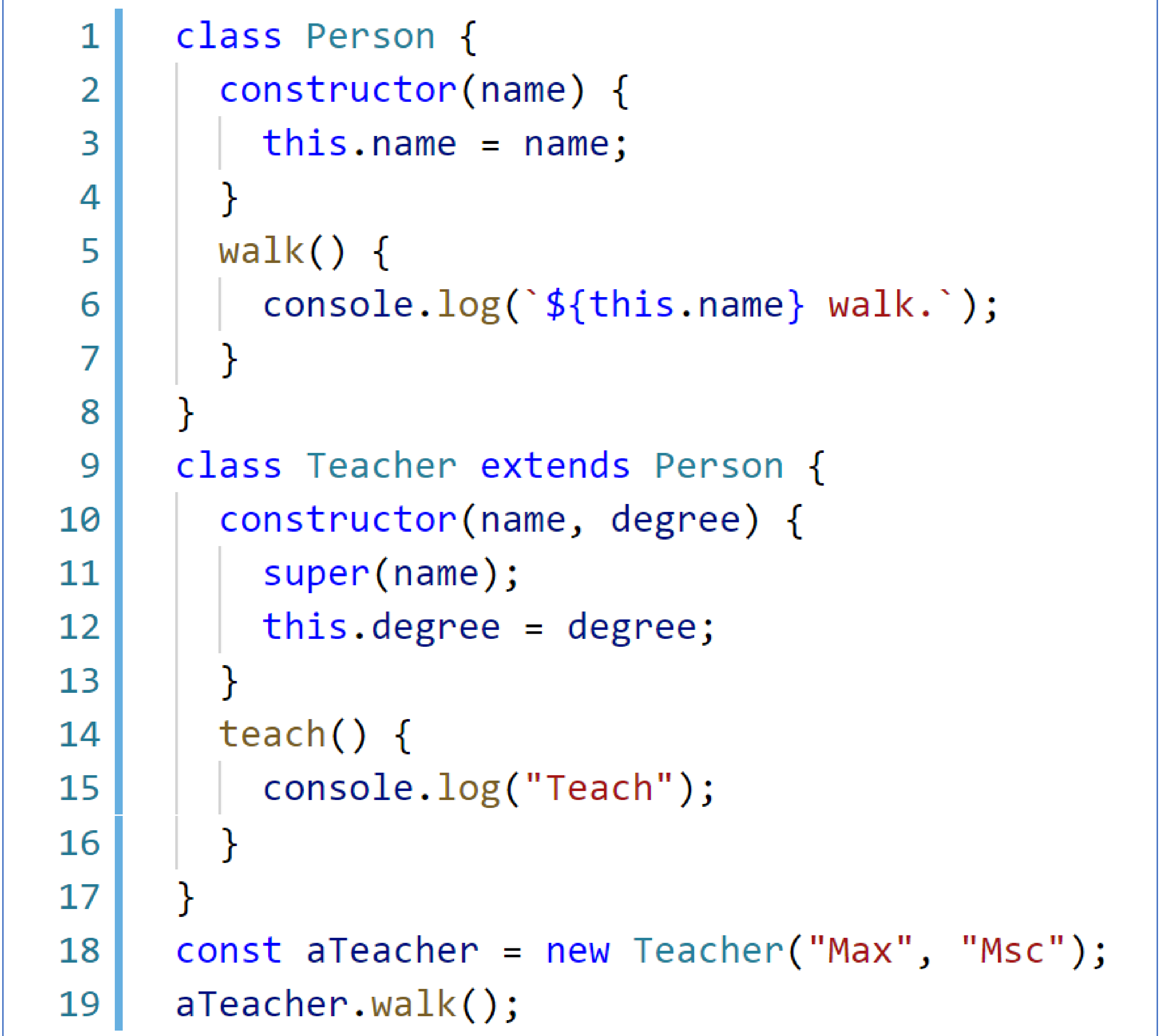
```javascript
const first = [1, 2, 3];
const second = [4, 5, 6];

const combined = first.concat(second);
const combinedSpread = [...first, "a", ...second, "b"];

console.log(combined);
console.log(combinedSpread);

const clone = [...first];
console.log(first);
console.log(clone);
```

```javascript
const first = { name: "Max" };
const second = { job: "Instructor" };

const combined = { ...first, ...second, age: 29 };
console.log(combined);
```

7. Classes & Inheritance



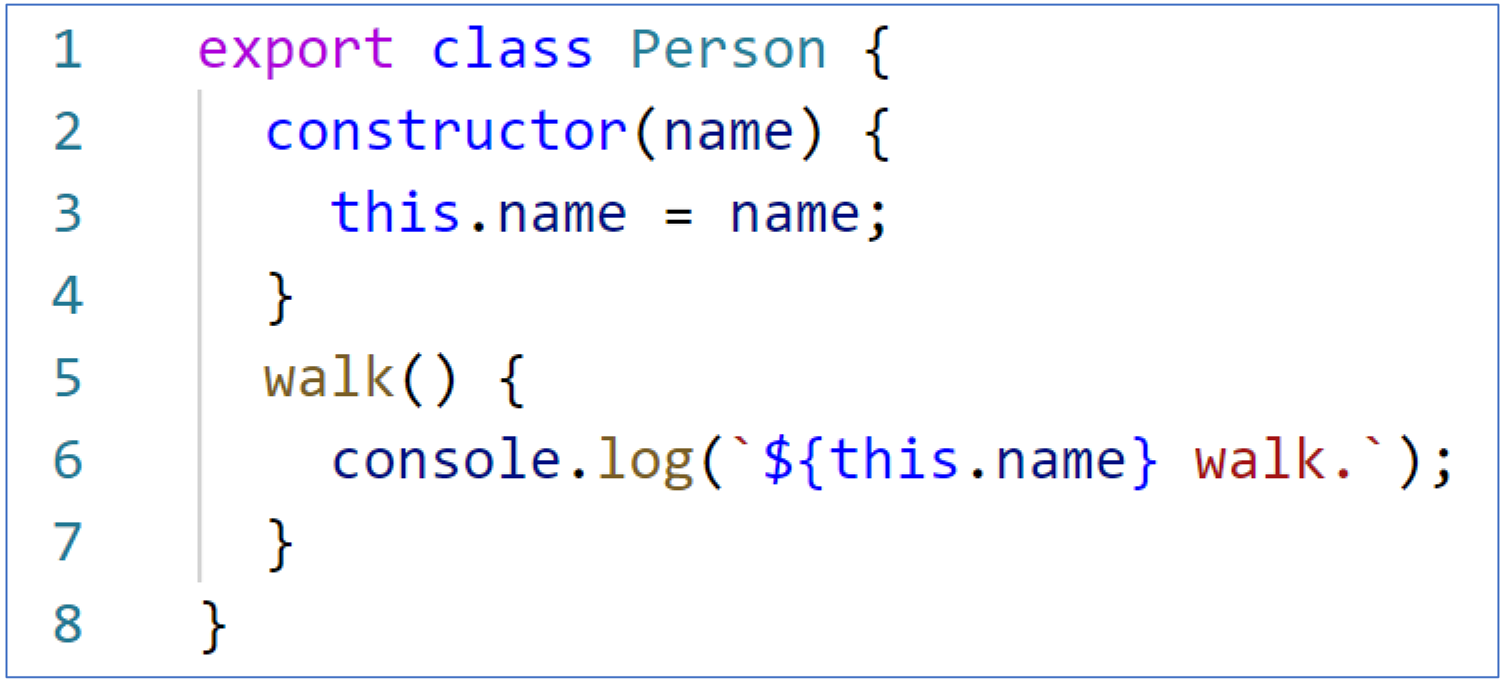
```
class Person {
  constructor(name) {
    this.name = name;
  }
  walk() {
    console.log(`${this.name} walk.`);
  }
}

const aPerson = new Person("Max");
aPerson.walk();
```



```javascript
class Person {
  constructor(name) {
    this.name = name;
  }
  walk() {
    console.log(`${this.name} walk.`);
  }
}
class Teacher extends Person {
  constructor(name, degree) {
    super(name);
    this.degree = degree;
  }
  teach() {
    console.log("Teach");
  }
}
const aTeacher = new Teacher("Max", "Msc");
aTeacher.walk();
```

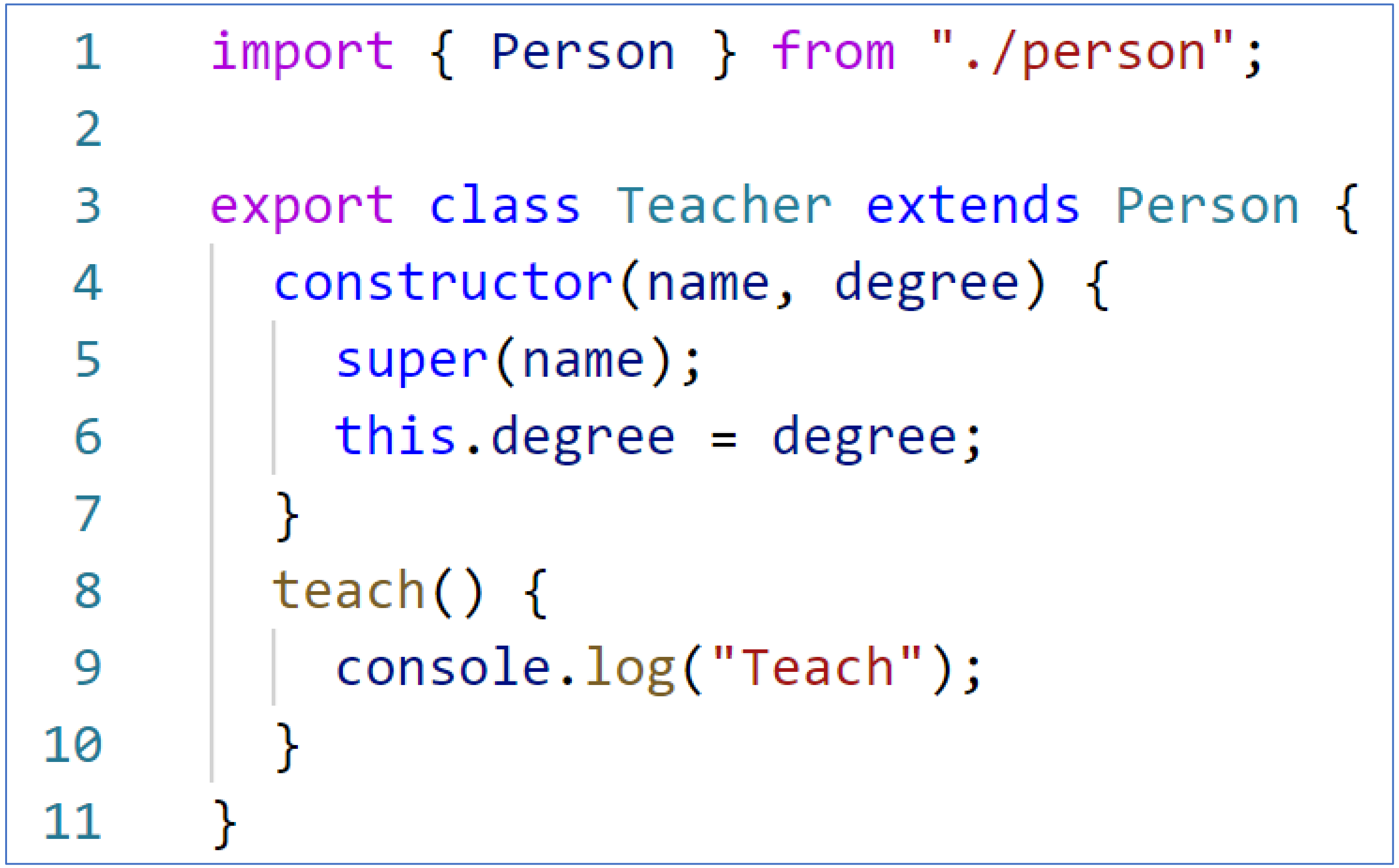
8. Modules

- Create a new file named “person.js”



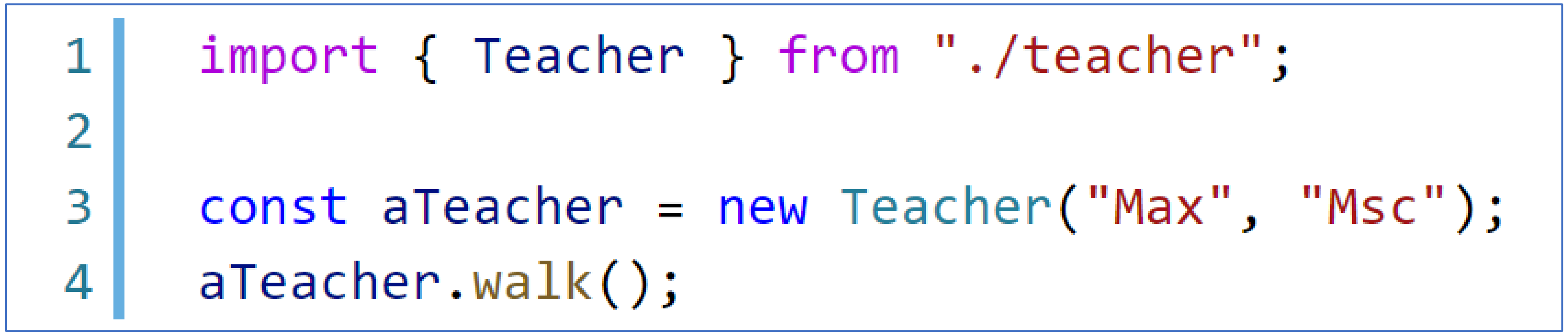
```javascript
export class Person {
  constructor(name) {
    this.name = name;
  }
  walk() {
    console.log(`${this.name} walk.`);
  }
}
```

- Create a new file named “teacher.js”



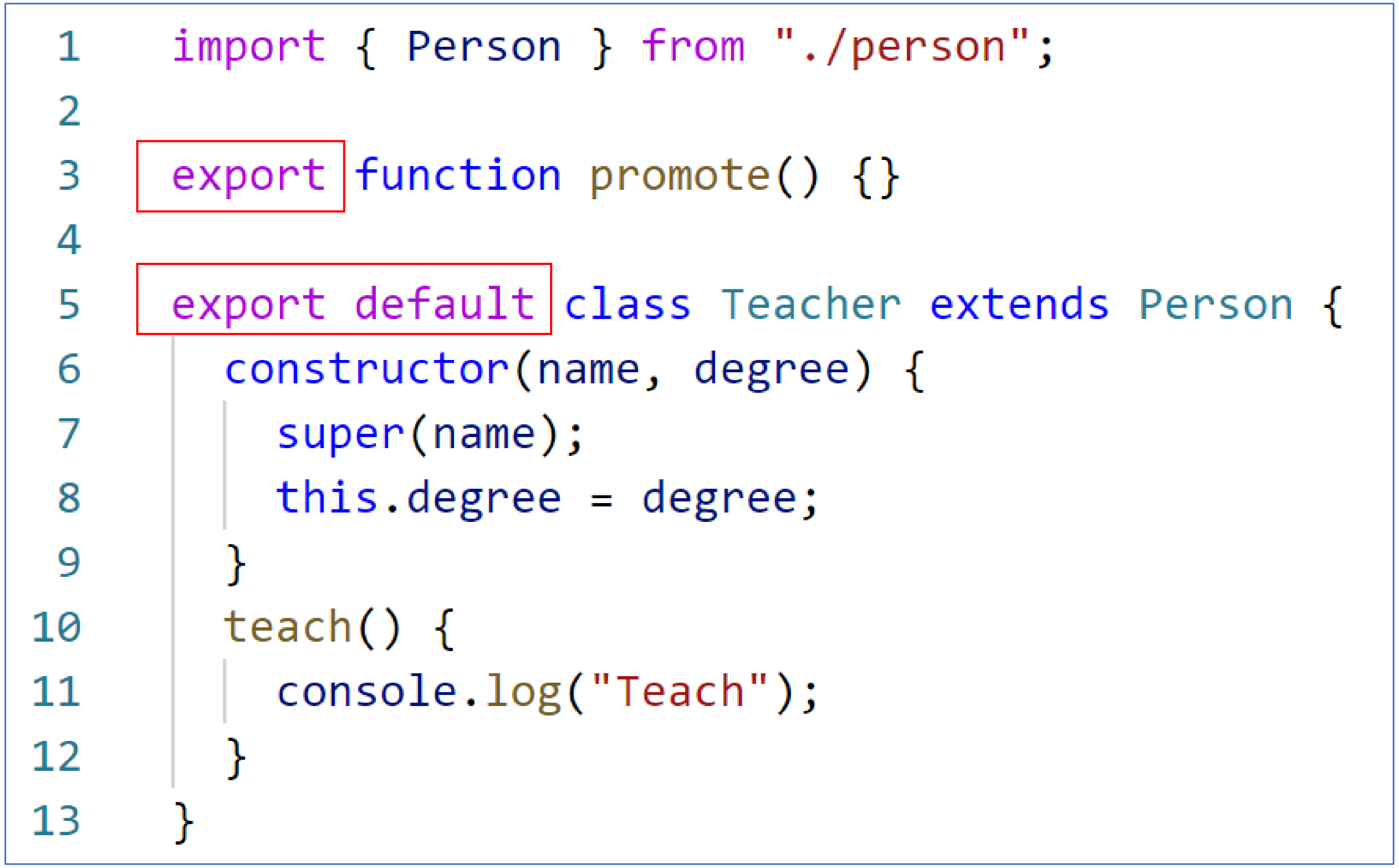
```
import { Person } from "./person";

export class Teacher extends Person {
  constructor(name, degree) {
    super(name);
    this.degree = degree;
  }
  teach() {
    console.log("Teach");
  }
}
```

- Code in “index.js”



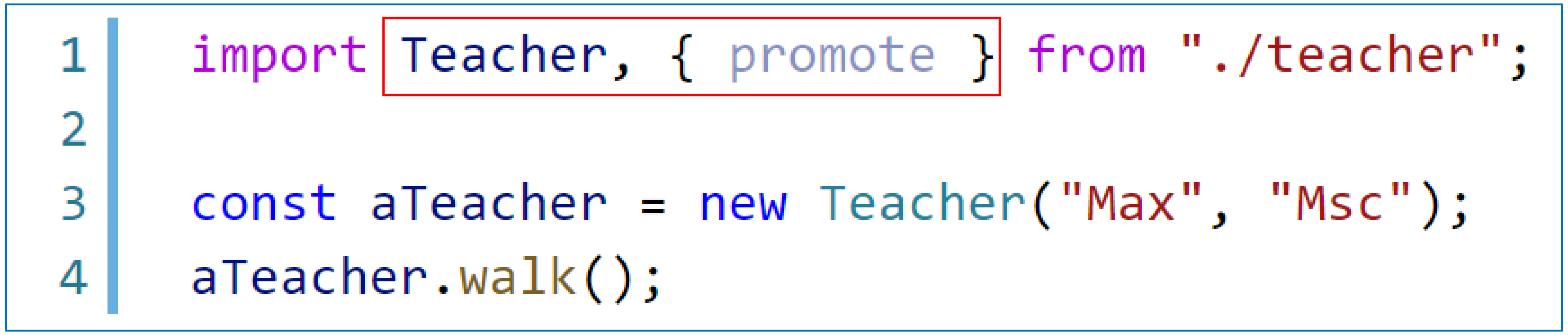
```
import { Teacher } from "./teacher";

const aTeacher = new Teacher("Max", "Msc");
aTeacher.walk();
```

9. Named and Default Exports



```javascript
import { Person } from "./person";

export function promote() {}

export default class Teacher extends Person {
  constructor(name, degree) {
    super(name);
    this.degree = degree;
  }
  teach() {
    console.log("Teach");
  }
}
```

- Code in “index.js”



```
import Teacher, { promote } from "./teacher";

const aTeacher = new Teacher("Max", "Msc");
aTeacher.walk();
```

10. Common useful methods

- Array.forEach()
- Array.map()
- Array.filter()
- Array.indexOf()
- Array.lastIndexOf()
- Array.find()
- Array.findIndex()
- Array.push()
- Array.pop()
- Array.unshift()
- Array.shift()
- Array.reverse()
- Array.sort()
- Array.slice()
- Array.splice()
- JSON.parse()
- JSON.stringify()

Reference

- https://www.tutorialspoint.com/es6/index.htm
- https://www.w3schools.com/jsref/jsref_obj_array.asp