Javascript Module Exercises

1. Determine what this Javascript code will print out (without running it):

   Answer : Undefined , 8 , 8 , 9, 10, 1

2. Define Global Scope and Local Scope in Javascript

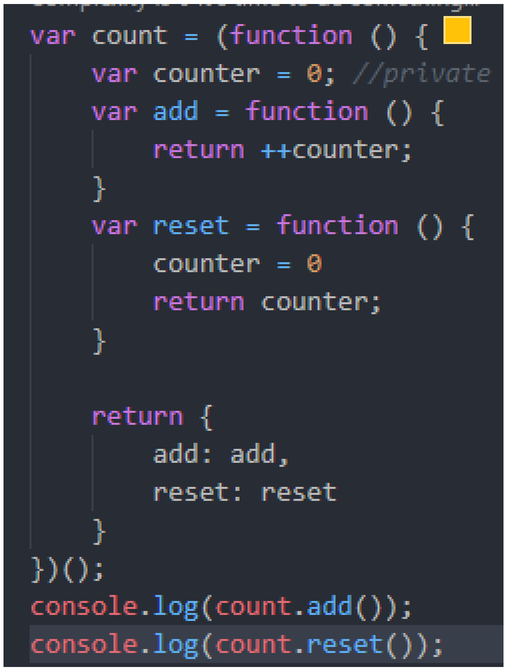
- **Local scope** is a scope within a function. If we have a function and declare values inside , those are locally scoped. They can only be accessed from within the function
- **Global scope -** any variable declared in windows level . Global variables are variables that are defined outside of functions. These variables have global scope, so they can be used by any function without passing them to the function as parameters.

3. Consider the following structure of Javascript code:

a) No
b) Yes
c) No
d) Yes
e) Yes

4. What will be printed by the following (answer without running it)?
   Answer : **81, 25,**
5. What will be printed by the following (answer without running it)?
   Answer: 10

6.



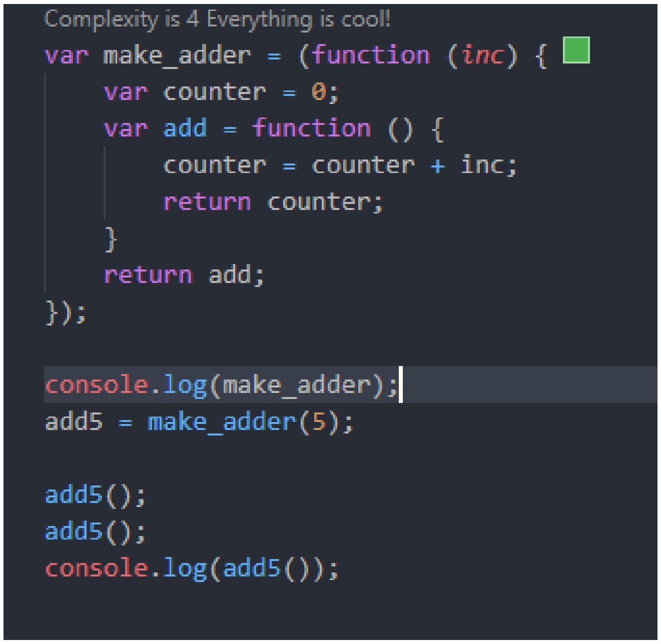
```
var count = (function () { 🟨
    var counter = 0; //private
    var add = function () {
        return ++counter;
    }
    var reset = function () {
        counter = 0
        return counter;
    }

    return {
        add: add,
        reset: reset
    }
})();
console.log(count.add());
console.log(count.reset());
```

7. Free Variable: counter

Free Variables are variables that are neither arguments nor local variables

8.



Complexity is 4 Everything is cool!

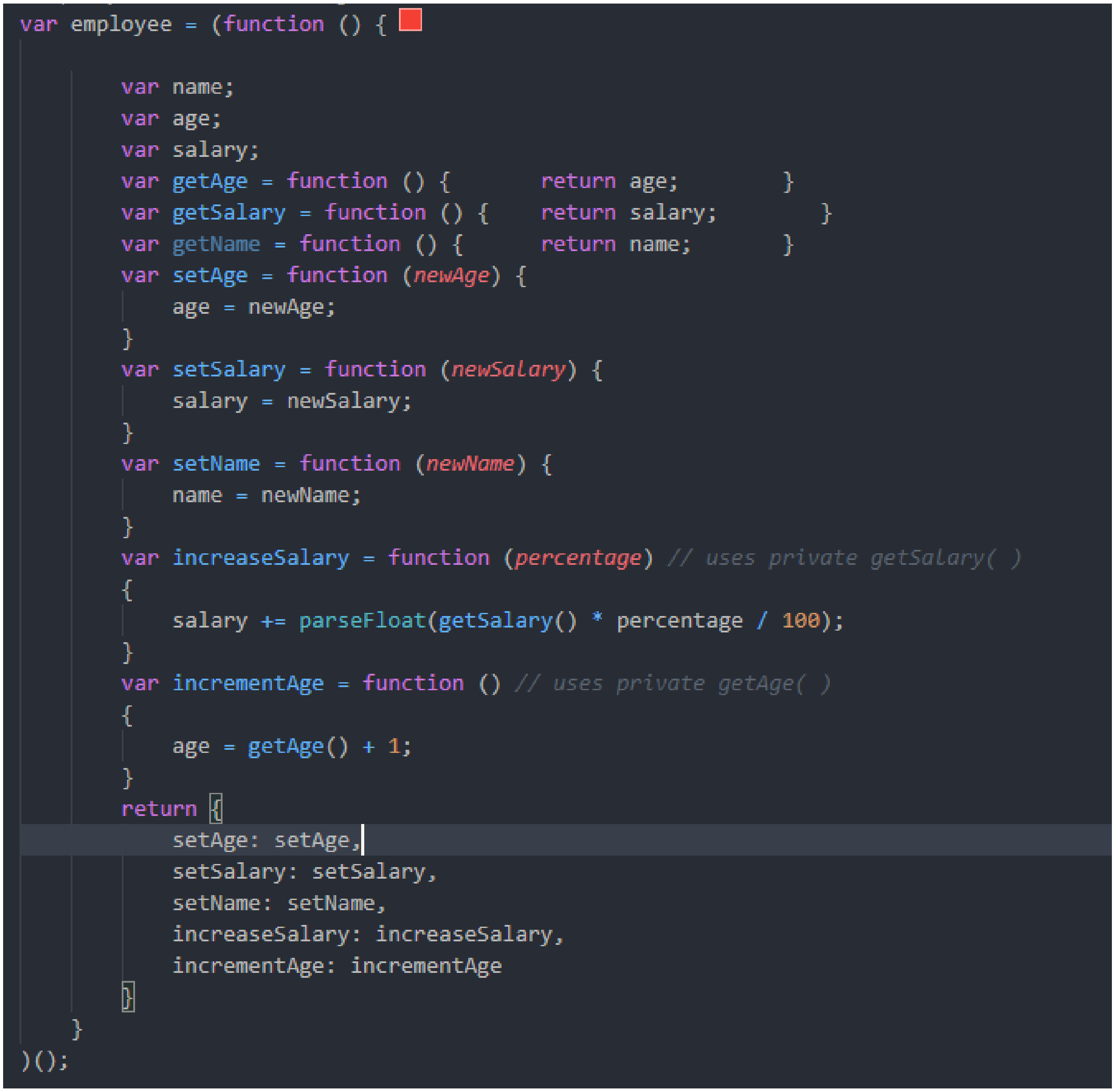
```
var make_adder = (function (inc) { 🟩
    var counter = 0;
    var add = function () {
        counter = counter + inc;
        return counter;
    }
    return add;
});

console.log(make_adder);
add5 = make_adder(5);

add5();
add5();
console.log(add5());
```

9. wrapping the code in module pattern design specifically revealing

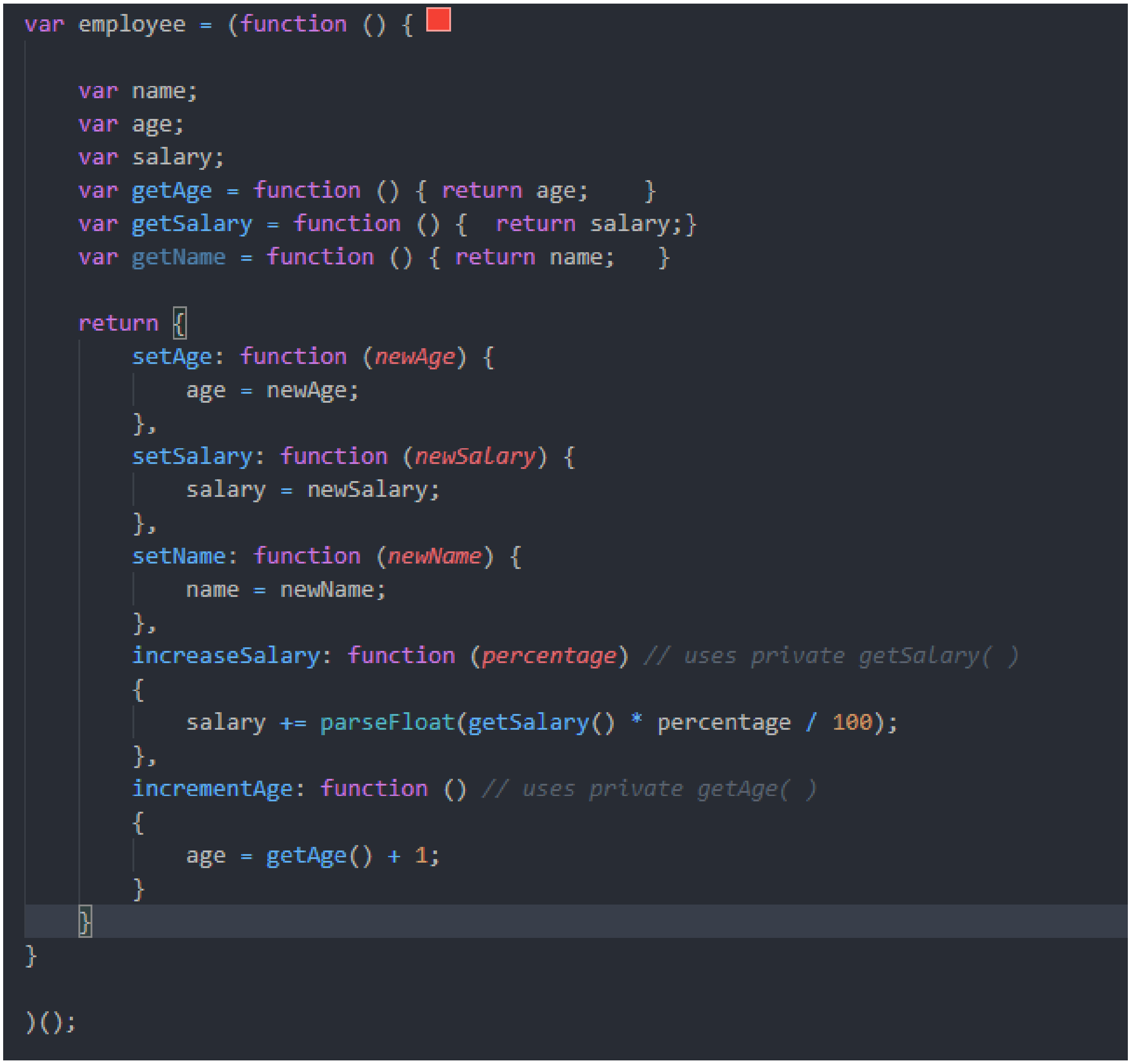
10.



```
var employee = (function () {

        var name;
        var age;
        var salary;
        var getAge = function () {      return age;        }
        var getSalary = function () {    return salary;        }
        var getName = function () {      return name;       }
        var setAge = function (newAge) {
            age = newAge;
        }
        var setSalary = function (newSalary) {
            salary = newSalary;
        }
        var setName = function (newName) {
            name = newName;
        }
        var increaseSalary = function (percentage) // uses private getSalary( )
        {
            salary += parseFloat(getSalary() * percentage / 100);
        }
        var incrementAge = function () // uses private getAge( )
        {
            age = getAge() + 1;
        }
        return {
            setAge: setAge,
            setSalary: setSalary,
            setName: setName,
            increaseSalary: increaseSalary,
            incrementAge: incrementAge
        }
    }
)();
```

11.



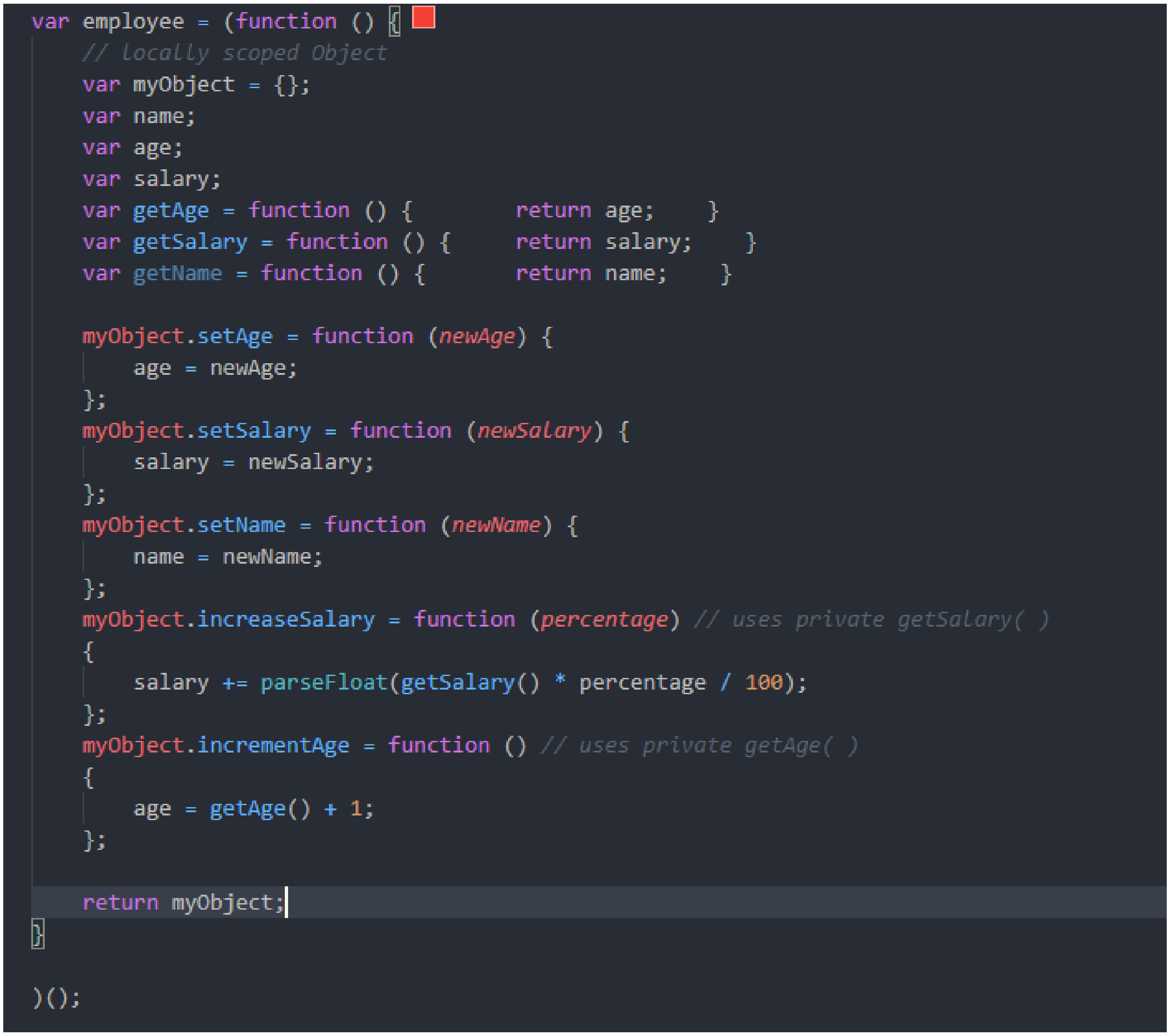
```
var employee = (function () {

    var name;
    var age;
    var salary;
    var getAge = function () { return age;    }
    var getSalary = function () {  return salary;}
    var getName = function () { return name;   }

    return {
        setAge: function (newAge) {
            age = newAge;
        },
        setSalary: function (newSalary) {
            salary = newSalary;
        },
        setName: function (newName) {
            name = newName;
        },
        increaseSalary: function (percentage) // uses private getSalary( )
        {
            salary += parseFloat(getSalary() * percentage / 100);
        },
        incrementAge: function () // uses private getAge( )
        {
            age = getAge() + 1;
        }
    }
}

)();
```

12.



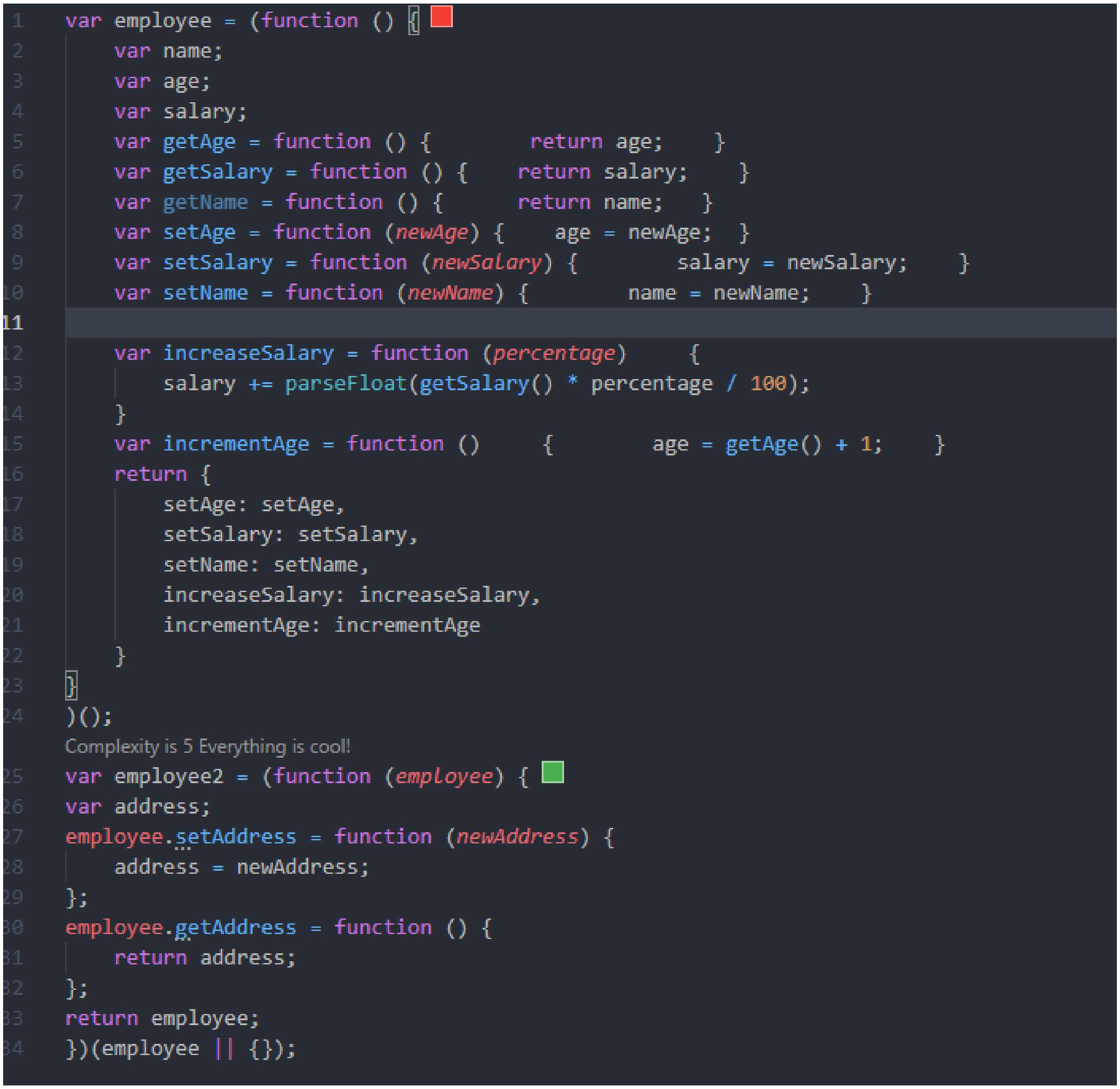
```
var employee = (function () {
    // locally scoped Object
    var myObject = {};
    var name;
    var age;
    var salary;
    var getAge = function () {        return age;    }
    var getSalary = function () {     return salary;    }
    var getName = function () {       return name;    }

    myObject.setAge = function (newAge) {
        age = newAge;
    };
    myObject.setSalary = function (newSalary) {
        salary = newSalary;
    };
    myObject.setName = function (newName) {
        name = newName;
    };
    myObject.increaseSalary = function (percentage) // uses private getSalary( )
    {
        salary += parseFloat(getSalary() * percentage / 100);
    };
    myObject.incrementAge = function () // uses private getAge( )
    {
        age = getAge() + 1;
    };

    return myObject;
}

)();
```

13.



```
var employee = (function () { 🟥
    var name;
    var age;
    var salary;
    var getAge = function () {        return age;    }
    var getSalary = function () {    return salary;    }
    var getName = function () {      return name;   }
    var setAge = function (newAge) {    age = newAge;  }
    var setSalary = function (newSalary) {        salary = newSalary;    }
    var setName = function (newName) {        name = newName;    }

    var increaseSalary = function (percentage)     {
        salary += parseFloat(getSalary() * percentage / 100);
    }
    var incrementAge = function ()     {        age = getAge() + 1;    }
    return {
        setAge: setAge,
        setSalary: setSalary,
        setName: setName,
        increaseSalary: increaseSalary,
        incrementAge: incrementAge
    }
}
)();
Complexity is 5 Everything is cool!
var employee2 = (function (employee) { 🟩
var address;
employee.setAddress = function (newAddress) {
    address = newAddress;
};
employee.getAddress = function () {
    return address;
};
return employee;
})(employee || {});
```

14. Error : Hattori

15. Success: Hattori

16. Success
error