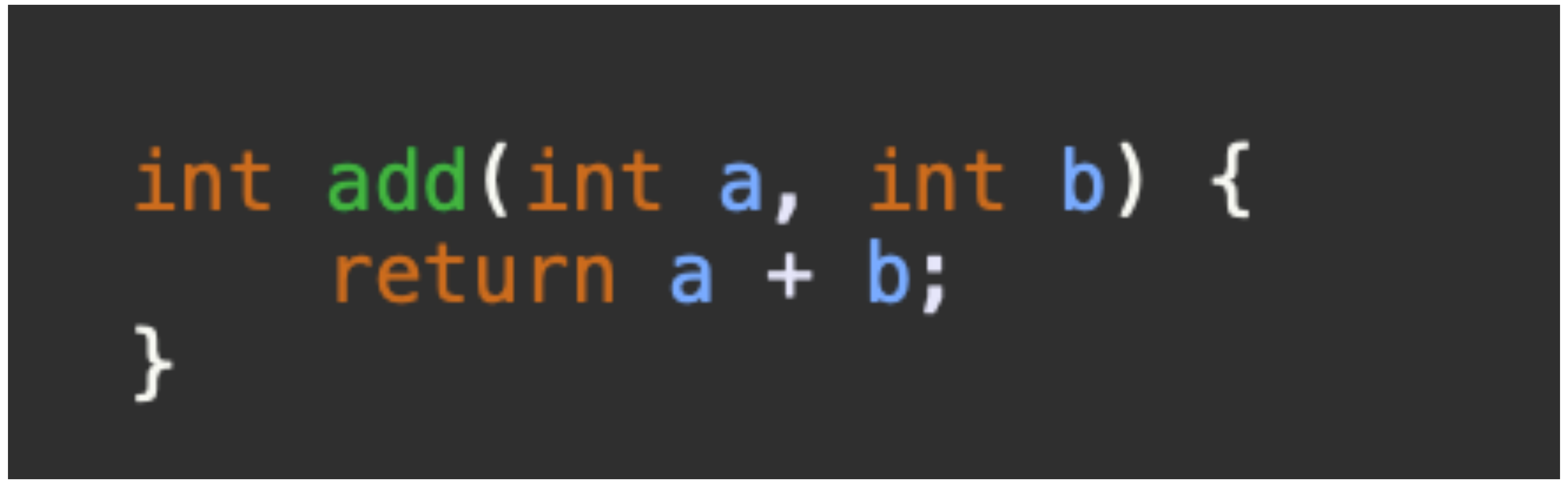
Java Course

Methods

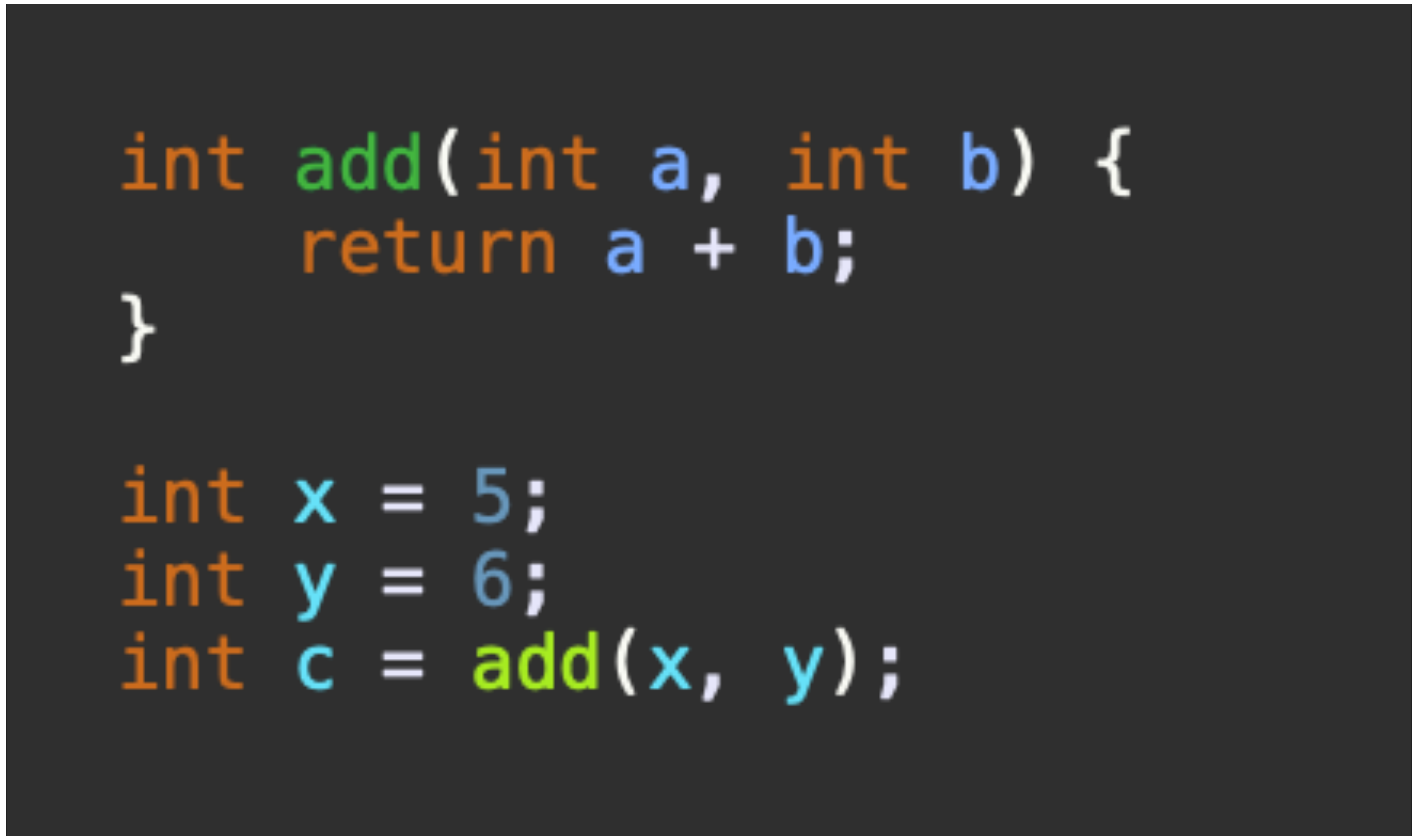
- A method is a block of code that performs a specific task
- Decomposition
- DRY principle
- Key elements of a method:
  - method definition
  - method call

Method definition

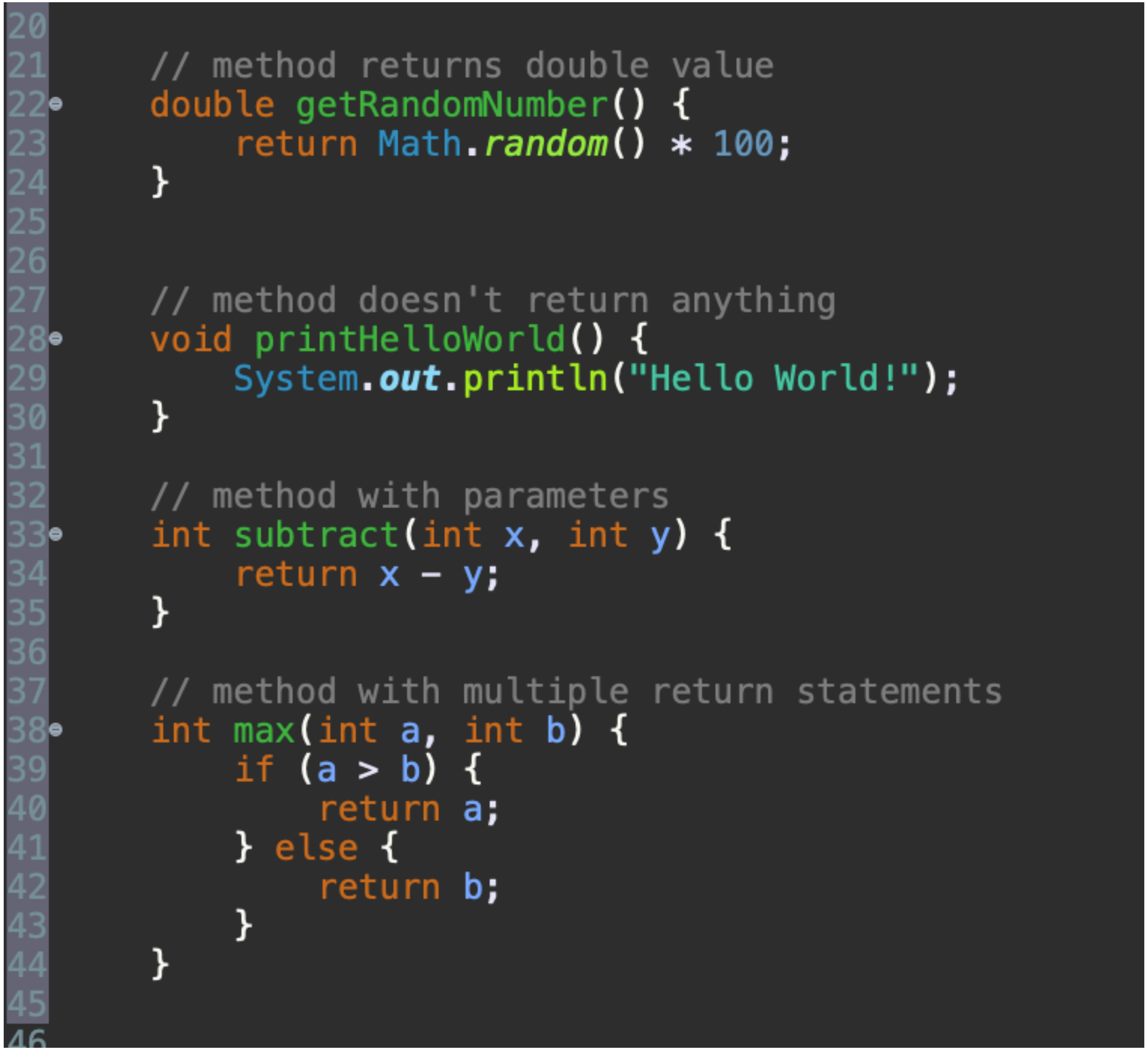
-
  ```
  return_type method_name(parameters) {
  …
  }
  ```
- `return_type` is type of data that is returned from method or `void` if method doesn’t return anything
- Method can return one value at most!
- Parameters are declared the same way we are declaring variables
  - Method can take zero parameters - just leave parentheses empty
- We can return a value from method with `return` keyword



```
int add(int a, int b) {
    return a + b;
}
```



```
int add(int a, int b) {
    return a + b;
}

int x = 5;
int y = 6;
int c = add(x, y);
```



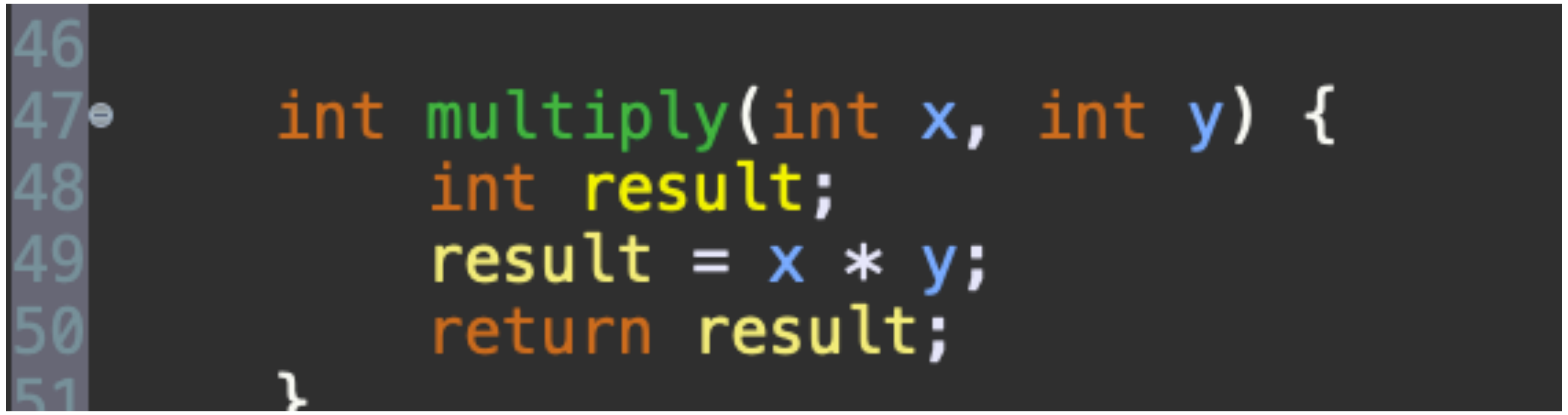
```java

    // method returns double value
    double getRandomNumber() {
        return Math.random() * 100;
    }


    // method doesn't return anything
    void printHelloWorld() {
        System.out.println("Hello World!");
    }

    // method with parameters
    int subtract(int x, int y) {
        return x - y;
    }

    // method with multiple return statements
    int max(int a, int b) {
        if (a > b) {
            return a;
        } else {
            return b;
        }
    }

```

Local variables

- Types of variables in Java
    - **Local**
    - Instance
    - Static

Local variables

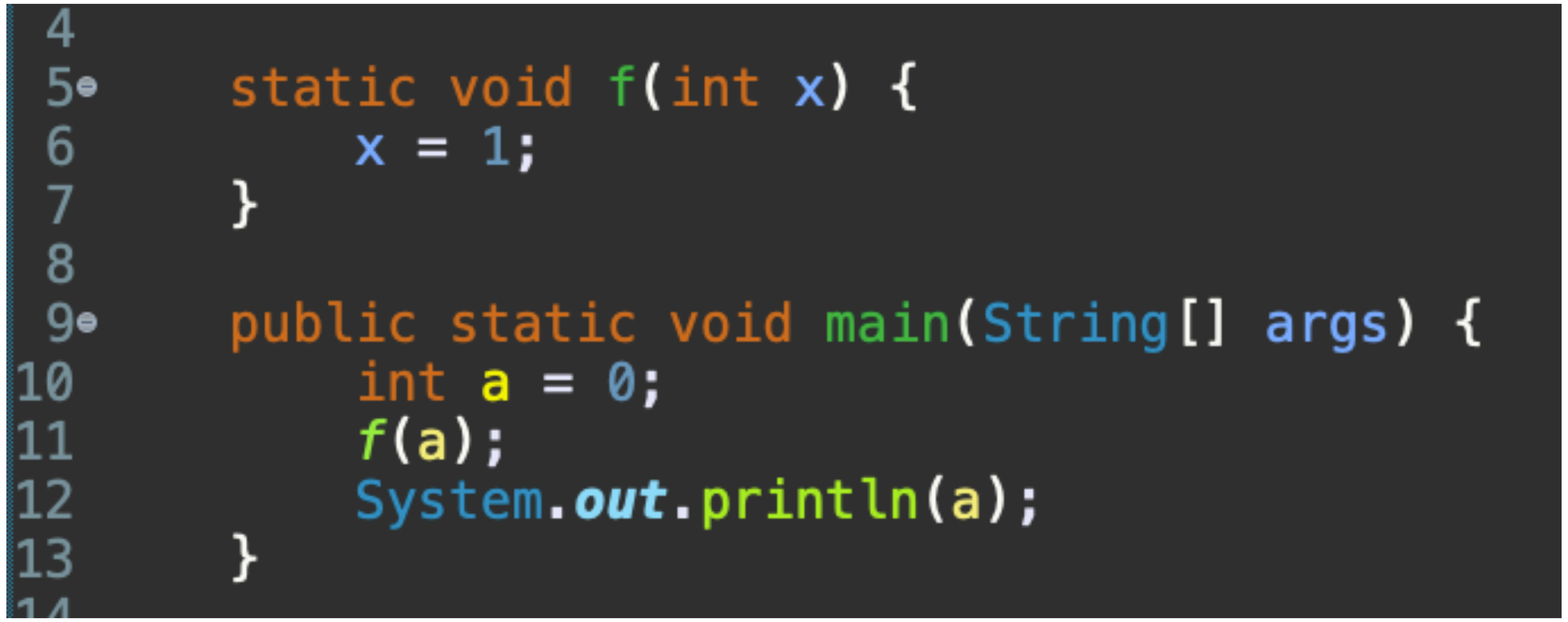
- Every variable declared in the body of a method is local variable
- Local variables in methods are created when the method is called and destroyed when the call returns from the method
- Local variables don't have a default value



```
int multiply(int x, int y) {
    int result;
    result = x * y;
    return result;
}
```

Method Parameters

- In Java, parameters are passed by value
- What does that mean?
  - copies of real parameters are created and then passed to a method
  - that means we can’t actually change the value of the argument that is passed to a method



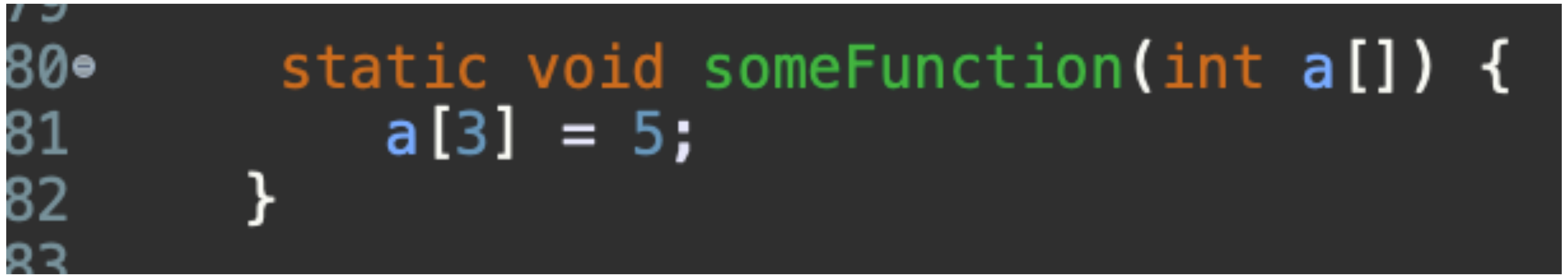
```java

    static void f(int x) {
        x = 1;
    }

    public static void main(String[] args) {
        int a = 0;
        f(a);
        System.out.println(a);
    }
```

Output

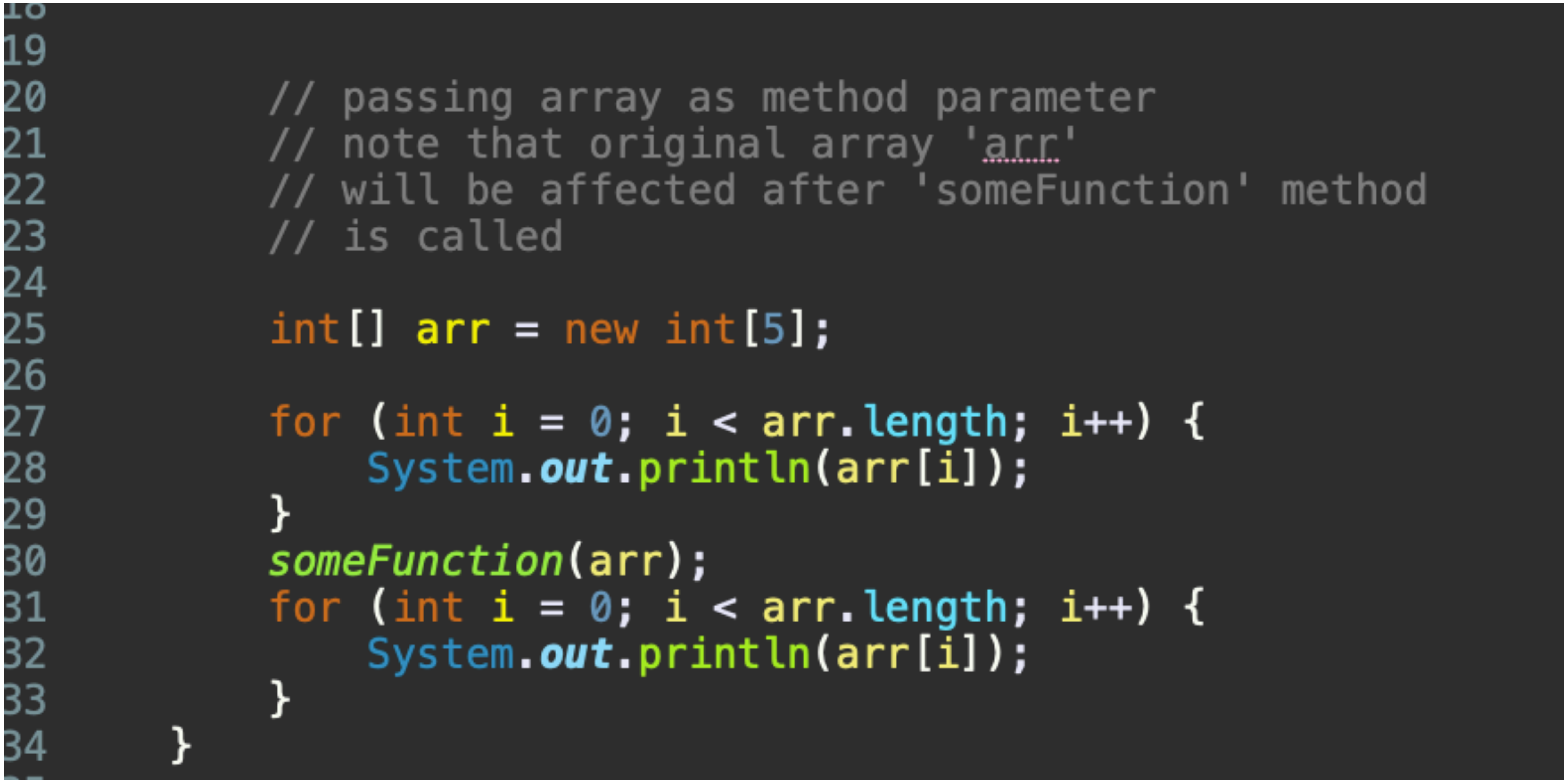
```
0
```

Arrays as Parameters

- When declaring a method that receives array as parameter, we are not specifying array dimensions
- Arrays are **not** passed by value - copy of data is **not** created
- What does that mean?
    - Original array can be changed inside of method



```
    static void someFunction(int a[]) {
        a[3] = 5;
    }
```



```
        // passing array as method parameter
        // note that original array 'arr'
        // will be affected after 'someFunction' method
        // is called

        int[] arr = new int[5];

        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
        someFunction(arr);
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }
    }
```

Variable number of parameters

- If we want to have a method that can take a variable number of parameters

-
  ```
  void f(int… params) {
      …
  }
  ```
- Parameters can be accessed the same way we are accessing array elements
- `int a = params[3];`

Exercise 1

- Write a Java method to compute and return the sum of three numbers
- Print the result in the main method
- Test data: 5, 6, 7
- Expected output: 18
- Expected print: “Sum is 18”

Exercise 2

- Write a Java method to count all the words in a string
- Print result in the main method
- Test data: “This is a Java sentence”
- Expected output: 5
- Expected print: “Sentence has 5 words”

Exercise 3

- Write a Java method to check if password string is valid
- Rules:
  - Password must have 8-30 characters
  - Password must have at least two digits
  - Password must contain only characters and digits
- Test data: “123”, “asdfgh12345!@#”, “asdf1234”
- Expected output: only third password is valid
- Hint: iterate over string with a for loop, and use `String.charAt()` method to access every letter
- Hint: we can check if letter is an alphabet character with `letter >= ‘a’ && letter <= ‘z’` for lower-case letters and `letter >= ‘A’ && letter <= ‘Z’` for upper case letters

Exercise 4

- Write a Java method that takes one string and converts it to upper-case
- Test data: “hEllO wOrlD”
- Expected output: “HELLO WORLD”

Exercise 5

- Write a Java method named `square` that takes one number and returns it's squared value.
- Write another Java method named `complexCalculation` that takes two numbers as input, sum those two numbers and than squares the sum (for sum use method from the first Exercise, and for square use method created in this Exercise).
- Test data: 5, 6
- Expected output: 121 (5+6)^2 = 11^2 = 121