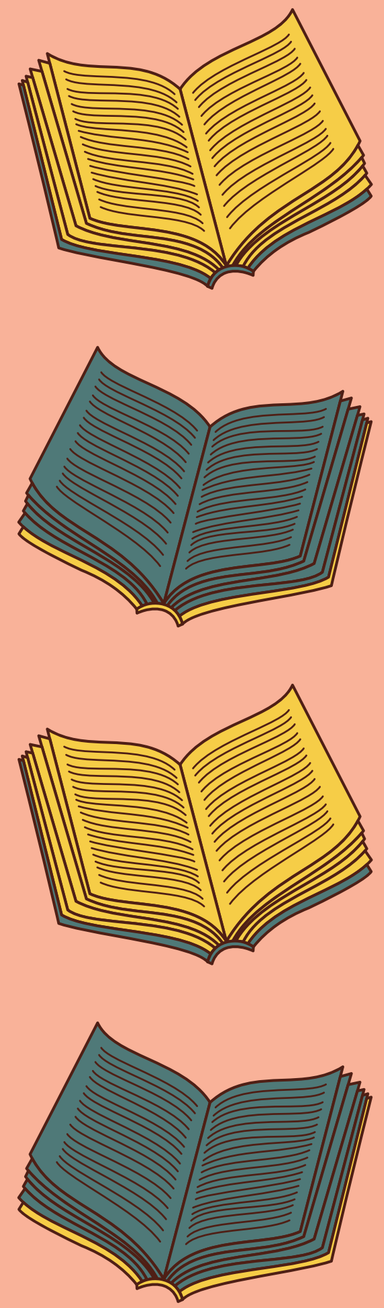
Advanced JavaScript

the Call Stack

Call Stack

The mechanism the JS interpreter uses to keep track of its place in a script that calls multiple functions.

How JS "knows" what function is currently being run and what functions are called from within that function, etc.

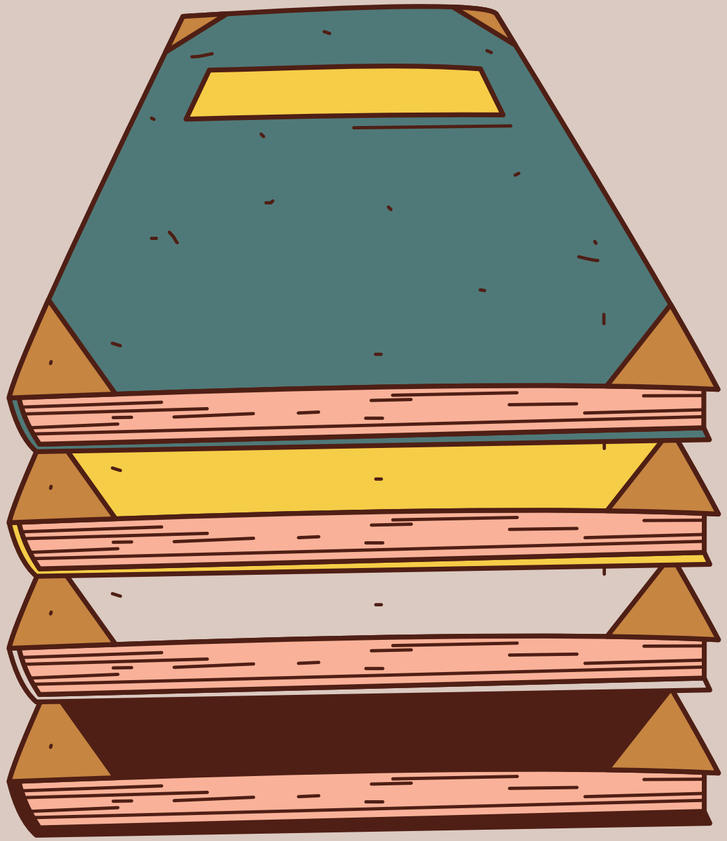


Call Stack

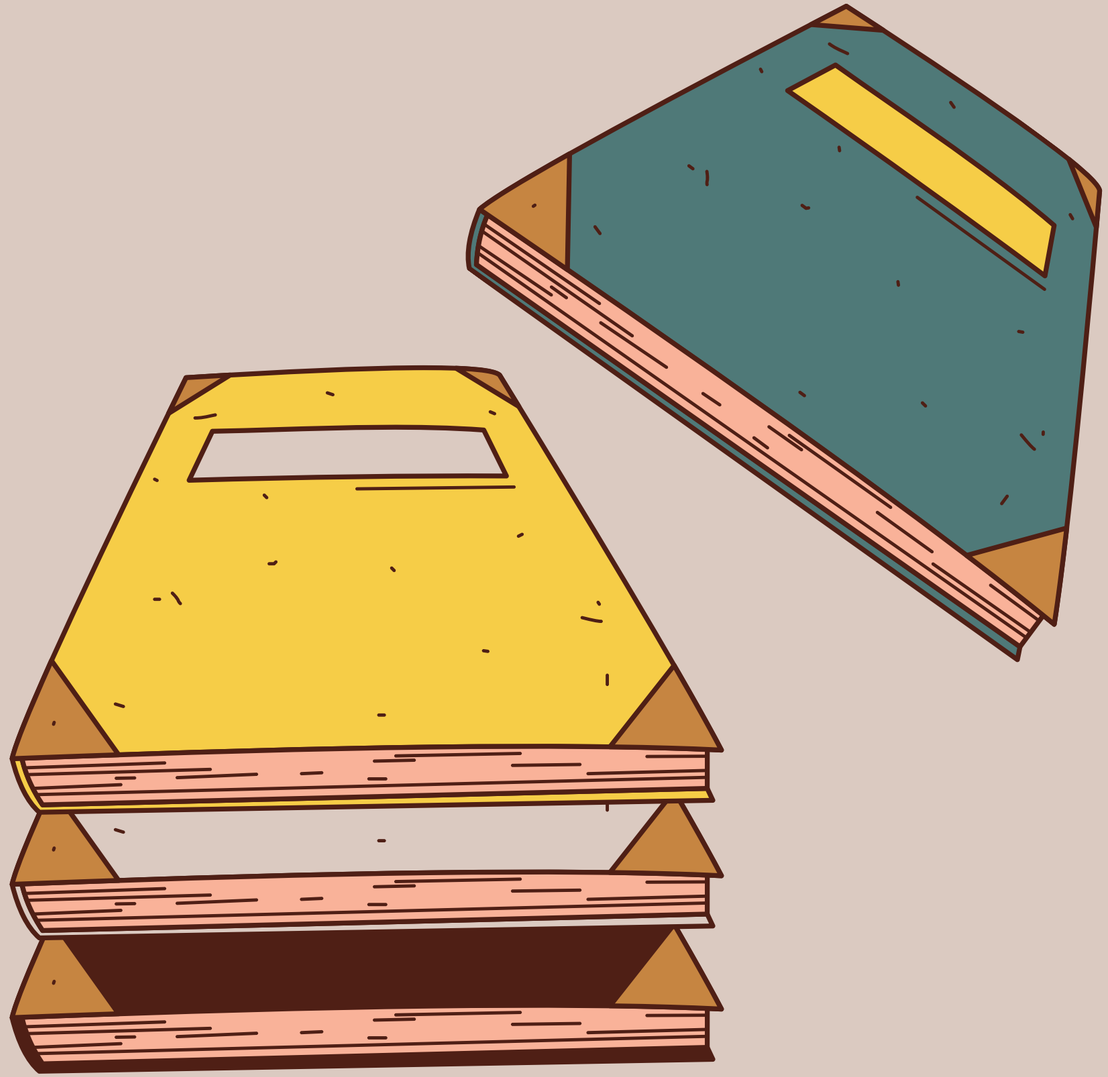
References?

Where was I?

Last Thing In...

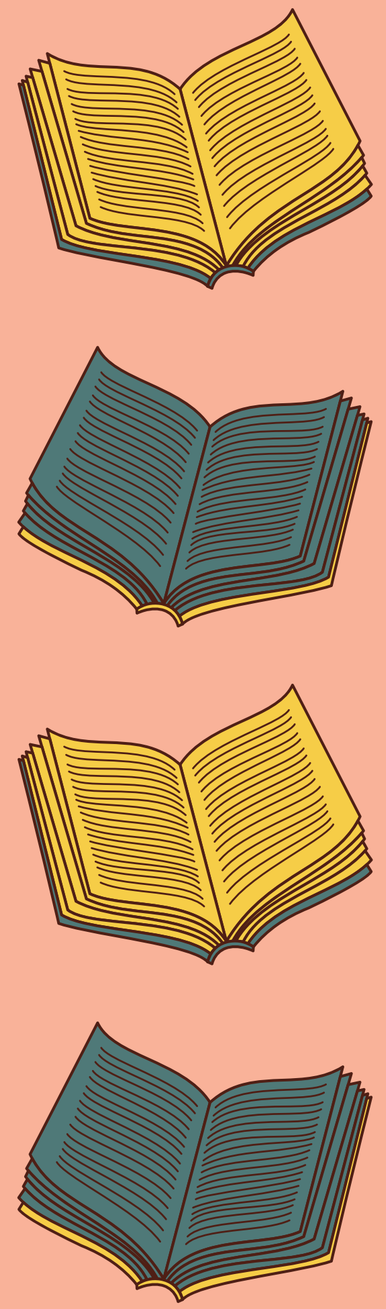


First Thing Out. . .

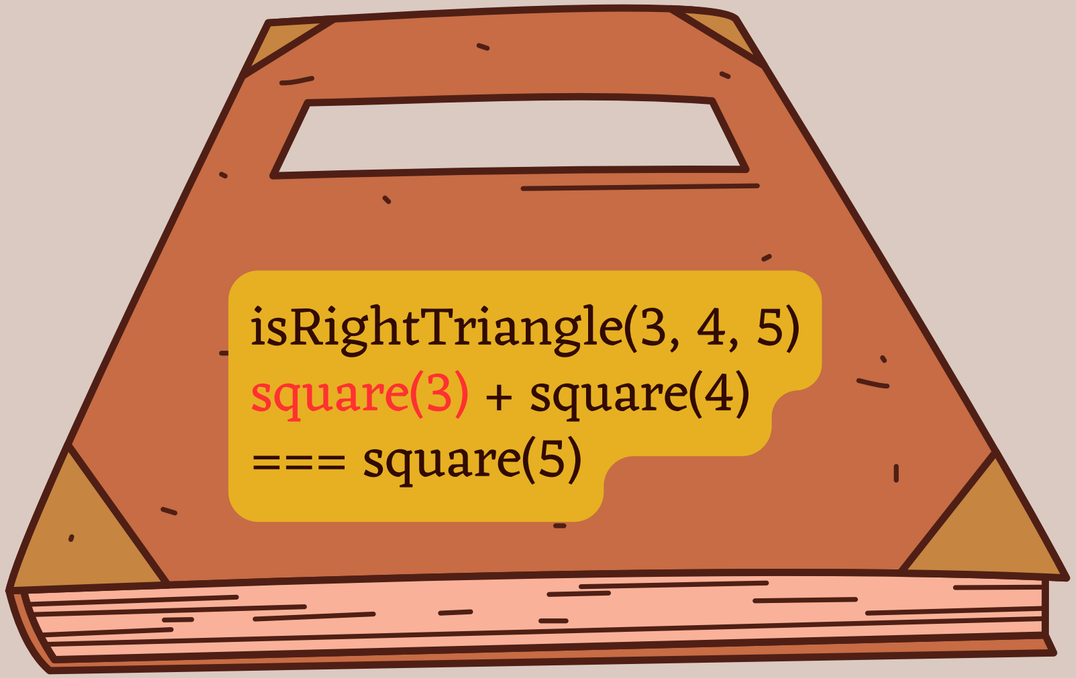


How It Works

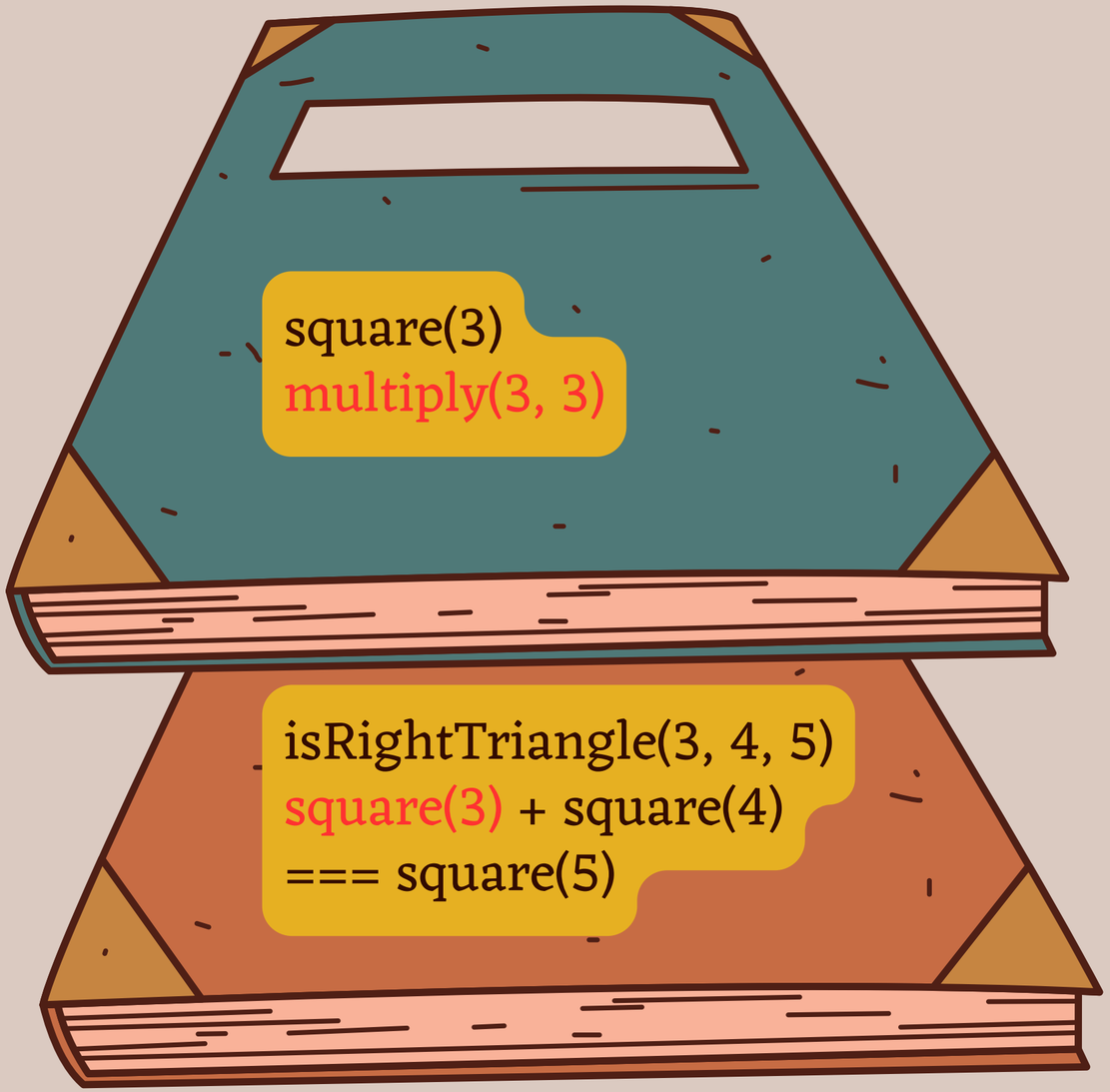
- When a script calls a function, the interpreter adds it to the call stack and then starts carrying out the function.
- Any functions that are called by that function are added to the call stack further up, and run where their calls are reached.
- When the current function is finished, the interpreter takes it out of the stack and resumes execution where it left off in the last code listing.



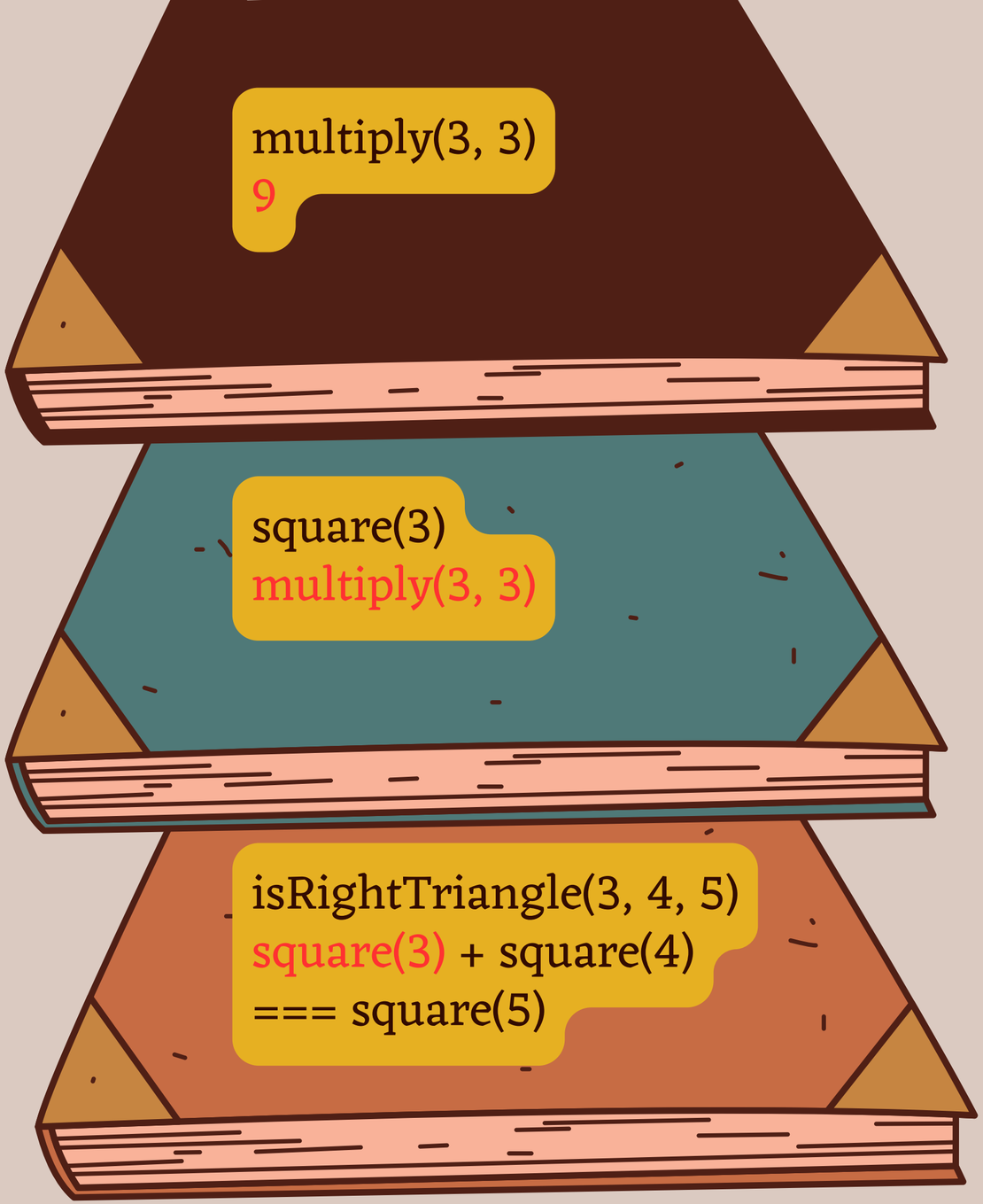
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



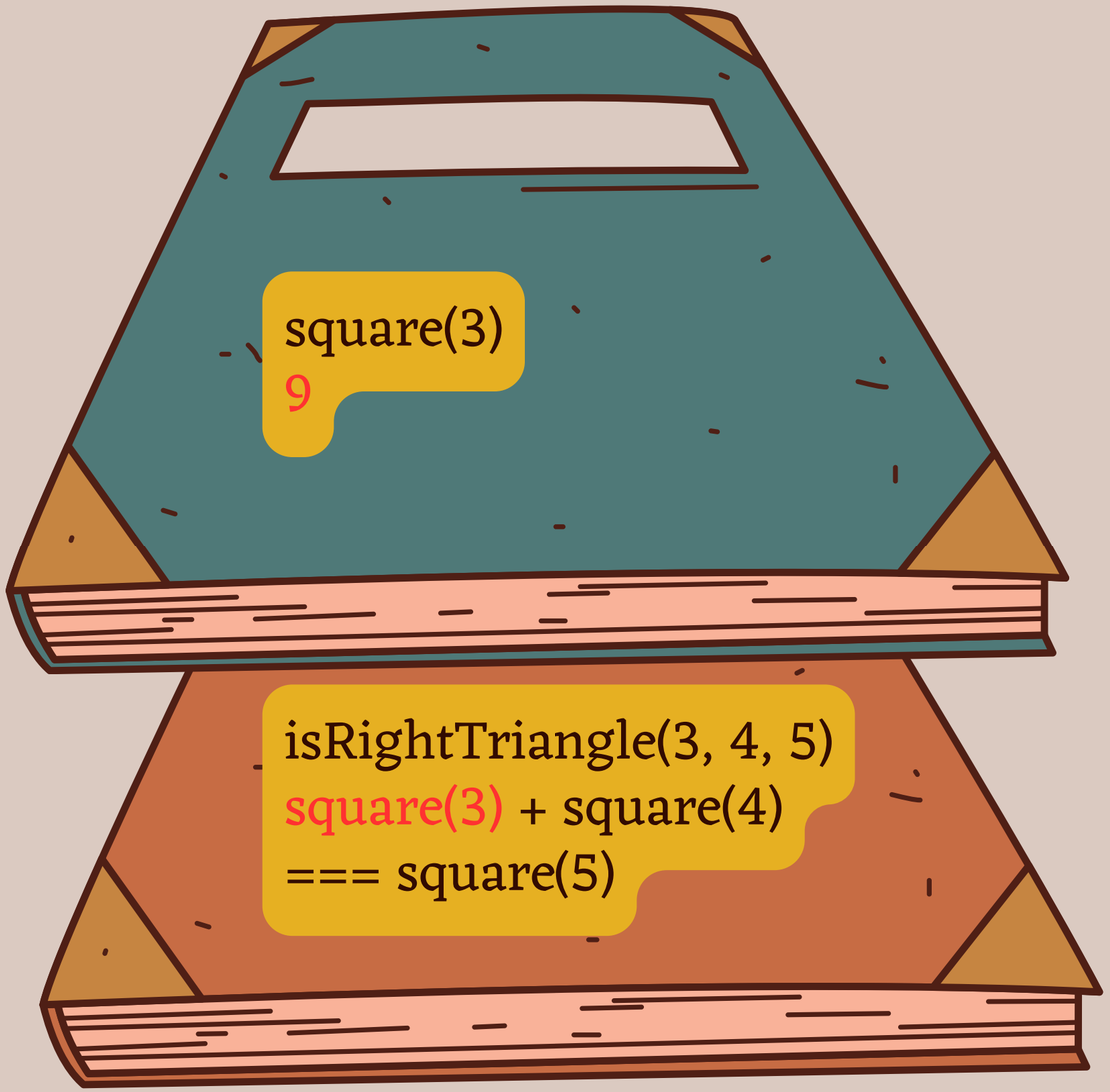
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



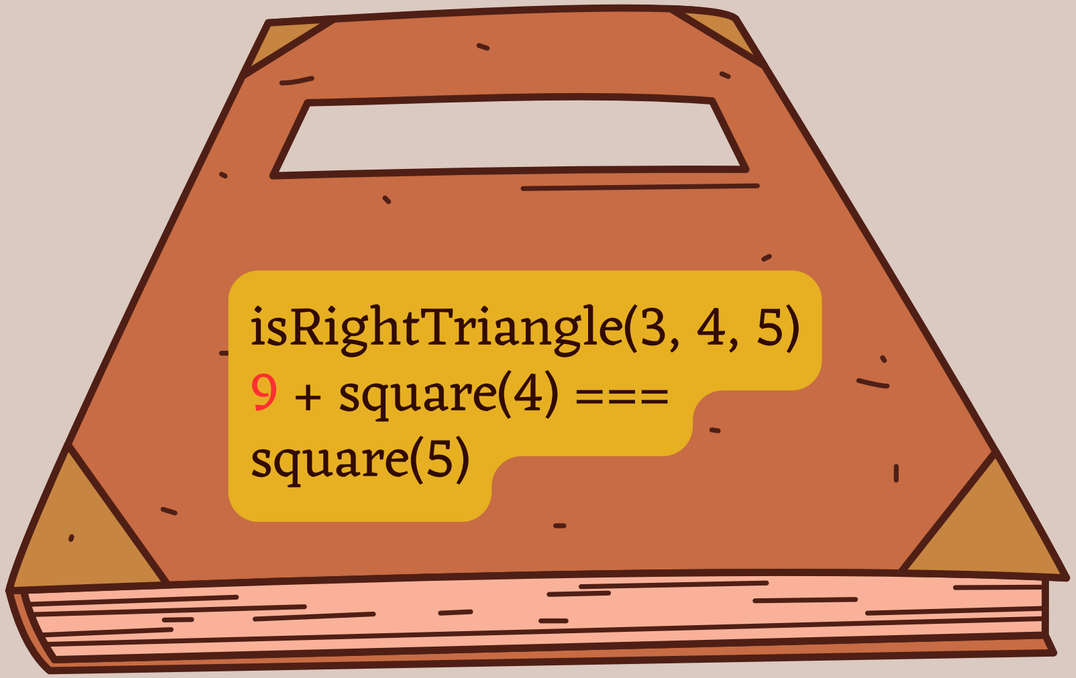
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



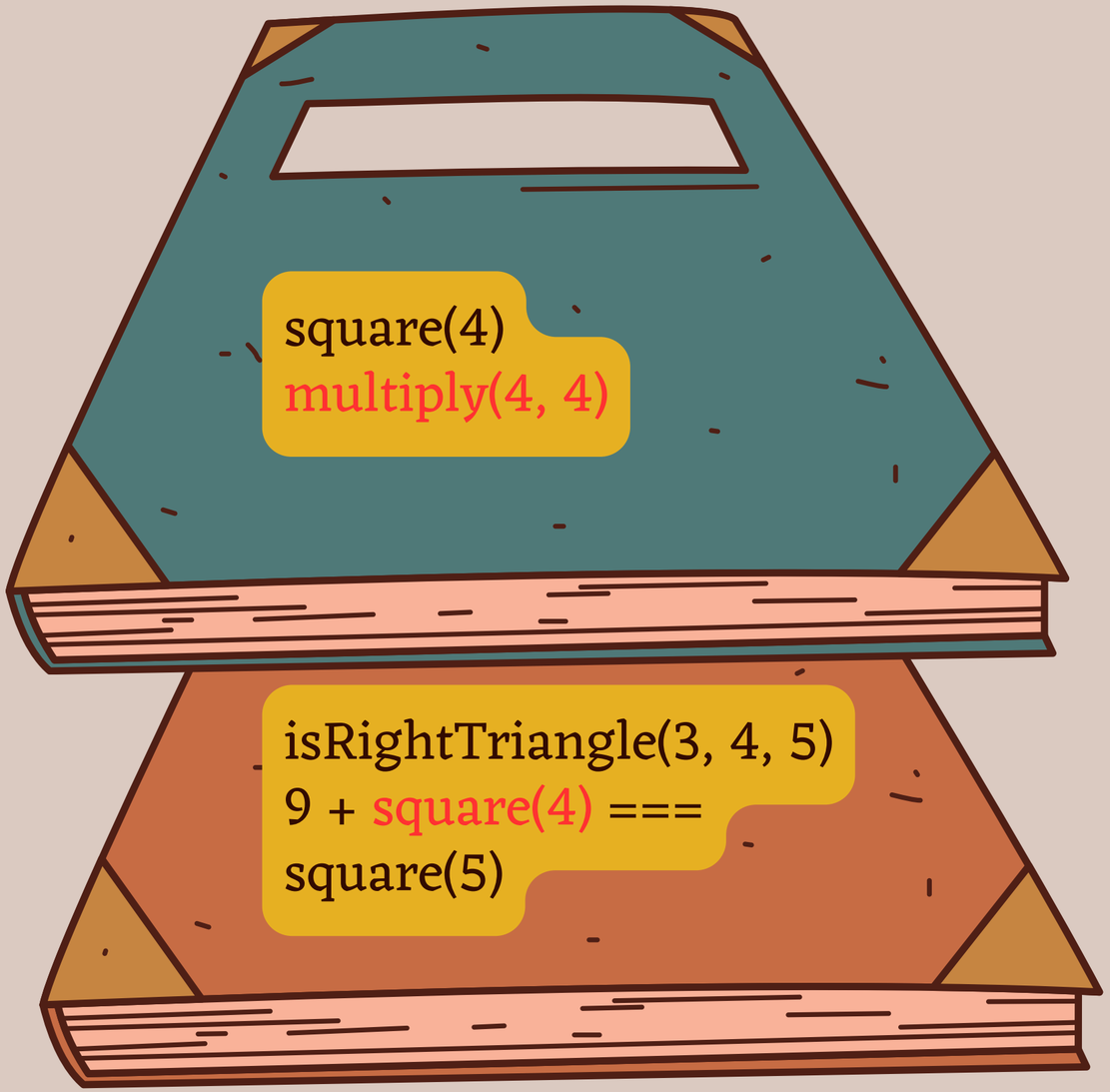
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



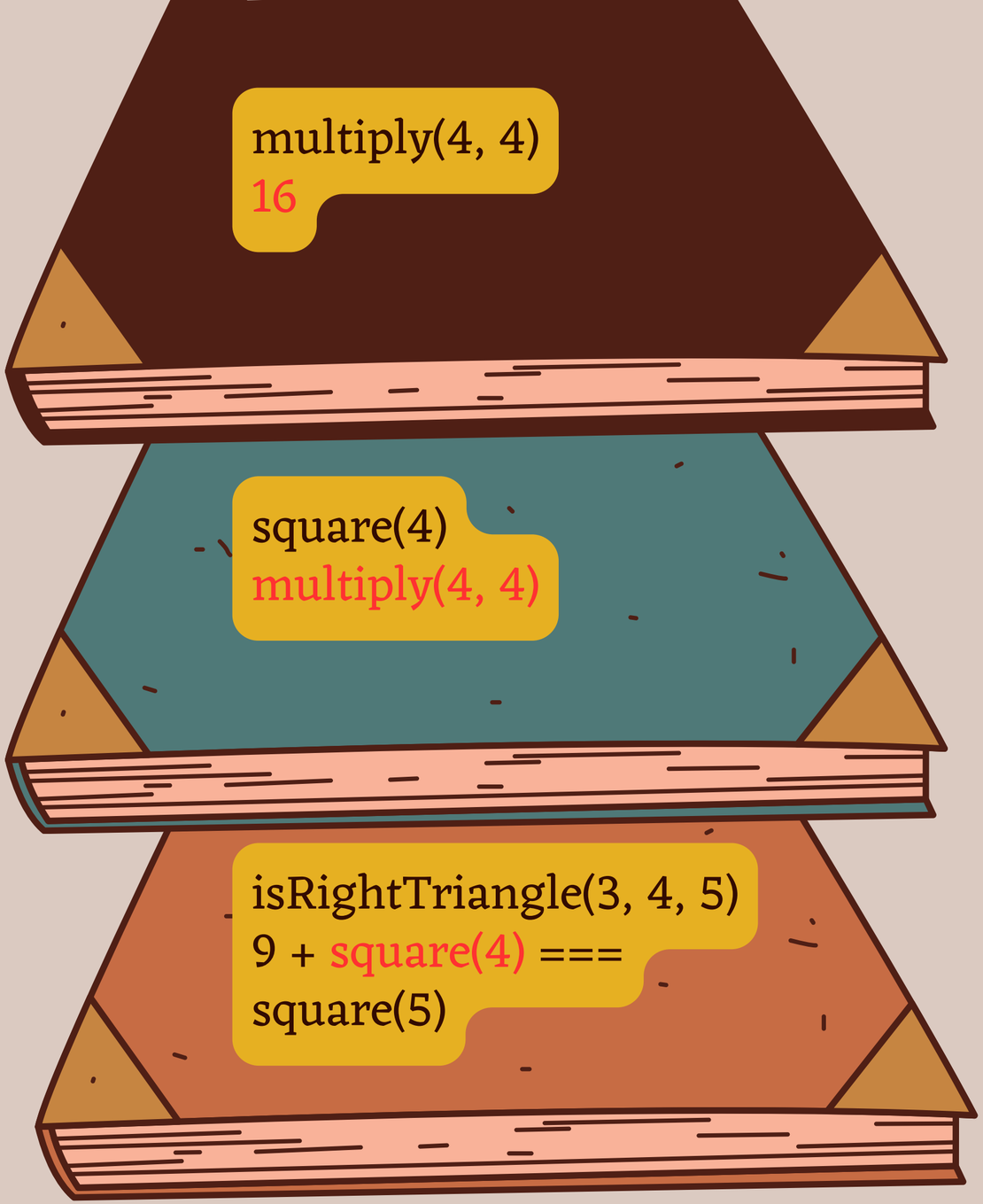
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



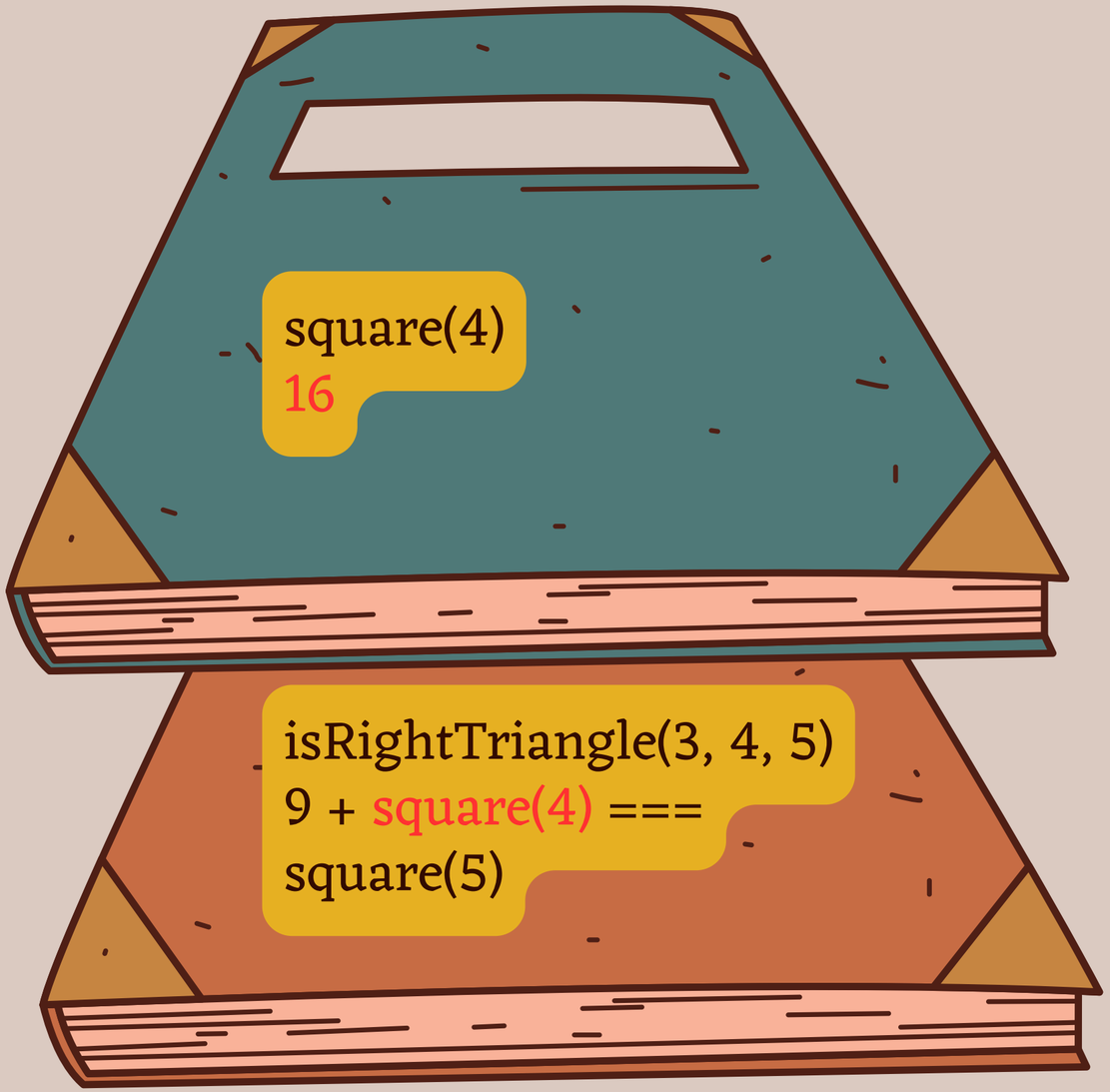
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



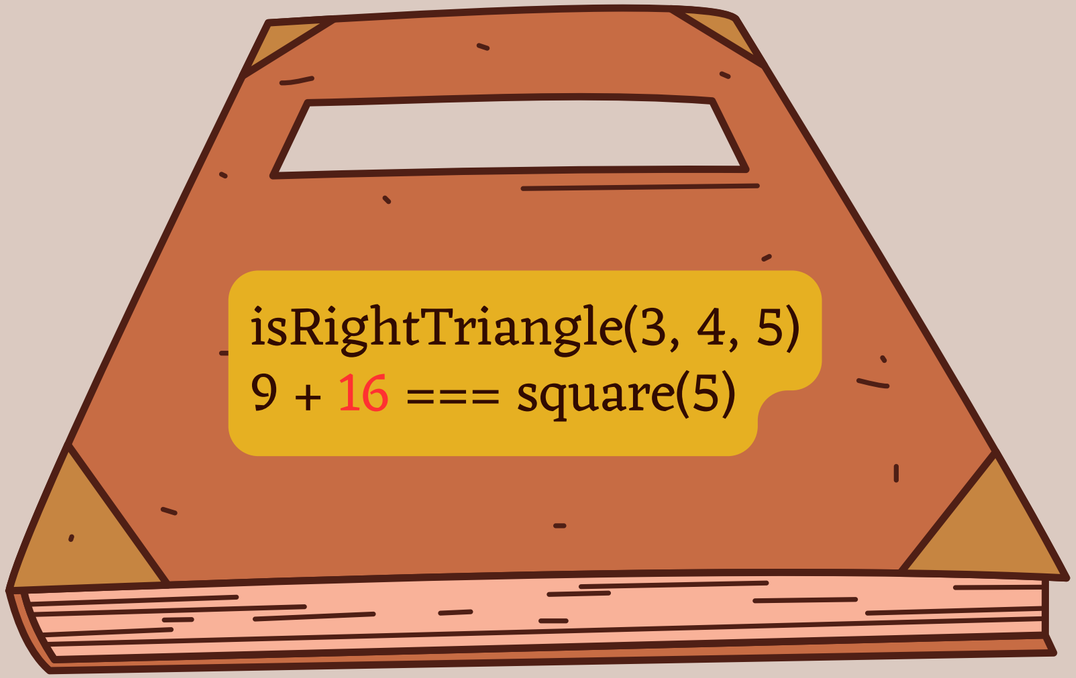
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



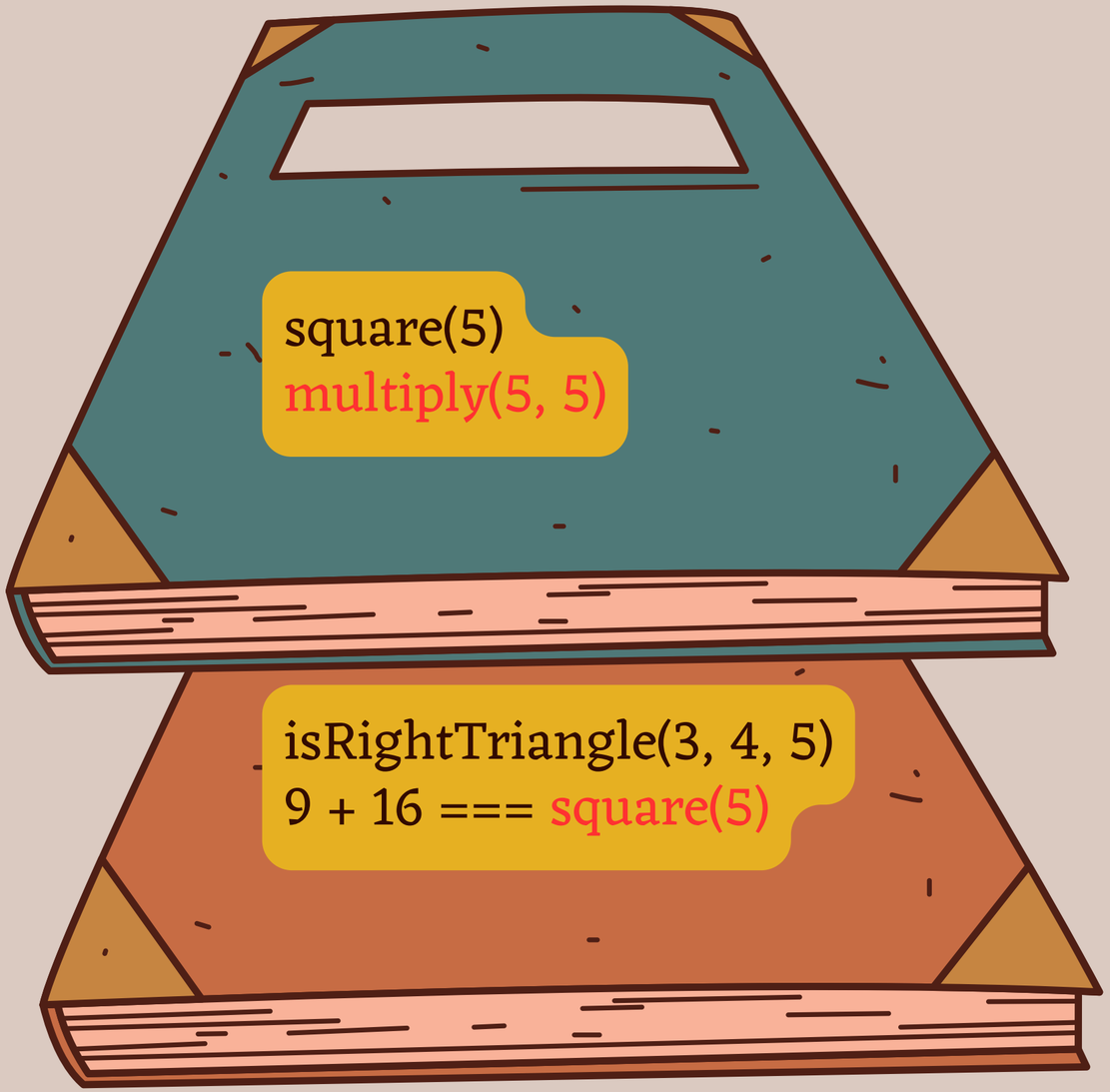
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```

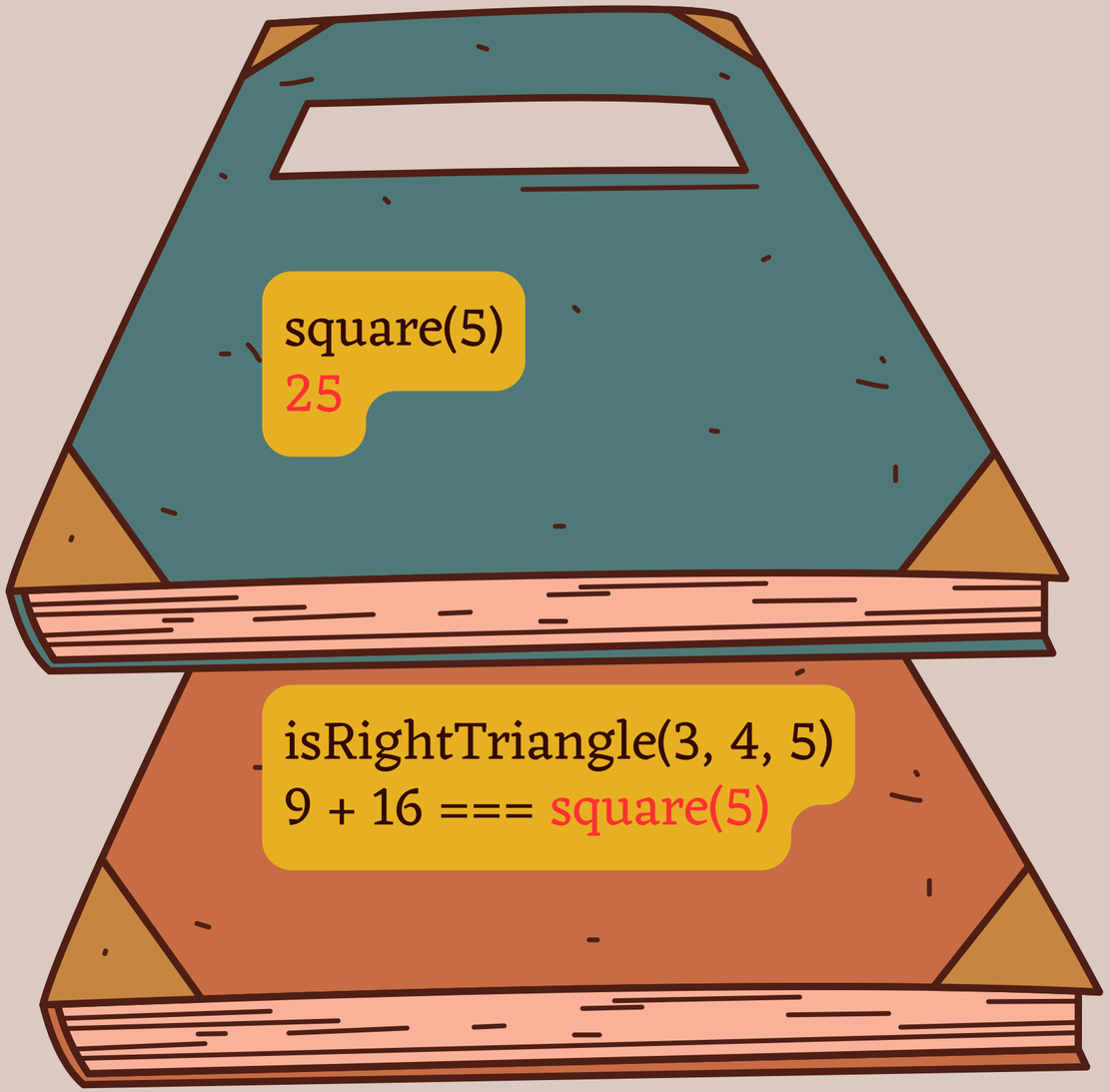


```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```

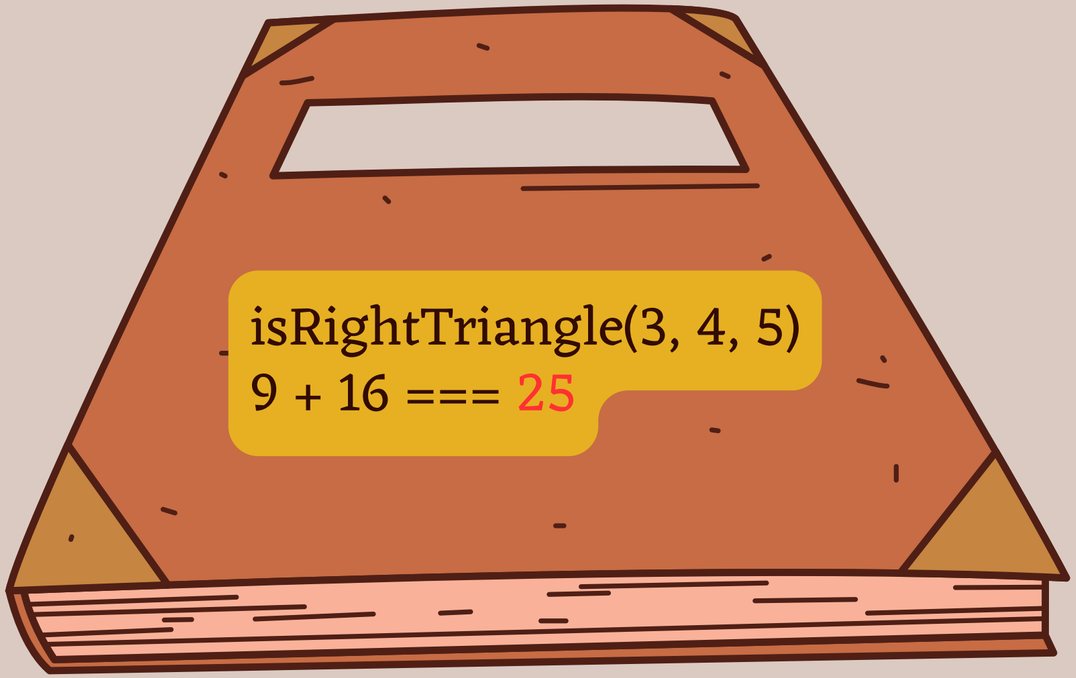


```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```

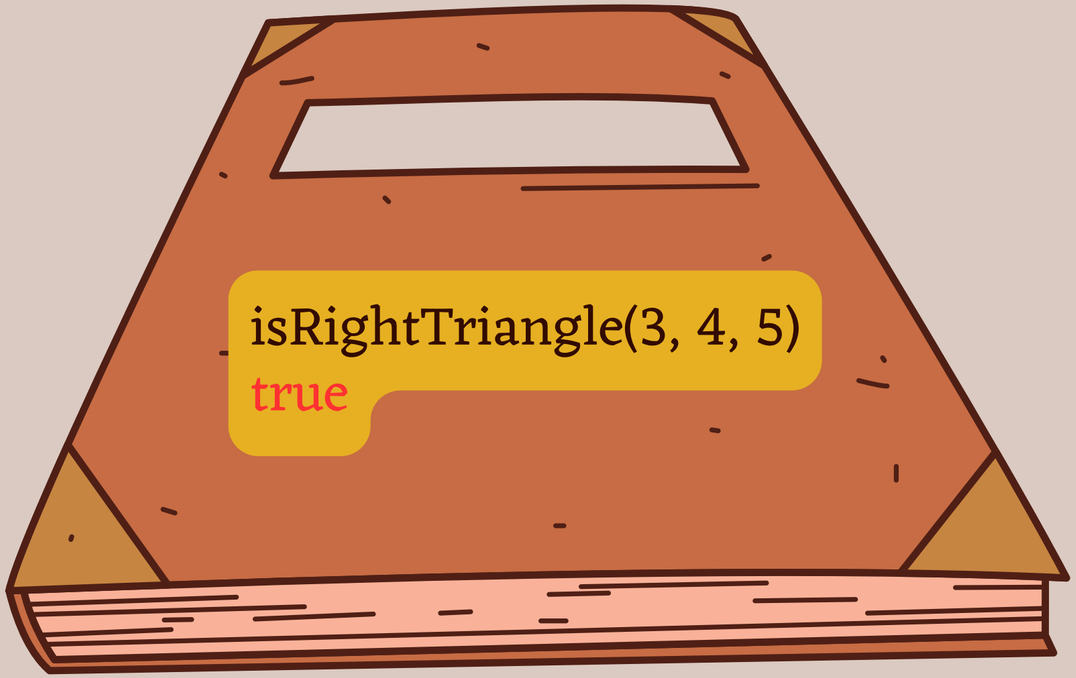
```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```



```
1  const multiply = (x, y) => x * y;
2
3  const square = (x) => multiply(x, x);
4
5  const isRightTriangle = (a, b, c) => {
6    return square(a) + square(b) === square(c)
7  }
8
9  isRightTriangle(3, 4, 5)
```

true