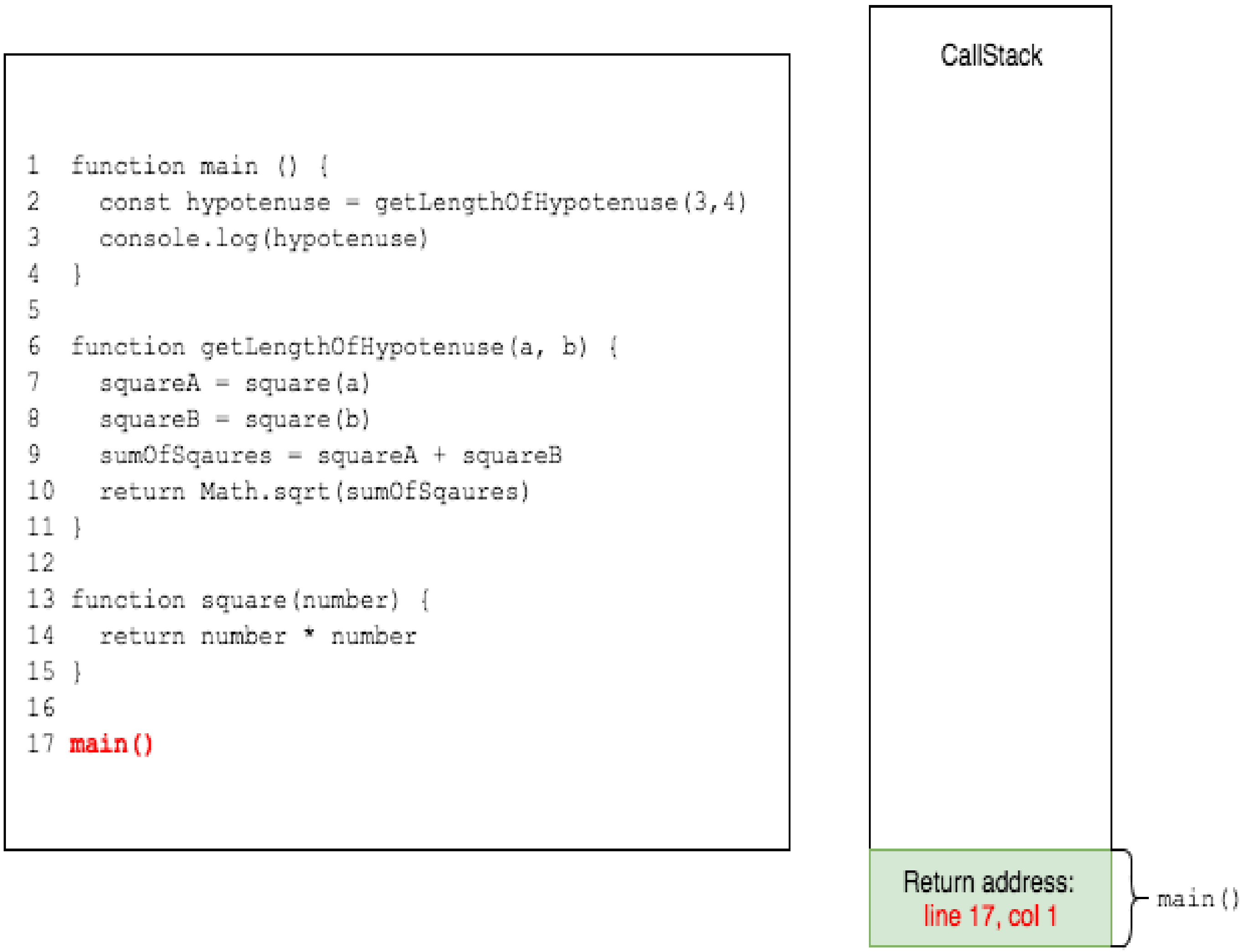
Stack

Whenever you call a function return address, parameters and local variables will be pushed to the stack. If you call another function from the currently running function, its contents will be pushed on top in the same manner as the previous one - with its return address.

For the sake of simplicity, I will say that 'a function is pushed' to the top of the stack from now on, even though it is not exactly correct.

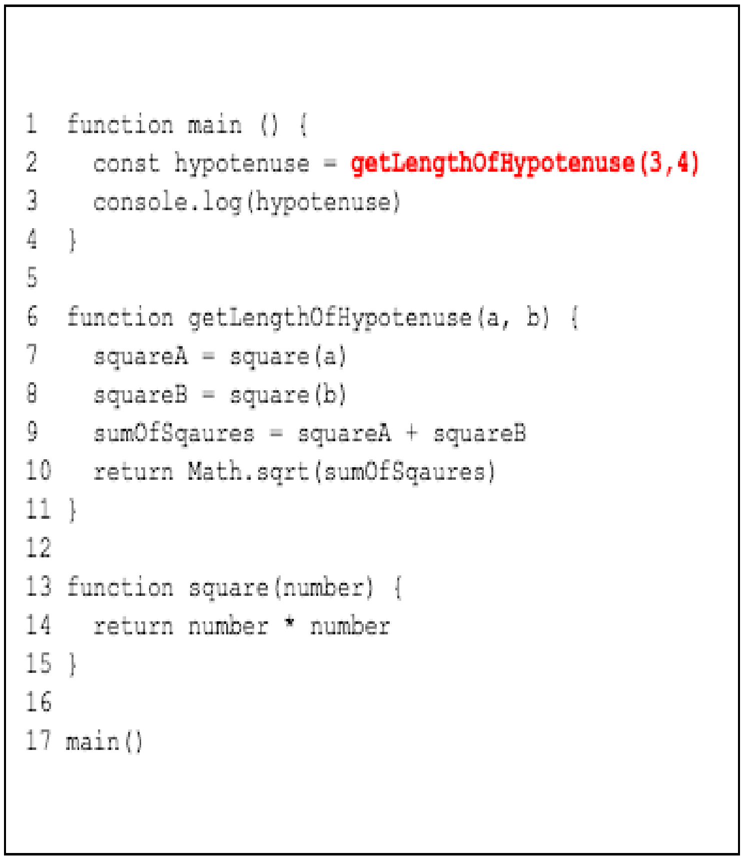
Let's take a look!

main is called first:

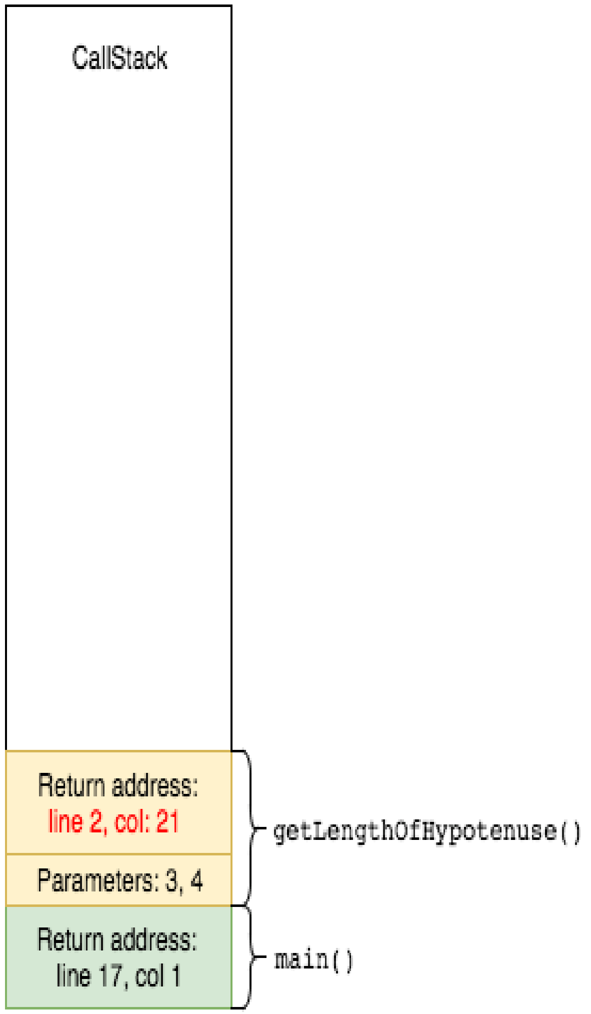


```
1   function main () {
2     const hypotenuse = getLengthOfHypotenuse(3,4)
3     console.log(hypotenuse)
4   }
5
6   function getLengthOfHypotenuse(a, b) {
7     squareA = square(a)
8     squareB = square(b)
9     sumOfSqaures = squareA + squareB
10    return Math.sqrt(sumOfSqaures)
11  }
12
13  function square(number) {
14    return number * number
15  }
16
17  main()
```

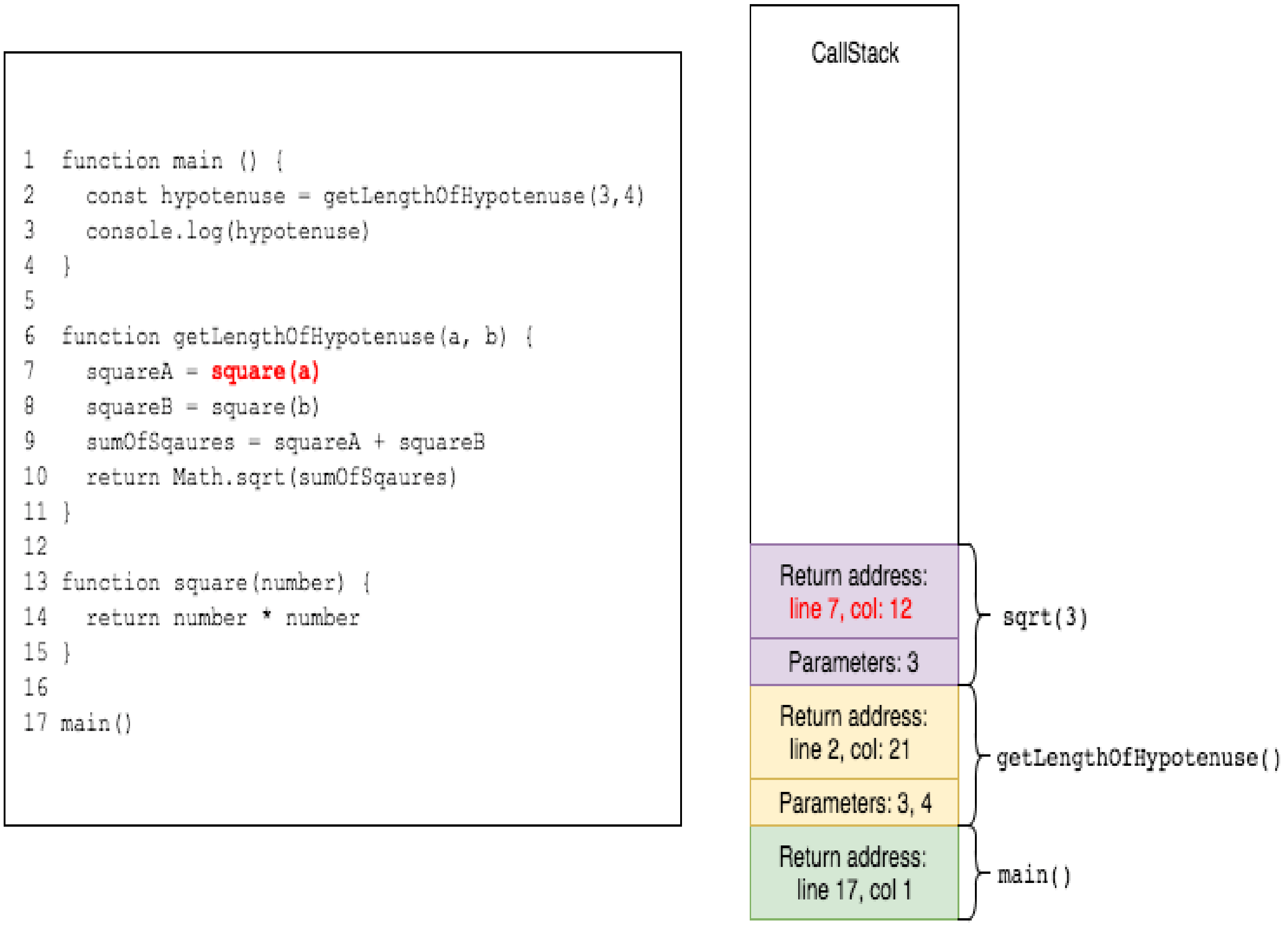
then main calls getLengthOfHypotenuse with 3 and 4 as arguments



```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```



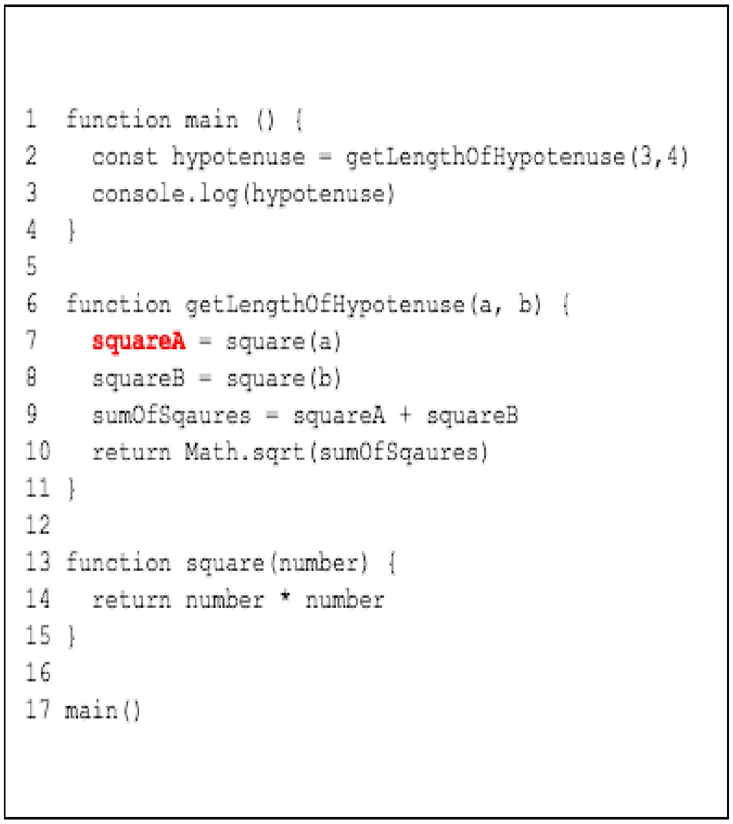
afterwards square is with the value of a



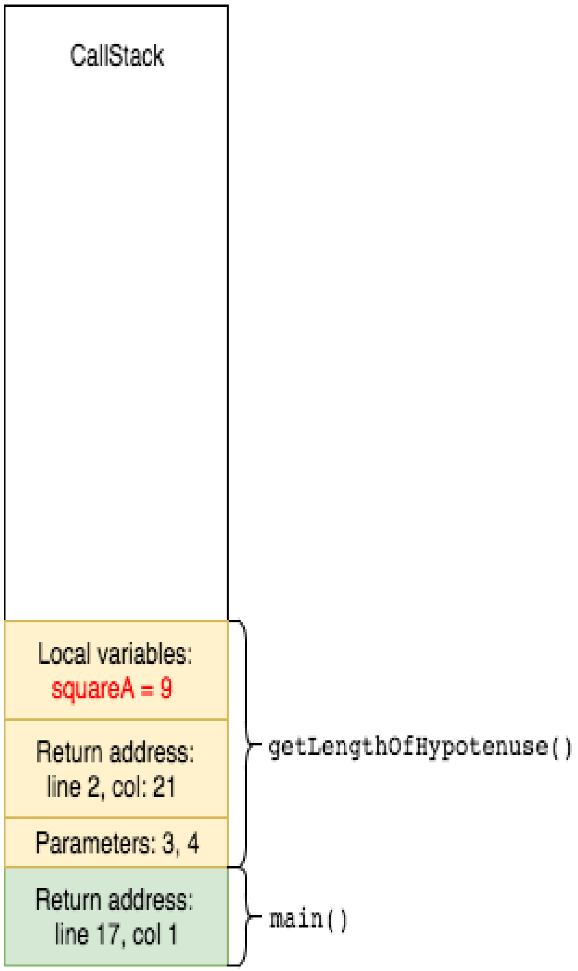
```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```

CallStack
Return address: line 7, col: 12
Parameters: 3
sqrt(3)
Return address: line 2, col: 21
Parameters: 3, 4
getLengthOfHypotenuse()
Return address: line 17, col 1
main()

when square returns, it is popped from the stack, and its return value is assigned to squareA. squareA is added to the stack frame of getLengthOfHypotenuse

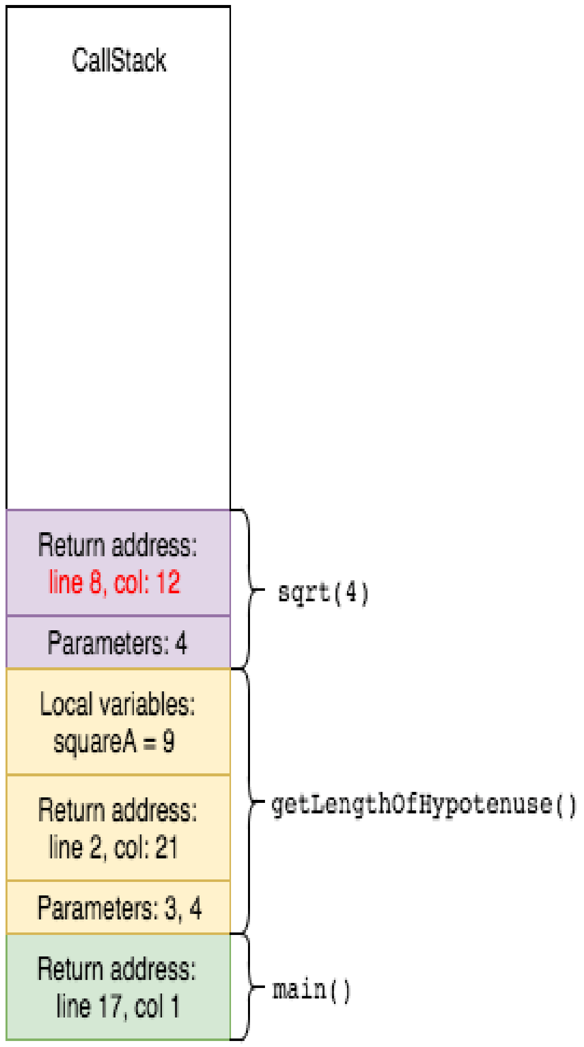


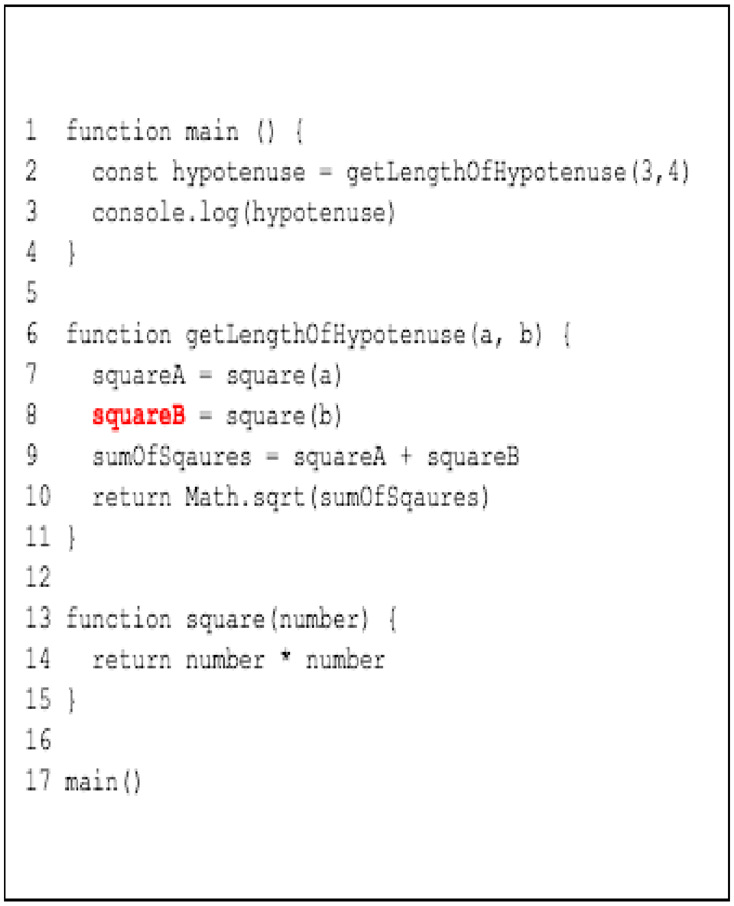
```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```



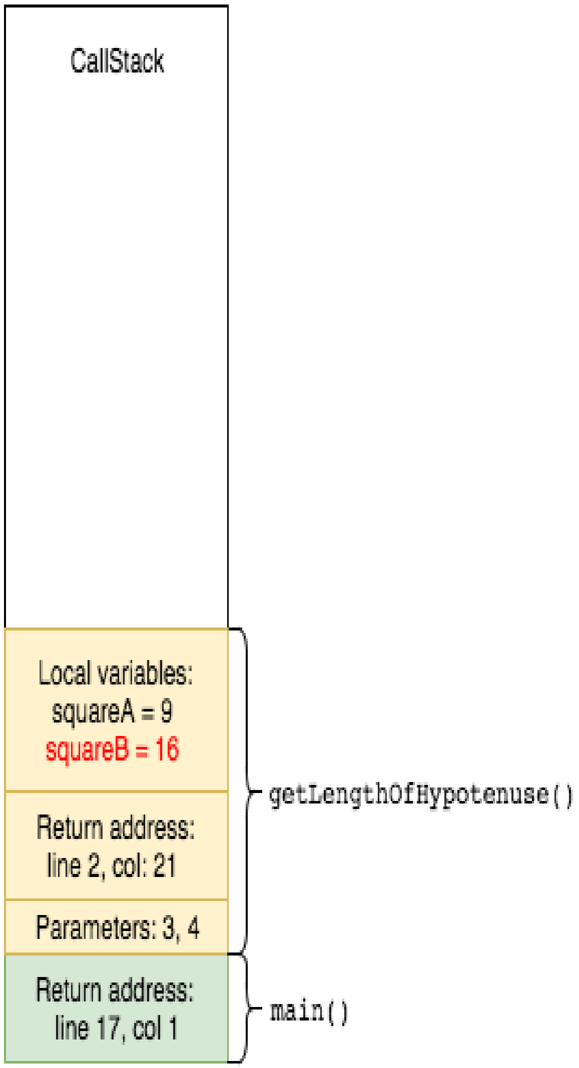
same goes for the next call to square

```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```

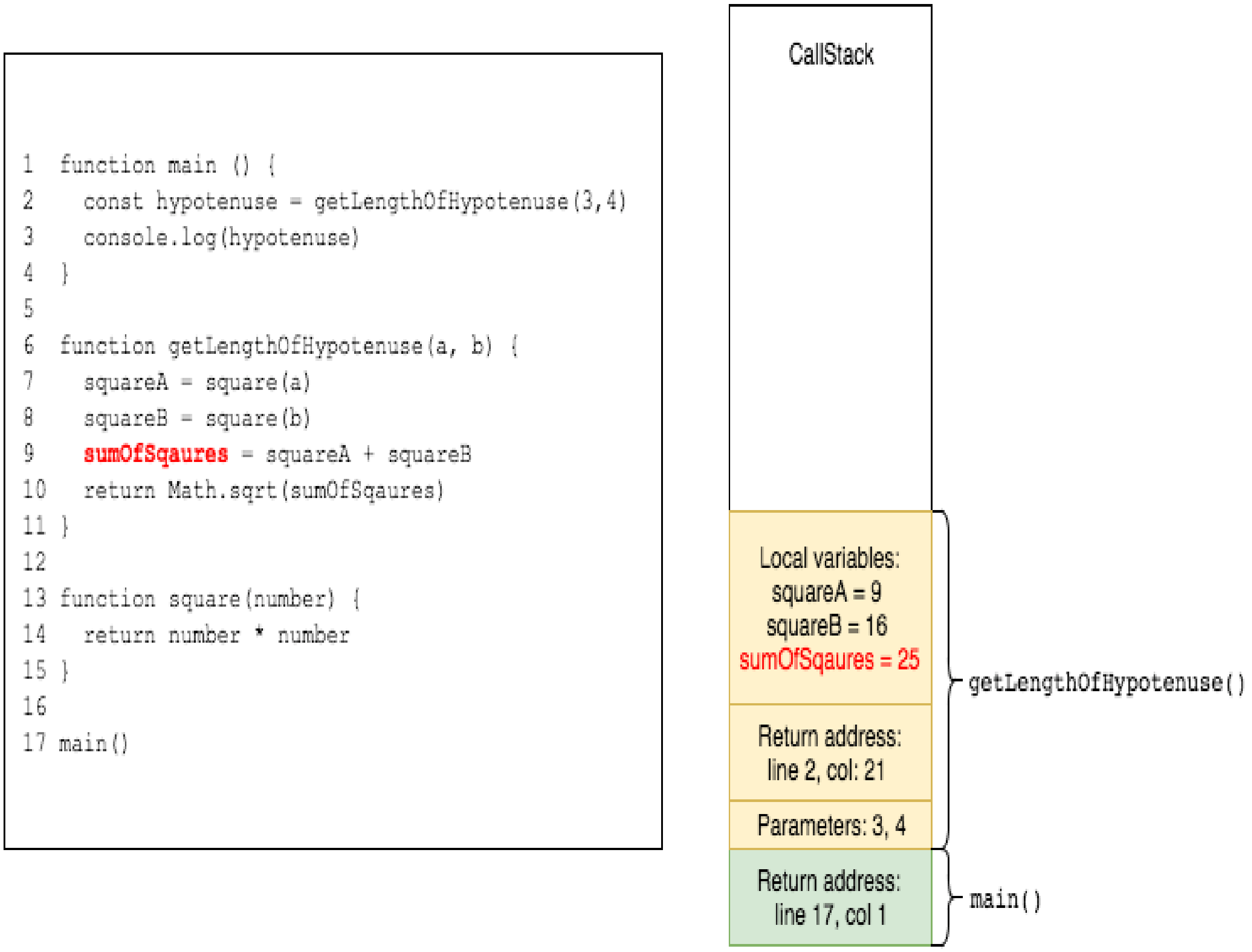

CallStack
Return address: line 8, col: 12
Parameters: 4
sqrt(4)
Local variables: squareA = 9
Return address: line 2, col: 21
Parameters: 3, 4
getLengthOfHypotenuse()
Return address: line 17, col 1
main()



```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```

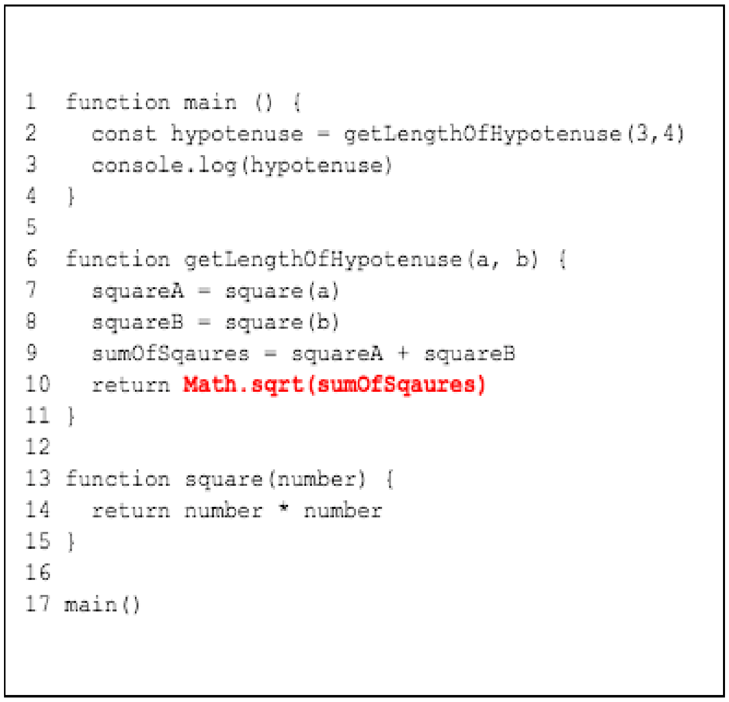


in the next line the expression squareA + squareB is evaluated

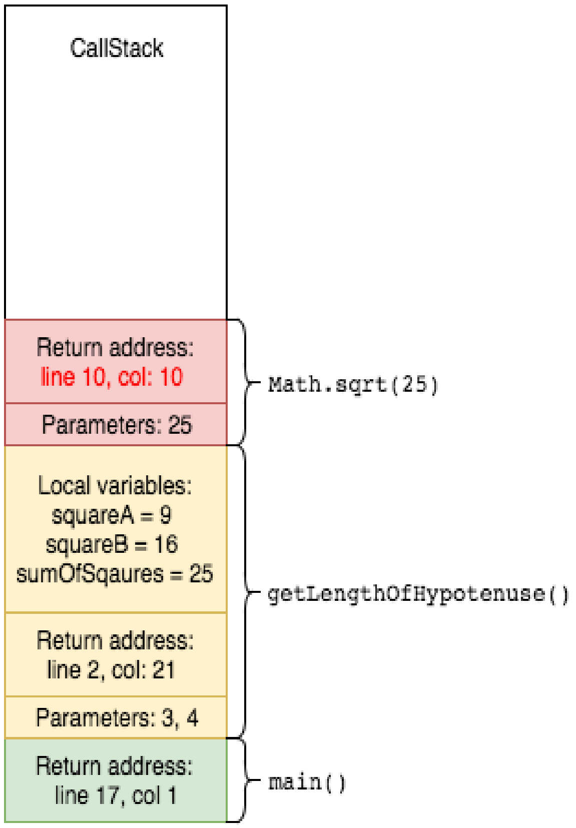


```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```

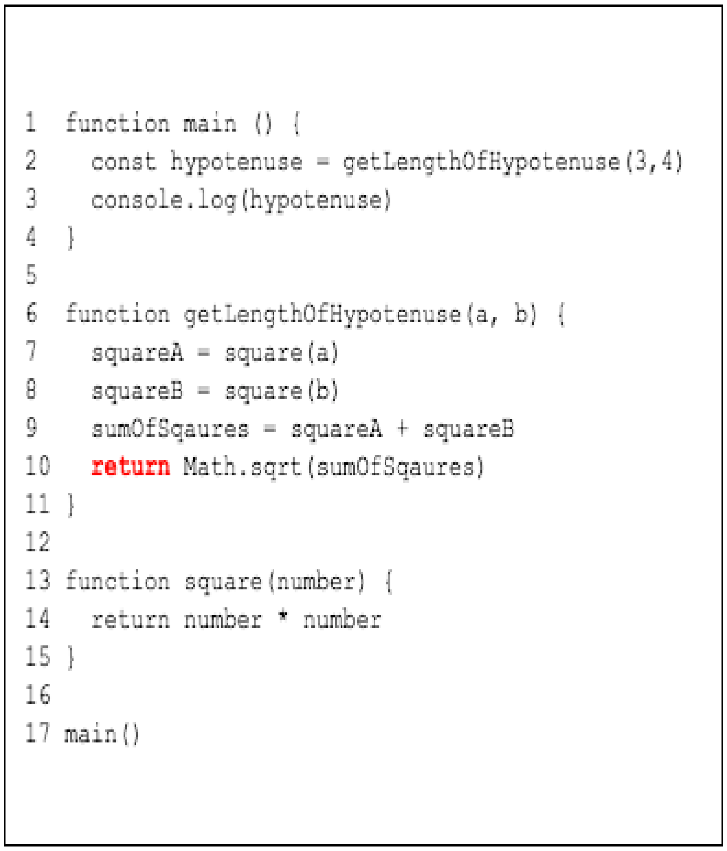
then Math.sqrt is called with sumOfSquares



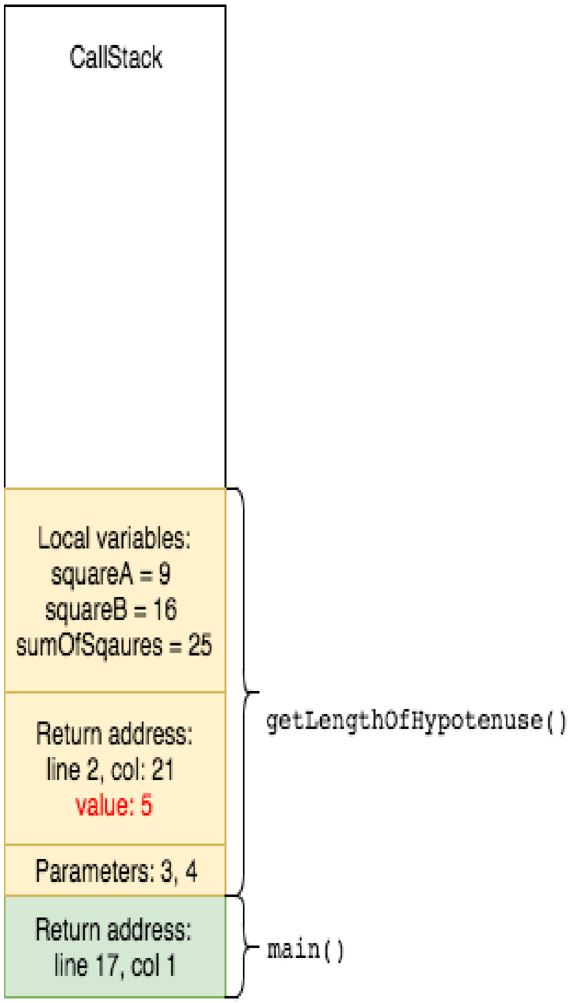
```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```


CallStack
Return address: line 10, col: 10
Parameters: 25
Math.sqrt(25)
Local variables: squareA = 9 squareB = 16 sumOfSqaures = 25
Return address: line 2, col: 21
Parameters: 3, 4
getLengthOfHypotenuse()
Return address: line 17, col 1
main()

now all is left for getLengthOfHypotenuse is to return the final value of its calculation

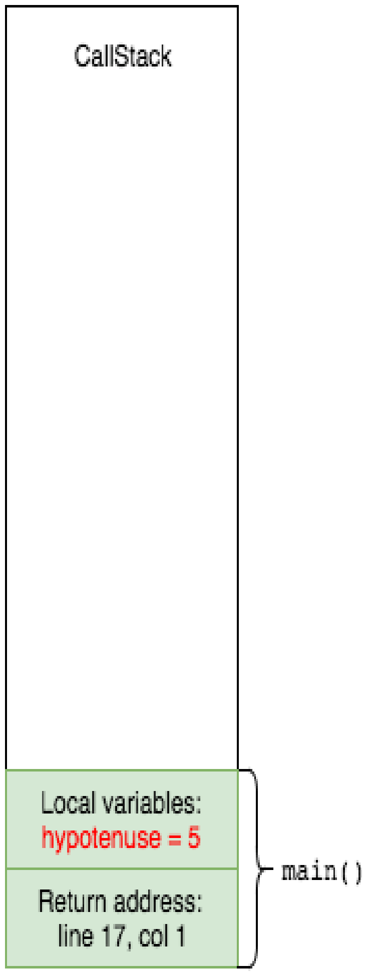


```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```

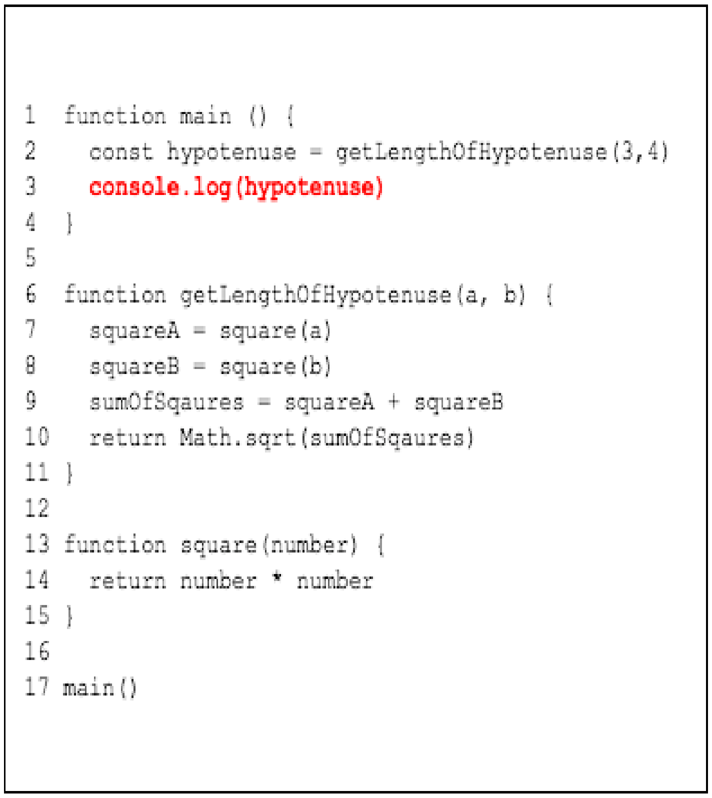


the returned value gets assigned to hypotenuse in main

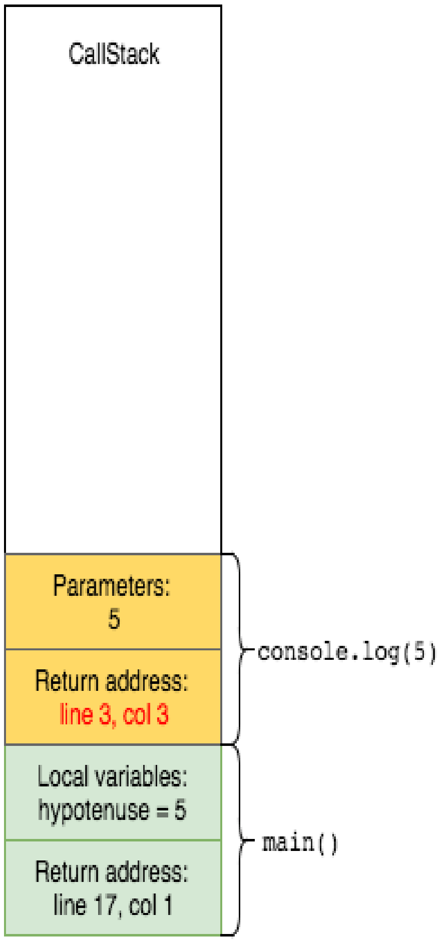
```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```



the value of hypotenuse is logged to console



```
1  function main () {
2    const hypotenuse = getLengthOfHypotenuse(3,4)
3    console.log(hypotenuse)
4  }
5
6  function getLengthOfHypotenuse(a, b) {
7    squareA = square(a)
8    squareB = square(b)
9    sumOfSqaures = squareA + squareB
10   return Math.sqrt(sumOfSqaures)
11 }
12
13 function square(number) {
14   return number * number
15 }
16
17 main()
```



finally, main returns without any value, gets popped from the stack leaving it empty

```
function main () {
  const hypotenuse = getLengthOfHypotenuse(3,4)
  console.log(hypotenuse)
}

function getLengthOfHypotenuse(a, b) {
  squareA = square(a)
  squareB = square(b)
  sumOfSqaures = squareA + squareB
  return Math.sqrt(sumOfSqaures)
}

function square(number) {
  return number * number
}

main()
```

CallStack