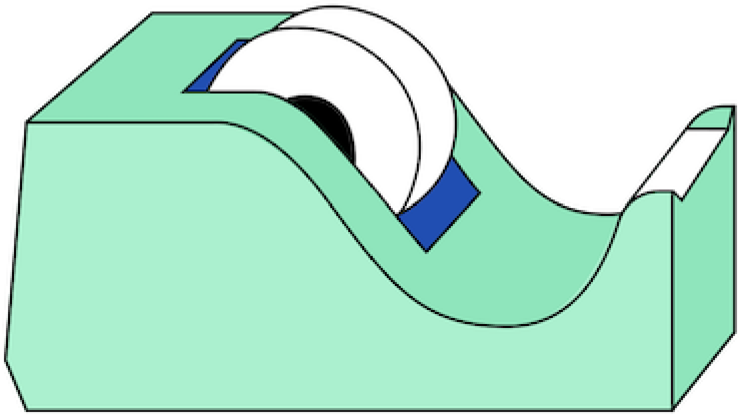


Coding with JS

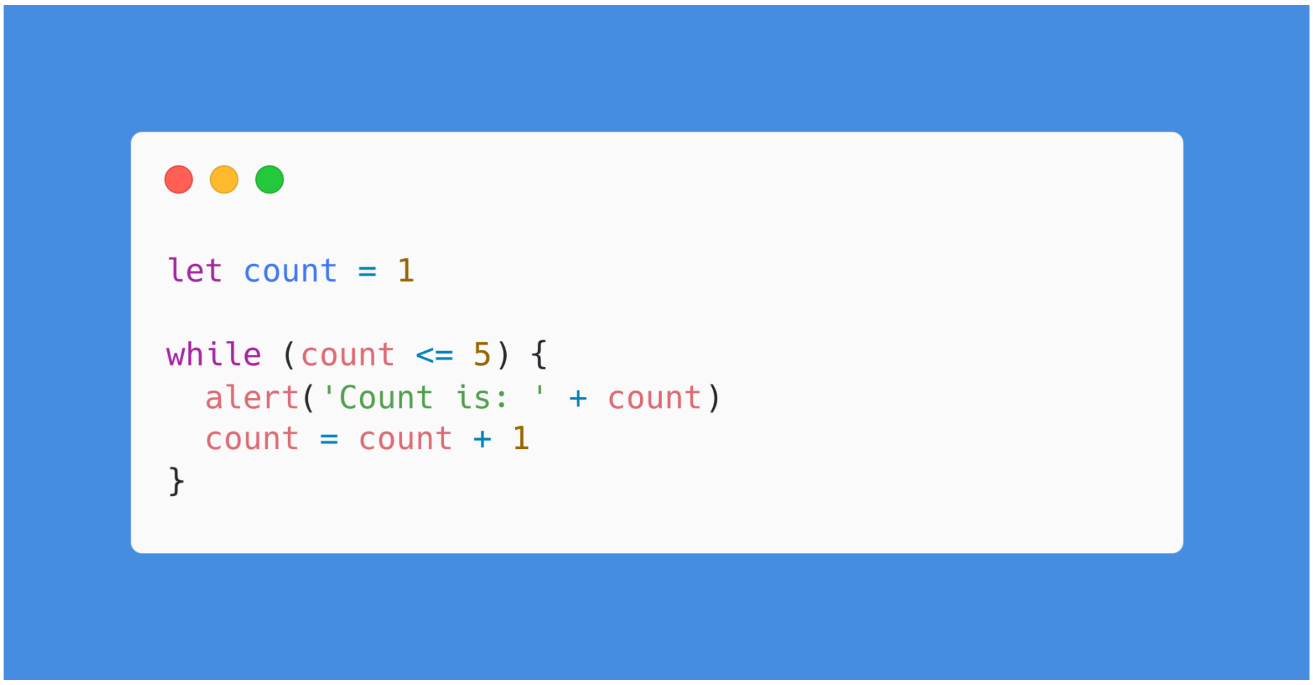
03

Iteration

Iteration with `while` loops

A `while` loop is a control structure that allows you to execute a block of code repeatedly as long as a specified condition is true. This is useful for performing tasks that require repetition until a certain condition is met.

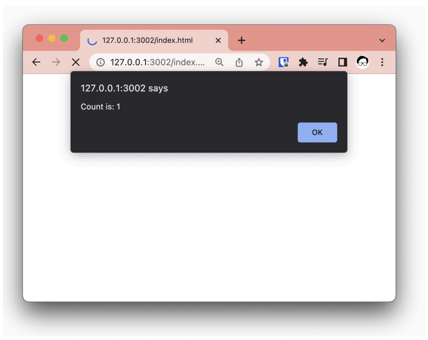
To create a while loop, you'll need to use the `while` keyword followed by a **condition** in brackets, and then a block of code inside curly brackets. The block of code will be executed repeatedly until the condition is no longer true. Here's an example:



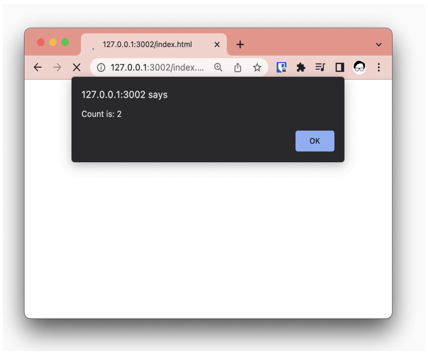
```
let count = 1

while (count <= 5) {
  alert('Count is: ' + count)
  count = count + 1
}
```

In this example, the loop will continue to execute as long as the count variable is less than or equal to 5. The output will be:

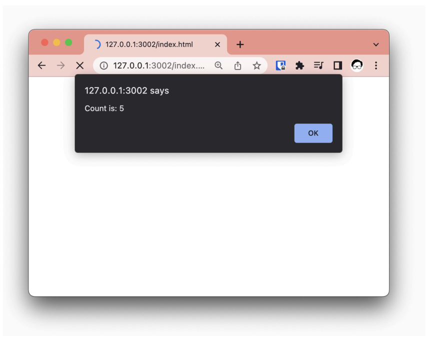


```
127.0.0.1:3002 says
Count is: 1
```



```
127.0.0.1:3002 says
Count is: 2
```

...



```
127.0.0.1:3002 says
Count is: 5
```

It's essential to ensure that the condition becomes false at some point, or else you'll create an infinite loop that will cause your program to crash. In our example, the `count = count + 1` statement inside the loop ensures that the value of `count` will increase with each iteration, eventually making the condition false.

By using while loops, you can easily repeat tasks in your code based on conditions, allowing for more dynamic and efficient programming.

Conditions in the brackets

Inside a while loop in JavaScript, we have a condition that decides whether the loop will keep running or not. This condition often compares two values using comparison operators like **==**, **!=**, **<**, **<=**, **>**, and **>=**. Here's what each one means:

== *(Equal To)*

This is like saying "Is this the same as that?" For example, 5 == 5 would be true because 5 is the same as 5.

!= *(Not Equal To)*

This is like saying "Is this different from that?" For example, 5 != 6 would be true because 5 is different from 6.

< *(Less Than)*

This is like saying "Is this smaller than that?" For example, 3 < 4 would be true because 3 is smaller than 4.

<= *(Less Than or Equal To)*

This is like saying "Is this smaller than or the same as that?" For example, 3 <= 3 would be true because 3 is the same as 3, and 3 <= 4 would also be true because 3 is smaller than 4.

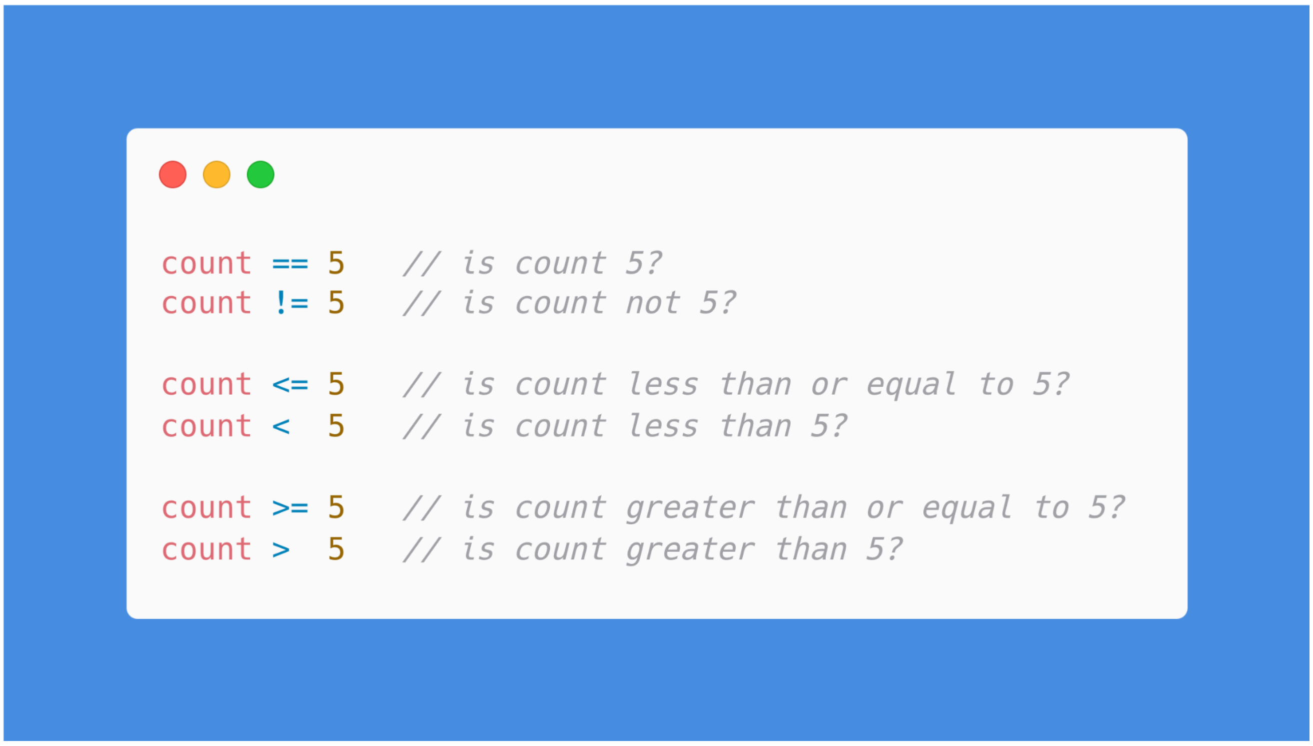
> *(Greater Than)*

This is like saying "Is this bigger than that?" For example, 5 > 4 would be true because 5 is bigger than 4.

>= *(Greater Than or Equal To)*

This is like saying "Is this bigger than or the same as that?" For example, 5 >= 5 would be true because 5 is the same as 5, and 5 >= 4 would also be true because 5 is bigger than 4.

So when we write a while loop, we use these comparisons to decide when to stop the loop. If the condition is true, the loop keeps going. If it's false, the loop stops.

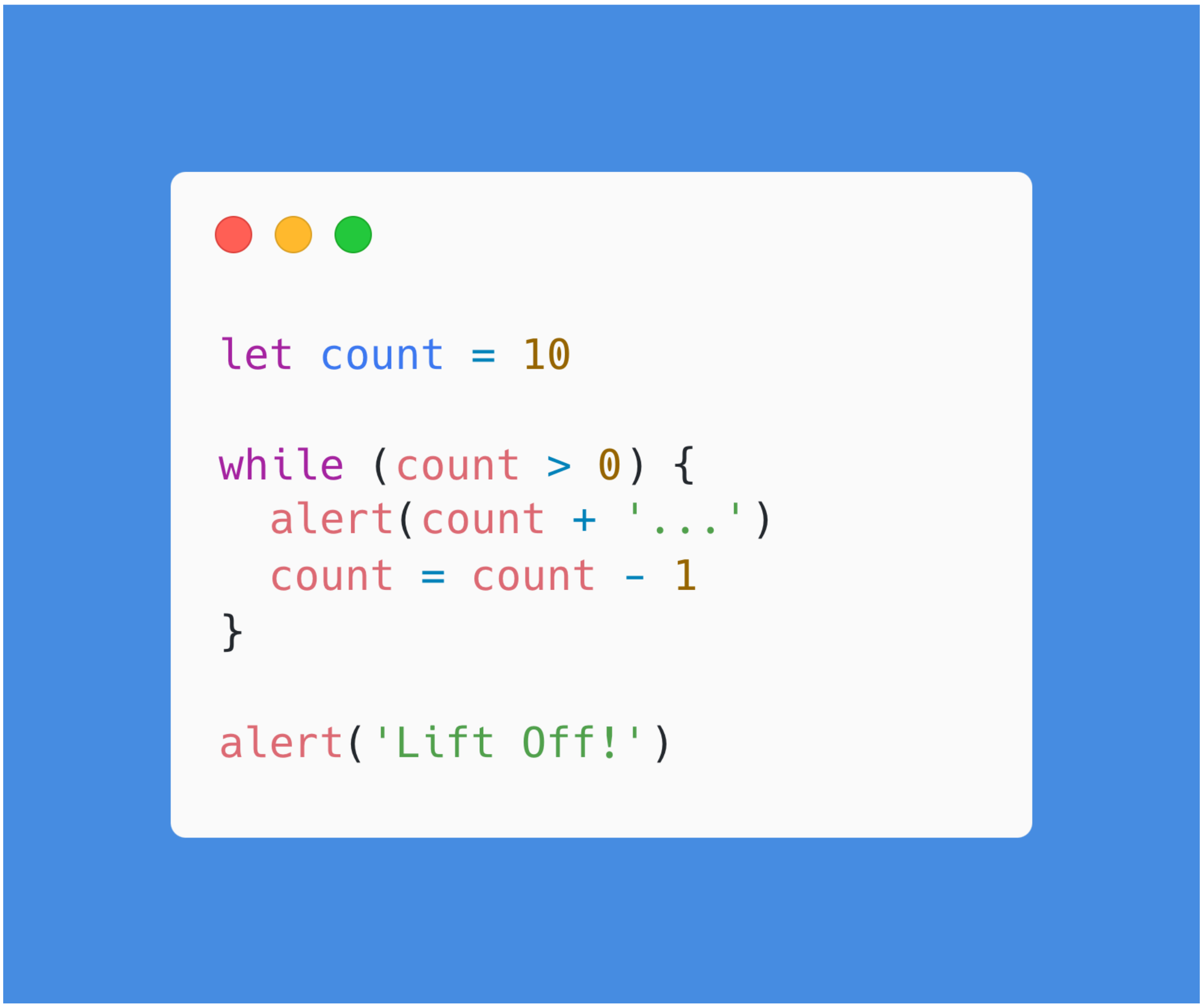


```
count == 5   // is count 5?
count != 5   // is count not 5?

count <= 5   // is count less than or equal to 5?
count <  5   // is count less than 5?

count >= 5   // is count greater than or equal to 5?
count >  5   // is count greater than 5?
```

🧑‍🎨 Lift Off



```
let count = 10

while (count > 0) {
  alert(count + '...')
  count = count - 1
}

alert('Lift Off!')
```

Predict

Read the code carefully. When you're ready, write what you predict it will do.

What do you think the code will do?

Run

Run the code, and say whether your prediction was right or not, and note any differences.

Did it do what you predicted?

✅ ❌

Differences

Investigate

1. Why are some of the lines indented a couple of spaces?

2. Why do you need an opening curly bracket `{` at the end of the `while` line?

3. Why do you need a closing curly bracket `}` on line 6?

4. What does `count > 0` do in this code?

5. Why is the last `alert` not inside the curly braces?

6. How many times will this loop run? How many times if we change the condition to `>=`?

7. What value will be stored in the `count` variable when the `while` loop ends?

Modify

Modify the program to ask the user the number to count down from.

Make

Write a program that starts at 0 and counts up to a maximum that the user enters.
It should count up in increments of another number the user puts in.

Example

```
Enter the maximum: 10
Enter the increment: 2
0
2
4
6
8
10
Done!
```

Ask another person to test your program.

Extension

Modify your program so that it tells the user whether there were any numbers left over.

Example

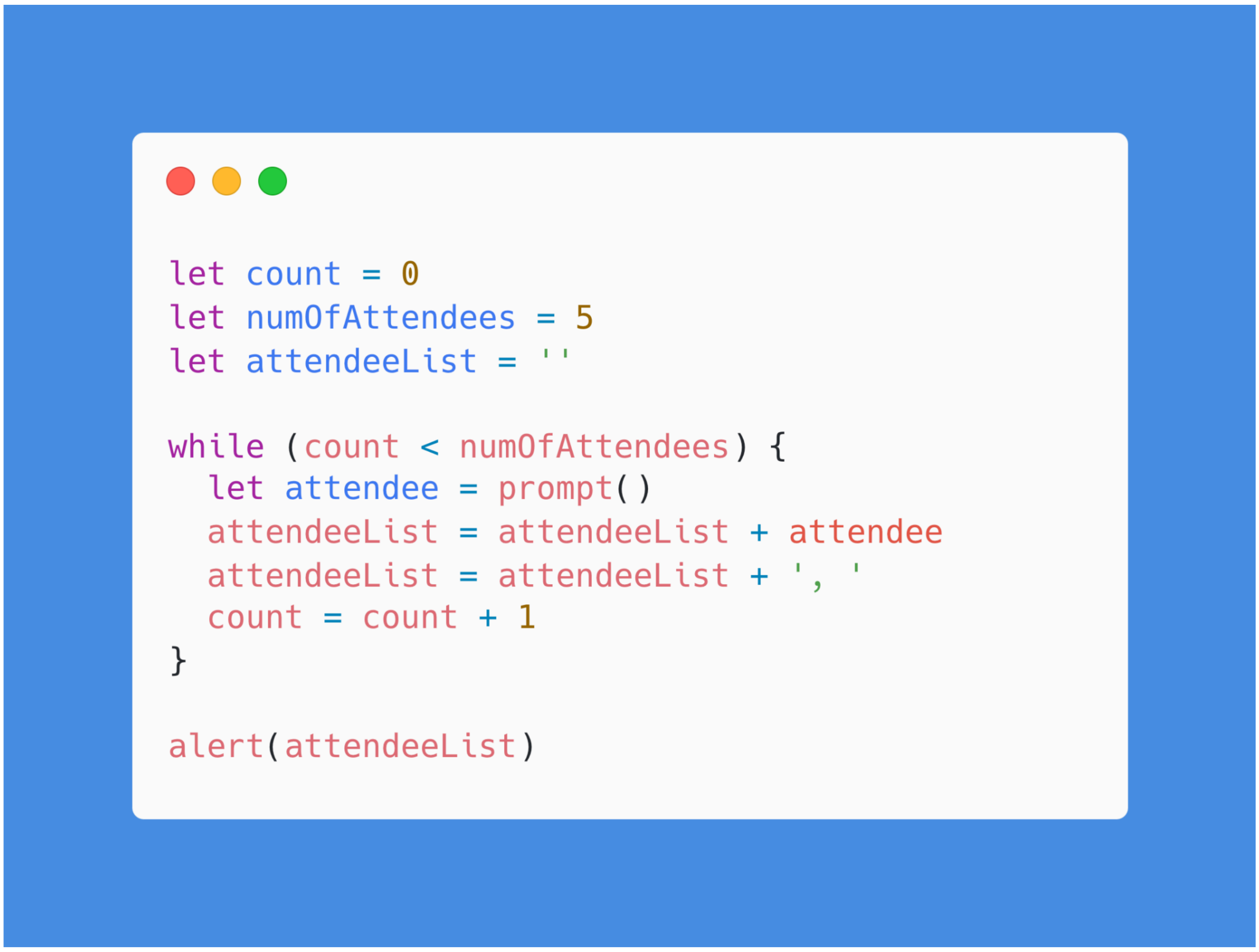
```
Enter the maximum: 10
Enter the increment: 3
0
3
6
9
Done, with 1 left over.
```

Matching lists

Draw a connection between the code on the left and the appropriate word on the right.

`let`	divide
`*`	output
`<`	equal
`-`	input
`prompt()`	less than
`>`	assignment
`=`	minus
`camelCase`	string
`/`	name
`!=`	multiply
`==`	concatenate
`alert()`	greater than
`while()`	not equal
`+`	loop
`` ` ` ``	variable

Attendees



```
let count = 0
let numOfAttendees = 5
let attendeeList = ''

while (count < numOfAttendees) {
  let attendee = prompt()
  attendeeList = attendeeList + attendee
  attendeeList = attendeeList + ', '
  count = count + 1
}

alert(attendeeList)
```

Predict

Read the code carefully. When you're ready, write what you predict it will do.

What do you think the code will do?

Run

Run the code, and say whether your prediction was right or not, and note any differences.

Did it do what you predicted?

✅ ❌

Differences

Investigate

1. Why is `count` set to `0` near the start? What would happen if that line wasn't there?

2. What would happen if `numOfAttendees` was set to `0` as well?

3. What does `<` do in this code?

4. What is the plus (`+`) being used for with the `attendeeList` variable?

Modify

1. Modify the program to ask the user to enter a name at the prompt.

2. Combine lines 12 and 13 into a single line that adds the attendee's name and a comma.

3. Add a prompt to get the number of attendees at the start of the program.

   Tip: you will need to convert the user's input to a number.

Make

4. Write a new program that uses the skills from the first one.

   Write a program that asks the user how many numbers they want to add together, and then asks them for the numbers, one at a time. At the end, your program should tell the user the sum of the numbers.

5. Ask another person to test your program.

Extension

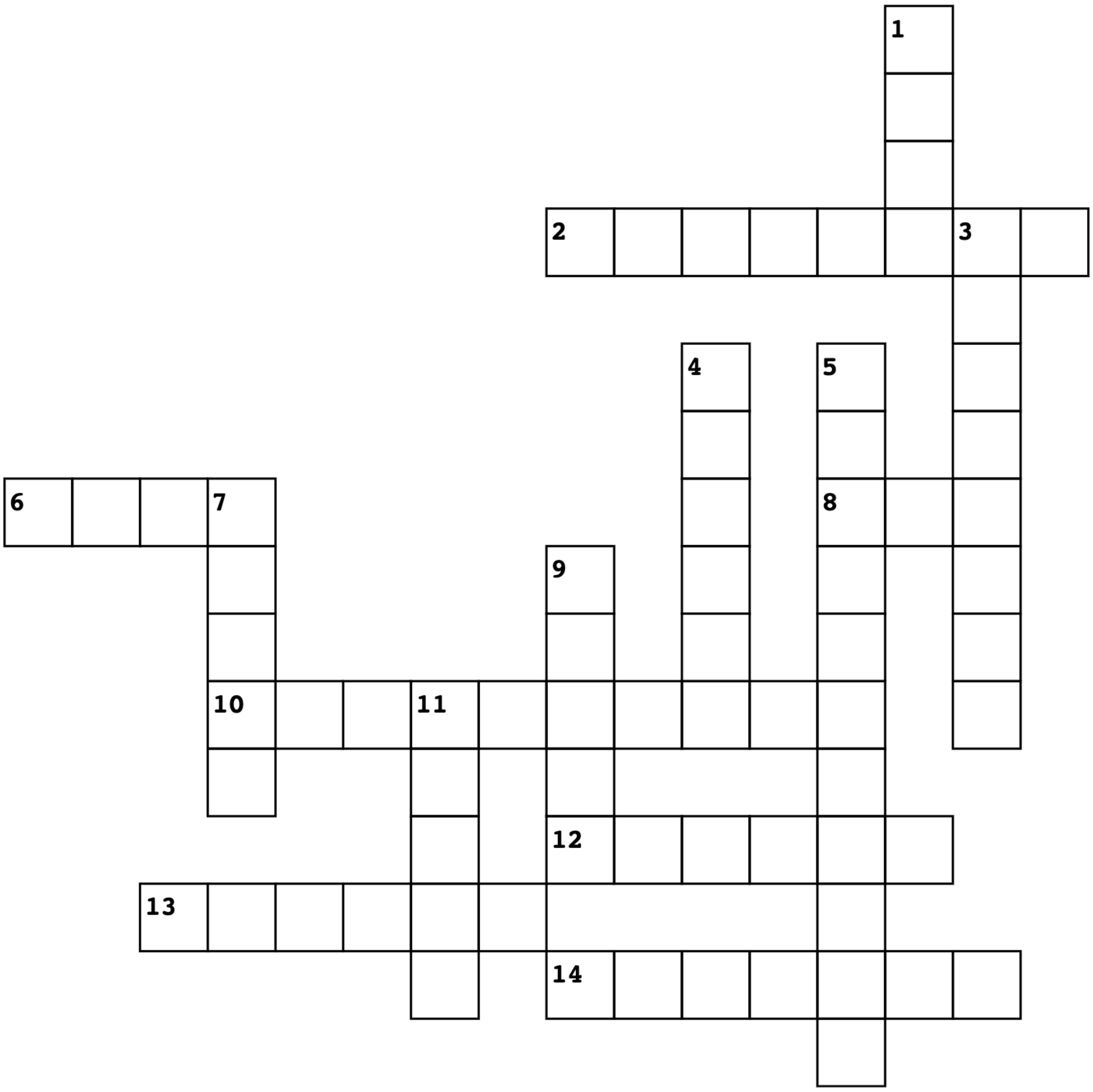
Modify the program to average the numbers after it has summed them.

Tip: you will need another variable to count how many numbers have been entered.

Summary

Term	What it means	How to do it
Iteration	When the program runs the same chunk of code repeatedly.	`while (num > 0) {` `alert(num)` `num = num - 1` `}`
Condition	The test inside a `while` statement.	`answer == 'Paris'   // test equality` `answer == 100       // test equality` `answer > 100        // test greater than` `answer < 100        // test less than` `answer >= 100 // test greater than or equal` `answer <= 100 // test less than or equal` `answer != 100       // test not equal`

Crossword



Across

2. *

6. camelCase

8. equal !=

10. =

12. ''

13. alert()

14. than >

Down

1. while()

3. <

4. /

5. +

7. ==

9. -

11. prompt()

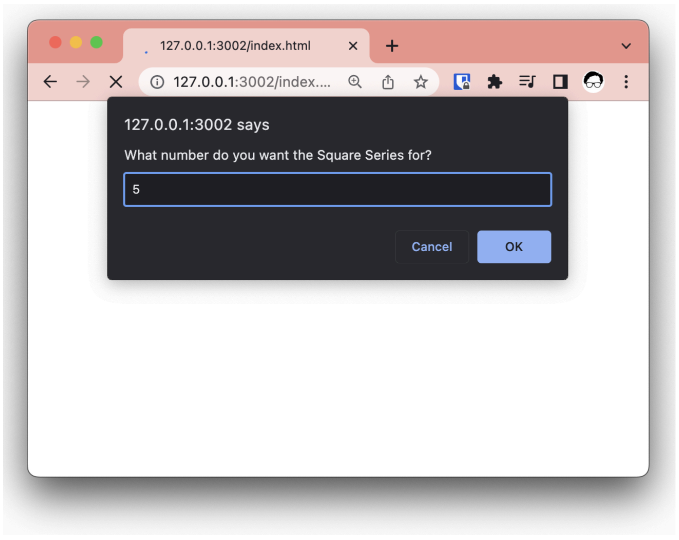
🧑‍🎨 Square Series

A square series is the sum of the squares of all the whole numbers up to the given number.

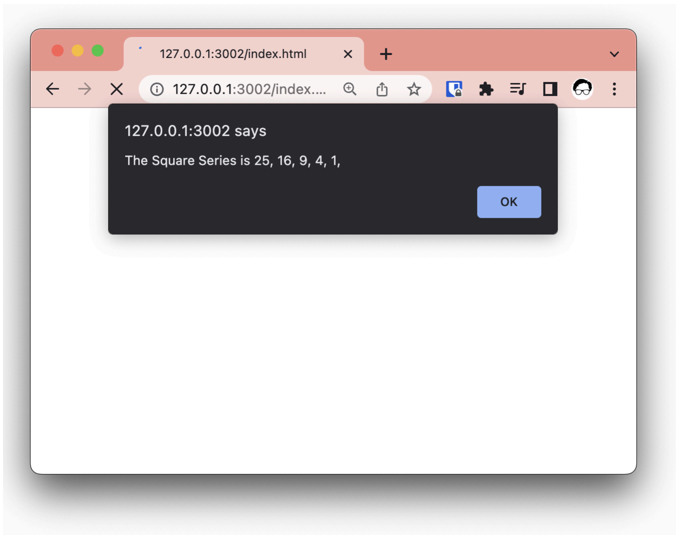
Task

Create a program that asks the user for a number,
and then outputs the square series up to that number.

Example output



```
127.0.0.1:3002 says
What number do you want the Square Series for?
5
```



```
127.0.0.1:3002 says
The Square Series is 25, 16, 9, 4, 1,
```

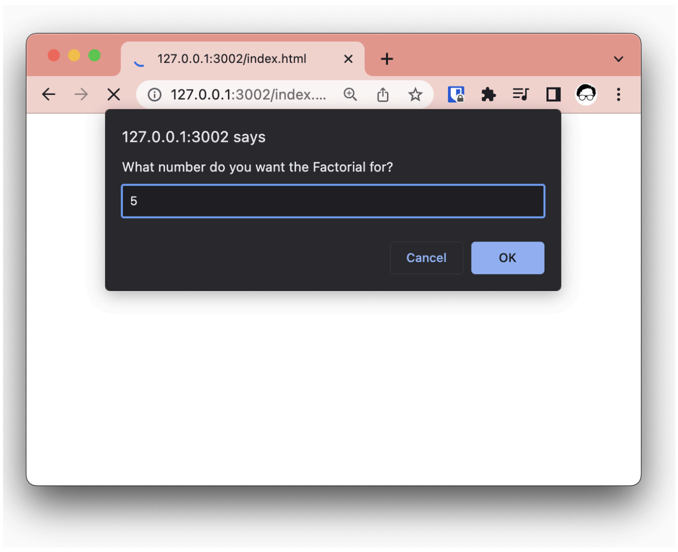
🧑‍🎨 Factorial

The factorial is every whole number multiplied together up to a given number.

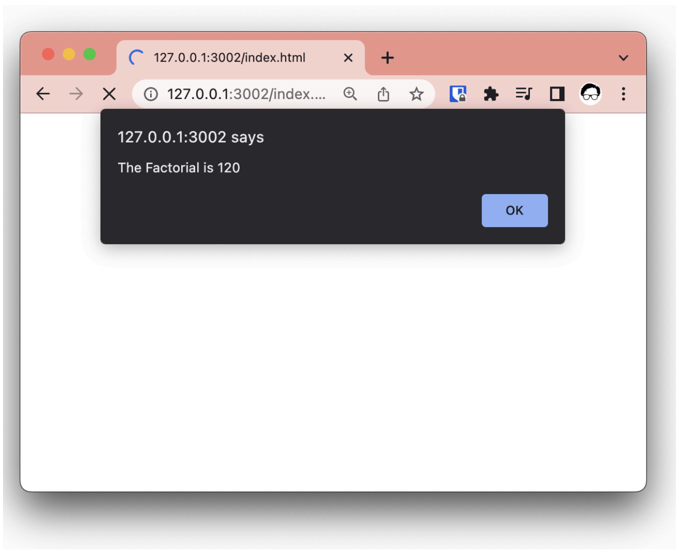
Task

Create a program that asks the user for a number and then gives them the factorial.

Example output



```
127.0.0.1:3002 says
What number do you want the Factorial for?
5
```



```
127.0.0.1:3002 says
The Factorial is 120
```

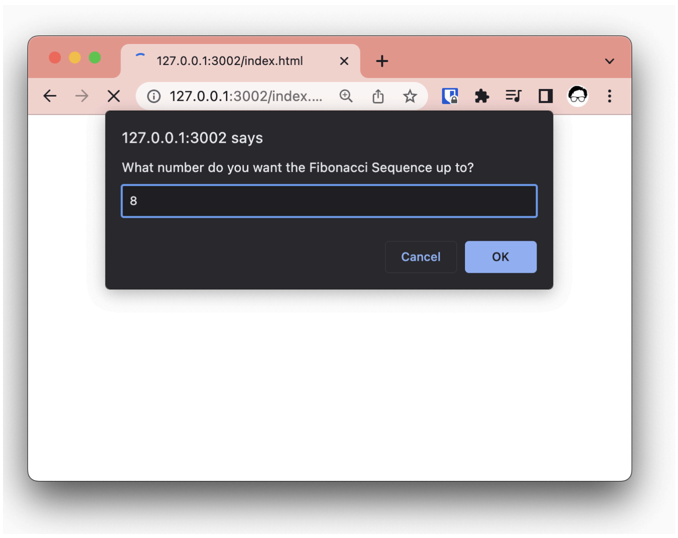
🧑‍🎨 Fibonacci

Fibonacci was an Italian mathematician who came up with a series that also appears in nature, for example, the number of petals on a flower, how branches in a tree spread, the shape of storms and shells, and even galaxies.

Task

Create a program that asks the user for a number and then gives them the Fibonacci sequence up to that number.

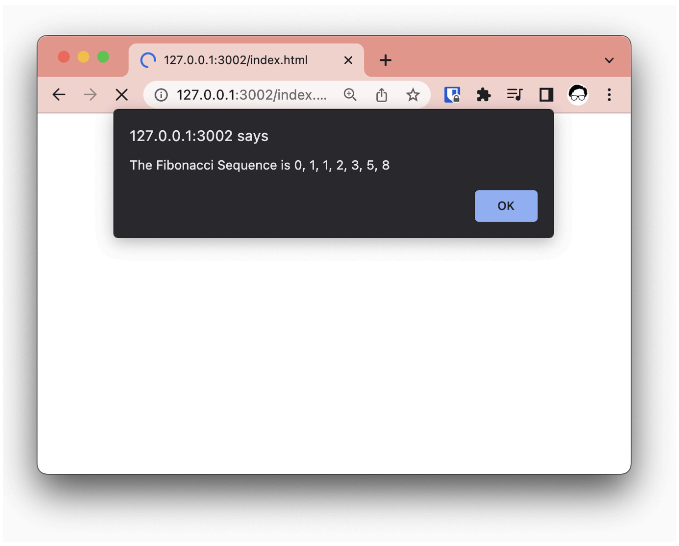
Example output



127.0.0.1:3002 says

What number do you want the Fibonacci Sequence up to?

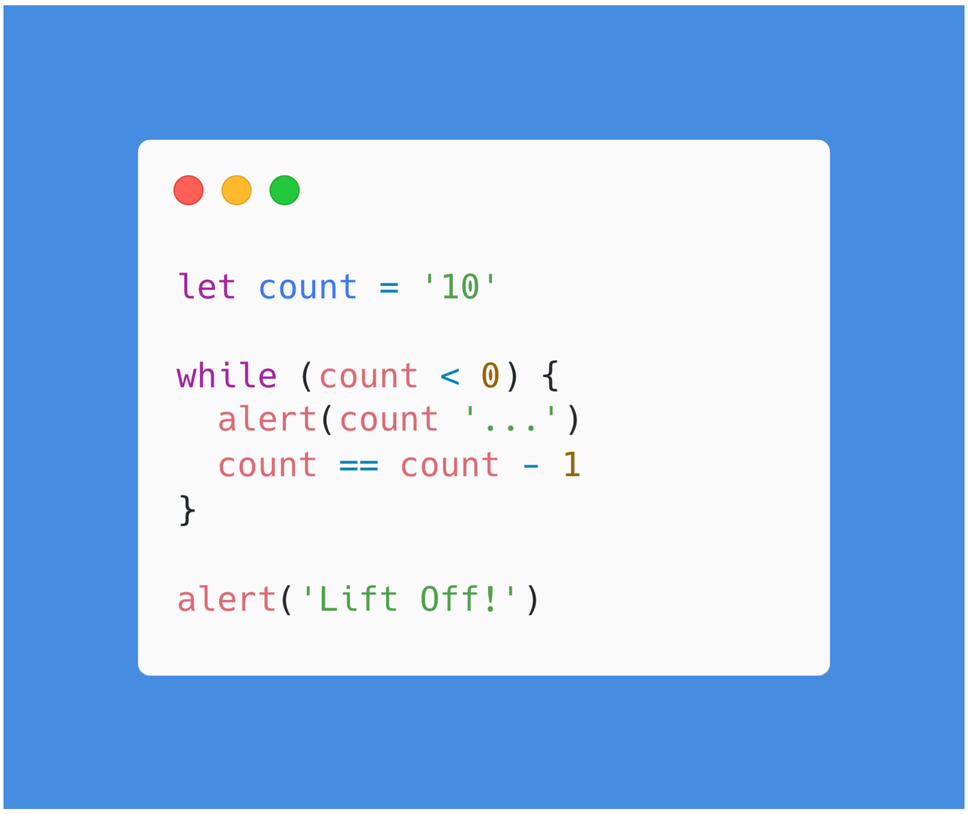
8



127.0.0.1:3002 says

The Fibonacci Sequence is 0, 1, 1, 2, 3, 5, 8

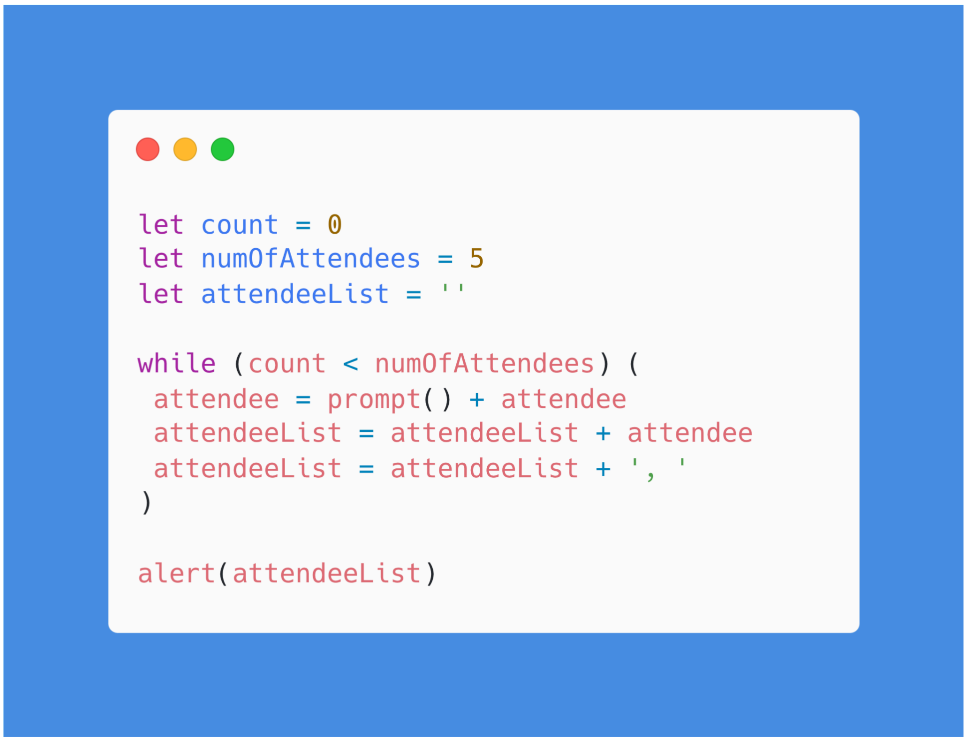
🧑‍🎨 Debugging



```
let count = '10'

while (count < 0) {
  alert(count '...')
  count == count - 1
}

alert('Lift Off!')
```



```
let count = 0
let numOfAttendees = 5
let attendeeList = ''

while (count < numOfAttendees) (
 attendee = prompt() + attendee
 attendeeList = attendeeList + attendee
 attendeeList = attendeeList + ', '
)

alert(attendeeList)
```