Functions:

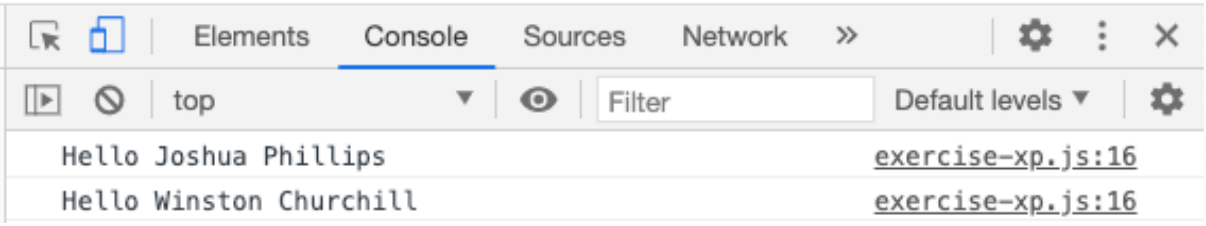
A **function** in **JavaScript** is similar to a procedure—a set of statements that performs a task or calculates a value, but for a procedure to qualify as a **function**, it should take some input and return an output where there is some obvious relationship between the input and the output.

```
function greet (name, lastName) {
    console.log('Hello ' + name + ' ' + lastName);
}
greet('Joshua', 'Phillips');
greet('Winston', Churchill);
```

What the code means:

greet	Function Name
( )	Commands within the function.
Console.log	Printing how you want the function to display.
Greet (“ “, “ “)	Function name plus data input.

Function Answer in console:



If Statements:

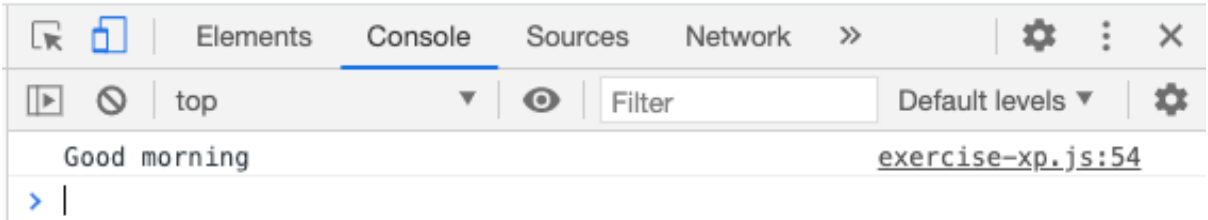
The **if/else statement** executes a block of code **if** a specified condition is true. **If** the condition is false, another block of code can be executed. The **if/else statement** is a part of **JavaScript's "Conditional" Statements**, which are used to perform different actions based on different conditions.

```
//Hour
// if hour is between 6am-12pm: Good morning!
// if it is between 12pm-6pm: Good afternoon!
// Otherwise: Good Evening!

let hour = 10;

if (hour >= 6 && hour < 12)
    console.log('Good morning');
else if (hour >= 12 && hour < 18)
    console.log('Good afternoon');
else
    console.log('Good evening');
```

What the code means:

hour	The variable	You decide a name
10	The data	(this can/will change)
If ---- else if ---- else	True or false statement	(will select an option)
(hour >= 6 && hour < 12)	The conditions for **if/ else if/ else**	(question)
Console.log('Good Morning');	The printed statement	(answer)

Answer in console: (Because the variable (hour) is 10;)



Loops:

For loops:

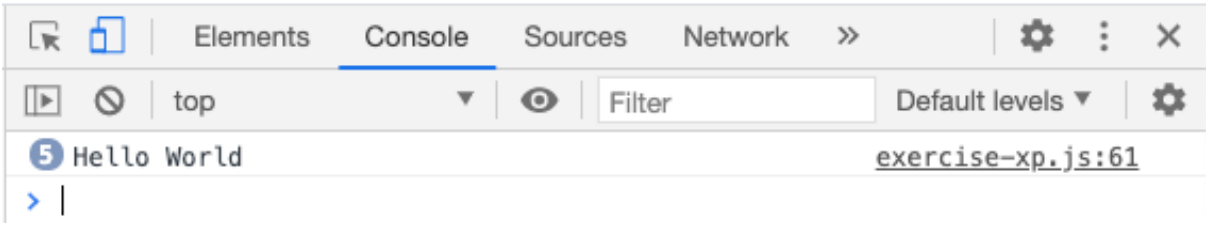
The for **statement** creates a **loop** that is executed as long as a condition is true.
The **loop** will continue to run as long as the condition is true. It will only stop when the condition becomes false. **JavaScript** supports different kinds of **loops**: ... for/in
- **loops** through the properties of an object.

```
for (let i = 0; i < 5; i++) {
    console.log('Hello World');
}
```

What the code means:

for	Type of loop	
i = 0; i < 5; i ++	The variable	
I = 0	Index starts at 0	**You must have a start**
I < 5	Index ends at 5	**You must have an end**
I ++	Adds 1 number	
Console.log('Hello World')	Printed 5 times	

Answer in the console: (Hello world is printed 5 times)



While Loops:

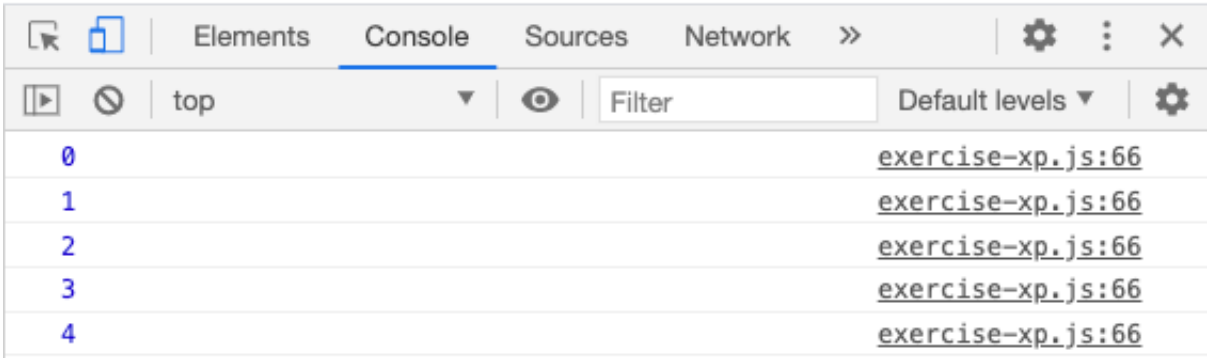
The **while statement** creates a **loop** that is executed **while** a specified condition is true. The **loop** will continue to run as long as the condition is true. It will only stop **when** the condition becomes false. **JavaScript** supports different kinds of **loops**: for - **loops** through a block of code a number of times.

```
let i = 0;
while (i < 5) {
    console.log(i);
    i++
}
```

What the code means:

i = 0	Variable	Outside the loop (starting number)
while	The loop	
i< 5	Index ends at 5	
Console.log(i);	Print i	Printing the variable
I ++	Adds 1 number	At the end

Answer in the console:



---

forEach Loops (Used with arrays):

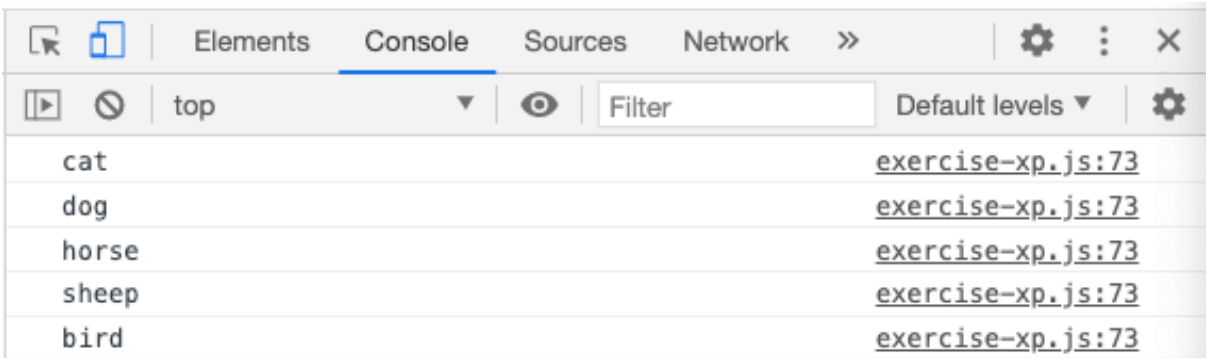
The **forEach**() method calls a function once **for each** element in an **array**, in order.
Note: the function is not executed for **array** elements without values.

```
const animals = ['cat', 'dog', 'horse', 'sheep', 'bird'];

animals.forEach(newVariable => {
    console.log(newVariable);
});
```

What the code means:

animals	The variable	
[“ “, “ “, “ “]	The array	
Animals.**forEach**	Variable & the loop	forEach loop
(newVariable =>{	Name of the new variable	Inside of the loop
Console.log(newVariable)	Print the new variable	

Answer in console: (loops the array into once for each element)



Array

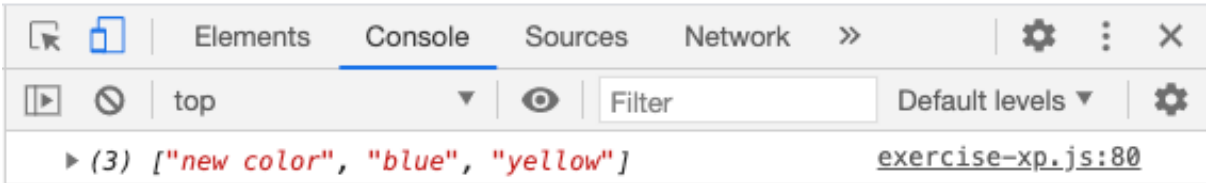
An array can hold many values under a single name, and you can access the values by referring to an index number. For example [0] or [300].

```
let selectedColors = ['red', 'blue', 'yellow'];
selectedColors[0] = 'new color';
console.log(selectedColors);
```

What the code means:

selectedColors	The variable
[“ “, “ “, “ “]	The array
selectedColors[0]	Position 1 in the array
selectedColors[0] = ‘new color’;	Replacing position 1 or [0] with ‘new color’
Console.log(selectedColors);	Print the adjustments to the array variable

Answer in the console: selectedColors[0] = ‘new color’; will replace red or [0].



Array methods:

Push()	add a new element to the end of an array
Pop()	removes the last element from an array
Splice()	add new items to an array/ remove items
Slice()	slices out a piece of an array into a new array.
toString()	converts an array to a string of (comma separated) array value.
Join()	join all the elements of an array
Shift()	To remove the first element
Unshift({item})	will add a new element to the head of the list. It will become index 0

Comparisons

Symbol	Meaning
=	is assignment
==	is comparison of value
===	is comparison of value and type

Symbol	Meaning
!=	Not equal
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
||	Or
&&	And
!	Not (if x is true, then x! is false)

Working with numbers

Operator	Example	Same As
++	x++	x = x + 1
--	x--	x = x - 1
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y