ReDISchool

Week 6

Logic & Control Flow

Conditions, comparisons and operators

Agenda

Week 6

Take Home Recap

General Feedback

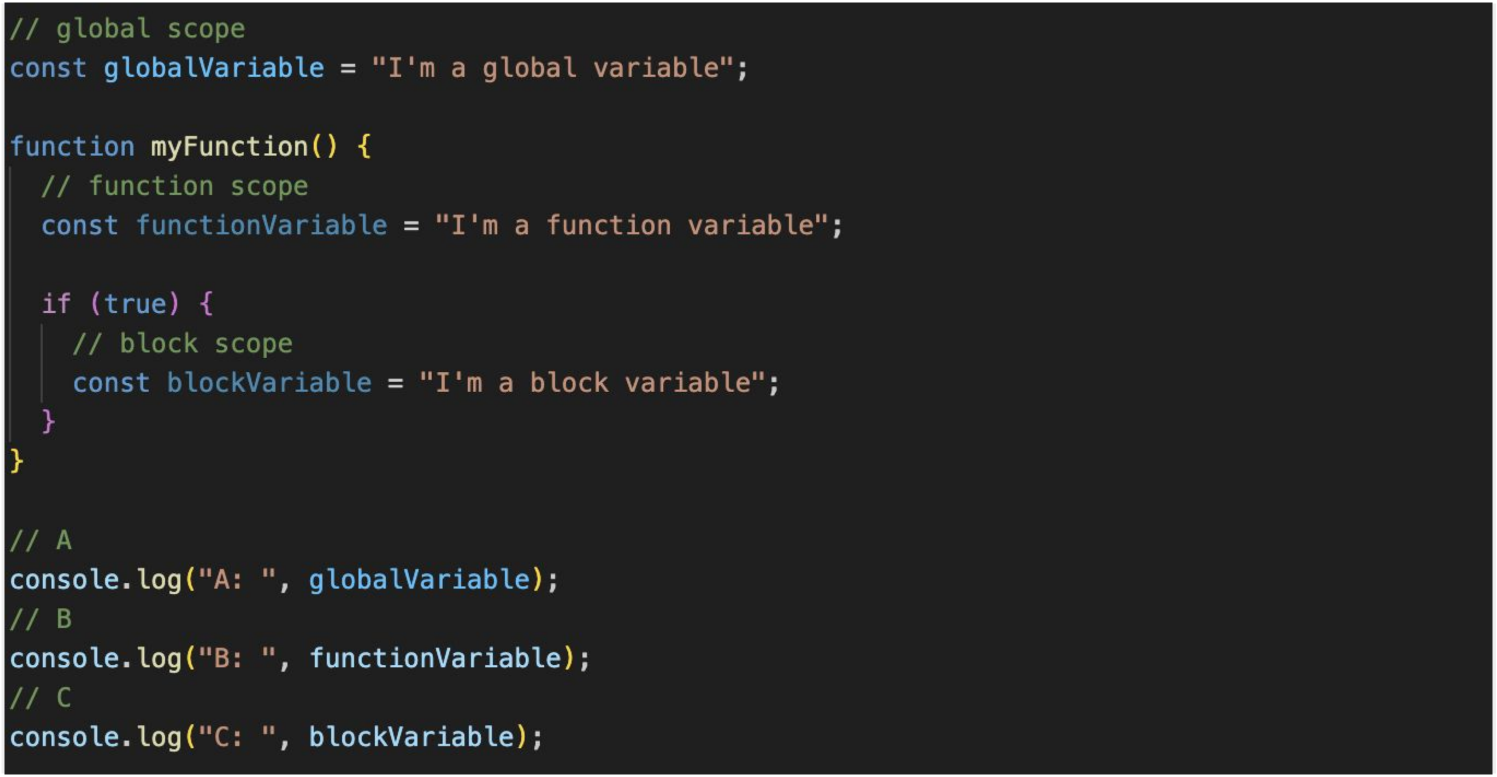
10 mins for questions and help

Week 6

Scope

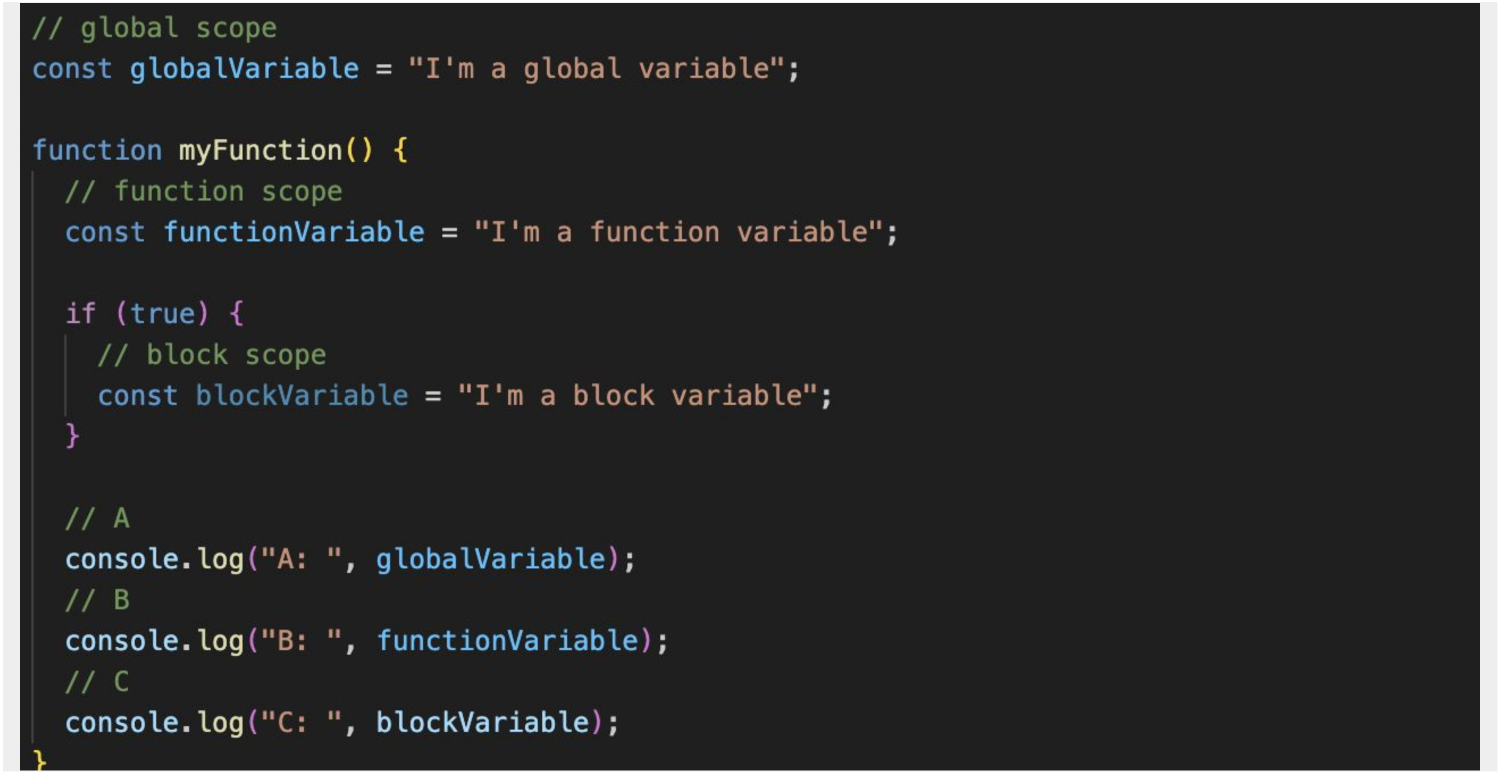
TURN TO CLEAR VISION
QUARTERS ONLY
U.S. Manufactured and Distributed by
THE TOWER OPTICAL COMPANY, INC.
Norwalk, CT
www.toweropticalco.com
50¢
TO OPERATE TURN HANDLE ONE FULL TURN
2254

What will print?



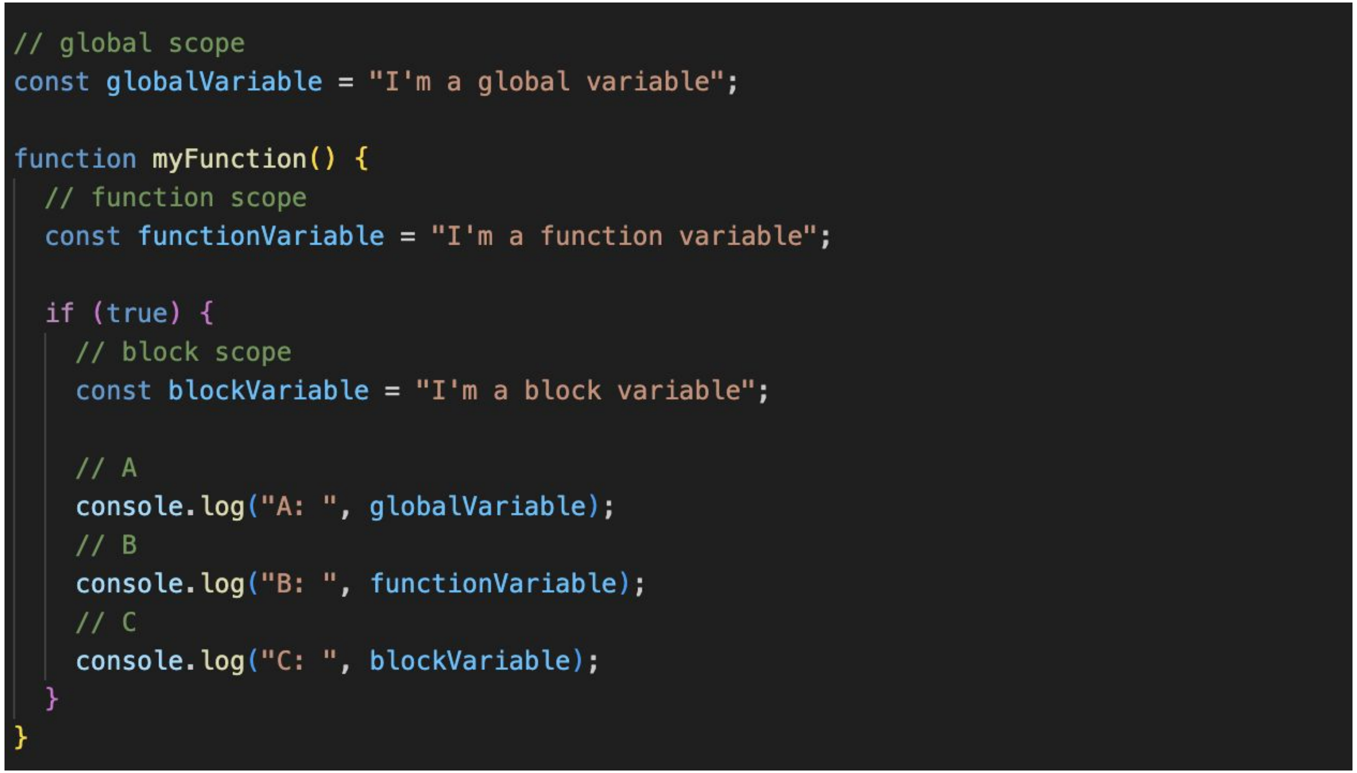
```javascript
// global scope
const globalVariable = "I'm a global variable";

function myFunction() {
  // function scope
  const functionVariable = "I'm a function variable";

  if (true) {
    // block scope
    const blockVariable = "I'm a block variable";
  }
}

// A
console.log("A: ", globalVariable);
// B
console.log("B: ", functionVariable);
// C
console.log("C: ", blockVariable);
```

What will print?



```
// global scope
const globalVariable = "I'm a global variable";

function myFunction() {
  // function scope
  const functionVariable = "I'm a function variable";

  if (true) {
    // block scope
    const blockVariable = "I'm a block variable";
  }

  // A
  console.log("A: ", globalVariable);
  // B
  console.log("B: ", functionVariable);
  // C
  console.log("C: ", blockVariable);
}
```

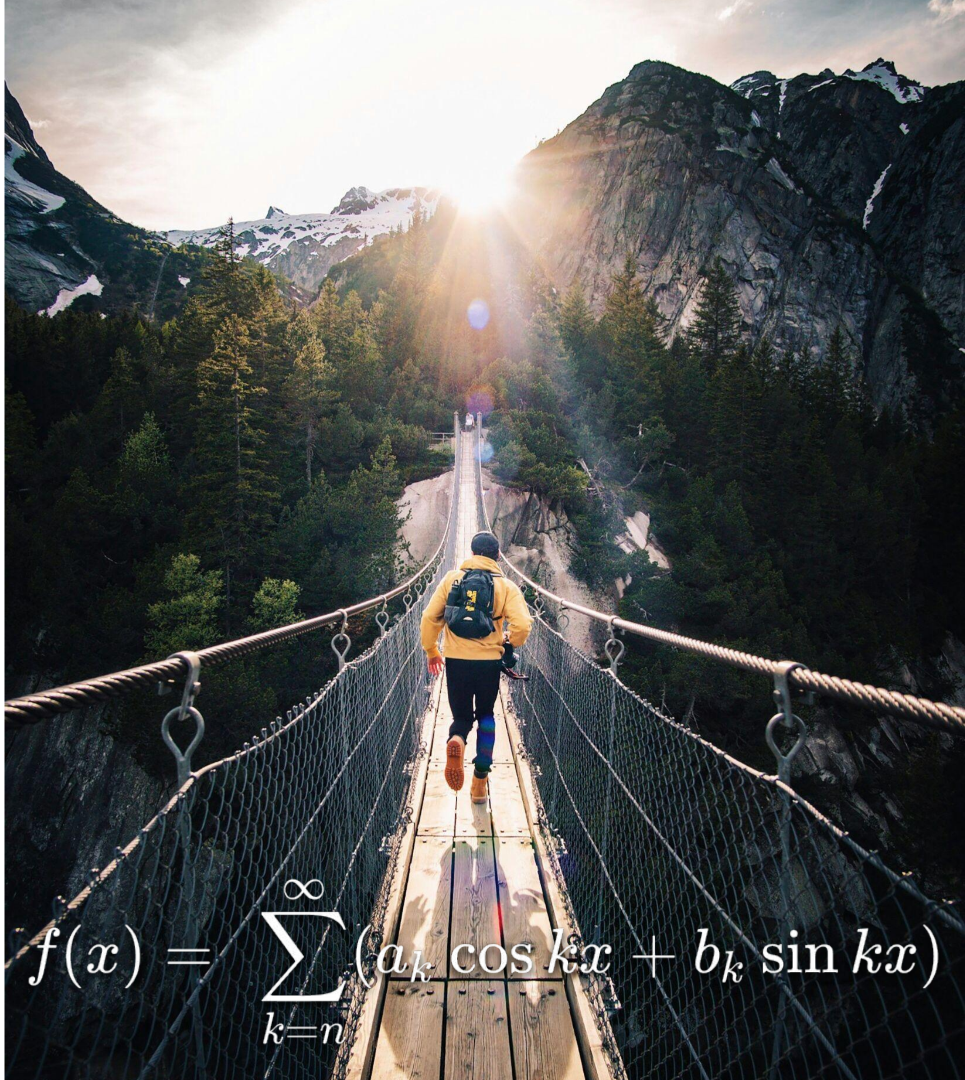
What will print?



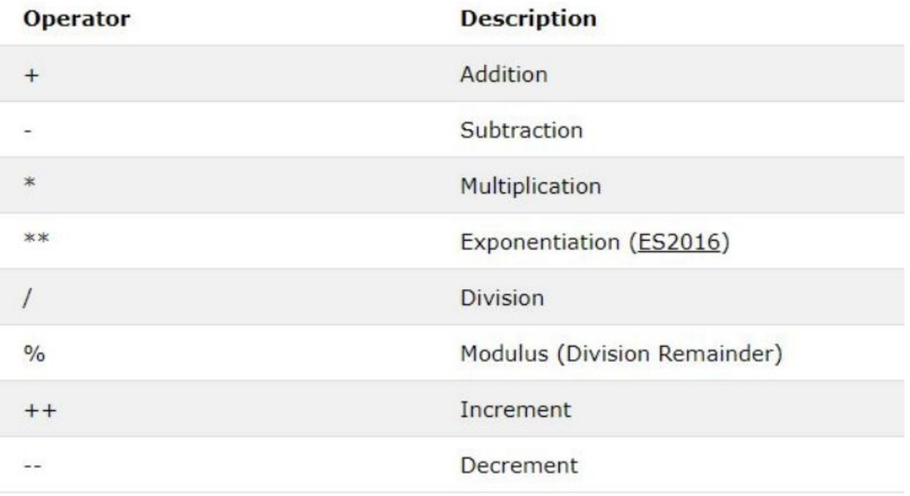
```
// global scope
const globalVariable = "I'm a global variable";

function myFunction() {
  // function scope
  const functionVariable = "I'm a function variable";

  if (true) {
    // block scope
    const blockVariable = "I'm a block variable";

    // A
    console.log("A: ", globalVariable);
    // B
    console.log("B: ", functionVariable);
    // C
    console.log("C: ", blockVariable);
  }
}
```

Week 6

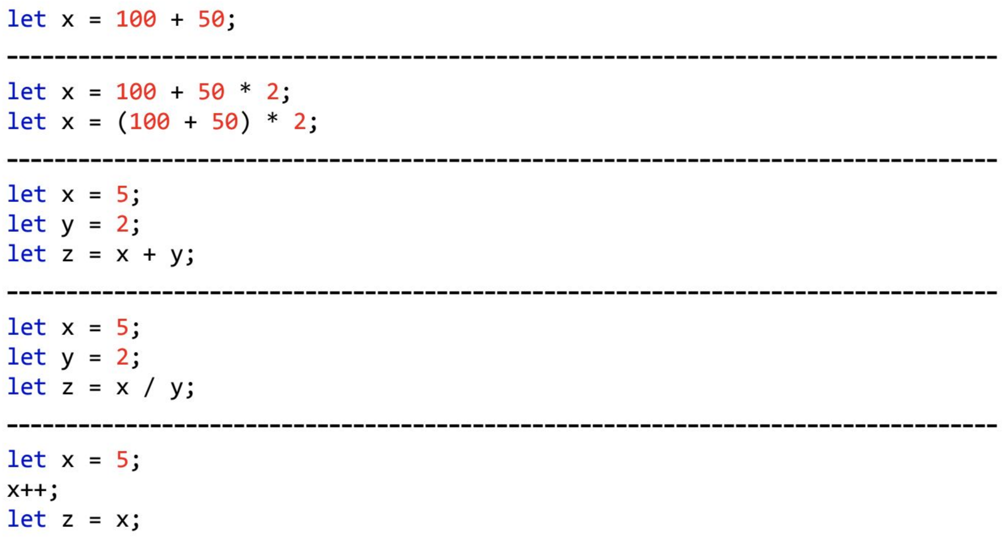
Oh no … Math 😱



Arithmetic Operators



Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
/	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement



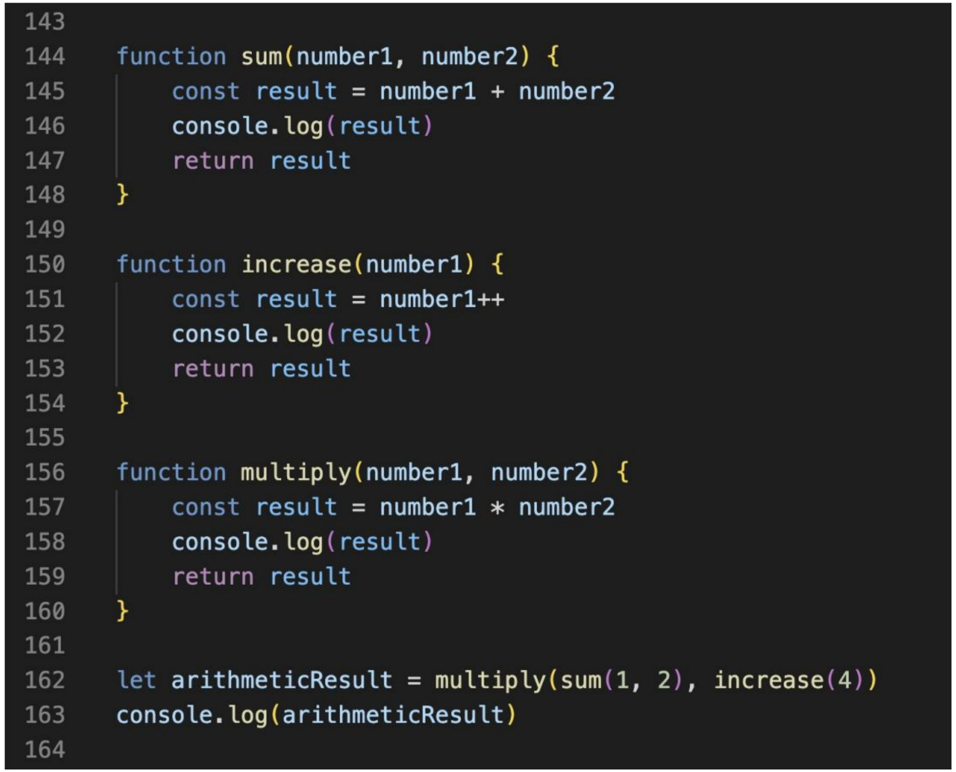
```
let x = 100 + 50;
--------------------------------------------------------------------------
let x = 100 + 50 * 2;
let x = (100 + 50) * 2;
--------------------------------------------------------------------------
let x = 5;
let y = 2;
let z = x + y;
--------------------------------------------------------------------------
let x = 5;
let y = 2;
let z = x / y;
--------------------------------------------------------------------------
let x = 5;
x++;
let z = x;
```

Exercise (10 min)

Speak to the person next to you and work together to:

1. Create a function that adds two numbers and returns the result. Add a console.log that prints the output outside of the function.
2. Create another function that increments the given parameter by 1 and return it. Add a console.log that prints the output outside of the function.
3. Create a third function that multiplies two numbers and return it. Add a console.log that prints the output outside of the function.
4. Use the multiplication function passing by parameter the result of the two previous functions with any values. Add a console.log that prints the output outside of the function.

Exercise - Solution



```
function sum(number1, number2) {
    const result = number1 + number2
    console.log(result)
    return result
}

function increase(number1) {
    const result = number1++
    console.log(result)
    return result
}

function multiply(number1, number2) {
    const result = number1 * number2
    console.log(result)
    return result
}

let arithmeticResult = multiply(sum(1, 2), increase(4))
console.log(arithmeticResult)
```

ReDISchool

Break

10 min

Week 6

Comparison

Comparison

Comparison operators in JavaScript compare **two values** (such as numbers or strings) to check if they are **equal**, **greater than**, or **less than** each other.

These comparisons return a boolean result: true if the condition is met and false if not.

Comparison

Greater than: a > b
Less Than: a < b

Greater or equal: a >= b,
Less Than or equal: a <= b.

Strict equality: a === b - strict equality
Equality: a == b - “normal” equality (BE CAREFUL!)

Strict inequality: a !== b - strict equality
Inequality: a != b - “normal” inequality (BE CAREFUL!)

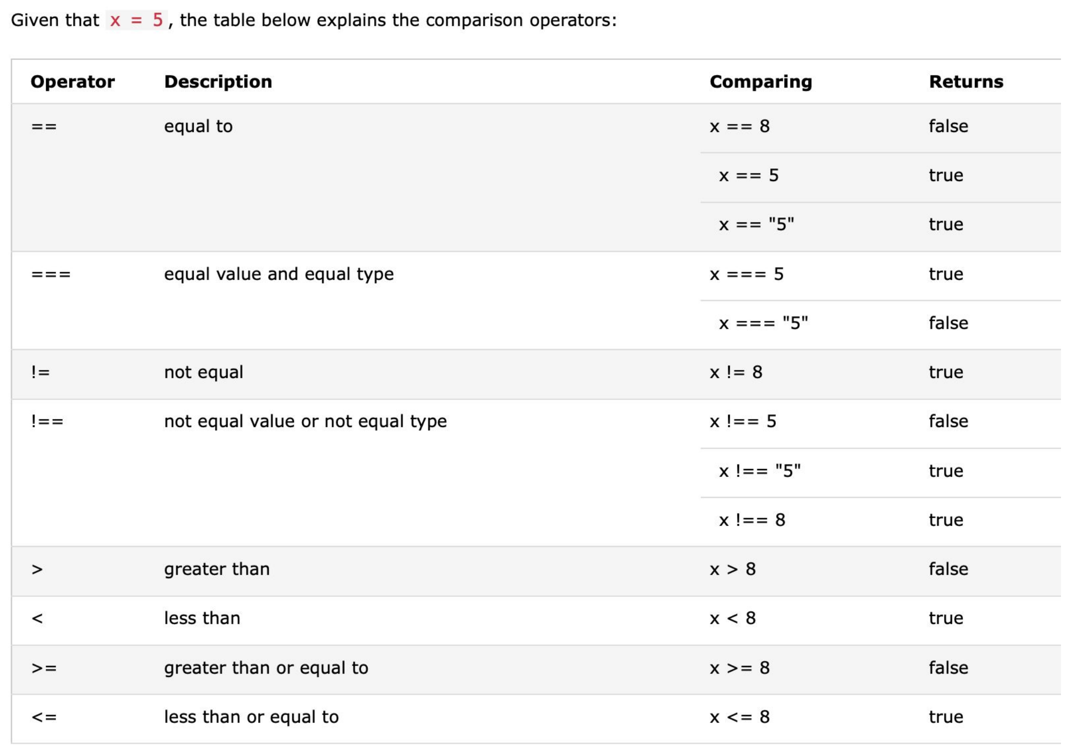
⚠️⚠️⚠️

a = b

is an Assignment, not a comparison!

⚠️⚠️⚠️

Comparison



Given that `x = 5`, the table below explains the comparison operators:

Operator	Description	Comparing	Returns
==	equal to	x == 8	false
		x == 5	true
		x == "5"	true
===	equal value and equal type	x === 5	true
		x === "5"	false
!=	not equal	x != 8	true
!==	not equal value or not equal type	x !== 5	false
		x !== "5"	true
		x !== 8	true
>	greater than	x > 8	false
<	less than	x < 8	true
>=	greater than or equal to	x >= 8	false
<=	less than or equal to	x <= 8	true

Examples

```
let  x  = 100 > 50;
```

```
let  x  = (100 + 50) < 10;
```

```
let  x  = 5;
let  y  = 2;
let  z  = x === y;
```

```
let  x  = 5;
let  y  = 2;
let  z  = x != y;
```

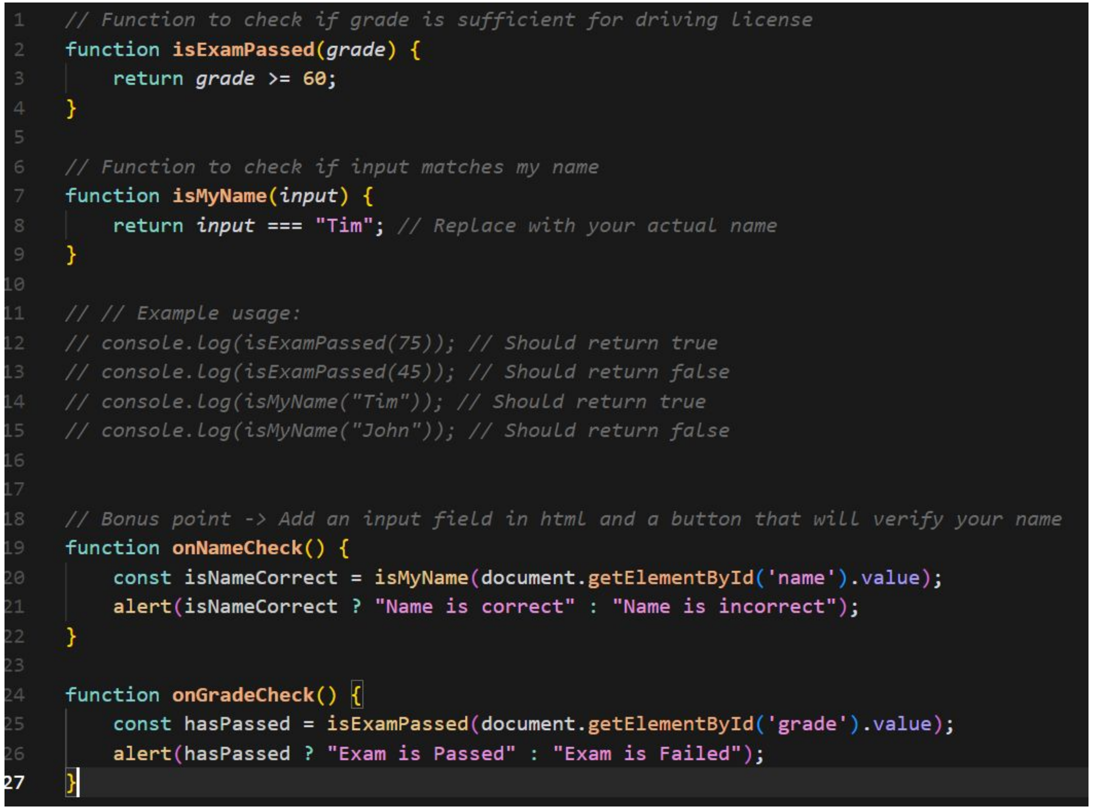
Exercise (10min)

1. Let's create a simple function that returns a boolean result. This function will be the one that decides whether we are going to pass the driving license exam.
   The function will receive one parameter which will be the grade. The function will return a boolean that evaluates if the grade is greater or equal than 60.

2. Now create another function that checks if the variable received is your name or not.

   Bonus point → try to get the input from the user for the function that checks if it is your name

Comparison operators: exercise answer



```javascript
// Function to check if grade is sufficient for driving license
function isExamPassed(grade) {
    return grade >= 60;
}

// Function to check if input matches my name
function isMyName(input) {
    return input === "Tim"; // Replace with your actual name
}

// // Example usage:
// console.log(isExamPassed(75)); // Should return true
// console.log(isExamPassed(45)); // Should return false
// console.log(isMyName("Tim")); // Should return true
// console.log(isMyName("John")); // Should return false


// Bonus point -> Add an input field in html and a button that will verify your name
function onNameCheck() {
    const isNameCorrect = isMyName(document.getElementById('name').value);
    alert(isNameCorrect ? "Name is correct" : "Name is incorrect");
}

function onGradeCheck() {
    const hasPassed = isExamPassed(document.getElementById('grade').value);
    alert(hasPassed ? "Exam is Passed" : "Exam is Failed");
}
```

Logical operators

Logical operators are used to determine the logic between variables or values. They will return a Boolean, just as the comparison operators. They come in really handy to create more complex conditions when used with comparison operators. The variables used in the operators are converted into Booleans.

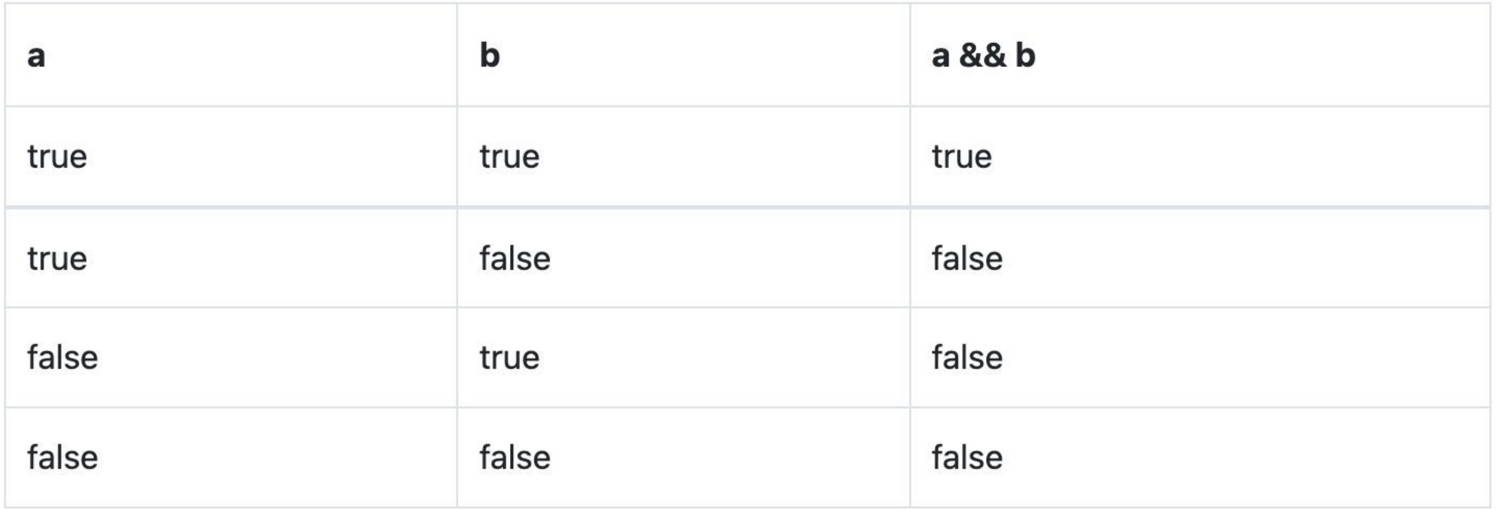
There are 3:

AND - &&
The result of the && operator is true only if both values are true, otherwise, it is false.

OR - II
The II operator returns false if both values evaluate to false. In case either value is true, the II operator returns true.

NOT - !
The ! operator can be applied to a single value of any type, not just a Boolean value. When you apply the ! operator to a boolean value, the ! returns true if the value is false and vice versa.

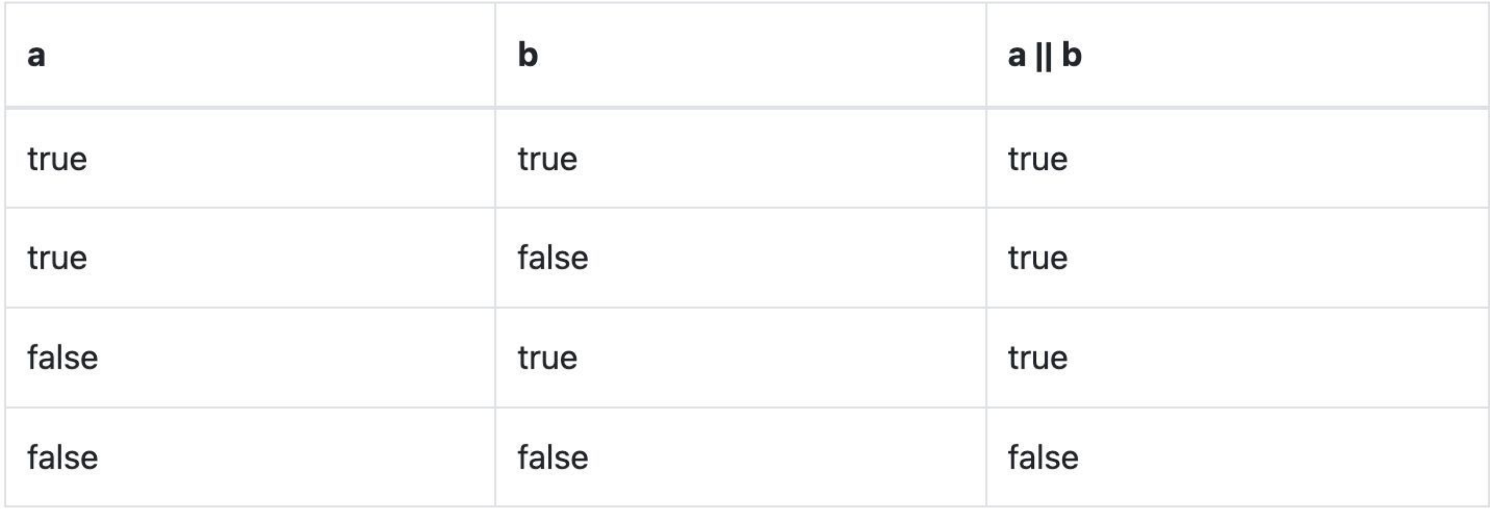
Logical operators - AND



a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

https://javascript.info/logical-operators
Logical operators are used to determine the logic between variables or values.

Logical operators - OR



a	b	a || b
true	true	true
true	false	true
false	true	true
false	false	false

https://javascript.info/logical-operators
Logical operators are used to determine the logic between variables or values.

Logical operators - examples

```
var  x  =  7;
var  y  =  2;
var  z  =  x  <  10   &&  y  >  1; // true
```

```
var  x  =  6;
var  y  =  3;
var  z  =  x  ===  5  ||  y  ===  5; // false
```

```
var  x  =  1;
var  y  =  3;
var  z  =  !(x  ===  y);
                     //
                     true
```

https://javascript.info/logical-operators
Logical operators are used to determine the logic between variables or values.

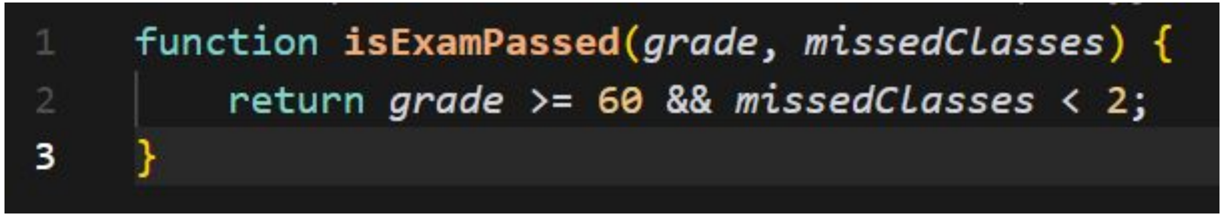
Exercise (5 min)

1. Let's create a stricter function for approving the drivers license. We already had that the grade should be greater or equal to 60. How about adding that the quantity of classes missed is less than 2.

2. Now let's make it even more complex. Apart from the above condition, you can also pass the exam if you have not missed any class and your grade is higher than 45.

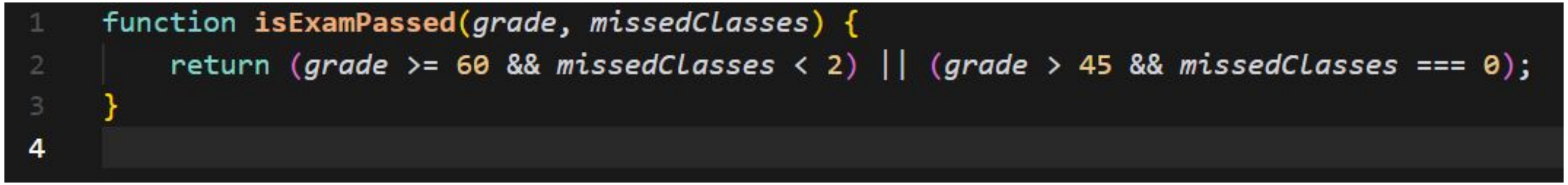
Exercise - Solution

1. Let's create a stricter function for approving the drivers license. We already had that the grade should be greater or equal to 60. How about adding that the quantity of classes missed is less than 2.



```
function isExamPassed(grade, missedClasses) {
    return grade >= 60 && missedClasses < 2;
}
```

2. Now let's make it even more complex. Apart from the above condition, you can also pass the exam if you have not missed any class and your grade is higher than 45.



```
function isExamPassed(grade, missedClasses) {
    return (grade >= 60 && missedClasses < 2) || (grade > 45 && missedClasses === 0);
}

```

Tiny Detour - Javascript is MADNESS

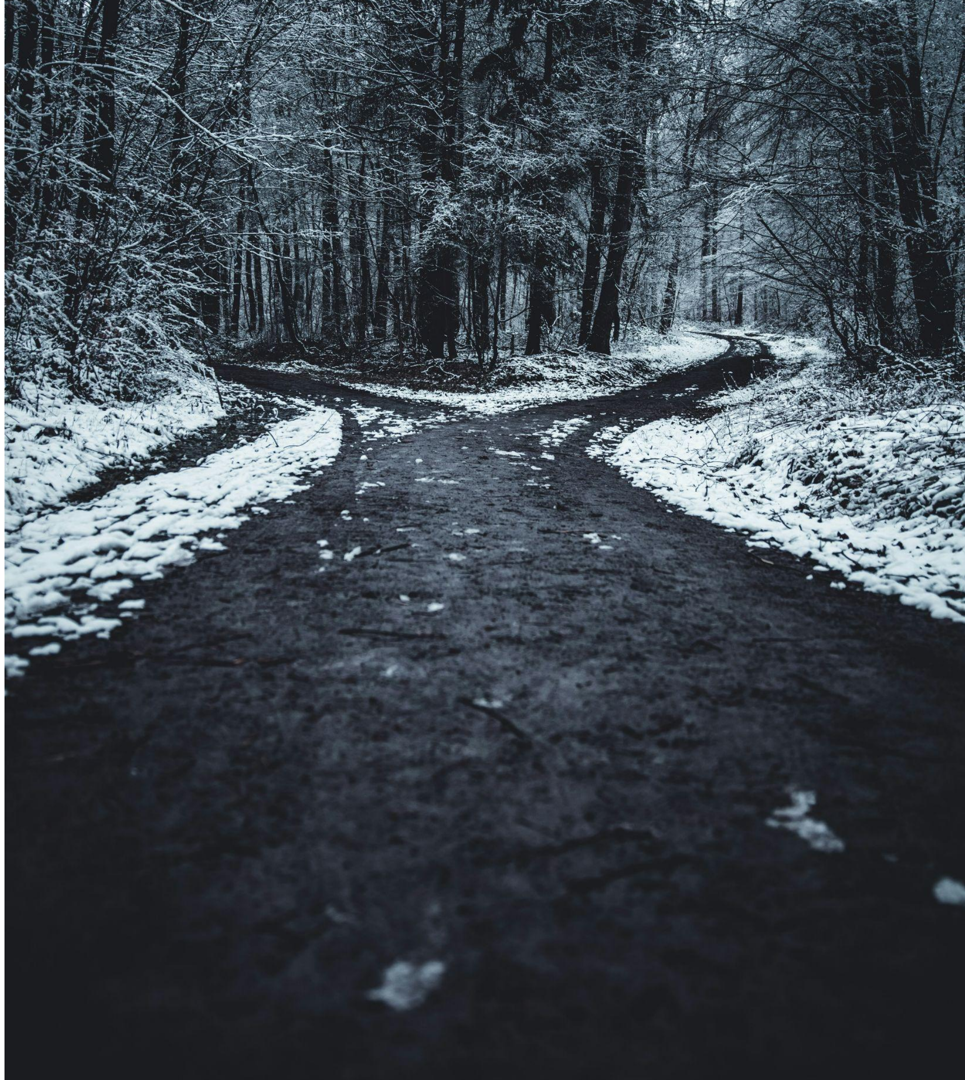
https://github.com/denysdovhan/wtfjs?tab=readme-ov-file#-is-equal-

Break

10 min

Week 6

Conditions



Conditions

Conditional statements control behavior in JavaScript and determine whether or not pieces of code can run.

Conditional statements are **decision making statements.**

There are multiple different types of conditionals in JavaScript including:

“If” statements

“Else” statements

“Else if” statements

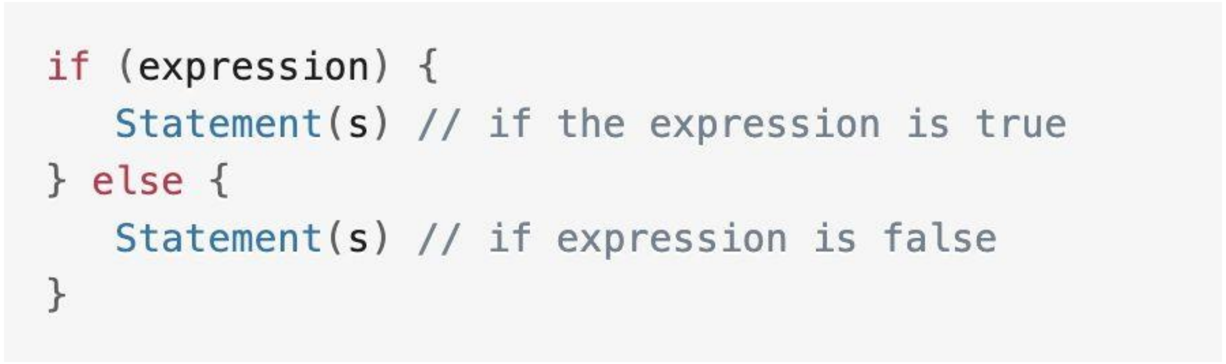
“Switch” statements

Conditions - if

An “if” statement means we can execute a statement **only if an expression is true. Note that the expression will always evaluate to a boolean.**

```
if (expression) {
   Statement(s) // if the expression is true
}
```

Conditions - if else

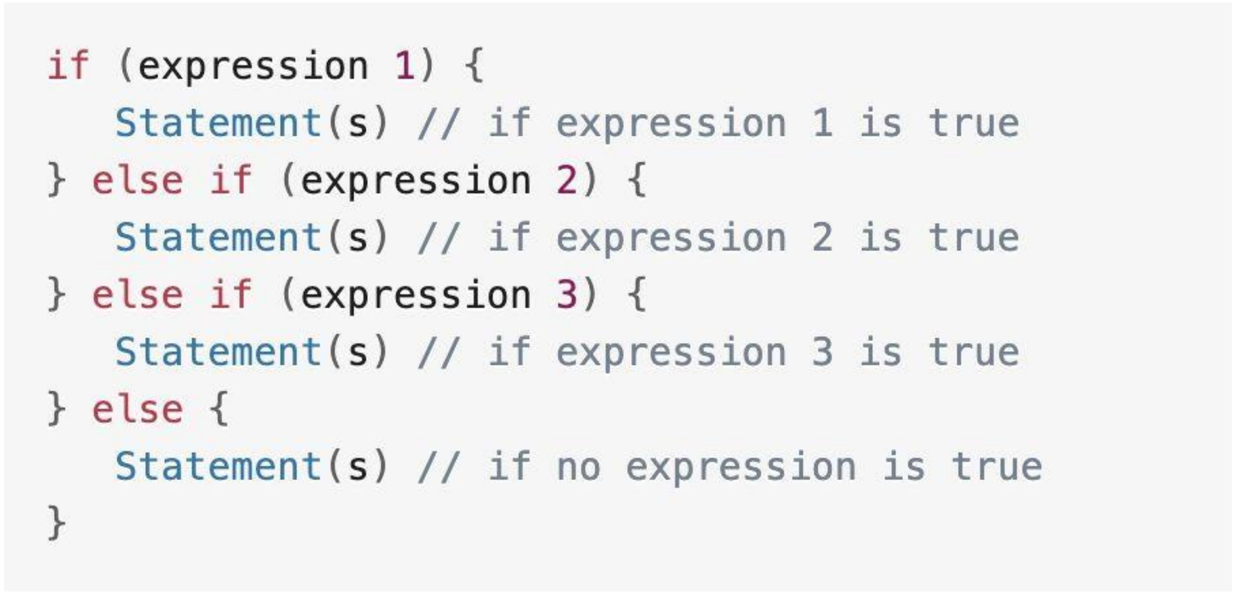


```
if (expression) {
   Statement(s) // if the expression is true
} else {
   Statement(s) // if expression is false
}
```

Conditions - else clauses

```
if (expression) {
   Statement(s) // if the expression is true
} else {
   Statement(s) // if expression is false
}
```

Conditions - else if clauses



```
if (expression 1) {
   Statement(s) // if expression 1 is true
} else if (expression 2) {
   Statement(s) // if expression 2 is true
} else if (expression 3) {
   Statement(s) // if expression 3 is true
} else {
   Statement(s) // if no expression is true
}
```

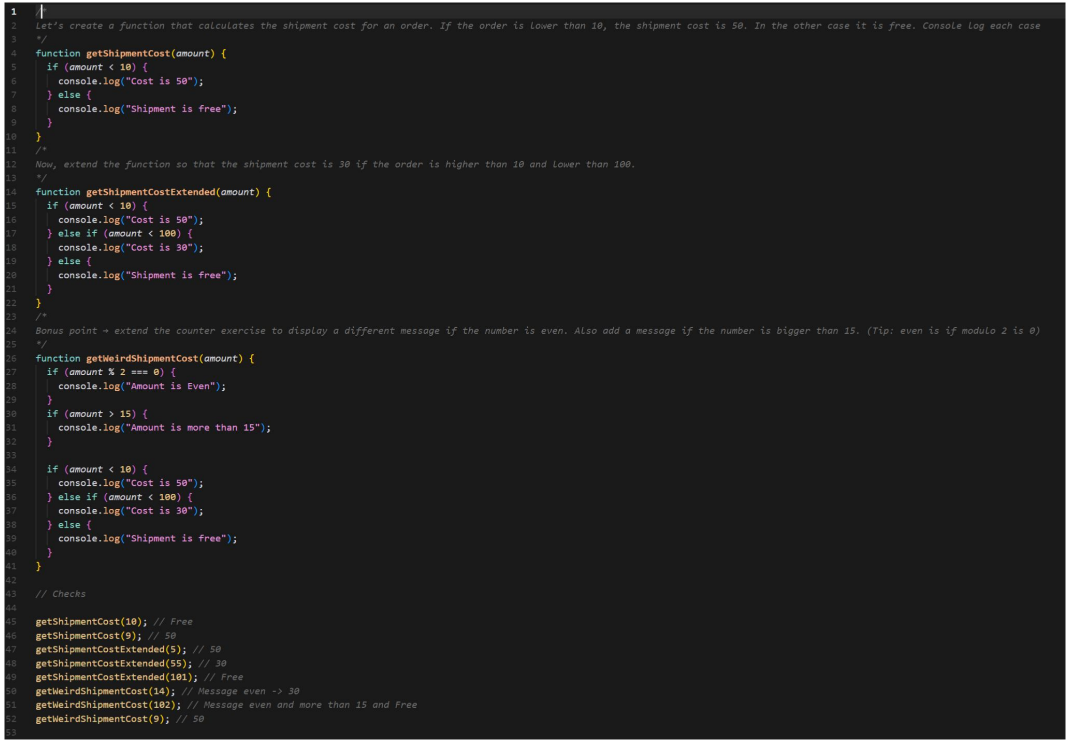
Exercise - All together(10 min)

1. Let's create a function that calculates the shipment cost for an order. If the order is lower than 10, the shipment cost is 50. In the other case it is free. Console log each case

2. Now, extend the function so that the ***shipment cost*** is 30 if the order is higher than 10 and lower than 100.

   Bonus point → extend the counter exercise to display a different message if the number is even. Also add a message if the number is bigger than 15.

Exercise - Solution



```javascript
/*
Let's create a function that calculates the shipment cost for an order. If the order is lower than 10, the shipment cost is 50. In the other case it is free. Console log each case
*/
function getShipmentCost(amount) {
  if (amount < 10) {
    console.log("Cost is 50");
  } else {
    console.log("Shipment is free");
  }
}
/*
Now, extend the function so that the shipment cost is 30 if the order is higher than 10 and lower than 100.
*/
function getShipmentCostExtended(amount) {
  if (amount < 10) {
    console.log("Cost is 50");
  } else if (amount < 100) {
    console.log("Cost is 30");
  } else {
    console.log("Shipment is free");
  }
}
/*
Bonus point → extend the counter exercise to display a different message if the number is even. Also add a message if the number is bigger than 15. (Tip: even is if modulo 2 is 0)
*/
function getWeirdShipmentCost(amount) {
  if (amount % 2 === 0) {
    console.log("Amount is Even");
  }
  if (amount > 15) {
    console.log("Amount is more than 15");
  }

  if (amount < 10) {
    console.log("Cost is 50");
  } else if (amount < 100) {
    console.log("Cost is 30");
  } else {
    console.log("Shipment is free");
  }
}

// Checks

getShipmentCost(10); // Free
getShipmentCost(9); // 50
getShipmentCostExtended(5); // 50
getShipmentCostExtended(55); // 30
getShipmentCostExtended(101); // Free
getWeirdShipmentCost(14); // Message even -> 30
getWeirdShipmentCost(102); // Message even and more than 15 and Free
getWeirdShipmentCost(9); // 50
```

Conditions

The ***switch*** statement looks a lot like an if statement; however, unlike if and else if, which check the condition on each line, the switch ***tests the condition once*** and then performs the relevant expression. A ***default*** statement is released if the condition isn’t met.

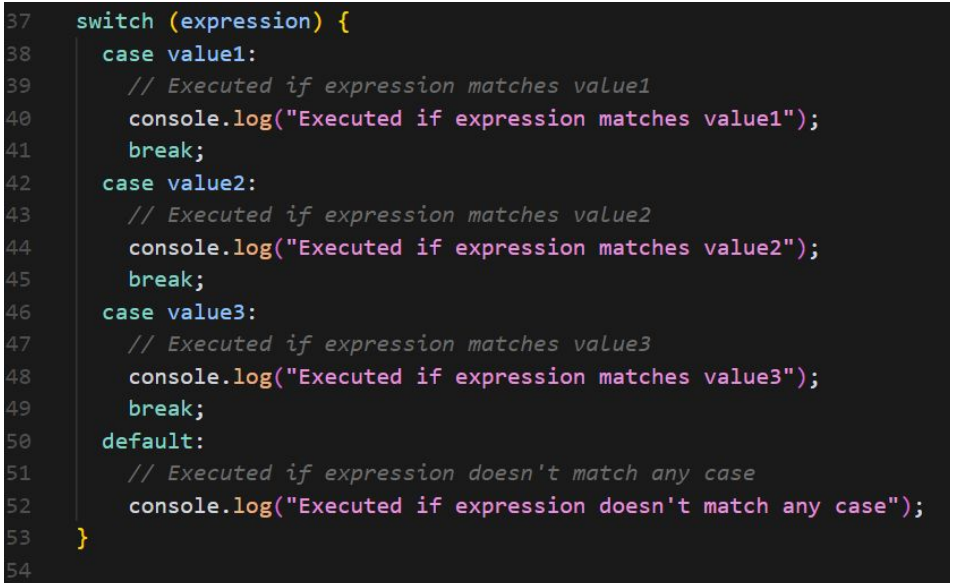
It is usually used when a variable can have multiple possible values.

Conditions - switch

The **switch** statement looks a lot like an if statement; however, unlike if and else if, which check the condition on each line, the switch tests the condition once and then performs the relevant expression. A default statement is released if the condition isn’t met.

It is usually used when a variable can have multiple possible values.

Conditions - switch

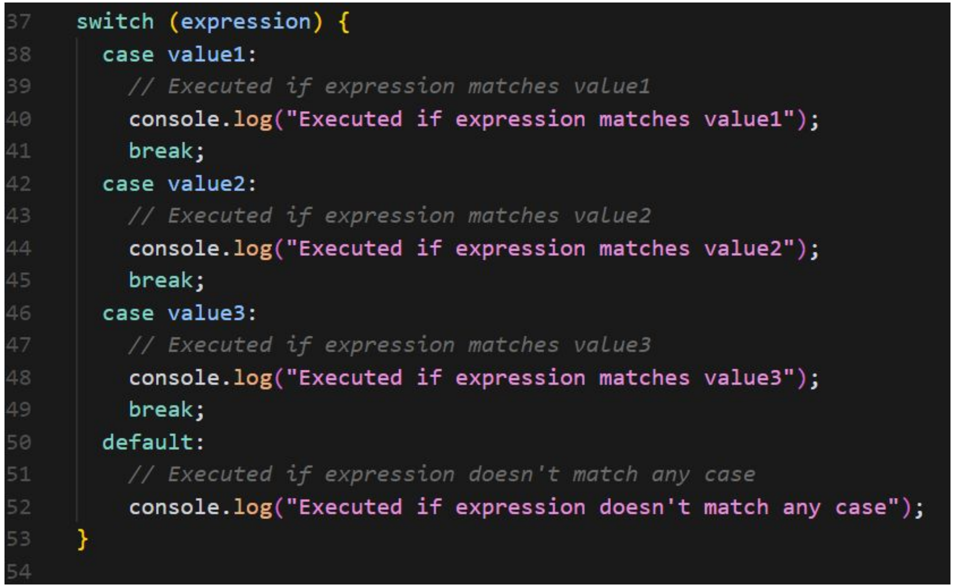


```
switch (expression) {
  case value1:
    // Executed if expression matches value1
    console.log("Executed if expression matches value1");
    break;
  case value2:
    // Executed if expression matches value2
    console.log("Executed if expression matches value2");
    break;
  case value3:
    // Executed if expression matches value3
    console.log("Executed if expression matches value3");
    break;
  default:
    // Executed if expression doesn't match any case
    console.log("Executed if expression doesn't match any case");
}
```

The break statement after every case statement is to let the control know the end of the statement. If the break is not added, the control will end up executing every statement.

Exercise(10 min)

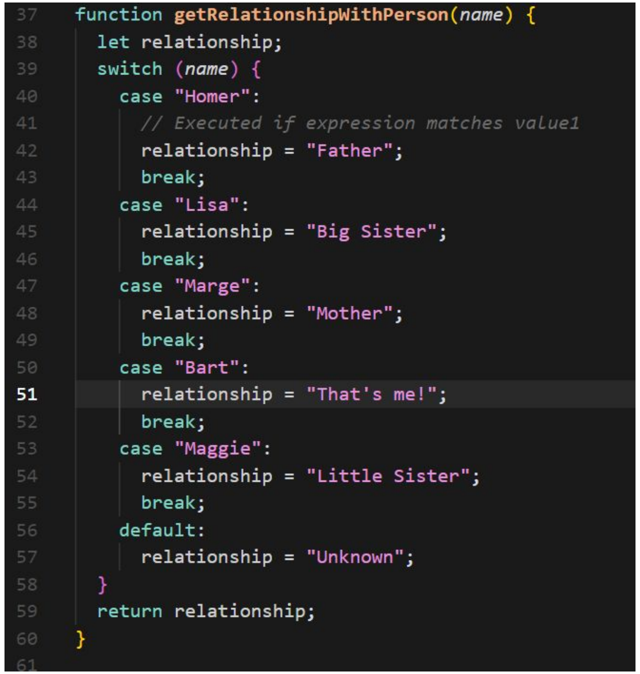
Create a function that **receives a name** and returns which family member (mom, dad, brother) it belongs to.



```
switch (expression) {
  case value1:
    // Executed if expression matches value1
    console.log("Executed if expression matches value1");
    break;
  case value2:
    // Executed if expression matches value2
    console.log("Executed if expression matches value2");
    break;
  case value3:
    // Executed if expression matches value3
    console.log("Executed if expression matches value3");
    break;
  default:
    // Executed if expression doesn't match any case
    console.log("Executed if expression doesn't match any case");
}
```

Exercise - Solution

Create a function that receives a name and tells you which family member it belongs to.



```
function getRelationshipWithPerson(name) {
  let relationship;
  switch (name) {
    case "Homer":
      // Executed if expression matches value1
      relationship = "Father";
      break;
    case "Lisa":
      relationship = "Big Sister";
      break;
    case "Marge":
      relationship = "Mother";
      break;
    case "Bart":
      relationship = "That's me!";
      break;
    case "Maggie":
      relationship = "Little Sister";
      break;
    default:
      relationship = "Unknown";
  }
  return relationship;
}
```

Homework

https://github.com/ReDISchoolDK/Spring25_Frontend/blob/main/Week-06_Operators-Conditionals/homework/README.md

ReDISchool

That’s it!