JavaScript Conditionals

What is a **conditional**?

Understanding the **operators** is key to working with **conditionals**

(and remembering how the differ from written mathematical notation)

```
1  if (x == y) {
2      // Do something
3  };
```

the code says:

if the value of x ***is equal to*** *the value of y then* ***do something***

```
1  if (arrivalTime > 15) {
2      getStarbucks();
3  };
```

the code says:

if when I get to the bus stop the arrival time
is more than the 15 minutes then I will grab a Starbucks

What do these **arithmetic operators** do?

+	-	*	/
add	subtract	multiply	divide

%	++	--
modulus	increment	decrement
(division remainder) *e.g. 13 % 5 = 3*	*(increase by 1)*	*(increase by 1)*

```
1  var val1 = 5;
2  var val2 = 3;
3
4  var val3 = val1 + val2;
```

What would happen in this example?

```
1  var val1 = ‘5’;
2  var val2 = 3;
3
4  var val3 = val1 + val2;
5
6  var val3 = parseInt(val1) + val2;
```

What about these ones?

What do these **assignment operators** do?

`=`	Is **equal to**, it assigns the value *(e.g. var x = 5;)*
`*=`	Keeps original value and **multiplies the new value to it**
`+=`	Keeps the original value and **adds the new value**
`/=`	As above, but **divides the existing value by the new**
`-=`	As above, but **subtracts the new value**
`%=`	x = 10 and y = 3, x %= y gives 1

What do these **comparison operators** do?

`==`	Returns **true** if **values are equal**
`!=`	Returns **true** if **values are not equal**
`===`	Returns **true** if the **values are equal** and **of the same type**
`!==`	Keeps original value and **multiplies the new value to it**
`>`	Returns **true** if the **left value is greater than the right value**
`>=`	Returns **true** if the **left value is greater than** or **equal to the right value**
`<`	Returns **true** if the **left value is less than the right value**
`<=`	Returns **true** if the **left value is less than** or **equal to the right value**

What do these **logical operators** do?

&&

Logical AND: two conditions have to be met

||

Logical OR: either condition has been met

!

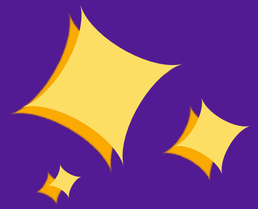
Logical NOT: neither condition is met

```
1   if ((a > b) && (x > y)) { //do something }
2
3   if ((a > b) || (x > y)) { //do something }
4
5   if (!moving) { //do something }
6
7   if (moving) { //do something }
```

Examples of logical operators

There are also special **bitwise operators** that work with **binary**

(you must work with binary numbers to use these, so for most operations we will do they are redundant)



Let’s look at an example

conditional charting

First and **last** condition are known as **operands**.
Then we have the **comparison operator**.

```
1  if (condition) {
2    //block of code to execute if condition is true
3  }
4
5  if (score >= 100) {
6    //do something
7  }
```

```
1  if (playing) {}
2  //shorthand for if playing equals true (for boolean statements)
3
4  if (!playing) {}
5  //shorthand for false (e.g. not playing)
```

```
1  if (score > 100) {
2      success();
3  } else {
4      encourage();
5  }
6
7  /* Using else we always get a decision being made
8  known as a default condition */
```

```
1   if (score > 100) {
2       console.log(‘success’);
3   } else if (score < 10) {
4       console.log(‘encourage’);
5   } else {
6       console.log(‘default’);
7   }
8
9   /* Using else we can also add another ‘if’ that gives
10  a second condition. We also have a default condition
11  which is the last else (this can be left out) */
```

```
1   // Switch Conditionals
2
3   switch (level) {
4     case 1:
5       console.log('success');
6       break;
7     case 2:
8       console.log('level 2, try to make it to level 3');
9       break;
10    case 3:
11      console.log('success');
12      break;
13    default:
14      console.log('good luck');
15      break;
16  }
```

Practice Quiz!

```
1  if (score > 100) {
2    //do something
3  }
```

Write the above if statement in ‘English’

```
1   var inputName = ‘’;  //users input
2   var savedName = ‘fred’;
3
4   var inputPw = ‘’;  //users input
5   var savedPw = ‘ixd2’;
```

Using the above variables, create a condition that checks if the user has logged in and if not forces them to.