



DATA TYPES & LOOPS

VARIABLES

Declaring a variable: `let age;`

Assigning a variable: `age = 29;`

Both in one step: `let age = 29;`

DATA TYPES

String	Literal characters, enclosed in quotes <code>"about"</code>
Number	Numbers treated as numeric values (not in quotes) <code>15</code>
Array	Collection of data <code>["Larry", "Curly", 15, 42]</code>

ARITHMETIC OPERATORS

<code>+</code>	<i>add (also concatenates strings)</i>
<code>-</code>	<i>subtract</i>
<code>*</code>	<i>multiply</i>
<code>/</code>	<i>divide</i>
<code>%</code>	<i>modulus (remainder)</i>

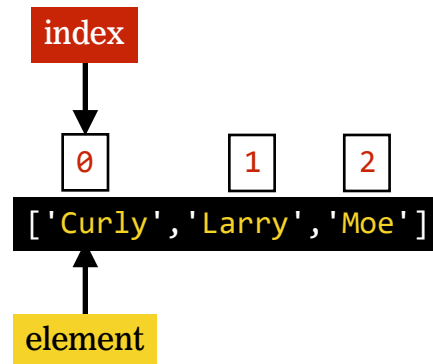
COMPOUND OPERATORS

<code>+=</code>	<i>adds a number to a variable and assigns the new value to the same variable</i>
<code>-=</code>	<i>subtracts a number from a variable and assigns the new value to the same variable</i>
<code>++</code>	<i>adds 1 to a value</i>
<code>--</code>	<i>subtracts 1 from a value</i>

EQUAL SIGNS

<code>=</code>	<i>assigns value on right to object on left</i>
<code>===</code>	<i>evaluates whether values on left and right are the same</i>

ARRAYS



SPECIAL NUMBER OPERATORS

<code>Math.pow(m,n)</code>	<i>Returns m to the power of n</i>
<code>Math.sqrt(n)</code>	<i>Returns the square root of n</i>
<code>Math.random()</code>	<i>Returns a random number between 0 (inclusive) and 1 (exclusive)</i>
<code>Math.floor(n)</code>	<i>Returns largest integer less than or equal to n</i>
<code>Math.ceil(n)</code>	<i>Returns smallest integer greater than or equal to n</i>

ARRAY HELPER METHODS

<code>toString()</code>	Returns a single string consisting of the array elements converted to strings and separated by commas	<code>reverse()</code>	Reverses the array
<code>join()</code>	Same as <code>toString()</code> , but allows you to pass a custom separator as an argument	<code>shift()</code>	Removes and returns the item at the start of the array
<code>pop()</code>	Removes and returns the item at the end of the array	<code>unshift()</code>	Adds one or more items to the start of the array
<code>push()</code>	Adds one or more items to the end of the array		

ARRAY ITERATOR METHODS

<code>forEach()</code>	Executes a provided function once per array element	<code>filter()</code>	Creates a new array with all elements that pass the test implemented by the provided function
<code>every()</code>	Tests whether all elements in the array pass the test implemented by the provided function	<code>map()</code>	Creates a new array with the results of calling a provided function on every element in this array
<code>some()</code>	Tests whether some element in the array passes the text implemented by the provided function		

Usage:

```
arrayName.method(function() {  
  // do something  
})
```

Example:

```
let friends = ['Curly','Larry','Moe'];  
  
friends.forEach(function() {  
  // do something  
})
```

LOOPS

for	Runs while a condition is true, and includes syntax to declare and customize the iterator at the start	<code>for (variable; condition; iteration) { // do something }</code>
while	Runs while a condition is true	<code>while (condition) { // do something }</code>
do while	Runs while a condition is true, and ensures that the code block is executed at least once	<code>do { // do something } while (condition)</code>
