Day 2 - Data Types

Escape Sequences in Strings

- \n: new line
- \t: Tab, means 8 spaces
- \\: Back slash
- \': Single quote (')
- \": Double quote(")

Template Literals (Template Strings)

We use (`) for these. They allow us to inject data (${}) inside a template string.

Example:

```
`String literal text`
`String literal text ${expression}`
```

We can also add expressions using string templates.

Example:

```
let a = 2
let b = 3
console.log(`${a} is greater than ${b}: ${a > b}`)
```

String Methods

- length: returns the number of characters in a string including empty space.

  ```
  let js = 'JavaScript'
  console.log(js.length)
  ```

- Accessing characters in a string: Each character in a string can be accessed using its index. First index is zero, last index is the length of the string minus one.

  ```
  let string = 'JavaScript'
  let firstLetter = string[0];
  let lastIndex = string.length - 1;
  let lastLetter = string[lastIndex];

  console.log(`First letter: ${firstLetter} | Last letter: ${lastLetter}`);
  ```

- toUpperCase(): Changes the string to uppercase letters.

  ```
  let string = 'JavaScript'

  console.log(string.toUpperCase())   // JAVASCRIPT
  ```

- toLowerCase(): Changes the string to lowercase letters.

```
let string = 'JavaSCript'

console.log(string.toLowerCase())   // javascript
```

- substr(): Takes two args, the starting index and number of characters to slice.

```
let string = 'JavaScript'
console.log(string.substr(4,6))    // Script
```

- substring(): Takes two args, the starting index and the stopping index, but doesn't include the character at the stopping index.

```
let country = 'Finland'

console.log(country.substring(0, 3))   // Fin
console.log(country.substring(3, 6))   // lan
console.log(country.substring(3, 7))   // land
console.log(country.substring(3))      // land
```

- split(): Splits a string at a specified place.

```
let string = '30 Days of JavaScript'

console.log(string.split())  // Changes to array -> ["30 days of JavaScript"]
console.log(string.split(' '))  // Split to an array at space -> ["30", "Days", "Of", "JavaScript"]

let firstName = 'Chris'

console.log(firstName.split())  // Changes to array -> ["Chris"]
console.log(firstName.split("")) // Split to an array at each letter -> ["C", "h", "r", "i", "s"]

let countries = 'Finland, Sweden, Norway, Denmark, and Iceland'
console.log(countries.split(','))    // Split to an array at every comma -> ['Finland', ' Sweden', ' Norway', ' Denmark', ' and Iceland
console.log(countries.split(', '))   //["Finland", "Sweden", "Norway", "Denmark", "and Iceland"]
```

- trim(): Removes trailing space in beginning or end of a string.

```
let string = '    30 Days of JavaScript   '
console.log(string.trim());   // "30 Days of JavaScript"
```

- includes(): Takes a substring arg and check if substring arg exists in string. Returns a boolean.

```
let string = '30 Days of JavaScript';

console.log(string.includes('Days'));  // true
console.log(string.includes('days'));  // false - case sensitive
```

- replace(): takes as a parameter(argument) the old substring and a new substring.

```
let string = '30 Days of JavaScript'
console.log(string.replace('JavaScript', 'Python'))
// 30 Days of Python
```

- chatAt(): Takes index and returns the value at that index.

```
let string = '30 Days of JavaScript'
console.log(string.charAt(0));   // 3
```

- charCodeAt(): Takes index and returns char code (ASCII number) of the value at that index

```
let string = '30 Days of JavaScript'
console.log(string.charCodeAt(3))   // D ASCII number is 68
```

- indexOf(): Takes substring and if substring exists in a string, it returns the first position of the substring. If it does not exist, returns -1.

```
let string = '30 Days of JavaScript'

console.log(string.indexOf('D'))   // 3
console.log(string.indexOf('Days'))  // 3
console.log(string.indexOf('days'))  // -1
```

- lastIndexOf(): Takes a substring and if the substring exists in the string it returns the last position of the substring. If it does not exist it returns -1.

```
let string = 'I love JavaScript. If you do not love JavaScript what else can you love.'

console.log(string.lastIndexOf('love'))   // 67
console.log(string.lastIndexOf('you'))    // 63
console.log(string.lastIndexOf('JavaScript')) // 38
```

- concat(): Takes many substrings and joins them

```
let string = '30'
console.log(string.concat("Days", "Of", "JavaScript"))
// 30DaysOfJavaScript
```

- startsWith(): Takes a substring as an argument and checks if string starts with that specified substring. Returns a boolean.

```
let string = 'I am hungry';

console.log(string.startsWith('I')) // true
console.log(string.startsWith('i')) // false
```

- endsWith(): Takes a substring as an argument and checks if the string ends with that specified substring. Returns a boolean.

```
let string = 'JavaScript'
console.log(string.endsWith('ipt'))  // true
console.log(string.endsWith('Script')) // true
console.log(string.endsWith('script')) // false
```

- search(): Takes a substring as an argument and returns the index of the first match. Search value can be a string or a regular expression pattern.

```
let string = 'I love JavaScript. If you do not love JavaScript what else can you love.'

console.log(string.search('love'))  // 2
console.log(string.search(/javascript/gi))  // 7
```

- match(): Takes a substring or regular expression pattern as an argument and returns an array if there's a match. If not, returns null.

```
let string = 'love'
let patternOne = /love/  // Without any flag
let patternTwo = /love/gi  // g - means to search in the whole text, i - case insensitive

// syntax
string.match(substring)
```

- repeat(): Takes a number as an argument and returns the repeated version of the string.

```
let string = 'taco'
console.log(string.repeat(3)) // tacotacotaco
```

Checking Data Types and Casting

Checking Data Types

To check the data type of a certain variable, we use the typeof method.

```
console.log(typeof 'Chris')  // string
console.log(typeof 10)  //string
console.log(typeof false)  //boolean
console.log(typeof NaN)  // number
console.log(typeof undefined)  // undefined
console.log(typeof null)  // object
```

Changing Data Type (Casting)

- Casting: Converting one data type to another data type. We use parseInt(), parseFloat(), Number(), + sign, str(). When doing arithmetic operations, string numbers should be first converted to integer or float, if not, it returns an error.

String to Int:

We can convert a string number to a number. Any number inside quotes is a string number. It can be converted to a number using the following methods:

- parseInt()
- Number()
- Plus sign(+)

```
let num = '10'
let numInt = parseInt(num)
console.log(numInt) // 10

let num2 = '11'
let numberInt = Number(num2)
console.log(numberInt)  // 11

let num3 = '12'
let numbInt = +num3
console.log(numbInt)  // 12
```

Float to Int:

We can convert float numbers to integers. We use the following method to convert float to int:

```
let num = 9.81
let numInt = parseInt(num)

console.log(numInt)  // 9
```