Java Script

- Javascript's 7 datatypes:
  Undefined- something that is not defined, means you might have variable set to nothing, **null**- nothing, like setting something (eg: variable) to nothing, **boolean**- true or false, **string**, **symbol**- immutable primitive value that is unique, **number**, and **object**- can store different key value pair.

```
var myName = "beau"
myName = 8

let ourName = "freeCodeCamp"

const pi = 3.14
```

- Storing values with the Assignment Operator

```
var a;

var b = 2;
console.log(a)
a = 7;
b = a;

console.log(a)
```

- Initialising variable w/ Assignment Operator

```
var a = 9;
//uninitialised variables
var b;
```

- Case Sensitivity in Variables

```
//declaration
var studlyCapVar;
//assignment
studlyCapVar = "hello world"

console.log(studlyCapVar)
```

- Basic Math

```
//addition
var sum = 3 + 4;
console.log(sum)
//subtraction
var sub = 89 - 46;
console.log(sub)
//muliplication
var product = 6 * 9;
console.log(product)
//division
var division = 66 / 33;
console.log(division)
//Incrementing number
var myVar = 68;
myVar++;
console.log(myVar)
//Decrementing number
var myVar2 = 70;
myVar2--;
console.log(myVar2)
//decimal
var ourDecimal = 5.5
//multiply decimals
var product2 = 2.0 * 4.9
console.log(product2)
//divide decimals
var quotient = 0.0/2.0
console.log(quotient)
//finding a remainder
var remainder;
remainder = 11 % 3;
console.log(remainder)
```

- Compound assignment with Augmented ‘basic math’

```
var a = 5
var b = 10
```

```
var c = 44
var d = 64
var e = 66
var f = 87

a += 4
b *=45
c -= 30
d /= 8
e %= 2


console.log(a)
console.log(b)
console.log(c)
console.log(d); console.log(e)
```

- Declare String Variables

```
var firstName= "Preet"
var lastName = "kaur"

console.log(firstName)
console.log(lastName)
```

- **Escaping Literal Quotes in Strings**

\ is the escape character

```
var myStr = "I am \"double quoted\" string inside \"double quotes\""
console.log(myStr)

Output:
"I am "double quoted" string inside "double quotes""
```

- Quoting string with single quotes

` (backtick) is used for single quote string

```
var myStr = `'I am \"double quoted\" string inside \"double quotes\"'`
console.log(myStr)
```

- Escape sequences in Strings

```
/****
CODE   OUTPUT
\'     single quote
\"     double quote
\\     backslash
\n     newline
\r     carriage return
\t     tab
\b     backepace
\f     form feed
******/

var myStr = "FirstLine\n\t\\SecondLine\nThirdLine"
console.log(myStr)
```

- Concatenate String- Plus Operator and with Augmented Plus Operator

```
var OutStr = "I come first\n" + "I come Second"
console.log(OutStr)

Augmented
var OutStr = "I come first\n"
OutStr += "I come Second"
console.log(OutStr)
```

- Constructing String with Variables

```
var MyName = "Preet"
var Introduction = "My name is " + MyName + " Kaur"
console.log(Introduction)
```

- Appending variables to Strings

```
var MyName = "Preet"
var Introduction = "My name is "
Introduction += MyName
console.log(Introduction)
```

- Finding the length of the string

```
var Introduction = "My name is "
Introlength = Introduction.length
console.log(Introlength)
```

- Bracket Notation

```
var Name = "Preet"
firstletteroftheName = Name[0];
console.log(firstletteroftheName)
```

Strings are immutable that means

In JavaScript, strings are immutable, which means that once a string is created, it cannot be modified. Instead, any operation on a string that appears to modify it actually creates a new string. This is because JavaScript strings are primitive values, which are immutable and cannot be changed. Therefore, any operation that appears to modify a string (such as using the `+=` operator) actually creates a new string that contains the original string plus the new content.

- Bracket notation to find Nth character & last character in String

```
var Name = "Preet"
firstletteroftheName = Name[4];
console.log(firstletteroftheName)

Last Character
var Name = "Preet"
firstletteroftheName = Name[Name.length - 1]; //subract length by 1
console.log(firstletteroftheName)
```

- Bracket Notation to find Nth-to-last character in string

```
'try to work it by yourself'
```

- Word Blanks (using functions)

```
function worldblank(noun, adj, verb, adverb) {

var result = "";
```

```
result += "The" + " " + adj + " " + noun + " " + verb + " " + "to the mall" + " " + adverb;

return result;
}
console.log(worldblank("dog", "big" , "ran" , "quickly"))
console.log(worldblank("boy", "medium", "walked", "slowed"))
```

- **Arrays-** Storing Multiple values with Arrays

```
var ourArray = ["John", 23];
console.log(ourArray)
```

- Nested Array

```
var ourArray = [["the universe", 42], ["John", 23]];
console.log(ourArray)

var myArray = [["bulls", 23], ["white sox", 45]]
console.log(myArray)
```

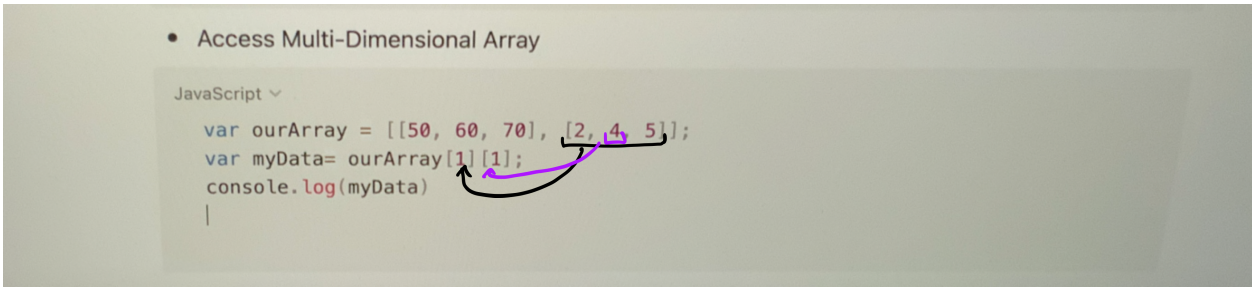
- Access Array Data with Indexes & modify Array data with Indexes

```
var ourArray = [50, 60, 70]
var myData = ourArray[1];
console.log(myData)

Modify Array data with Indexes

var ourArray = [50, 60, 70];
ourArray[1] = 45;
console.log(ourArray)
```

- Access Multi-Dimensional Array



- Manipulate Array with Push & Pop

```
var ourArray = [50, "Aadi", "Preet"];
ourArray.push("nobody", "cares");
console.log(ourArray)

Pop
var ourArray = [50, "Aadi", "Preet"];
var remove50 = ourArray.pop();
console.log(ourArray)

Output:
It'll pop the last element like the output in this one is [50, "Aadi"]
```

- Manipulate Arrays with Shift & Unshift

```
var ourArray = [50, "bye", "Preet"];
var remove50 = ourArray.shift();
console.log(ourArray)

Output:
It is similar to pop fn. but only difference is that it removes the first function
from the list (element on 0th index)
["bye","Preet"]

unshift
var ourArray = [50, "bye", "Preet"];
ourArray.unshift("work");
console.log(ourArray)

Output:
It is similar to push fn. but the only difference is that it adds element on the
Oth index.
["work",50,"bye","Preet"]
```

- Write Useable with Function

```
function reusablefn() {
 console.log("Hi world");
}
reusablefn();
```