Quick Lab 2 - JavaScript Operators

Objectives

* To understand how operators work in JavaScript.

Activity

1. In VSCode, create the files index.html and **operators.js**
2. Create a **script** tag to import **operators.js**
3. Copy the contents of **operators.js** from https://gist.github.com/JHarry444/ac2985664ca9e1ccffb7f77c8f6ede80.
4. Open **index.html** using the live server

The browser window will open, but there will be nothing to view!

1. Examine the following operations and write the result you expect before you look at any code output!

|  |  |
| --- | --- |
| Arithmetic operators | |
| Operation | Result |
| console.log(5 + 5); | 10 |
| console.log(5 \* 10); | 50 |
| console.log(10 % 3); | 1 |
| console.log(5 + 10 / 2 \* 5 - 10); | 20 |
| console.log((6 + 10) / 2 \* 5 - 10); | 30 |

1. Uncomment the Arithmetic Operators section of the code - highlight the required lines to uncomment and press CTRL + / (or CMD + / on MacOS).
2. Save the file and check the console output against what you expected.
3. Repeat for assignment operators, assuming x is initialised as 0 and the statements are processed sequentially.

|  |  |
| --- | --- |
| Assignment operators | |
| Operation | Result |
| console.log(x = x + 1); | 1 (adds one) |
| console.log(x+= 1); | 2 (adds one) |
| console.log(x++); | 2 (print then increment) |
| console.log(++x); | 4 (increment then print) |

1. Now, we will move onto relational operators. Every expression will evaluate as either true or false.

|  |  |
| --- | --- |
| Relational operators | |
| Operation | Result |
| console.log(5 > 3); | True |
| console.log(3 != 3); | False |
| console.log(3 <= 2 && 5 >2); | False |
| console.log(!5>3); | false |

1. Finally, we will explore what happens with mismatched types.

|  |  |
| --- | --- |
| Mismatched types | |
| Operation | Result |
| console.log(5 + "5"); | 55 |
| console.log(5 + true); | 6 |
| console.log(5 \* "5"); | 25 |
| console.log(1 == true); | true |
| console.log(1 === true); | False  When using triple equals === in JavaScript, we are testing for strict equality. This means both the type and the value we are comparing have to be the same. |

This is the end of Quick Lab 2