Fundamentals of JS-II

Arithmetic Operators:

Arithmetic operators are used to perform mathematical operations on numerical values.

1. Addition (+): Adds two values.
   Example:

```
let a = 5, b = 3;
console.log(a + b); // Output: 8
```

2. Subtraction (-): Subtracts the right operand from the left operand.
   Example:

```
console.log(a - b); // Output: 2
```

3. Multiplication (*): Multiplies two values.
   Example:

```
console.log(a * b); // Output: 15
```

4. Division (/): Divides the left operand by the right operand.
   Example:

```
console.log(a / b); // Output: 1.6666666666666667
```

5. Modulus (%): Returns the remainder of the division of the left operand by the right operand.
   Example:

```
console.log(a % b); // Output: 2
```

6. Exponentiation (**): Raises the left operand to the power of the right operand.
   Example:

```
console.log(a ** b); // Output: 125
```

Comparison Operators:

Comparison operators are used to compare two values and return a Boolean result.

1. Equal (==): Returns true if the values on both sides are equal.
   Example:

```
console.log(a == b); // Output: false
```

2. Not Equal (!=): Returns true if the values on both sides are not equal.
   Example:

```
console.log(a != b); // Output: true
```

3. Greater Than (>): Returns true if the left operand is greater than the right operand. Example:

```
console.log(a > b); // Output: true
```

4. Less Than (<): Returns true if the left operand is less than the right operand.
   Example:

```
console.log(a < b); // Output: false
```

5. Greater Than or Equal To (>=): Returns true if the left operand is greater than or equal to the right operand.
   Example:

```
console.log(a >= b); // Output: true
```

6. Less Than or Equal To (<=): Returns true if the left operand is less than or equal to the right operand.
   Example:

```
console.log(a <= b); // Output: false
```

Logical Operators:

Logical operators are used to perform logical operations on Boolean values.

1. AND (&&): Returns true if both the left and right operands are true.
   Example

```
let x = true, y = false;
console.log(x && y); // Output: false
```

2. OR (||): Returns true if at least one of the operands is true.
   Example:

```
console.log(x || y); // Output: true
```

3. NOT (!): Returns true if the operand is false and vice versa.
   Example:

```
console.log(!x); // Output: false
```

Type Conversion:

Type conversion is the process of converting one data type to another.

1. Implicit Type Conversion (Coercion): Automatically performed by the interpreter.
   Example:

```
let int_var = 5 + 2.0;
console.log(int_var); // Output: 7.0
```

   (Here, the integer 5 is implicitly converted to a float)

2. Explicit Type Conversion (Casting): Done by the programmer using predefined functions. Example:

```
let str_var = String(42);
console.log(str_var); // Output: "42"
```

   (Here, the integer 42 is explicitly converted to a string)

Type Coercion:

Type coercion is the automatic conversion of one data type to another.
Example of Type Coercion:

```
const value1 = "5";
const value2 = 9;
let sum = value1 + value2;
console.log(sum);
//output: "59"
```

When using the + operator between a string and a number, JavaScript coerces the number (9) into a string ("9") and then performs string concatenation.

Overall Importance of this lecture

- These concepts are foundational for building algorithms, making decisions, and performing various operations in JavaScript.
- They are essential for creating dynamic and interactive web applications.
- Understanding operator precedence helps in writing expressions that are evaluated as intended.
- Type conversion and coercion play a crucial role in managing different types of data.

In summary, mastering these concepts is fundamental for anyone working with JavaScript, whether for web development, server-side programming, or any other application where JavaScript is used. They provide the tools needed to manipulate data and control the flow of a program effectively.

Reference

- **Arithmetic Operators** - https://javascript.info/comparison
- **Comparison Operators** - https://javascript.info/comparison
- **Logical Operators** - https://javascript.info/logical-operators
- **Bitwise Operators** - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Bitwise_AND
- **Type Conversion** - https://developer.mozilla.org/en-US/docs/Glossary/Type_Conversion
- **Type Coercion** - https://developer.mozilla.org/en-US/docs/Glossary/Type_coercion