Set 23: Math Operations

Skill 23.01: Review arithmetic operations

Skill 23.01 Concepts

The basic arithmetic operations are as follows,

Operator	Description
+	addition
-	substraction
*	multiplication
/	division
%	modulus

Examples of how each of the above operators can be applied are illustrated below,

Addition

```
var x = 1;
var y = 2;
var z = x + y;  //3 is assigned to z
x = x + z;  //4 is re-assigned to x
console.log(z);   //3 is printed to the console
console.log(x + y);   //6 is printed to the console
console.log(x + 10);   //14 is printed to the console
console.log(10 + 10); //20 is printed to the console
```

Subtraction

```
var x = 1;
var y = 2;
var z = x - y;   //-1 is assigned to z
x = x - z;   //2 is re-assigned to x
console.log(z);   //-1 is printed to the console
console.log(x - y);   //0 is printed to the console
console.log(x - 10);   //-8 is printed to the console
console.log(10 - 10);   //0 is printed to the console
```

Multiplication

```
var x = 1;
var y = 2;
var z = x * y;  //2 is assigned to z
x = x * z;  //2 is re-assigned to x
console.log(z);  //2 is printed to the console
console.log(x * y);  //4 is printed to the console
console.log(x * 10);   //20 is printed to the console
console.log(10 * 10);   //100 is printed to the console
```

Division

```
var x = 1;
var y = 2;
var z = y/x;   //2 is assigned to z
x = z/x;   //2 is re-assigned to x
console.log(z);   //2 is printed to the console
console.log(x/y);   //1 is printed to the console
console.log(x/10);   //.2 is printed to the console
console.log(10/10);   //1 is printed to the console
```

Modulus

Modulus prints the remainder of a division operation. For example, console.log(5%3); will print 2. This is because when 5 is divided by 3, the remainder is 2. Modulus gives the remainder. Modulus also handles negatives. The answer to a%b has the same sign as a. The sign of b is ignored.

```
var x = 1;
var y = 2;
var z = x%y;     //1 is assigned to z
x = x%z;     //0 is re-assigned to x
console.log(z);   //1 is printed to the console
console.log(x%y);   //0 is printed to the console
console.log(x%10);   //0 is printed to the console
console.log(10%10);   //0 is printed to the console
```

Skill 23.01 Exercise 1

Skill 23.02: Apply unary operators

Skill 23.02 Concepts

The unary operators are operations that require only one operand; they perform various operations such as incrementing/decrementing a value by one, negating an expression, or inverting the value of a boolean (a true/false variable type).

Incrementing a value by 1

The code below illustrates how to increment the variable x by 1

```
var x = 1;
x = x + 1;   //x is now 2
```

Incrementing a value by 1 can also be done using the *++* operator,

```
var x = 1;
console.log(x++);  //1 is printed to the consol, then x is incremented
console.log(++x);  //x is incremented first, then it's value, 3, is printed to the c
onsol.
```

Notice in the above example that *++* can come before or after the variable. If it comes *before* the variable, the variable is first incremented then printed. If it comes *after* the variable, the variable is first printed then incremented.

Decrementing a value by 1

The code below illustrates how to decrement the variable y by 1

```
var y = 10;
y = y - 1;  //y is now 9
```

Decrementing a value by 1 can also be done using the -- operator,

```
var y = 10;
console.log(y--);  //10 is printed to the consol, then y is decremented
console.log(--
y);  //y is decremented first, then it's value, 8, is printed to the console
```

Notice in the above example that -- can come before or after the variable. If it comes *before* the variable, the variable is first decremented then printed. If it comes *after* the variable, the variable is first printed then decremented.

Skill 23.02 Exercise 1

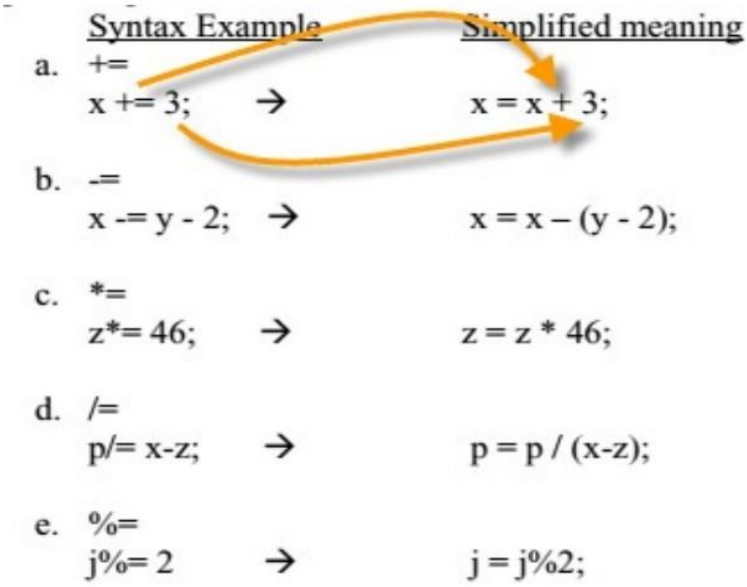
Skill 23.03: Apply compound operators

Skill 23.03 Concepts

A compound assignment operator is an operator that performs a calculation and an assignment at the same time. In the below example, x can be re-assigned explicitely using *x = x + 5*; x can also be re-assigned using the addition compound operator.

```
var x = 10;
x = x + 5;  //x is 15
x += 5;   //x is now 20
```

Compound operators can be applied to all the arithmetic operations. How this is done is illustrated below,



[Skill 23.03 Exercise 1]

Skill 23.04: Use the Math object to perform mathematical operations

Skill 23.04 Concepts

The built in Math object in JavaScript allows us to perform calculations that go beyond the simple arithmetic operations we've seen. An example of how the Math object can be applied is illustrated below. The example below computes the square root of 17. The result is assigned to the variable p.

```
var p = Math.square(17); //prints 4.123105625617661
```

In the above example,

- *var p* is the variable to which the result of the Math operation is assigned
- *Math. is* the notation we use to access the library of Math functions in javascript
- *square(17)* is the operation we want to perform on the number 17. In this case, it is the square root.

JavaScript provides an extensive library of Math operations. Below is a description of some of them.

Operation	Example	Description
abs	Math.abs(n);	returns the absolute value of n
pow	Math.pow(n, p);	Returns the the number n raised to the p power
sqrt	Math.sqrt(n);	Returns the square root of n
ceil	Math.ceil(n);	Returns the highest whole number from n
floor	Math.floor(n);	Returns the lowest whole number form n
min	Math.min(a, b);	Returns the smaller of a and b
max	Math.max(a, b);	Returns the larger of a and b
random	Math.random();	Returns a random number in the range (0 <= r < 1)
round	Math.round(n);	Returns n rounded to the nearest whole number
PI	Math.PI	Returns 3.141592653589793

[Skill 23.04 Exercises 1]

Skill 23.05: Apply random() to create a random number in a specified range

Skill 23.05 Concepts

Many applications you will create will require a random number. For example, what if you needed to write a program to generate a number that represented a face from a 6-sided die, or a card from a 52 card deck?

The random() function generates a random number between 0 and 1, where 0 is inclusive, but 1 is not. The below code is illustrative,

```
console.log(Math.random()); //prints a random var from 0 up to 1
```

To create a number in a different range, say 0 up to 10, simply multiply the result of Math.random() by the desired range. An example is shown below,

```
console.log(Math.random()*10); //prints a random var from 0 up to 10
```

Recall, however that the random() method returns a decimal number. The below code illustrates how to generate a random integer from 0 up to 10,

```
var randomNumber = Math.random()*10;
console.log(Math.floor(randomNumber)); //prints a random integer from 0 up to 10
```

The previous examples illustrate how to scale the random() method to a specified range. The example below illustrates how to shift the range of the random number.

```
var randomNumber1 = (Math.random()*10) + 100;
console.log(randomNumber); //prints a random number from 100 to 110

var randomNumber2 = (Math.random()*10) - 100;
console.log(randomNumber2);  //prints a random number from -100 to -90

var randomNumber2 = (Math.random()*10) - 5;
console.log(randomNumber2);  //prints a random number from -50 to 5
```

Skill 23.05 Exercises 1