Intro to JS - Part 2 - Day 1: Sept 16

Spread/Rest Operator

denoted by three dots (...)

Spread Operator

concatenating arrays:

```
const arr1 = [1, 2, 3];       //recall arrays have “elements”
const arr2 = [4, 5, 6];
const combinedArr = [...arr1, ...arr2];
console.log(combinedArr);
// Output: [1, 2, 3, 4, 5, 6]
```

combining objects:

```
const obj1 = {a: 1, b: 2};   //recall objects have “properties”
const obj2 = {c: 3, d: 4};
const newObj = {...obj1, ...obj2};
console.log(newObj);
// Output: {a: 1, b: 2, c: 3, d: 4}
```

copy an array and add elements:

```
const originalArr = [1, 2, 3];
const arrCopy = [...originalArr, 4, 5, 6, 7];
console.log(arrCopy);
// Output: [1, 2, 3, 4, 5, 6, 7]
```

Copy an object and add properties:

```
const originalObj = {a:1,b:2,c:3,d:4};
const objCopy = {...originalObj, e:5, f:6, g:7};
console.log(objCopy);
// Output: {a:1, b:2, c:3, d:4, e:5, f:6, g:7}
```

Rest Operator

used in function parameters

```
function add4567(...numbers) {
  arrCopy = [...numbers, 4, 5, 6, 7];
  return arrCopy;
}
console.log(add4567(1, 2, 3));
```

Practice exercise : write a function that combines 2 arrays

```
function combine(firstarray, lastarray) {
  combinedArr = [...firstarray, ...lastarray];
  return combinedArr;
}

arr1=[1, 2, 3];
arr2=[4, 5, 6];
console.log(combine(arr1, arr2));
```

For…of loops [ARRAYS]

More concise than a regular for loop.
Fyi ES6 = EcmaScript which is a set of standards for scripting languages like JS.

In this kind of loop, we iterate through an array (or string), to execute some code for **each element in the array** (or component of string)

```
const colors = ['red', 'green', 'blue'];

for (const color of colors) {
  console.log(color);
} // Output: red green blue
```

Practice Exercise: function that returns the sum of array elements using **for…of**

```
function ArraySum(numbers) {
  let mySum = 0;
  for (const number of numbers) {
    mySum += number;
  }
  console.log(mySum);
}

let myNumbers = [21, 5, 0, 2, -2];
ArraySum(myNumbers);
```

For…in loops [OBJECTS]

Iterates over the keys or property names of an object

Practice exercise: Function takes object as argument and returns property names.

```
function propertyArray(someObject){
const myPropertyArray = [];
for (const property in someObject) {
    myPropertyArray.push(property);
  }
console.log(myPropertyArray);
}

const tshirt = {
  size: 'Large',
  color: 'Red',
  slogan: 'I do what I want',
  slogancolor: 'White'
  };

propertyArray(tshirt);
```

SetInterval / SetTimeout / IntervalID

Practice exercise: function logs message every 5 seconds.

```
function logMessageevery5() {
 console.log("Life is good!");
}

 const intervalID = setInterval(logMessageevery5, 5000);
 setTimeout(function(){clearInterval(intervalID);}, 25000);
 logMessageevery5();
```

Object Destructuring

extract object properties, make them variables of the same name

```
const person = { firstName: 'John', lastName: 'Doe', age: 30 };
const {firstName, lastName, age} = person;
```

Set default value

```
const person = { firstName: 'John', lastName: 'Doe' };
const {firstName, lastName, age=30} = person;
```

Rename variables

```
const person = { firstName: 'John', lastName: 'Doe' };
const {firstName: fName, lastName: lName} = person;
```

Nested objects

```
const person = { name: { firstName: 'John', lastName: 'Doe' }, age: 30 };
const {name: {firstName, lastName}, age} = person;
```

Array Destructuring

extract array elements, make them variables, set the variable names

```
const numbers = [1, 2, 3];
const [firstNumber, secondNumber, thirdNumber] = numbers;
console.log(firstNumber); // Output: 1
console.log(secondNumber); // Output: 2
console.log(thirdNumber); // Output: 3
```

Set default value

```
const numbers = [1, 2];
const [firstNumber, secondNumber, thirdNumber = 3] = numbers;
```

Skipping elements

```
const numbers = [1, 2, 3];
const [firstNumber, , thirdNumber] = numbers;
```

Swapping values

```
let a = 1;
let b = 2;
[a, b] = [b, a];
```

Nested arrays

extract the elements from both the outer and inner arrays

```
const numbers = [1, [2, 3], 4];
const [firstNumber, [secondNumber, thirdNumber], fourthNumber] = numbers;
```

Math methods

Math.floor()

returns the largest integer less than or equal to a given number. The returned value is rounded **down** (hence, floor) to the nearest integer.

```
const num = 5.9;
const roundedDown = Math.floor(num);
console.log(roundedDown); // Output: 5
```

Math.ceil()

returns the smallest integer greater than or equal to a given number, and the returned value is rounded up to the nearest integer.

```
const num = 5.1;
const roundedUp = Math.ceil(num);
console.log(roundedUp); // Output: 6
```

Math.random()

returns a random decimal number between 0 and 1.

```
const randomNum = Math.random();
console.log(randomNum); // Output: a random decimal number between 0 and 1
```

Math.abs()

returns the absolute value of a number

```
const num = -10;
const absoluteNum = Math.abs(num);
console.log(absoluteNum); // Output: 10
```

AEL 1.2.1

```
// Practice Time 1
const person={
  firstName: "Jane",
  lastName: "Doe",
  age: 75,
  address: "123 someStreet, someTown"
};
const {firstName, lastName, age, address} = person;
console.log(firstName);
console.log(address);

//Practice Time 2
const colors=["red", "green", "blue"];
const [color1, color2, color3]=colors;
console.log(color1);
console.log(color3);

//Practice Time 3
const person3={
  firstName: "Jane",
  lastName: "Doe",
  age: 75,
};

const {firstName:fname, lastName:lname} = person3;

console.log(fname + " " + lname);

//Practice Time 4
const numbers = [1, 2, 3, 4, 5];

const [firstNum, secondNum, thirdNum] = numbers;
console.log(firstNum);
console.log(secondNum);
console.log(thirdNum);
```

```javascript
//Practice Time 5
let students = ["John", "Jane", "Jim"];
let [first, middle, last] = students;
console.log("before, first was :" + first);
console.log("before, last was :" + last);
[first, last] = [last, first];
console.log("after, first is :" + first);
console.log("after, last is :" + last);
```