Javascript

1. Basic Syntax

1.1 출력

```
console.log("a", "b");
```

1.2 식별자

상수명 : snake_case (대문자)

변수명 : camelCase

```
var TOTAL_COUNT = 50;
var firstName = "Doojin",
    lastName  = "Park";
console.log(firstName, lastName, TOTAL_COUNT);
```

1.3 변수선언 및 연산

```
var a = 1;
var b = 2;
var result = a + b;
console.log(result);
```

1.4 연산자

+, -, *, /, %, ++, --

```
var i = 2;
i++;
console.log(i);
i--;
```

```
console.log(i);
console.log(5/3);
```

1.5 연산자 우선순위

```
var c, d;
c = 3;
d = 4;
var result = (c - d) * 5;
console.log(result);
```

1.6 데이터 타입

```
var a = 1,              // number
    b = 1.9,            // number
    c = "data",         // string
    d = [1,2,3],        // object
    e = {"a":1, "b":2}, // object
    f = true,           // boolean
    g = false;          // boolean
console.log(typeof a, typeof b, typeof c, typeof d, typeof e , typeof f, typeof g);
```

1.7 null, undefined, NaN

```
console.log(null); // 값이 없음을 지정
console.log(undefined); // 값이 지정되지 않음
console.log(NaN); // 존재하지 않는 데이터 형태
var a = null,
    b,
    c = 0/0;
```

```
console.log(a, b, c);
```

1.8 비교 연산자

1.8.1 == 값을 비교

```
console.log(1==1);      // true
console.log(1=='1');    // true
```

=== 은 데이터 타입까지 비교해준다.

=== 로 쓰는것을 권한다.

```
console.log(1===1);     // true
console.log(1==='1');   // false
```

1.8.2 ! 값을 비교

```
console.log(1!=1);      // false
console.log(1!='1');    // false
```

데이터 타입까지 비교

```
console.log(1!==1);     // false
console.log(1!=='1');   // ture
```

```
console.log(1>2);   // fasle
console.log(1<2);   // true
console.log(1>=1);  // ture
console.log(1<=2);  // ture
```

1.8.3 NaN은 비교연산으로 사용되지 않는다.

비교연산에서 어느 한쪽이 NaN이면 무조건 false

```
console.log(NaN===NaN);
```

1.9 할당 연산자

```
var a = 1;
a += 2;
console.log(a);
a -= 1;
console.log(a);
a *= 6;
console.log(a);
a /= 2;
console.log(a);
a %= 5;
console.log(a);
```

1.10 논리 연산자

1.10.1 && : 모두 참일때 참 (and)

```
console.log(true && true);      // true
console.log(true && false);     // false
console.log(false && false);    // false
```

1.10.2 || : 하나라도 참이면 참 (or)

```
console.log(true || true);      // true
console.log(true || false);     // true
console.log(false || false);    // false
```

2. Condition - 조건문

2.1 if, else if, else

```
if(true){
    console.log("hello javascript");
}

if(false){
    console.log("hello javascript");
} else {
    console.log("hello datascience");
}

if(false){
    console.log("hello javascript");
} else if(true){
    console.log("hello html");
} else {
    console.log("hello datascience");
}
```

2.2 false로 간주되는 데이터 형

```
if(null || undefined || NaN || 0 || ""){
    console.log("hello false");
}
```

2.3 ture로 간주되는 데이터형

```
if([] && {}){
    console.log("hello true");
}
```

문제. 점수를 입력하면 학점이 나오는 코드를 작성하시오.

3. function - 함수

3.1 선언 및 호출

```
function add(a, b){
    return a + b;
}
var result = add(3, 5);
console.log(result);
```

3.2 함수를 변수로 사용

```
var add = function(a, b){
    return a + b;
};
var result = add(3, 5);
console.log(result);
console.log(typeof add);
```

4. loop - 반복문

4.1 while

```
var a = 0;
while(a < 5){
    a++;
    console.log(a);
}
```

4.2 for

```
for(var i = 0; i < 3; i++){
    console.log(i);
}
```

4.3 break

```
var a = 0;
while(a < 5){
    a++;
    if(a === 3){
        break;
    }
    console.log(a);
}
```

4.4 contiune

```
var a = 0;
while(a < 5){
```

```
    a++;
    if(a === 3){
        continue;
    }
    console.log(a);
}
```

문제 1. 구구단 가로 (for, while)

```
for(var num1=2; num1<10; num1++){
    for(var num2=1; num2<10; num2++){
        console.log(num1 + "*" + num2 + "=" + num1*num2);
    }
    console.log();
}
```

문제 2. 구구단 세로 (for, while)

줄바꿈 안되게 출력 process.stdout.write()

```
for(var num2=1; num2<10; num2++){
    for(var num1=2; num1<10; num1++){
        process.stdout.write(num1 + "*" + num2 + "=" + num1*num2 + "\t");
    }
    console.log();
}
```

5. Array - 배열

5.1 배열 선언

```
var arr = ['a','b','c','d','e'];
```

5.2 특정 위치 데이터 가져오기

```
console.log(arr[2]);
```

5.3 배열의 크기 (length)

```
console.log(arr.length);
```

5.4 배열 추가

```
arr.push('f'); // 뒤에 추가
console.log(arr);
arr.unshift('z'); // 앞에 추가
console.log(arr);
```

5.5 제거

```
arr.shift(); // 첫번째 배열 제거
console.log(arr);
arr.pop(); // 마지막 배열 제거
console.log(arr);
```

5.6 정렬

```
arr.reverse(); // 역순으로 정렬
console.log(arr);
arr.sort(); // 오름차순으로 정렬
```

```
console.log(arr);
```

5.7 배열 자르기

```
arr.splice(2,1); // 2번에서 1개 자름
console.log(arr, arr.length);
delete arr[2]; // 2번이 삭제 하지만 데이터만 삭제, 공간은 남아있음
console.log(arr, arr.length);
```

5.8 배열 데이터 하나씩 사용하기

```
for(var i = 0; i < arr.length; i++){
    console.log(arr[i]);
}
```

6. Object - 객체

6.1 객체 생성

```
var obj = {};
obj.math = 92;
obj.english = 97;
obj.science = 85;
console.log(obj);
console.log(obj.english);
```

6.2 객체 출력

```
for(var key in obj){
    console.log(key, obj[key]);
}
```

6.3 객체에 함수 담기

Object.keys : 객체의 키값 리스트로 가져오기

toFixed(number) : 반올림해서 number 자리수까지 출력

```
var pointsObj = {
    'points':{'math': 91, 'science': 98, 'english': 86},
    'total': function(){
        var total = 0;
        for(var key in this.points){
            total += this.points[key];
        }
        return total;
    },
    'avg': function(){
        return this.total() / Object.keys(this.points).length;
    }
};

console.log(pointsObj.total());
console.log(pointsObj.avg());
console.log(pointsObj.avg().toFixed(2));
```

7. Scope - 스코프

7.1 함수 밖에서 선언된 변수를 함수 안에서 사용

```
var a = 'hello';
function func(){
```

```
    console.log(a);
}
func();
```

7.2 함수 안에서 var를 사용해서 선언

```
var a = 'hello';
function func(){
    var a = 'javascript';
    console.log('inner', a);
}
func();
console.log('outer', a);
```

7.3 함수 안에서 var를 사용하지 않고 선언

```
var a = 'hello';
function func(){
    a = 'javascript'; // global로 선언이 된다.
    console.log('inner',a);
}
func();
console.log('outer',a);
```

7.4 전역변수를 사용하지 않는 방법

익명함수 : 모든 코드를 익명함수 안에서 처리한다.

```
(function(){
    var a = 'hello';
    console.log(a);
```

```
}());
console.log(a);
```

8. callback - 콜백함수

함수내에서 모든 동작이 끝나고 실행시키는 함수를 파라미터로 넘겨서 사용하는 방법

웹에서 비동기 통신을 할때 많이 사용됨

```
function add(a, b, callback){
    var result = a + b;
    callback(result);
}

function disp(data){
    console.log(data);
}

add(5, 8, disp);
```