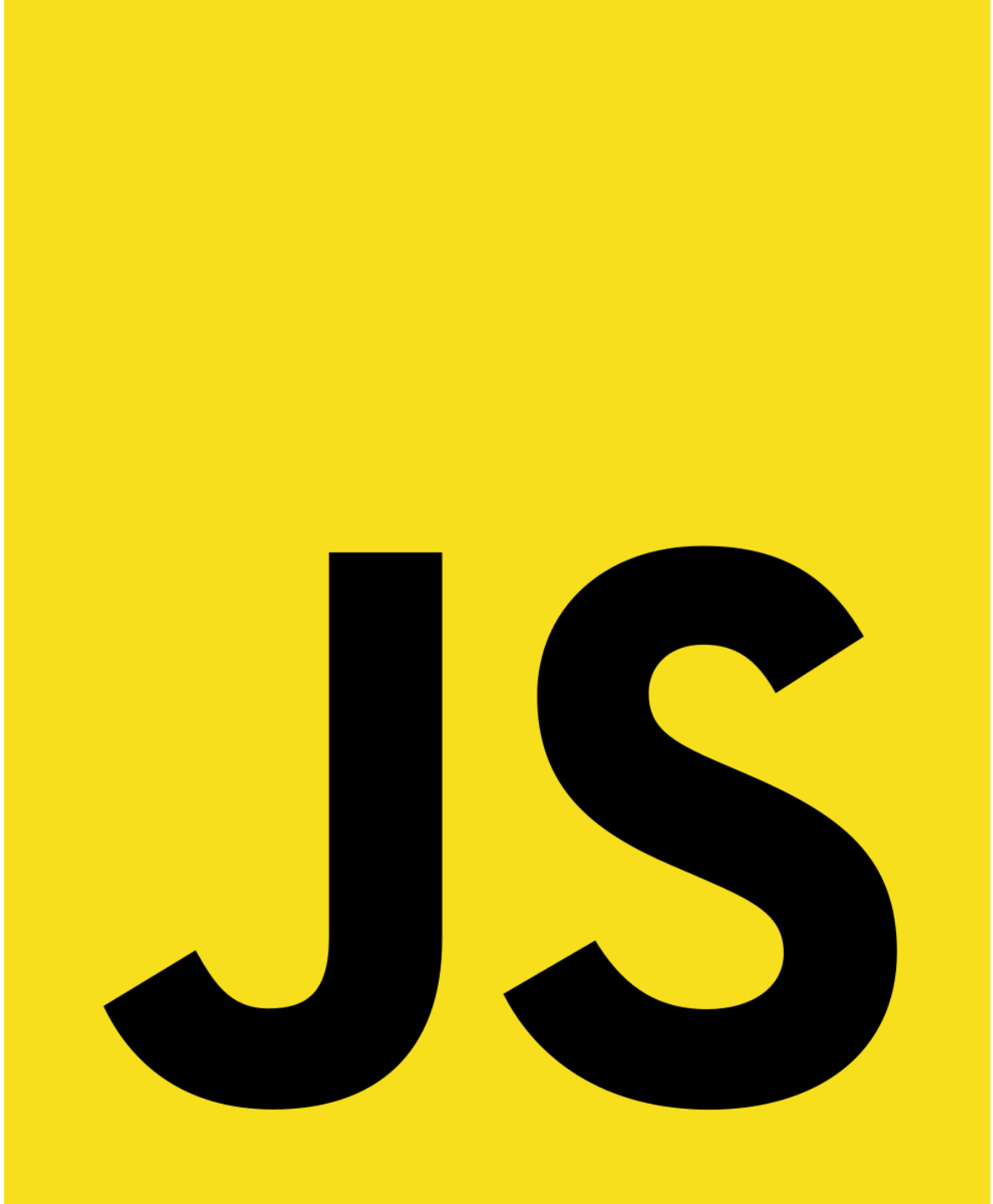
JavaScript _ 변수, 연산자, 함수

OYR 2조 _ 김주환, 오다혜, 이정화

OYR _ JavaScript _ contents

JavaScript?

HTML, CSS와 함께 웹을 구성하는 요소 중 하나
객체(object) 기반의 스크립트 언어
웹페이지가 동작하는 것을 담당



;

Semicolon

문장 뒤에 세미콜론으로 문장 마침 표시

' '

** **

스크립트는 공백(띄어쓰기) 무시

{ }

Curly Bracketes

중괄호를 통해 그룹화

‘ C언어 문법과 유사 ’

변수

—

변수 _ 바뀔 수 있는 값

변수 선언 _ let

상수 _ 고정된 값

상수 선언 _ const

선언은 보통 맨 윗줄에 함

변수 선언에 var라는 키워드를 사용하여 선언하는 방법도 있으나 모던 자바스크립트에서는 더 이상 사용하지 않음

```
let a,b;
const c = 20;
a = 10;
b = a;
c = a; // 오류 발생
```

변수 이름의 주의사항

—

변수 명의 첫 번째 글자에 숫자 사용은 불가 (ex. 1number - 불가능)

_ , $ 외의 특수 기호 사용 불가, 띄어쓰기 불가

주로 구체적인 이름 사용 (ex. saveMoney)

예약어 사용 불가 (ex. var, function 등)

대소문자 구분해서 사용 (ex. Name != name)

대입 연산자 _ ' = '

정수, 실수, 문자열 모두 대입 가능
표현식에 변수 값 포함 가능

—

할당 연산자 _ ' =, +=, −=, *= ' 등

뒤에 있는 식을 앞에 있는 식과 계산
a += b + c − d * e == a = a + (b + c − d * e)

```
var tmp, a;
tmp = 10;

a = 100;
a = 9.51;
a = tmp;
a = "hello";
```

산술 연산자

+ - * / _ 사칙 연산

** _ 지수 표현 (ex. 2**3 = 8)

% _ 나머지 표현 (ex. 10 % 3 = 1)

++ _ 1 증가 (a++ == a = a + 1)

-- _ 1 감소

문자열 연산자

문자열도 + 기호로 붙일 수 있음.

문자열 + 숫자 → 문자열 반환됨
(ex. Hello + 5 = Hello5)

비교 연산자와 논리 연산자

== _ 같다

>= _ 크거나 같다

<= _ 작거나 같다

!= _ 같지 않다

=== _ 값도 같고 타입도 같은지

!== _ 값이 안 같거나 타입이 안 같거나(ex. “5” == 5 , “5” !== 5)

&& _ and

|| _ or

! _ not

```
function functionName(parameters,parameters){

  return parameters^^2;

}
```

함수 정의 / 함수 선언문

```
let a = function (parameters, parameters){return parameters % 3};
```

함수 표현식

```
let a = new Function("parameter1","parameter2","return parameter1 * parameter2");
```

생성자를 통한 선언

전역

—

함수 밖에서 선언되며,
프로그램 전체에서 접근, 사용

지역

—

함수 내부에서 선언되며, 선언된 함수 안에서만 사용되고 함수가 종료되면 사라진다.

```
let a; //전역 변수

function myFunction(){

  let b; //지역 변수

}
```

```
function myFunction(){// 전역 함수

    function myFunction(){ //지역 함수



    }
}
```

재귀함수

함수 내에서 자기 자신을 호출하는 함수

보통은 탈출 조건을 걸어줌
→ 탈출 조건이 없으면 무한 루프

C언어의 while 문과 유사

```
let cnt = 0;
function myFunction(){

  cnt++; //함수 실행 시 cnt 증가

  document.write(cnt,"<br>");

  if (cnt == 10) { //cnt의 값이 10이면 종료됩니다.
    return;
  }
  myFunction(); //cnt 10되기 전까지 다시 호출 -> 재귀

}

myFunction(); //함수 호출(처음 실행)
```

즉시 실행 함수

정의하자마가 바로 호출하는 함수 - 실행문 필요 X

기존의 함수는 변수에 함수를 저장하고 실행함 → 실행문 필요

Function을 ( ) 괄호로 묶고 뒤에 ( )를 붙여서 함

```
(function sqare(x){

  console.log(x*x);

})(2); //result - 4
```

OYR _ JavaScript _ 04 modal window

```
<script>
    document.getElementById("modal_open_btn").onclick = function() {
        document.getElementById("modal").style.display="block";
    }

    document.getElementById("modal_close_btn").onclick = function() {
        document.getElementById("modal").style.display="none";
    }
</script>
```