1주차 2차시

5장_표현식과_문.pdf

6장 데이터 타입.pdf

7장_연산자.pdf

8장_제어문.pdf

5장 과제

1. 아래의 코드에서 표현식인 부분과 표현식이 아닌 부분에 대해서 구분하시오

```
var x;
x=100;
//*
----------------------------
*/
var a = y = 100;
console.log(a);
//*
----------------------------
*/
var foo = var x
```

A:

표현식: x = 100;, var a = y = 100;

표현식 아님: var x;, console.log(a);, var foo = var x

6장 과제

1. 그렇다면 위에 설명과 같이 다 실수로 측정한다면 2진수, 8진수, 16진수를 출력하면 어떤식으로 될까?

```
var binary = 0b01000001;
var octal = 0o101;
var hex = 0x41;
console.log(binary,octal,hex);
if(binary === hex) console.log(true);
if(binary === octal) console.log(true);
```

A:

65 65 65

true

true

2. 다 실수라면 아래의 비교문의 결과는 어떻게 나올까?

```
console.log(1 === 1.0);
console.log(4 / 2);
console.log(3 / 2);
```

A:

true

2

1.5

3. 세가지를 console을 이용하여 도출해보세요.
    - Infinity: 양의 무한대
    - -Infinity: 음의 무한대
    - NaN(Not a Number): 산술 연산 불가

A:

Infinity → console.log(1/0);

-Infinity → console.log(-1/0);

NaN → console.log(1/0 + (-1/0));

4. 만약 NaN이 아닌 nan, NAN 같이 변수에 대입하면 어떤 식으로 나올까요?

A: 정의되지 않았기 때문에 오류가 발생한다.

5. “” 안의 ‘’(single quote)은 뭘로 인식되고 ‘’(single quote) 안의 “”은 뭘로 인식될까?

A: 문자로 취급되고 “’a’” → ‘a’, ‘”a”’ → “a” 로 출력 된다.

6. 아래의 코드는 어떤식으로 다를까?

```
console.log(`a + b = ${1 + 2}`);
console.log('a + b = ${1 + 2}');
```

A:

첫번째 출력은 표현식을 사용할 수 있기 때문에 ${1 + 2} 부분이 3으로 출력되고

두번째 출력은 문자로 인식되기 때문에 ${1 + 2} 그대로 출력된다.

7. 의도적 부재를 왜 사용할까?

A:

8. 과연 아래의 사용법이 옳은 선택일까? 다른방법으로 변수를 소멸시키는게 좋지 않을까?

A: 변수의 lifecycle을 짧게 만들어 변수 자체를 소멸시키는 것이 좋을 것 같다.

9. ECMAScript 사양은 문자열과 숫자 타입 외에는 명시적으로 규정하고 있지 않은데 그렇다면 해당 데이터 타입들 외에는 어떤 식으로 계산되고 있는가?

A:

10. 심벌 테이블 이라는 뜻을 알아보시오

A: 심벌 테이블이란 컴파일러 또는 인터프리터에서 사용되는 데이터 구조로, 일반적으로 해시 테이블을 사용해 구현된다. 심벌 테이블은 데이터 객체의 이름을 저장하는 테이블이다.

11. 대표적인 동적/정적 언어를 조사해보시오

A:

동적 언어: JavaScript, Python, Ruby

정적 언어: C, C++, C#, Java

7장 과제

1. 아래의 코드를 실행해라

```
var a = '1';
console.log(+a, typeof +a);
console.log(a, typeof a);
a = true;
console.log(+a, typeof +a);
console.log(a, typeof a);
a = false;
console.log(+a, typeof +a);
console.log(a, typeof a);
a = 'Hi';
console.log(+a, typeof +a);
console.log(a, typeof a);
```

A:

1 ‘number’

1 string

1 ‘number’

true ‘boolean’

0 ‘number’

false ‘boolean’

NaN ‘number’

Hi string

2. **암묵적 타입 변환** 또는 **타입 강제 변환**에 대해서 알아보시오.

A:

암묵적 타입 변환: 예상치 못한 타입을 받았을 때 예상 가능한 타입으로 바꿔준다. 이를 통해 숫자 값을 넘겨야 하는 곳에 문자열을 넣을 수 있고, 문자열을 넣어야 하는 곳에 객체를 넘길 수도 있다. → 자바스크립트의 주요한 기능 중 하나인 동시에 가장 피해야 할 기능 중 하나.

타입 강제 변환: 타입이 다른 타입으로 변환되는 것. 명시적 변환은 Number(), String() 등을 사용하여 타입을 강제로 변환하는 경우이다. 암시적 변환은 특정 연산자 또는 표현식이 사용될 때 동작한다. ex) 1 == '1' → true

3. 아래의 비교가 뭐가 다른지 알아보시오.

```
5 == 5;
5 == '5';
// ================================================
5 === 5;
5 === '5';
// ================================================
'0' == '';
0 == '';
0 == '0';
// ================================================
false == 'false';
false == '0';
false == null;
false == undefined;
// ---
NaN === NaN
0 == -0
0 === -0
```

A:

== 은 느슨한 동등 비교로 양변에 타입 변환을 통해 양변의 값만 같으면 true 를 반환하고 같지 않으면 false를 반환한다.

=== 은 엄격한 동등 비교로 타입 변환을 시도하지 않고 같음을 비교한다.

결과:

true

true

true

false

false

true

true

false

true

false

false

false

true

true

4. -0 === 0;
   Object.is(-0,0)

   NaN === NaN;
   Object.is(NaN,NaN);
   의 결과가 왜 다를까?

A:

===은 값의 같음을 비교하지만 Object.is는 기능의 같음을 비교하기 때문이다.

-0과 0은 값으로는 동일하지만 1 / 0 == Infinity, 1 / -0 == -Infinity 와 같이 기능적으로는 다르기 때문에 Object.is를 사용할 경우 false가 출력된다.

NaN은 값 자체끼리는 같이 않지만, 숫자가 아닌 것을 의미하는 기능적으로는 동일하기 때문에 Object.is를 사용할 경우 true가 출력된다.

5. 위에 있는 반환 값을 다 나타내보시오.

```
typeof 1; // number
typeof 'a'; // string
typeof true; // boolean
var undefinedValue;
typeof undefinedValue; // undefined
typeof Symbol(); // symbol
```

```
typeof Object(); // object
typeof alert; // function
```

8장 과제