2주차 5차시

12장_함수.pdf

12장 과제

1. **아래의 빈칸과 서로 차이점을 서술하시오.**

```
function add(x, y) // x, y를 뭐라하는가: 매개변수
{
  return x + y;
}
add(2,5) // 들어가는 값에 대한 단어: 인수
```

2. **본인이 생각하기에 이상적인 개발자는 어떤 형태인가?**
   해당 질문에 대해서 진지하게 한번쯤은 고민할 만 합니다. 보통 프로그래머로 취업을 한다면 인터뷰에서 무조건 물어보는 내용이기도 합니다.
   a. 제게 있어서 이상적인 개발자란 미지의 공간을 탐험하는 모험가와도 같습니다. 주어진 도구를 이용해 장애물을 넘어가고, 막힌 길을 뚫으며 가끔은 도구를 만들어 내기도 합니다. 그렇게 울고 웃으며 앞으로 나아간 끝에 마침내 문제를 해결해내는 모습이 진짜 개발자라고 생각합니다.
3. **선언문에서는 함수 이름을 생략할 수 없다. 만약 함수 이름을 생략하면 나오는 에러는 어떤건지 확인해보세요.**

```
function (x,y)
{
    return x + y;
}
VM414:1 Uncaught SyntaxError: Function statements require a function name
```

4. **{}는 블록문일까 객체 리터럴일까? 본인의 생각을 쓰고 그 이유에 대해서 서술하시오.**
   둘 다 가능하다고 생각한다. 반복문이나 조건문의 {}는 블록문이지만, 함수를 호출하거나 변수를 할당하는 것이 있어서 {}는 객체 리터럴이기 때문이다.
5. **하단의 에러는 왜 날까?**

```
var add1 = (function()
{
  var a = 10;
  return function (x, y)
  {
    return x + y + a;
  };
}());

console.log(add1(1,2)); // 13

var add2 = (function()
{
  var a = 10;
  return new Function('x', 'y', 'return x + y + a;')
}());

console.log(add2(1,2)); // ReferenceError: a is not defined
```

new 키워드를 사용해 만들어진 함수는 자신 이외의 공간에 접근할 수 없기 때문에 new로 함수가 만들어질 때 포함되지 않은 a는 정의되어 있지 않으므로 에러가 발생한다.

6. **아래 함수를 실행해보고 결과 값을 적으시오.**

```
function add(x,y)
{
  console.log(x,y);
  return x + y;
}

add(2, 5);

console.log(x, y);
```

VM133:9 Uncaught ReferenceError: x is not defined
at <anonymous>:9:13

```
function add(x, y)
{
  return x + y;
}

console.log(add(2));
```

undefined

```
function add(x, y)
{
  console.log(arguments);
  return x + y;
}

console.log(add(2,5,10));
```

undefined

7. **재귀함수로 팩토리얼을 구현해보시오. 그리고 해당 코드에 대한 리뷰를 해 보세요.**

```
function factorial(n)
{
    // n == 0 또는 n == 1 일 때 팩토리얼의 값은 1
    if(n == 0 || n == 1)
    {
        return 1;
    }
    // n과 n - 1의 팩토리얼 값을 곱함(재귀)
    return n * factorial(n - 1);
}
```

8. **callback 지옥이라는 말이 유명하다. 직접 지옥을 만들어보자.**
   그리고 callback 지옥이 왜 위험한지 서술하시오.

```
function randomTime()
{
    return Math.floor(Math.random() * 10) * 1000;
}
function getChicken()
{
    setTimeout(() =>
        {
            console.log("동묘시장->chicken");
        }, randomTime());
}
function getEgg()
{
    setTimeout(() =>
        {
            console.log("동묘시장->chicken->egg");
        }, randomTime());
}
function getMeal()
{
    setTimeout(() =>
        {
            console.log("동묘시장->chicken->egg->fried egg");
```

```
        }, randomTime());
}

getChicken();
getEgg();
getMeal();
```

callback 지옥은 가독성이 매우 떨어지고, 디버깅이 매우 어려워진다.

9. **아래의 코드 중 어떤 것이 순수 함수이며 어떤 것이 비순수 함수인지 서술하시오.**

```
var count = 0;

function increase(n)
{
  return ++n;
}

count = increase(count);
console.log(count);

count = increase(count);
console.log(count);
```

순수 함수이다.

외부 상태에 전혀 영향을 받지 않고 매개변수 n 을 통해서만 값을 생성하기 때문이다.

```
var count = 0;

function increase()
{
  return ++count;
}

count = increase(count);
console.log(count);

count = increase(count);
console.log(count);
```

비순수 함수이다.

전역변수에 접근해 값을 생성했기 때문이다.

Call By Reference & Call By Value

Call By Reference와 Call By Value는 모두 함수의 호출 방식이다.

Call By Value의 경우는 함수 호출 시 전달되는 변수의 값을 복사하여 함수의 인자로 전달한다.

이렇게 복사된 인자는 함수 안에서 지역적으로 사용되는 local value의 특성을 가진다.

그렇기 때문에 함수 내부에서 인자의 값을 변경해도 외부의 변수 값은 변경되지 않는다.

Call By Reference의 경우는 참조에 의한 호출 방식으로 함수를 호출 했을 때 인자로 전달되는 변수의 레퍼런스를 직접 전달한다.(해당 변수를 가르킨다.)

그렇기 때문에 함수 내부에서 인자의 값을 변경하면 외부의 변수 값도 변경된다.