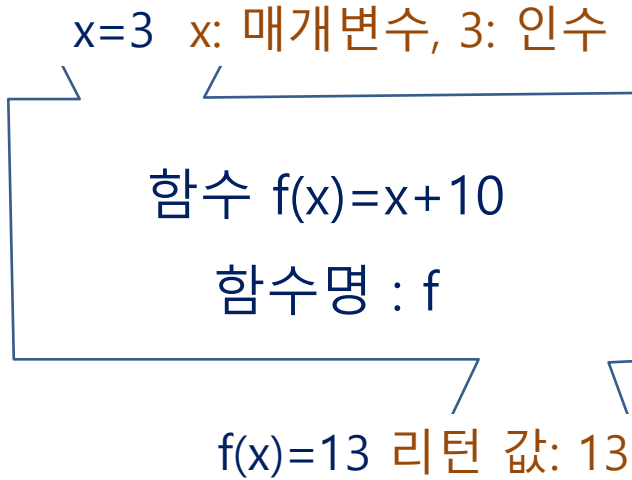


04

# 함수

## ❖ 함수(function) : 특정 기능을 수행하는 코드들의 집합



```
function f(x){
    return x+10;
}
console.log(f(3));
```

함수도 하나의 자료형

### 함수 선언

```
function 함수명(매개변수){
    함수코드
    return();
}
```

### 함수실행(호출 한다)

```
함수명(인수:매개변수 값);
```

## ❖ 함수(function) 선언

함수 정의 방식1 : 선언문 형태

```
function myFunc(param){  
  console.log(param + " run!");  
}
```

함수 정의 방식2 : 함수 표현식(리터럴 방식)

```
const myFunc2 = function(param){  
  console.log(param + " run!");  
}
```

함수 사용 : 호출

```
myFunc("func1");  
myFunc2("func2");
```

## ❖ 매개변수와 리턴값

- 매개변수의 개수가 맞지 않아도 허용
  - 추가된 매개변수 : 무시
  - 지정되지 않은 매개변수 : undefined 입력

```
function sum(a=0, b=0){ => 기본값 지정
```

```
    return a+b;
```

전개 연산자 : ... 변수명

```
}
```

```
console.log(sum(4,5));
```

```
console.log(sum(4));
```

```
console.log(sum());
```

- return : 함수 실행도중 호출한 곳으로 돌아가라는 의미
  - 아무 값도 리턴하지 않은 경우 자료형과 값 모두 undefined

## ❖ 함수(function)

- 함수 사용의 목적은 코드 재사용성
- 익명함수 : 이름이 없는 함수

```
function (){ };
```

```
let 함수명 = function (){ } : 변수에 넣어 사용
```

- 선언적 함수 : 이름이 있는 함수

```
function 함수명( ){ }
```

- 자기 호출 함수(즉시실행) : 함수를 정의함과 동시에 실행하는 함수

```
( function() { } ) ( ); → 변수에 할당 가능
```

- 콜백 함수 : 다른 함수의 매개변수로 전달하는 함수
- 화살표 함수 : 함수 선언을 간단하게 작성

```
( ) => { }
```

## ❖ 익명 함수와 선언적 함수

// 익명함수 - 선언 전에 호출하면 오류 발생

```
func();
```

```
var func = function(){alert('func A')};
```

```
var func = function(){alert('func B')};
```

// 선언적 함수 - 함수 선언이 먼저 실행 되므로 호출 가능

```
func(); - ③
```

```
function func(){alert('func A')};
```

- ①

```
function func(){alert('func B')};
```

- ②

// 실행순서 ①②③, 선언함수

## ❖ 익명 함수와 선언적 함수

// 함수 생성

var func = function(){alert('func A')}; - ②

function func(){alert('func B')}; - ①

// 함수호출

func(); - ③

// 실행 결과는?

127.0.0.1:5500 내용:

func A

확인

## ❖ 화살표 함수(익명함수에서만 사용)

```
let sum = function(a, b){  
  return a + b;  
}
```

```
console.log(sum(5, 10));
```

```
const hi = function(){  
  return "hi?";  
}
```

```
let hi = function(user){  
  alert('hi?');  
}
```

```
let sum = (a, b) => {return a + b}
```

```
let sum = (a , b) => a + b;
```

```
console.log(sum(5, 10));
```

```
const hi = () => {return "hi?"}  
const hi = () => "hi?"
```

```
let hi = (user) => alert('hi?');
```



## ❖ 내부함수

- 다른 개발자와 충돌을 방지 하기 위해 사용(함수명, 변수 등)
- pyta() 함수 외부에서 내부함수 square() 함수를 사용할 수 없다

```
function pyta(width, height){  
    function square(x){  
        return x*x;  
    }  
    return Math.sqrt(square(width)+square(height));  
}  
  
let result = pyta(4, 9);  
console.log(result);
```

## ❖ 콜백 함수

- 함수를 매개변수로 전달되는 함수
- 함수도 하나의 자료형이므로 전달가능

```
function callTenTimes(callback){  
    for ( let i = 0; i<10; i++){  
        callback();  
    }  
}  
  
let callback = function(){  
    console.log('함수호출')  
}  
  
callTenTimes(callback);
```

## ❖ 리턴되는 함수 - 클로저를 위해 사용

- 함수의 결과 값으로 리턴 되는 함수

```
function returnFunction(){  
    return function(){  
        console.log('hello Function');  
    };  
}  
returnFunction();
```

```
returnfunction();  
⇒ ( function(){ console.log('hello Function')} )();
```

## ❖ 클로저

- 지역변수는 함수 외부에서 사용할 수 없고, 함수가 종료되면 제거
- 클로저 : - 지역변수를 남겨두는 현상, 남겨진 지역변수
  - 내부 변수들이 남아있는 함수 공간,
  - 리턴된 함수 자체
- 반드시 리턴된 클로저 함수를 통해 지역변수 사용

```
function test(name){  
  let output = 'hello' + name;  
}  
console.log(output);  
// output -> undefined
```

```
function test(name){  
  let output = 'hello' + name;  
  
  return function(){  
    console.log(output);  
  };  
}  
test('javascript')();
```

## ❖ 클로저

```
function test(name){  
  let output = 'hello! ' + name;  
  
  return function(){  
    console.log(output);  
  };  
}  
  
let test1 = test('web');  
let test2 = test('javascript');  
test1();  
test2();
```

```
test1() : (function(){  
  console.log('hello! web');  
})();  
  
test2() : (function(){  
  console.log('hello! javascript');  
})();
```

## ❖ 커링(curring)

**curring 함수 : 한번에 인수를 하나만 받아 처리하는 함수**

```
function orderSet(burger){  
  return function orderBeverage(beverage){  
    return function orderSide(side){  
      return function orderKetchup(yn){  
        return function orderChicken(count){  
          console.log("세트:"+burger+" , " + beverage + " , "+ side + " , 케첩  
            (" + yn + " ), 조각치킨(" + count + "개");  
  
          } } } } }  
let order = orderSet ("치즈버거")("콜라");  
order("프렌치후라이")("y")("2");
```

## ❖ 커링(curring)

```
const orderSet = burger => beverage => side => yn => count => {  
  console.log("세트:" + burger + " , " + beverage + " , " + side + " , 케첩  
    (" + yn + " ), 조각치킨(" + count + "개)");  
}  
  
let order = orderSet ("치즈버거")("콜라");  
order("프렌치후라이")("y")("2");
```

=> 실행구조를 단순화 해서 쉽게 읽을 수 있는 것이 장점

## ❖ 펼침연산자(spread operator)

```
let calc =function(x, y, ...restparams){  
    return x + y + restparams.reduce(function(sum, param){  
        return sum + param;  
    });  
}  
  
let arr = [0,1];  
console.log(calc(-1,...arr, 2, ...[3]));  
console.log(calc(-1,0, 1,2,3));  
let arr2=[1,2,3,4,5,6];  
console.log(calc(...arr2));  
console.log(calc(null, ...arr2));
```



## ❖ 변수의 스코프(scope)

```
var a=1;
var b=5;
```

```
function outfunc(){
    function innerfunc(){
        a=b;
    }

```

```
    console.log(a);

```

```
    a=3;

```

```
    b=4;

```

```
    innerfunc();

```

```
    console.log(a);

```

```
    var b= 2; console.log(b); ----- ?

```

```
    }

```

전역 a=1, b=5

출력 a=1

전역 a=3, 지역 b=4

전역 a=4, 지역 b=4

출력 a=4

지역 b=2

outfunc(); console.log(b);-----?

## ❖ 함수 용어 정리

용어	설명
함수	함수는 하나의 자료형
호출	선언된 함수의 내부 코드를 실행하는 것
매개변수	함수 내부로 자료 값을 넘기기 위해 사용하는 변수
리턴	함수 실행 결과값을 호출한 곳으로 넘기는 것
인수	매개변수로 전달되는 값
익명함수	이름 없이 선언된 함수, 변수에 대입가능
선언적 함수	이름이 있는 함수
콜백 함수	매개변수로 전달되는 함수
클로저	리턴된 함수를 통해 지역변수에 접근하기 위한 것

## ❖ 도전! 문제

### 문제1

학생 3명의 국어, 영어, 수학 점수를 넘겨 받아 평균을 계산하는 프로그램을 작성하세요.

- 평균을 구하는 함수명 : sungJuk

<콘솔창 출력 내용>

홍길동		70		60		90		평균: 73.33333333333333
-----	--	----	--	----	--	----	--	-----------------------

성춘향		90		70		90		평균: 83.33333333333333
-----	--	----	--	----	--	----	--	-----------------------

이몽룡		80		60		70		평균: 70
-----	--	----	--	----	--	----	--	--------

## ❖ 도전! 문제 풀이

### 문제1

```
function sungJuk(kor, eng, mat){  
  let avg = (kor + eng + mat)/arguments.length;  
  return avg;  
}  
  
console.log(`홍길동 | 70 | 60 | 90 | 평균: ${sungJuk(70, 60, 90)}`);  
console.log(`성춘향 | 90 | 70 | 90 | 평균: ${sungJuk(90, 70, 90)}`);  
console.log(`이몽룡 | 80 | 60 | 70 | 평균: ${sungJuk(80, 60, 70)}`);
```