**11회차(1)**

< 함수(function) vs 메소드 >

객체지향 언어에서 클래스 내에서 정의되는 함수가 메소드

자바는 반드시 클래스가 있어야 한다 -> 자바에서는 전부 메소드라고 부른다

우리가 알고 있는 함수가 맞다. 그러나 엄밀한 수학적 정의의 함수를 그대로 사용 x, 차용해서 사용

수학에서의 함수는 어떤 집합 X와 Y 사이에 대한 관계

프로그래밍 언어에서는 프로그램을 이루는 기본 단위

- 명령어들의 집합

- 소스코드 관리의 편리성 [재활용성]

자바에서 메소드의 기본 형태

[반환타입] [메소드 이름](타입 변수명, 타입 변수명, ...){

메소드 블록

}

자바에서 main 메소드는 특별한 메소드

- 누군가에 의해 자동으로 실행됨

- 자바에서 모든 코드들은 전부 main 메소드부터 실행 시작됨

메소드에서 반드시 리턴값을 정의해줄 필요는 없다.

- 반환 타입 (double, int, ...) -> void로 변경

- return (kor + eng + math) / 3.0; 대신

-> 출력식 System.out.println((kor + eng + math) / 3.0);

메소드에서 리턴은 한 번만 쓸 수 있다

- 단 if(true) return ~;

else return ~; 은 가능하다

변수의 타입에 따라 입력값 전달

- 자바에서 변수의 종류는 두 가지

- 기본형 변수(boolean, byte, char, short, int, long, float, double)

- 참조형 변수(String, Array, ...)

- 값에 의한 참조 Call by value, 참조에 의한 참조 Call by reference

기본형 변수의 전달

- 지역 변수: 메소드 내에서 정의된 변수

- 유효기간 존재 \_ 메소드가 실행되는 동안에만 유효

참조형 변수의 동작은 기본형 변수와는 다르다

- 서로 다른 두 지역변수가 참조하는 변수가 같은 변수가 된다

참조형 변수의 반환

- 입력값 전달과 마찬가지로 동일하게 동작을 한다

**11회차(2)**

package classname;

import java.util.Scanner;

public class YourClassNameHere {

/// 국어, 영어, 수학 점수를 입력 받아 평균을 구하는 메소드

static double ave(int kor, int eng, int math) // static 작성해주어야 한다!, double형으로 수정

{

return (kor + eng + math) / 3.0; // return : 계산된 값을 호출한 이에게 돌려줌

}

public static void main(String[] args) {

// 메소드를 실행하려면 정의된 메소드의 이름으로 호출해줘야 한다.

// 반드시 정의된 매개변수의 개수와 순서 맞춰주어야 함

Scanner sc = new Scanner(System.in);

System.out.print("국어 점수 : ");

int kor = sc.nextInt();

System.out.print("영어 점수 : ");

int eng = sc.nextInt();

System.out.print("수학 점수 : ");

int math = sc.nextInt();

// 1) 새 변수 만들어 대입하기

double average = ave(kor, eng, math);

System.out.println("평균 : "+average);

// 2) 출력식에 그대로 대입하기

System.out.println("평균 : "+ave(kor, eng, math));

}

}

**11회차(3)**

package classname;

public class MethodArray {

// 국영수 점수를 하나의 배열로 입력 받아 평균 전달

static double ave(int[] score)

{

return (score[0] + score[1] + score[2]) / 3.0;

}

public static void main(String[] args) {

int[] arr = {20, 50, 70};

double average = ave(arr);

System.out.println("평균 : "+average);

}

}

**11회차(4)**

package classname;

public class MethodArray1 {

static void pri\_mthod(int a, int b)

{

System.out.printf("Before a : [%d], b : [%d]\n", a, b);

// a와 b의 값을 교환해보자(1)

System.out.printf("After a : [%d], b : [%d]", b, a);

}

public static void main(String[] args) {

int a = 10;

int b = 20;

// a와 b의 값을 교환해보자(2)

int num;

int num1;

num = a; // a의 값을 담아둘 변수

num1 = b; // b의 값을 담아둘 변수

b = num; // b에 num 대입

a = num1; // a에 num1 대입

// a와 b의 값을 교환해보자(3)

int tmp;

tmp = a;

a = b;

b = tmp;

pri\_mthod(a, b);

}

}

**11회차(5)**

package classname;

import java.util.Arrays;

public class MethodArray2 {

static void refer\_mthod(int[] arr)

{

// 전달 받은 배열을 정렬

Arrays.sort(arr);

}

public static void main(String[] args) {

int[] arr = {5, 4, 3, 2, 1};

System.out.println("함수를 호출하기 전");

System.out.println(Arrays.toString(arr));

refer\_mthod(arr);

System.out.println("함수를 호출한 후");

System.out.println(Arrays.toString(arr));

}

}

**11회차(6)**

package classname;

import java.util.Arrays;

public class MethodArray3 {

static int[] average(int kor, int eng, int math)

{

int[] arr = {kor, eng, math};

return arr;

}

public static void main(String[] args) {

int[] arr = average(20, 50, 70);

System.out.println(Arrays.toString(arr));

}

}