객체지향 프로그래밍

□이론 //자바 자동 줄띄움 : Ctrl+Shift+f

* 객체 지향 프로그래밍이란? : 모든 데이터를 오브젝트로 취급하여 프로그래밍 하는 방법으로, 처리 요구를 받은 객체가 자기 자신의 안에 있는 내용을 가지고 처리하는 방식이다.

* JRE : 자바 프로그램을 실행하기 위한 일반 사용자들이 설치하는 환경을 꾸미는 것
* JDK : 자바 언어로 프로그램을 개발하기 위한 도구를 설치하는 것,
  자바를 실행하기 위해 JDK에 JRE가 포함되어 있다.

* Class : 클래스를 만드는 것은 오브젝트의 특징을 만드는 것이다, 계급 (붕어빵 틀)
  / 클래스 파일에는 반드시 하나의 자바 클래스만이 들어있다.

* 패키지 : 클래스를 모아놓은 덩어리

* 메소드 : 클래스에서 쓰는 방법, 수단, C에서 배운 함수, 다른 이름으로 클래스 펑션 (붕어빵을 수행하는 기능)

* 객체 : 대상화화 함 (붕어빵)

* Call by value(콜바이벨류) : 자바에서 사용, 값을 줘서 계산하라고함.
  (병규에게 맴매/상훈이에게 병규 때리라고명령 / 상훈에게 가서 맞으라고 병규에게 명령함)
  - 기본형 변수(byte, float, int, long 등)가 벨류에서 사용.
  - 다른 메소드로 넘기면 값 자체가 통째로 복사되 새로운 변수를 만듬
  - 그렇기에 원본과 복사된 것은 서로 다른 entity임.

* Call by reference(콜바이레퍼런스) : C프로그래밍, 값을 주지 않고 주소 값을 줘서 계산하라고함.(자바에선 안 씀)
  (상훈이에게 주소 값을 줘서 병규를 때리라고함)
  - 참조 : 값을 주지 않고 포인트로 계산해봐라 하는 것이 참조
  - 참조형 변수가 레퍼런스에서 사용
  - 코드 상에서 확인할 수 있는 값들이 아닌 특정 인스턴스(또는 배열)을 가리킴.
  - 나이, 점수, 키 등 값이 아니라 '메모리에 올라가 있는 덩어리들의 위치'를 나타내는 값(참조값)임.
  - 다른 메소드에서 바꾸면 원본도 같이 바뀜으로 이해.

○ 장점
  - 자바는 링크 과정이 없음/ 클래스들이 독립적/ 가비지컬렉터(안쓰는 메모리를 자동처리)
  - 멀티스레드 기능 (여러 class를 만들어 사용 가능)
○ 단점 : 언제 가비지컬렉터가 작동될지 모름(그렇기에 게임에선 사용X)

특징
  - main()메소드로 시작 (메인은 하나만 가능) / 논리타입이 다르다 / 문자열 리터럴은 string 사용
  - 자바는 문자 타입이 2바이트, 1바이트 대용하는 것이 byte(당 16비트)
  - 컴퓨터의 타입에 따라 변하지 않음
  - 자바에서 null은 객체로 사용하므로 int n = null 사용불가
  - 상수 선언 : final 키워드 사용,값 변경 불가.선언시 초기값 지정/final int PRICE=10;

* 확장자 : .java -> .class

* 자바가 어느 운영체제에서든 쓸 수 있는 이유 :
  컴퓨터에서 직접 사용하지 않고 자바 가상기계를 사용하기 때문. / WORA(한번쓰면 어디서든 실행가능)
* 바이트 코드 : 자바 소스를 컴파일한 목적 코드, CPU에 종속적이지 않음

* 정적링크(라이브러리) : 컴파일 타입/ 실행파일에 포함(컴파일시간에 이루어지는 모든 것)
=> 자바에서는 정적링크를 쓰지 않음.
* 동적링크(라이브러리) : 실행을 하며 가져다 씀.(실행시간에 이루어지는 모든 것) / dll은 동적.
=> 동적링크를 쓰는 이유는 파일크기가 커지기 때문. => dll을 씀.

* 식별자 : 특수문자는 불가능하나 \와 _는 식별가능

* 상수 선언 : final 키워드 사용(값 변경 불가)
자동타입 변환 바이트가 자동 변환
강제 타입 변환 / byte a; int price; a=(byte)price; // (byte가 강제변환타입)

int : 정수형 인트
Int : 객체형 인트

C언어	Java
stdio.h를 include (#include <stdio.h>)	java.io를 import (import java.io.*;) : *는 전부다 라는 뜻 / java.io에있는 클래스를 *(전부다) import하라
printf("Hello World"); //출력	System.out.println("Hello World"); System.out.printf();
	try문이 실행문구/ 시도 input에서 오류가 생기면 catch (IOException e)가 오류출력
scanf //입력받음	InputStreamReader scanner.next(); // 다음 입력을 스캔함 공백문자단위를 끊어서 읽어줌(덩어리로 읽어줄때는 스캐너가 편함)

연산자 종류	연산자
증감	++ --
산술	+ - * / %
시프트	✓ >> << >>> // 00001101 >> 00000110 ✓ // b= a/2; b= a>>1 //b=a*2; b=a<<1 ✓ >>는 1비트 오른쪽으로 시프트한때 2로 나누기 ✓ <<는 1비트 시프트시 2로 곱하는 결과 ✓ >>>는 시프트 시 최상위 비트에 항상 1이 삽입
비교	▶ > < > = <= == !=
비트	& l ^ ~
논리	&& ll ! ^
조건 : 계산을 빠르게 할 때 사용	? :
대입	= *= /= += -= &= ^= l= <<= >>= >>>=

EX) 저장시 꼭 .java로 저장/ 실행하면 class 자동생성

```
Public class Hellojava { //헬로자바클래스가 공개되있다. class 중요
        Public static void main(String[] arg) {
                System.out.println("Hello World"); // 헬로월드를 시스템에 출력한다.
        }
}
```

폴더생성/ java파일생성 .java붙이고

cmd 실행

cd \ // 최상위 파일로 이동

cd Hello // java폴더로 이동(주소)

javaC Hello.java // 클래스 생성

java Hello // 클래스 실행 => Hellojava 실행(출력)

EX2)

```
public class Hello2 {    //클래스(처음~끝)
        public static int sum(int n, int m){ /* 메소드(방법) public(공개된 방법)
                return n+m;  //호출된 메소드로 리턴    */ private(비공개된 방법)
        {

                // main()메소드에서 실행 시작함.
        public static void main(String[] args) {
                int I = 20;
                int s;
                char a;

                s = sum(i, 10);  // public static int sum(int n, int m)으로 들어감
                a = '?';
                System.out.println(a); // print 'ln' 은 자동줄바꿈
                System.out.println("Hello2");
                System.out.println(s);
        }
}
```

EX3) InputStreamReader rd 실행문 // 바이트 스트림을 캐릭터 스트림으로 바꾸어주는 가교 역할을 하는 클래스.

```
import java.io.*;       // java.io

public class systemIn {
        public static void main(String[] arg) {
                InputStreamReader rd = new InputStreamReader(System.in);

                System.out.println("input your string");
                try {
                        while(true) {
                        int c = rd.read();
                        System.out.println(c);
                        }
                }
                catch(IOException e){
                        System.out.println("Input Error");
                }
        }

}
// a를 입력 시 97 13 10 이 출력 / 1 입력시 49 13 10 이 출력
```

실습) Scanner을 이용해 삼각형 밑변, 높이를 입력하여 빗변의 길이를 구하라.

```
import java.util.*;
import java.lang.*;

public class systemIn {
	public static void main(String[] arg) {
		Scanner scanner = new Scanner(System.in); //스캐너 실행

		System.out.println("input two lengths : "); //사용자에게 입력받음

		double len1 = scanner.nextDouble(); // 더블 렌1
		double len2 = scanner.nextDouble(); // 더블 렌2
		double anotherLen = Math.sqrt(len1*len1+len2*len2); //sqrt는 루트
		scanner.close(); // 스캐너를 꼭 닫아야함.

		System.out.print("빗변의 길이 = ");
		System.out.println(anotherLen);
		}
	}

}
```

EX4) 비트 연산자와 시프트 연사자의 예

```
public class BitShiftOperator{
  public static void main (String[] args){
     short a = (short)0x55ff;  // ff
     short b = 0x00ff; // 55ff


    //비트 연산
     System.out.printf("%x\n",a & b); // 5500
     System.out.printf("%x\n",a | b); // ffffaa00
     System.out.printf("%x\n",a ^ b); // 80
     System.out.printf("%x\n", ~a); // 5
```

실습) 평균값과 등급 구하기

```
import java.util.*;

public class systemIn {
	public static void main(String[] arg) {
		Scanner scanner = new Scanner(System.in);

		System.out.println("국어,영어,수학,자연 점수를 입력하시오 : ");
//국어 영어 산수를 입력받고 평균을 구해서 그 평균을 ABCD평균으로 나눠라

		double len1 = scanner.nextDouble();
		double len2 = scanner.nextDouble();
		double len3 = scanner.nextDouble();
		double len4 = scanner.nextDouble();
		double score = (len1+len2+len3+len4)/4;
		char grade;

		if(score>=90) grade='A';
		else if(score>=80) grade='B';
		else if(score>=70) grade='C';
		else if(score>=60) grade='D';
		else grade = 'F';

		scanner.close();

		System.out.print("평균 : ");
		System.out.println(score);
		System.out.print("평점 : ");
		System.out.println(grade);
		}
	}
```

4주차

ㅇ배열

배열 선언 int intArray[]; or int[] intArray;

배열 생성 (동적으로 메모리 할당을 해야함.: new)
*포 인트를 사용하기 위해선 하나당 4바이트를 쓴다. / 10개의 배열을 만드는데는 44바이트가 든다.
 -> 10개의 배열과 주소 값 4바이트도 있기 때문.
배열 선언은 주소 값 하나만 만들 수 있는 공간이 만들어 지는 것.//나중에 실제 배열을 가리킬 것이다.

int intArray[]= new int[5];
int myArray[] = intArray;를 하면 intArray의 주소 값을 바라본다.
(주소+1은 그 다음공간이 됨)

* C에서는 배열을 10개 잡고 11번째 숫자를 불러오면 실행은 하나 메모리의 엑세스 충돌이 일어날 수 있다.
  자바에서는 아예 이렇게 되질 않는다.
 배열의 크기는 배열 레퍼런스 변수를 선언할 때 결정되지 않음/ 배열 생성 시에 결정되고 바꿀수 없다/
 배열의 크기는length에 저장된다.

*** for-each문** - 배열이나 나열의 각 원소를 순차적으로 접근하는데 유용한 for문
ex) for(int k : num) // 반복될 때마다 k는 num[0]~num[1]~... 값으로 설정
      sum += k;

ㅇ2차원 배열

int intArray[][] = {{0,1}{3,4}{6,7}} // 3 * 2 배열
I.length = 3
[0] = 2
[1] = 2
[][이 값에 숫자를 점점 더하면]비 정방형 length를 설정할 수 있다.

```
//2차원 배열에 값들을 넣고 그 수를 다 더함//
public class jkm{
        public static int[] makeArray(int size){ //1차원 배열(주소)을 리턴
                int [] arr = new int[size];
                for(int i=0; i<arr.length; i++) { // 0, 0,1 , 0,1,2 ,...
                        arr[i]= i; // 배열안의 배열의 칸에 값을 넣음
                }
                return arr;
        } //변수의 범위규칙 : 모든 변수는 자기 블록 안에서만 엑서스 가능함.

public static void main(String[] argv) {
        int arr[][];

        int numArrays = 5;
        arr = new int[numArrays][];
        // create arrays with makeArray
        for(int i=0; i<arr.length;i++){
                arr[i] = makeArray(i+1); //arr[0]은 1칸 arr[1]은 2칸...
        }

        // copute the sum of all elements in arr
        int sum=0;
        for(int a[] : arr){
                for(int e : a){
                        sum+= e;
                }
        /*for(int i=0; i<arr[i].length; i++){ // arr[i].length를 numArrays를 넣으면 C에서 사용하는 방식
                for(int j=0; j<arr[i].length;j++){ //arr[i].length의 값은 위에 설정한 5가 됨
                        sum += arr[i][j];
                }*/
        }
        System.out.println("Sum = " + sum);
} }
```

* static : 실행 하기 전에 미리 존재하는 것들. / 객체 생성 전부터 호출 가능! / 정적으로 만들어져 있어야함.
* (String[] args) : String[] 은 문자열 배열
 ex) dir /s 라면 dir에 /s를 집어넣어준다. 이게 args
* cmd창에서 java String을 받으면 String을 더블로 바꿔줌.

```
public class Arguments {

	public static void main(String[] args){
		if(args.length<3) return; // args 를 3개가 들어오도록 설정

		double operand1 = Double.parseDouble(args[0]);
		// parseDouble은 어떤 문자를 Double로 해독하는 것
		// java Arguments 3.4 + 5.4를 하면 3.4가 args[0], 5.4가 args[2]
		double operand2 = Double.parseDouble(args[2]);

		if(args[1].equals("+"))System.out.println(operand1+operand2); /*equals는 내용이 같은지를
		if(args[1].equals("-"))System.out.println(operand1-operand2); 볼 때 씀==는 사용할 수 없음.
		if(args[1].equals("*"))System.out.println(operand1*operand2); 주소 값이기 때문
		if(args[1].equals("/"))System.out.println(operand1/operand2); */
	}
}
```

* 예외처리

- try-catch-finally 문 사용
- finally는 생략 가능
- 나누는 동작이 try/divisor이 0이므로 예외발생을 catch
	try{
	… /*예외가 발생하는 순간 바로
	실행문 catch문으로 넘어감*/
	…
	}
	catch //오류가 발생한 것은 실행x
	System.out.println(“0으로 나눌 수 없습니다”);

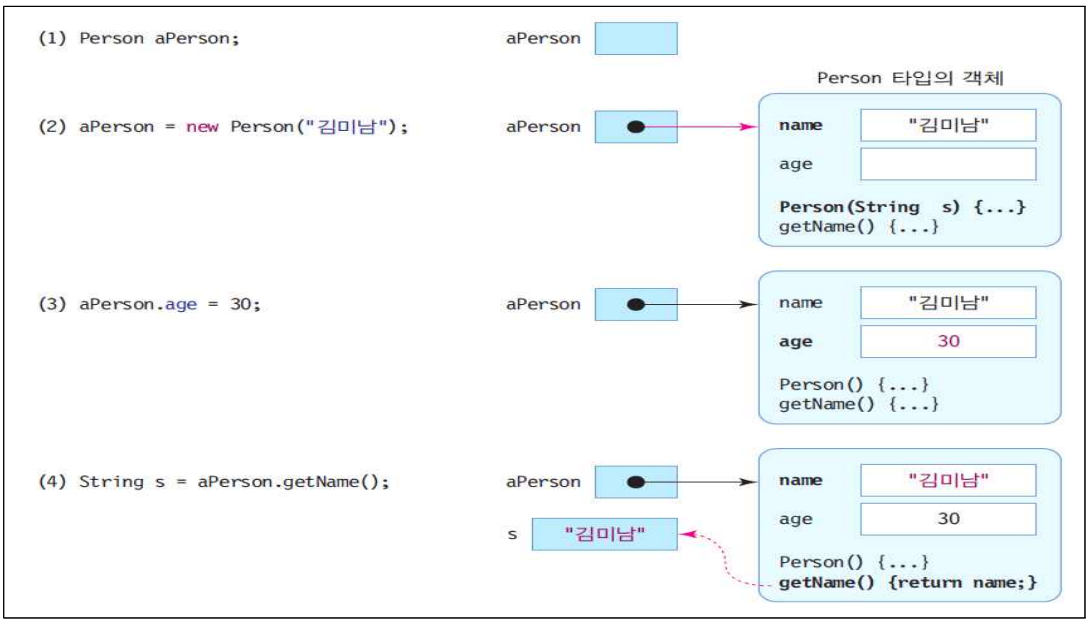
//실습) 계산기 예외처리//

```
public class Arguments {

	public static void main(String[] args){
		if(args.length<3) return; //args 를 3개가 들어오도록 설정

		double operand1=0, operand2=0;

		try{ //try안에 double을 사용하면 밖에서 사용할 수 없기에 위에 따로 생성해준다.
		operand1 = Double.parseDouble(args[0]); //parseDouble은 어떤 문자를 Double로 해독하는 것
		// java Arguments 3.4 + 5.4를 하면 3.4가 args[0], 5.4가 args[2]
		operand2 = Double.parseDouble(args[2]);
		if(args[1].equals("+"))System.out.println(operand1+operand2); //equals는 == 임.
		if(args[1].equals("-"))System.out.println(operand1-operand2);
		if(args[1].equals("*"))System.out.println(operand1*operand2);
		if(args[1].equals("/"))System.out.println(operand1/operand2);
		}
		catch (NumberFormatException e){
		System.out.println("사용할 수 없는 형식의 인자 입니다.");
		} } }
```

* **오버 로딩 : 여러 개의 동작들을 정의해 놓는 것 // ‘치다’ 라는 것에 피아노를 치다. 사람을 치다 등으로 뜻이 여러개**
* **오버 라이딩 : 기존에 있는 것을 덮어쓰는 것 // 동물의 cry를 휴먼의 cry로 고쳐쓰겠다는 것.**
* 다형성 폴리모피지(?) : 짖는다는 것은 똑같지만 각각의 동물이 다른 울음소리는 알아서 내는 것
  추상의 ex) 애국 : 추상화를 하는 것이 클래스

함수	내용
data.length	배열의 길이(갯수)
arr[i] = Integer.*parseInt*(data[i]);	String을 Int형으로 바꾸어줌

* 클래스의 이름과 동일한 메소드 / 객체가
* ex) Person(클래스의 이름) aPerson(객체의 이름) // 레퍼런스 변수 aPerson선언(자바에서는 주소 값만 가지는게 생성)
  aPerson = new Person("김미남"); // Person 객체 생성 (주소 값만 가지기 때문에 Call by reference가 됨)
  age = 30;; // 이게 aPerson의 주소값에 불려가기 때문에 Call by reference가 됨



* 메인은 무조건 한번 실행, 하나만 존재하기 때문에 'static'으로 표현
* **지수**

```
//지수 예//
import java.io.*;

public class test2{
        int base; // 아무것도 없이 선언하면 private임.
        int exp;

        int getValue(){ //이것을 지수라고 함
                int res = 1;
                for(int i=0; i<exp; i++) res *= base;
                return res;
        }
        public static void main(String[] arg){
                test2 exp1 = new test2(); //()가 빈공간이면 default(아무 값도 없음)
                exp1.base = 2;
                exp1.exp = 3;

                System.out.println("value = "+ exp1.getValue());
        }
}
```

* **재귀호출** : 자기자신을 부름

```
import java.io.*;

public class test2{
        double base; // 아무것도 없이 선언하면 private임.
        int exp;

        double power(double b, int e){
                if(e==1) return b;
                if(e<1) return 1.0;
                else {
                        double t = power(b,e/2);  //재귀호출을 함
                        return t*t*((e%2==0)?1:b);
                }
        }
        double getValue(){ //이것을 지수라고 함
                return power(base,exp);
        }
        public static void main(String[] arg){
                test2 exp1 = new test2(); //()가 빈공간이면 defalt(아무값도 없음)
                exp1.base = 1.0000000001;
                exp1.exp = 1000000000;

                System.out.println("value = "+ exp1.getValue());
        }
}
```

❑ 공개 여부

* public : 공개적임
* default : 패키지 private임. (패키지 안에서는 쓸 수 있음) //아무것도 쓰지 않을 때
* private : 사용하는 곳 이외에서는 쓸 수 없음
* Protect : 막아줌

```
public class Program{
        public static void main(String[] arg){
                test2 exp1 = new test2();
                 // test2 클래스 타입으로 exp1을 새로 생성하라(동적생성)
                exp1.base = 2; //default 는 패키지 Private
                exp1.exp = 3;

                System.out.println("value = "+ exp1.getValue());
        }
}    *동적이란 프로그램을 실행시키면서 사용되는 것
```

실습) 배열 10개를 생성하고 모든 요소에 대해(참조에 대해)객체를 만듬// 파일2

```
public class test2{
        double base; // 아무것도 없이 선언하면 private임.
        int exp;

        double power(double b, int e){
                if(e==1) return b;
                if(e<1) return 1.0;
                else {
                        double t = power(b,e/2);
                        return t*t*((e%2==0)?1:b);
                }
        }
        double getValue(){ //이것을 지수라고 함
                return power(base,exp);
        }
}
```

//파일2

```
public class Program{
        public static void main(String[] arg){
                Person[] expArr;// = new Person[10]; 배열 생성함
                expArr = new Person[10];
                //실제로 Person클래스를 생성한게 아니라 배열만 생성
                //Person클래스 객체 10개를 담은 배열을 생성
                int i=0;
                for(Person e : expArr){ //for(int i=0; i<expArr.length; i++){
                        e = new Person(); //Person 클래스의 e를 expArr에 넣음
                        e.base = expArr.length-i;
                        e.exp = i;
                        i++;
                        System.out.println(e.base + "^" + e.exp+ "=" + e.getValue());
                }
        }
}
```

//실습//배열 / 써서 사람의 수를 지정, 국어,영어,수학 의 점수를 각각의 사람에 넣고 총점 1등인 사람 출력 (국어, 수학, 영어점수를 알려주는 것 필요) (토탈 스코어 메소드 필요)

```
public class Student {
	private String name;
	private int Korea;
	private int English;
	private int Math;

	void setName(String n){
		name = n;
	}
	void setScore(int course, int score){
		switch(course){
		case 0 : Korea = score; break;
		case 1 : English = score; break;
		case 2 : Math = score; break;
		default : System.out.println("Invalid course ID");
		}
	}

	int getTotalScore(){
		return Korea+English+Math;
	}
	String getName(){
		return name;
	}

	public void display(){
		System.out.println(name + ":" + Korea + "," + English + "," + Math);
	}
}
```

```
import java.util.Scanner;

public class Program {

	public static void main(String[] arg){
		Scanner sc = new Scanner(System.in);

		System.out.print("How many Student? ");
		int nStudent = sc.nextInt();

		Student[] aArr = new Student[nStudent]; //c++에서는 여기서 완결
		//자바는 이게 주소값을 가르켜서 새로 객체처럼 만들어줘야함
		for(int i=0; i<nStudent; i++){
			aArr[i] = new Student(); //객체 처럼 만든게 이것임
			System.out.println("이름 : ");aArr[i].setName(sc.next());
			System.out.println("국어 : ");aArr[i].setScore(0, sc.nextInt());
			System.out.println("영어 : ");aArr[i].setScore(1, sc.nextInt());
			System.out.println("수학 : ");aArr[i].setScore(2, sc.nextInt());
		}

		Student bestStudent = aArr[0]; // bestStudent의 주소값만을 가짐
		for(Student s : aArr){//for(int i=0;i<nStudent; i++){
			if(s.getTotalScore() > bestStudent.getTotalScore()){
				bestStudent = s;
			}
		}

		for(int i=0;i<nStudent; i++){
			aArr[i].display();
		}
		System.out.println("Best Student = "+bestStudent.getName());
	}
} // 다음주 클래스 사용 시험!!
```

// 오버라이딩(동적바인딩), 생성자//

```
public class ShapeArea {

	public static void main(String[] arg) {
		Shape cir1 = new Circle(2.0);
		Shape cir2 = new Circle(3.0);
		Shape rect1 = new Rectangle(1.3, 10);
		Shape rect2 = new Rectangle(2.1, 3.1);

		System.out.println(cir1.toString() + " cir1 Area = " + cir1.Area());
		System.out.println(cir2.toString() + " cir2 Area = " + cir2.Area());
		System.out.println(rect1.toString() + " rect1 Area = " + rect1.Area());
		System.out.println(rect2.toString() + " rect2 Area = " + rect2.Area());

		// AreaSum (obj1, obj2)
		System.out.println("cir1  + cir2  = " + AreaSum(cir1,cir2));
		System.out.println("cir1  + rect1 = " + AreaSum(cir1,rect1));
		System.out.println("rect2 + cir2  = " + AreaSum(rect2,cir2));
		System.out.println("rect1 + rect2 = " + AreaSum(rect1,rect2));

	}

	public static double AreaSum(Shape a, Shape b) {
		return a.Area() + b.Area();
	}
}
```

```
public abstract class Shape {
	public abstract double Area();
}
```

```
public class Rectangle extends Shape {
	double width;
	double height;

	public Rectangle(double w, double h) {
		width = w; height = h;
	}

	public double Area() {
		return width*height;
	}
}
```

```
public class Circle extends Shape{
	double radius;

	public Circle(double r) {
		radius = r;
	}
	public double Area() {
		return 3.14*radius*radius;
	}
}
```

* 과제1 : java find max/min 5 7 9 10 4 8 중 최대 값,최소 값을 짜는 것 / 개수는 unknown
입력된 정수 리스트에서 최대 값 혹은 최소 값을 찾는 프로그램 find를 작성하라.
첫 번째 인자가 최대 값 혹은 최소 값을 찾을지를 결정하는 인자이며 두 번째 인자부터 개수가 정해지지 않은 정수의 리스트가 들어온다.

```
import java.util.Scanner;
import java.io.*;

public class find {
	public static void main(String[] arg)throws Exception{
		Scanner sr = new Scanner(System.in);
		InputStreamReader rd = new InputStreamReader(System.in);
		int MAX=0,MIN=0;

		System.out.println("-- MAX와 MIN를 선택하시오 ( M / N )--");
		char a = (char)rd.read();

		a = a=='M' ? 'M' : a=='N' ? 'N' : 'X';

		if(a=='M'||a=='N')
		{
			System.out.print("비교할 숫자들을 입력하시오 : ");
			String input = sr.nextLine();
			String data[] = input.split(" ");

			int[] arr = new int[data.length];

			for(int i=0; i<data.length; i++){
				arr[i] = Integer.parseInt(data[i]);
			}

			switch(a)
			{
			case 'M':
				for(int i=0; i<data.length-1; i++){
					if(i==0){MAX=arr[i];}
					if(i>=0){
						if(MAX<=arr[i+1]){
							MAX = arr[i+1];
						}
					}
				}
				System.out.println("MAX값 : "+MAX);

			case 'N':
				for(int i=0; i<data.length-1; i++){
					if(i==0){MIN=arr[i];}
					if(i>=0){
						if(MIN>arr[i+1]){
							MIN = arr[i+1];
						}
					}
				}
				System.out.println("MIN값 : "+MIN);
			}
			System.out.println("완료되었습니다.");
		}else {System.out.println("잘못 입력하셨습니다. 프로그램을 종료합니다.");}	}	}
```

```
import java.util.Scanner; //MAX,MIN동시에 구하기
public class Arguments {
	public static void main(String[] args){
		Scanner scn = new Scanner(System.in);
		int input;
		System.out.println("---나열할 갯수 입력---");
		input = scn.nextInt();
		int arr[] = new int[input];
		System.out.println("---숫자 입력---");
		for(int i=0; i<arr.length; i++){
			arr[i] = scn.nextInt();
		}
		int MAX=0, MIN=0;
		for(int i=0; i<arr.length; i++){
			if(i==0){
				MAX=arr[i];
				MIN=arr[i];
				System.out.println("초기MAX값 : " +MAX);
				System.out.println("초기MIN값 : " +MIN);
			}
			if(i>=0){
				if(MAX<=arr[i+1]){
					MAX = arr[i+1];
					System.out.println("--MAX값 적용!--");
				}
				else if(MIN>=arr[i+1]){
					MIN = arr[i+1];
					System.out.println("--MIN값 적용!--");
				}
				else{System.out.println("--적용값 없음!--");}
			}
			System.out.println("현재 MAX : "+MAX);
			System.out.println("현재 MIN : "+MIN);
		}	}	}
```

```
//MAX값 MIN값 오름차순 내림차순으로 만들기//
                if(z=='M'){
                        int MAX=0;
                        for(int a=0; a<1; a++){
                                for(int b=a+1; b<dt.length; b++){
                                        if(a==0){MAX = arr[a];}
                                        if(arr[a]<=arr[b])
                                        {
                                                int c=arr[a];
                                                arr[a]=arr[b];
                                                arr[b]=c;
                                                MAX = arr[a];
                                        }
                                }
                                System.out.println("MAX값 : "+MAX);
                        }
                }
                else if(z=='N')
                {
                        int MIN=0;
                        for(int a=0; a<1; a++){
                                for(int b=a+1; b<dt.length; b++){
                                        if(a==0){MIN = arr[a];}
                                        if(arr[a]>=arr[b])
                                        {
                                                int c=arr[a];
                                                arr[a]=arr[b];
                                                arr[b]=c;
                                                MIN = arr[a];
                                        }
                                }
                                System.out.println("MIN값 : "+MIN);        }          }
```

```
//MAX값 MIN값 오름차순 내림차순으로 만들기//단어로입력도가능//
import java.util.Scanner;
import java.io.*;

public class test {
        public static void main(String[] arg)throws Exception{
                Scanner sr = new Scanner(System.in);
                InputStreamReader rd = new InputStreamReader(System.in);
                int MAX=0,MIN=0;

                System.out.println("MAX와 MIN를 선택하시오 (MAX / MIN)");
                BufferedReader x = null;
            x = new BufferedReader(new InputStreamReader(System.in));
            String a = x.readLine();

            String MAX1 = "MAX";
            String MIN1 = "MIN";
                if(a.equals(MAX1) || a.equals(MIN1))
                {
                        System.out.print("비교할 숫자들을 입력하시오 : ");
                        String input = sr.nextLine();
                        String data[] = input.split(" ");

                        int[] arr = new int[data.length];

                        for(int i=0; i<data.length; i++){
                                arr[i] = Integer.parseInt(data[i]);
                        }

                        if(a.equals(MAX1))
                        {
                                for(int i=0; i<data.length-1; i++){
                                        if(i==0){MAX=arr[i];}
                                        if(i>=0){
                                                if(MAX<=arr[i+1]){
                                                        MAX = arr[i+1];
                                                }
                                        }
                                }
                                System.out.println("MAX값 : "+MAX);
                        }
                        else if(a.equals(MIN1)){
                                for(int i=0; i<data.length-1; i++){
                                        if(i==0){MIN=arr[i];}
                                        if(i>=0){
                                                if(MIN>arr[i+1]){
                                                        MIN = arr[i+1];
                                                }
                                        }
                                }
                                System.out.println("MIN값 : "+MIN);
                        }
                        System.out.println("완료되었습니다.");
                        }else {System.out.println("잘못 입력하셨습니다. 프로그램을 종료합니다.");}   }   }
```

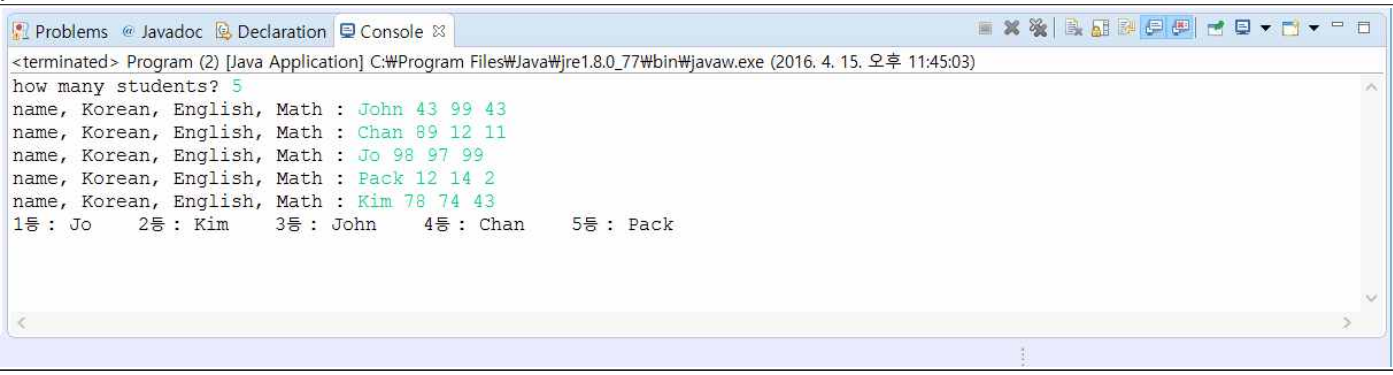
* 과제2 : 5-4를 활용하여 입력된 학생들을 성적순서대로 정렬하는 프로그램을 작성하라
ex) java find max 1 3 6 2 8 3 5 7 6
max = 8 출력된 결과를 캡쳐하기

//Program.java//

```
import java.lang.reflect.Array;
import java.util.Arrays;
import java.util.Scanner;

public class Program {

	public static void main(String[] arg) {
		Scanner sc = new Scanner(System.in);

		System.out.print("how many students? ");
		int nStudents = sc.nextInt();

		Student student[] = new Student[nStudents];

		for(int i=0;i<nStudents; i++) {
			student[i] = new Student();
			System.out.print("name, Korean, English, Math : ");
			student[i].setName(sc.next());
			student[i].setScore(0, sc.nextInt());
			student[i].setScore(1, sc.nextInt());
			student[i].setScore(2, sc.nextInt());
		}


		for (int i=0; i<student.length; i++){
			Student List = student[i];
			for (int j=i+1; j<student.length; j++){
				if ( student[j].getTotalScore() >= List.getTotalScore() ) {
					Student c = student[j];
					student[j] = List;
					List = c;
				}
			}
			System.out.print((i+1)+"등 : " + List.getName()+"      ");
		}	}	}
```

//Student.java//

```
import java.util.Arrays;

public class Student {
	private String name;
	private int Korean;
	private int English;
	private int Math;

	public void setName(String n) {
		name = n;
	}
	public void setScore(int course, int score) {
		switch(course) {
		case 0 : Korean = score; break;
		case 1 : English = score; break;
		case 2 : Math = score; break;
		default: System.out.println("Invalid course ID");
		}
	}

	public int getTotalScore() {
		return Korean+English+Math;
	}

	public String getName() {
		return name;
	}

	public void display() {
		System.out.println(name + ":" + Korean + "," + English + "," + Math);
	}

}
```



중간고사 예제

1) 객체지향 프로그래밍이란 무엇인가?

- 객체 지향 프로그래밍이란, 해결할 과제를 실제 세상의 객체와
  객체 간의 상호 관계로 모델링하여 인간의 사고에 가깝게 표현한 프로그래밍이다.

2) 자바가상기계(JVM, Java Virtual Machine)란 무엇이며, 이것 때문에 자바(Java)가 다른 언어와 어떤 차이점을 갖게 되는가?

-자바가상기계란, 자바로 작성된 응용프로그램을 윈도우나 유닉스와 같은 컴퓨터 운영체계에서 원활히 운용될 수 있도록 하는 소프트웨어이다.
/ C/C++같은 기존 언어는 플랫폼 종속적이지만 자바는 WORA로 어느 플랫폼에서나 실행가능하다.

3) JDK와 JRE의 차이는 무엇인가?

- JDK는 자바 프로그램 개발을 위해 설치하는 환경이고,
  JRE는 일반 사용자들이 자바 실행을 위해 사용하는 환경이다.

4) 두 개의 정수를 입력 받아 큰 수를 출력하는 프로그램을 자바(Java)로 작성하라.

- 아래에 작성

5) 값에 의한 호출(call by value)와 참조에 의한 호출(call by reference)의 차이를 설명하고, 자바(Java)에서 호출은 어떻게 이루어지는지 설명하라.

- call by value는 함수를 호출할 때 단순히 값을 전달하는 함수 호출이고,
  call by reference는 메모리의 접근에 사용되는 주소 값을 전달하는 형태의 함수 호출을 말한다.
  call by reference(주소 값)을 사용하는 이유는 메모리 사용을 줄이기 위해서이다.

[int형 배열 10만개를 저장하면 메모리가 40만 바이트가 필요한데, 주소 값을 사용하면 하나만 가지면 된다.]

자바에서는 call by reference(주소값)는 구현되지 않는다. 따라서 call by value 호출을 사용하며, 기본형타입과 참조형타입은 인수를 넘길 때 값 복사가 일어난다. 즉, 참조 값끼리 아무리 복사를 하더라도 객체 내부의 메모리끼리의 복사는 이루어지지 않기 때문에, 단지 참조 값만이 복사되어 호출된다.

// 자바에서는 call by value만이 존재한다. 기본 타입은 말 할 것도 없고, 참조타입은 call by reference와 비슷하지한 효과가 구현되지만, 엄밀히 말하면 call by value이다.

6) 클래스(Class)와 객체(instance), 그리고 메소드(method)에 대해 설명하라.

- 클래스 : 사물의 특성을 소프트웨어적으로 추상화하는 설계도.(객체에 대한 정의)
  객체 : 사물의 특성을 소프트웨어적으로 추상화한 것(사물 또는 개념)(클래스의 인스턴스(실체))
  메소드 : 클래스를 수행하는 기능

7) 오버라이딩(overriding)은 무엇이며, 오버로딩(overloading)과 어떻게 다른지 설명하라. 간단한 두 개의 클래스 class A와 class B를 실제로 만들고, 이 두 클래스에 메소드를 자바 코드로 작성하여 오버로딩(overloading)과 오버라이딩(overriding)을 표현해 보라.

- 오버라이딩 : 기존에 있던 것을 덮어 쓴 것.(재정의하는 것)
  오버로딩은 : 여러개의 동작들을 만든 것

8) 정적(static) 메소드는 어떤 특성을 갖는가?

- static 멤버는 non-static멤버와 달리 이미 외부에 별도로 존재하며 이를 공유한다.
  (new를 써서 객체가 만들어지기 전에 존재하여야 한다)
- static 메소드는 오직 static멤버만 접근이 가능하다.

- 종속되어 있는 클래스의 인스턴스들의 생성이나 활용을 도우는 목적으로 사용된다.

- 클래스 내에서 인스턴스끼리 공유하는 변수이다.(공유 변수)

- 메서드에 static을 붙여 사용하는 경우 메서드의 기능이 인스턴스의 상태와 상관없이 수행 할 수 있는 경우이다.

- static 클래스가 로딩 된 시점에서 메모리 공간을 가지고 있기 때문에
  인스턴스 생성 없이 변수나 상수나 메소드를 클래스 이름 뒤에 참조연산자 .(마침표)를 이용해서 바로 접근이 가능하게 된다. ex) StaticInner.Inner = new StaticInner.Inner();

4) 두 개의 정수를 입력 받아 큰 수를 출력하는 프로그램을 자바(Java)로 작성하라.

```
import java.util.*;

public class jkm {
	public static void main(String[] arg) {
		Scanner scanner = new Scanner (System.in);

		System.out.println("두 개의 정수를 입력하시오. : ");

		int len1 = scanner.nextInt();
		int len2 = scanner.nextInt();

		if(len1<len2){
			System.out.printf("%d가 %d보다 작습니다.\n", len1, len2);
		} else if(len1==len2){
			System.out.printf("%d와 %d가 같습니다.\n", len1, len2);
		} else if(len1>len2);
			System.out.printf("%d가 %d보다 큽니다.\n", len1, len2);
		}
	}
}
```

7주차(16.04.21)

❑ 오버로딩

*

* 메소드의 이름이 동일해야함
* 인자의 개수가 서로 다르거나, 인자 타입이 서로 달라야함

```
// 메소드 오버로딩이 성공한 사례
class MethodOverloading {
  public int getSum(int i, int j) {
    return i + j;
  }
  public int getSum(int i, int j, int k) {
    return i + j + k;
  }
  public double getSum(double i, double j) {
    return i + j;
  }
}
```

```
// 메소드 오버로딩이 실패한 사례
class MethodOverloadingFail {
  public int getSum(int i, int j) {
    return i + j;
  }
  public double getSum(int i, int j) {
    return (double)(i + j);
  }
}
```

❑ this

- this 레퍼런스

* 현재 객체 자기 자신을 가리킴
  - 자기 자신에 대한 레퍼런스
  - 같은 클래스 내에서 클래스 멤버, 변수를 접근할 때 객체의 이름이 없으면 묵시적으로 this로 가정
  - 객체의 멤버 변수와 메소드 변수이 이름이 같은 경우
  - 객체 자신을 메소드에 전달 또는 반환할 때
* int id;

  void set(int x) {this.id = x;}

  void set(3); id 주소값이 3으로 바뀜
* this(), 같은 클래스의 다른 생성자 호출
* 생성자 내에서만 사용 가능 // 다른 메소드에서는 사용 불가

- this()
  - 같은 클래스의 다른 생성자 호출
  - 생성자 내에서만 사용 가능
    - 다른 메소드에서는 사용 불가
  - 반드시 생성자 코드의 제일 처음에 수행

```
public class Book {
  String title;
  String author;
  int ISBN;

  public Book(String title, String author, int ISBN) {
  this.title = title;
  this.author = author;
  this.ISBN = ISBN;
  }
  public Book(String title, int ISBN) {
    this(title, "Anonymous", ISBN);
  }
  public Book() {
    this(null, null, 0);
    System.out.println("생성자가 호출되었음");
  }
  public static void main(String [] args) {
    Book javaBook = new Book("Java JDK", "황기태", 3333);
    Book holyBible = new Book("Holy Bible", 1);
    Book emptyBook = new Book();
  }
}
```

title = "Holy Bible"
author = "Anonymous"
ISBN = 1

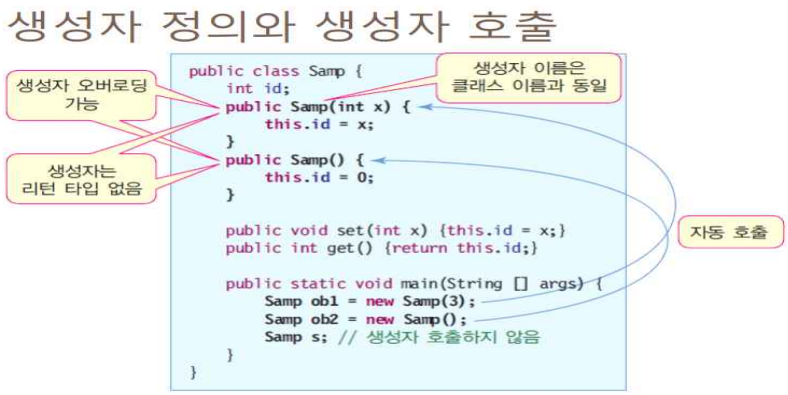
title = "Holy Bible"
ISBN = 1

❑ 객체의 지환

* 객체의 치환은 객체가 복사되는 것이 아니며 레퍼런스가 복사된다.

❑ 생성자

* 생성자의 특징
  - 생성자는 메소드 (new로 한 번만 부를 수 있다)
  - 생성자 이름은 클래스 이름과 동일
  - 생성자는 **new를 통해 객체를 생성할 때만 호출됨**
  - 생성자도 오버로딩 가능
  - 생성자는 리턴(return) 타입을 지정할 수 없다. **// 생성자의 주소만 리턴하기 때문.**
  - 생성자는 하나 이상 선언되어야 함
* 개발자가 생성자를 정의하지 않으면 자동으로 기본 생성자가 정의됨
  - 컴파일러에 의해 자동 생성 (아예 없을 때 생성)



❑ 가비지 컬렉터

* 값을 바라보는 객체가 없으면 그 값은 가비지가 됨.

❑ 멤버 접근자

default (또는 package-private)	**•같은 패키지 내에서 접근 가능**
public	•패키지 내부, 외부 클래스에서 접근 가능
private	•정의된 클래스 내에서만 접근 가능 •상속 받은 하위 클래스에서도 접근 불가
protected	•같은 패키지 내에서 접근 가능 •다른 패키지에서 접근은 불가하나 상속을 받은 경우 하위 클래스에서는 접근 가능

멤버에 접근하는 클래스	멤버의 접근 지정자			
	default	private	protected	public
같은 패키지의 클래스	O	X	O	O
다른 패키지의 클래스	X	X	X	O

❑ static

* 종속되어 있는 클래스의 인스턴스들의 생성이나 활용을 도우는 목적으로 사용된다.

>> 다중 상속을 써야할 경우 interface(implements)를 쓴다.

// inter 파일 //

```
/*iSender 인터페이스를 사용할 경우 iSender에 있는 모든 메소드를 구현하지 않으면
 추상 클래스가 되어 class에 abstract를 붙여줘야 한다. 아래 클래스inter 에서 다른 인터페이스
 void Send(String strReceiver)와 void Receive(String strSender)가 구현되어 있지
 않으므로 abstract를 붙여 추상클래스로 선언했다. */
public abstract class inter extends CChildclass implements iSender, iReceiver
{
	String strSender;
	String strReceiver;

	public inter()
	{
	}

	public void SetSender(String strSender)
	{
		this.strSender = strSender; //this는 현재 클래스의 멤버 변수 strSender이며,
		 //값으로 쓰인 strSender는 SetSender함수의 Argument인자인 strSender이다.
	}

	public void SetReceiver(String strAreceiver)
	{
		this.strReceiver = strAreceiver;
	}
}
```

// iSender 파일 //

```
public interface iSender {
	void SetSender(String strSender);
	void Snd(String strReceiver);
}
```

// iReceiver 파일 //

```
public interface iReceiver{
	void SetReceiver(String strReceiver);
	void Receive(String strSender);
}
```

// CChildclass파일 //

```
public class CChildclass {
}
```