Day12 (H1)

20150825

Access Modifier
Inheritance

default

명시하지 않더라도 public

누구나 액세스 가능

```
class Student {
    public int Kor;
    public int Eng;
    public int Math;

    Student(                        ) { Kor= 0; Eng=0; Math=0;
    Student(int x, int y, int z) { Kor= x; Eng= y; Math= z


    int getKor ()  { return Kor; }
    int getEng ()  { return Eng; }
    int getMath() { return Math; }

    void setKor ( int x ) { Kor= x; }
    void setEng ( int x ) { Eng= x; }
    void setMath( int x ) { Math= x; }
```

..... main() .....

new 를 사용하여
객체 생성 후
객체 참조 변수를
사용하여

```
S[0].Kor = 20;
S[0].Eng = 30;
S[0].Math = 40;

S[1].setKor( 10 );
S[1].setEng( 10 );
S[1].setMath( 10 );
```

ok.

access 가능

↑
객체참조변수

private → 객체 참조변수를 통해서 Read / ~~write~~

```
class Student {
    private int Kor;
    private int Eng;
    private int Math;

    Student(                    ) { Kor= 0; Eng=0; Math=0;
    Student(int x, int y, int z) { Kor= x; Eng= y; Math= z


    int getKor ()  { return Kor; }
    int getEng ()  { return Eng; }
    int getMath() { return Math; }

    void setKor ( int x ) { Kor= x; }
    void setEng ( int x ) { Eng= x; }
    void setMath( int x ) { Math= x; }
```

→ 같은 class (Student) 내의 member 변수는 항상 read / write 사용가능

private data는 같은 class내의 method 들만 사용가능

read/write

private method는 같은 class내의 method 들만 사용가능

calling

Accessor & Mutator

```
int getKor()
int getEng()
int getMath()
```

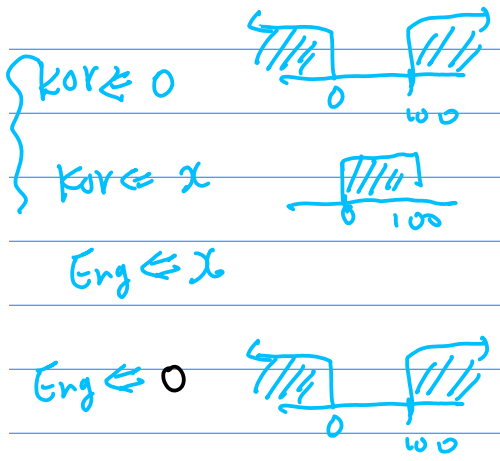
```
void setKor(int x)
void setEng(int x)
void setMath(int x)
```

Accessor
field 값을 얻어오는 method들

Mutator.
field 값을 설정하는 method들

Mutator에 부가기능 추가

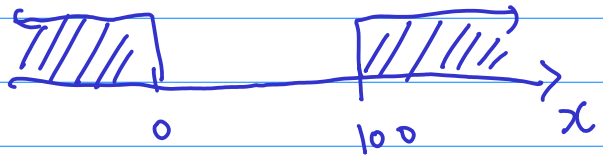
```
void setKor ( int x ) {
    if (x<0 || x>100) Kor = 0;
    else              Kor= x;
}
void setEng ( int x ) {
    Eng= x;
    if (x<0 || x>100) Eng = 0;
}
void setMath( int x ) {
    if (0<x && x<=100) Math= x;
    else               Math= 0;
}
```



OR

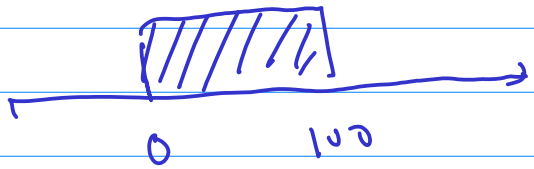
```
void setKor ( int x ) {
    if (x<0 || x>100) Kor = 0;
    else              Kor= x;
}
void setEng ( int x ) {
    Eng= x;
    if (x<0 || x>100) Eng = 0;
}
void setMath( int x ) {
    if (0<x && x<=100) Math= x;
    else               Math= 0;
}
```

if $x<0 \;||\; x>100$



Kor ⇐ 0

else $0 \le x \le 100$



Kor ⇐ x

NOT ((x<0) OR (x>100))

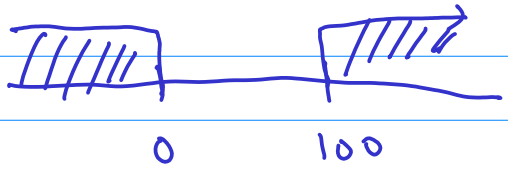
= (NOT (x<0)) AND (NOT (x>100))

= (x>=0) AND (x<=100)

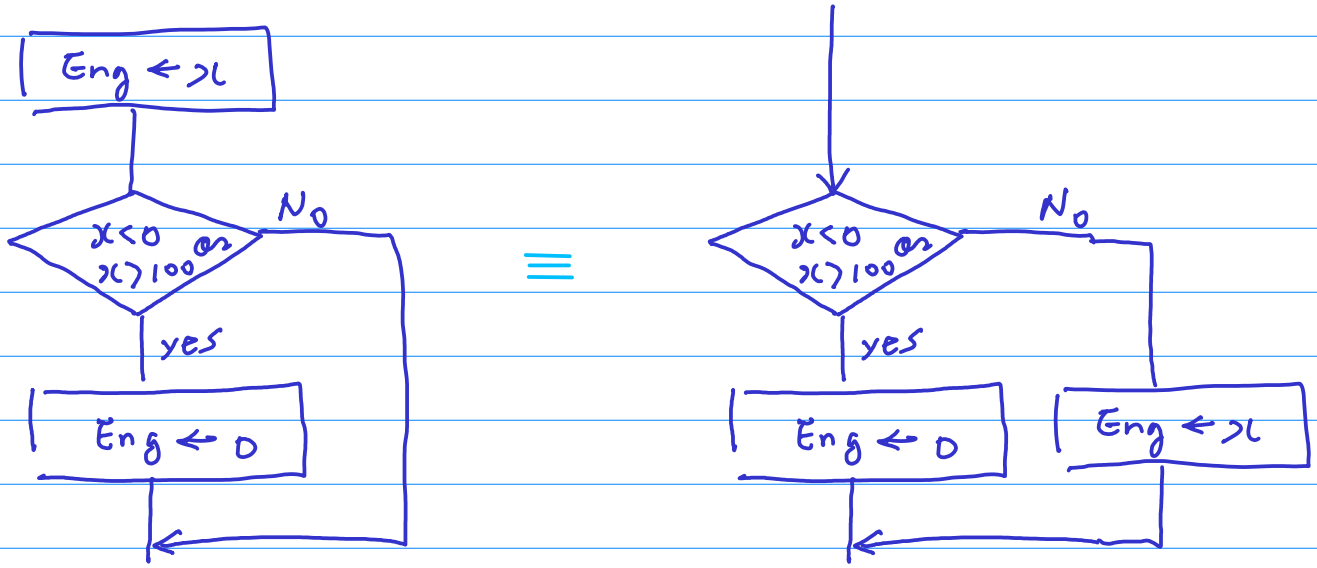
```
void setKor ( int x ) {
    if (x<0 || x>100) Kor = 0;
    else              Kor= x;
}
void setEng ( int x ) {
    Eng= x;
    if (x<0 || x>100) Eng = 0;
}
void setMath( int x ) {
    if (0<x && x<=100) Math= x;
    else               Math= 0;
}
```

① Eng ⇐ x

② (x<0) || (x>100)



Eng ⇐ 0



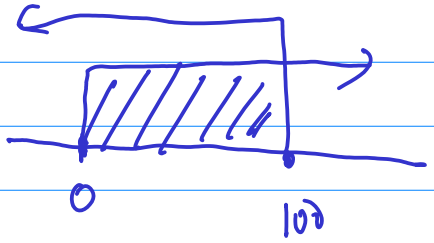
```
void setKor ( int x ) {
    if (x<0 || x>100) Kor = 0;
    else              Kor= x;
}
void setEng ( int x ) {
    Eng= x;
    if (x<0 || x>100) Eng = 0;
}
void setMath( int x ) {
    if (0<=x && x<=100) Math= x;
    else                Math= 0;
}
```

$\sim(x<0 \;||\; x>100)$

$\sim(x<0) \;||\; \sim(x>100)$

$(x \geq 0) \;\&\&\; (x \leq 100)$

① if



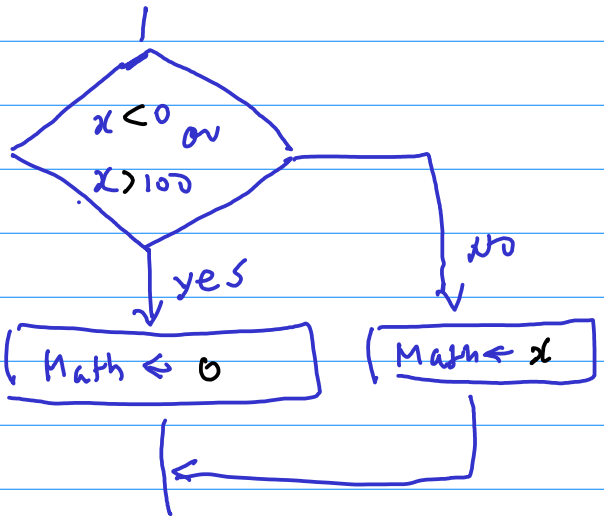
$0 \leq x$ and $x \leq 100$

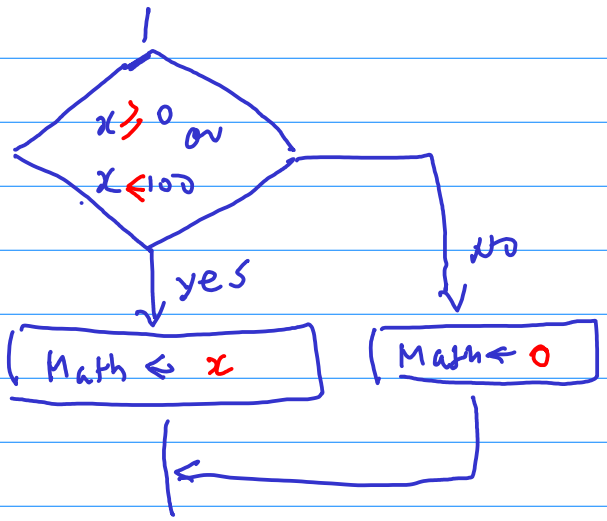
$Math \Leftarrow x$

②

$x<0$ or $x>100$

$Math <= 0$





Class StudentTest2 의 main 함수에서

Student class 의 객체를 참조하는 참조변수

```
S[0].Kor = 20;
S[0].Eng = 30;
S[0].Math = 40;

S[1].setKor( 10 );
S[1].setEng( 10 );
S[1].setMath( 10 );
```

→ private 선언된 field (member data) 는 access 불가능

Error!

private → 객체 참조변수를 통해서 Read / Write

```
class Student {
    private int Kor;
    private int Eng;
    private int Math;

    Student(                        ) { Kor= 0; Eng=0; Math=0;
    Student(int x, int y, int z) { Kor= x; Eng= y; Math= z


    int getKor ()  { return Kor; }
    int getEng ()  { return Eng; }
    int getMath() { return Math; }

    void setKor ( int x ) { Kor= x; }
    void setEng ( int x ) { Eng= x; }
    void setMath( int x ) { Math= x; }
```

→ 같은 class (Student) 내의 member 함수 는 read / write 사용가능

Extends

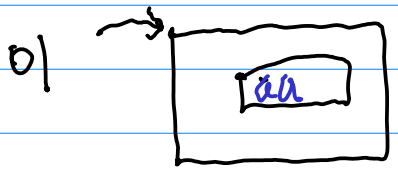
```
class AA {
  int aa;
}
```

```
class BB {
  int bb;
}
```

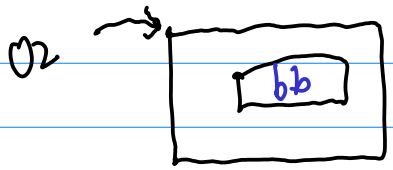
```
class CC {
  int aa;
  int bb;
}
```

```
class DD extends AA {
  int bb;
}
```

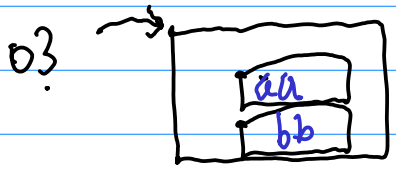
```
AA o1 = new AA();
```



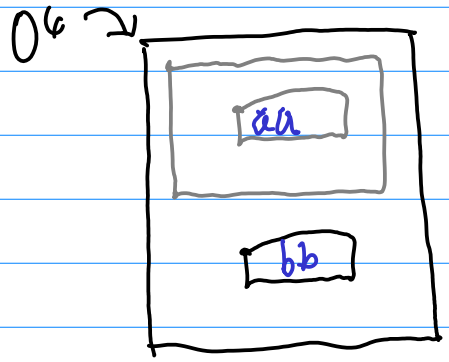

```
BB o2 = new BB();
```




```
CC o3 = new CC();
```




```
DD o4 = new DD();
```



StudentTest2.java

```
class Student {
    ⋮
}

public class StudentTest2{
    ⋮
  -main() ···
    ⋮
}
```

Student.java

```
class Student {
    ⋮
}
```

StudentTest2.java

```
public class StudentTest2{
    ⋮
  -main() ···
    ⋮
}
```

① new class → "Student" → Student.java

```java
public class Student {
     int Kor;
     int Eng;
     int Math;

    Student(                    ) { setKor(0); setEng(0); setMath(0); }
    Student(int x, int y, int z) { setKor(x); setEng(y); setMath(z); }

    int getKor ()  { return Kor; }
    int getEng ()  { return Eng; }
    int getMath() { return Math; }

    void setKor ( int x ) {
        if (x<0 || x>100) Kor = 0;
        else              Kor= x;
    }
    void setEng ( int x ) {
        Eng= x;
        if (x<0 || x>100) Eng = 0;
    }
    void setMath( int x ) {
        if (0<=x && x<=100) Math= x;
        else                Math= 0;
    }
```

```java
static void avg_mode( Student[] X, int mode) {
    double avg= 0.0; int i;

    for (i=0; i<X.length; ++i) {
        switch (mode) {
        case 0:  avg += X[i].Kor;  break; // mode=0
        case 1:  avg += X[i].Eng;  break; // mode=1
        case 2:  avg += X[i].Math; break; // mode=2
        default: avg = 0;  break;
        }
    }
    avg /= X.length;

    String str;
    switch (mode) {
    case 0: str = "** Kor =";  break; // mode=0
    case 1: str = "** Eng =";  break; // mode=1
    case 2: str = "** Math=";  break; // mode=2
    default:str = "** Wrong mode!!!";  break;
    }

    System.out.println(str + avg);

}
```

```java
static void avg_kor( Student[] X ) {
    double avg= 0.0; int i;
    for (i=0; i<X.length; ++i) avg += X[i].Kor;
    avg /= X.length;
    System.out.println("** Kor Avg = " + avg);
}
static void avg_eng( Student[] X ) {
    double avg= 0.0; int i;
    for (i=0; i<X.length; ++i) avg += X[i].Eng;
    avg /= X.length;
    System.out.println("** Eng Avg = " + avg);
}
static void avg_math( Student[] X ) {
    double avg= 0.0; int i;
    for (i=0; i<X.length; ++i) avg += X[i].Math;
    avg /= X.length;
    System.out.println("** Math Avg = " + avg);
}
```

```java
public class Student {
     int Kor;
     int Eng;
     int Math;

    Student(                    ) { setKor(0); setEng(0); setMath(0); }
    Student(int x, int y, int z) { setKor(x); setEng(y); setMath(z); }

    int getKor ()  { return Kor; }
    int getEng ()  { return Eng; }
    int getMath() { return Math; }

    void setKor ( int x ) { 
    void setEng ( int x ) { 
    void setMath( int x ) { 

    double Avg() { return (Kor+Eng+Math) / 3.0; }

    void disp() {

    static void avg_mode( Student[] X, int mode) {

    static void avg_kor( Student[] X ) {
    static void avg_eng( Student[] X ) {
    static void avg_math( Student[] X ) {
    
}
```

```java
class MyStudent extends Student {
    int StID;
    String Name;

    MyStudent() { setKor(0); setEng(0); setMath(0);}
    MyStudent(int x, int y , int z) { setKor(x); setEng(y); setMath(z); }

    int    getStID() { return StID; }
    String getName() { return Name; }

    void setStID( int x )   { StID= x; }
    void setName( String x) { Name= x; }

    void disp() {
        System.out.println( "----------------------");
        System.out.println( "Name= " + Name    );
        System.out.println( "StID= " + StID    );

        System.out.println( "Kor= " + Kor    );
        System.out.println( "Eng= " + Eng    );
        System.out.println( "Math=" + Math   );
        System.out.println( "GPA= " + Avg()  );

    }

}
```

```java
public class StudentTest2 {

    /**
     * @param args
     */
    public static void main(String[] args) {

}
```

```java
public static void main(String[] args) {
    // TODO Auto-generated method stub

    MyStudent[] S = new MyStudent[5]; // S[i] : reference var

    S[0] = new MyStudent(99, 45, 50);
    S[1] = new MyStudent(88, 55, 80);
    S[2] = new MyStudent(77, 65, 90);
    S[3] = new MyStudent(66, 75, 80);
    S[4] = new MyStudent(55, 85, 90);



    S[0].setKor( 0 );
    S[0].setEng( 0 );
    S[0].setMath( 0 );

    S[0].Kor = 20;
    S[0].Eng = 30;
    S[0].Math = 40;

    S[1].setKor( 10 );
    S[1].setEng( 10 );
    S[1].setMath( 10 );
```

```
        S[1].setKor( 10 );
        S[1].setEng( 10 );
        S[1].setMath( 10 );


        S[0].setName("Park");
        S[1].setName("Kim");
        S[2].setName("Lee");
        S[3].setName("Baker");
        S[4].setName("John");

        S[0].setStID(20150001);
        S[1].setStID(20150002);
        S[2].setStID(20150003);
        S[3].setStID(20150004);
        S[4].setStID(20150005);

        S[0].disp();
        S[1].disp();
        S[2].disp();
        S[3].disp();
        S[4].disp();


         Student.avg_kor( S );
         Student.avg_eng( S );
         Student.avg_math( S );

         Student.avg_mode( S, 0);
         Student.avg_mode( S, 1);
         Student.avg_mode( S, 2);
    }

}
```