1811943 황성미 4일차 과제

▼ 3.2 Class

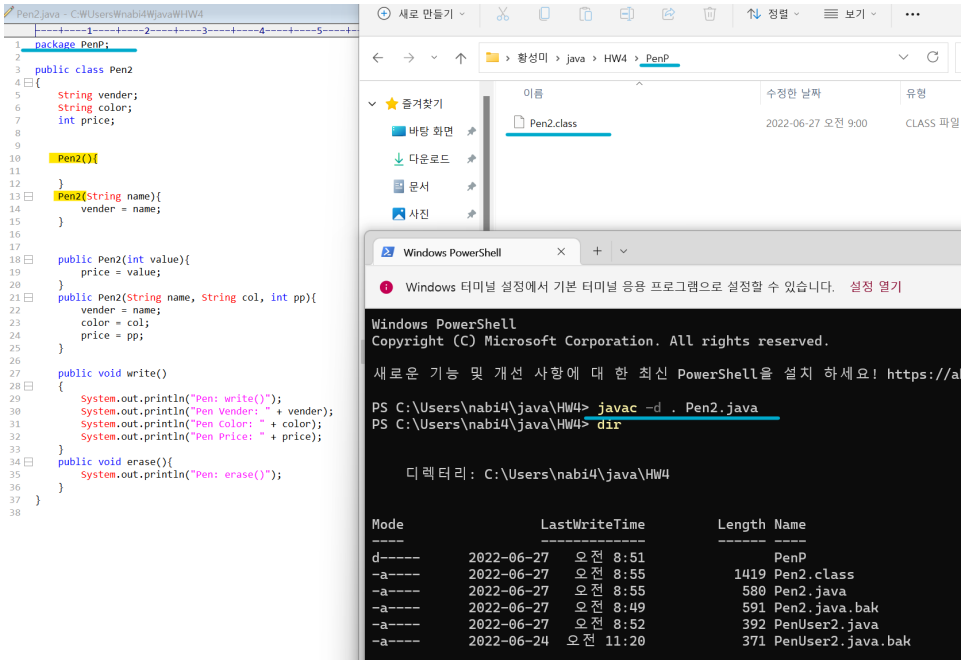
▼ 패키지

: 공유할 수 있도록 클래스 셋을 만들때 사용(현 강의에서는 자바 프로그램 하나를 돌리는걸 배우는 과정이기 때문에 필요없음)

▼ how?

: 첫번째 라인에 패키지를 빌드하고 import를 이용해 사용 가능

▼ Lab1-4



```
Pen2.java - C:\Users\nabi4\java\HW4
package PenP;

public class Pen2
{
    String vender;
    String color;
    int price;


    Pen2(){

    }
    Pen2(String name){
        vender = name;
    }


    public Pen2(int value){
        price = value;
    }
    public Pen2(String name, String col, int pp){
        vender = name;
        color = col;
        price = pp;
    }

    public void write()
    {
        System.out.println("Pen: write()");
        System.out.println("Pen Vender: " + vender);
        System.out.println("Pen Color: " + color);
        System.out.println("Pen Price: " + price);
    }
    public void erase(){
        System.out.println("Pen: erase()");
    }
}
```

```
새로 만들기 | 정렬 | 보기
황성미 › java › HW4 › PenP
즐겨찾기
바탕 화면
다운로드
문서
사진
이름 | 수정한 날짜 | 유형
Pen2.class | 2022-06-27 오전 9:00 | CLASS 파일

Windows PowerShell
Windows 터미널 설정에서 기본 터미널 응용 프로그램으로 설정할 수 있습니다. 설정 열기

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

새로운 기능 및 개선 사항에 대한 최신 PowerShell을 설치하세요! https://a

PS C:\Users\nabi4\java\HW4> javac -d . Pen2.java
PS C:\Users\nabi4\java\HW4> dir


    디렉터리: C:\Users\nabi4\java\HW4


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        2022-06-27   오전 8:51              PenP
-a----        2022-06-27   오전 8:55           1419 Pen2.class
-a----        2022-06-27   오전 8:55            580 Pen2.java
-a----        2022-06-27   오전 8:49            591 Pen2.java.bak
-a----        2022-06-27   오전 8:52            392 PenUser2.java
-a----        2022-06-24  오전 11:20            371 PenUser2.java.bak
```

Pen2를 PenP라는 패키지로 선언한 후 cmd창에서 다음과 같이 입력하면 PenP라는 폴더 안에 Pen2.class가 생긴 것을 볼 수 있음

PenUser2.java - C:₩Users₩nabi4₩java₩HW4

```java
import PenP.Pen2;

class PenUser2
{
    public static void main(String[] args)
    {
        System.out.println("Hello Pen!");

        Pen2 mypen1 = new Pen2();
        mypen1.write();

        Pen2 mypen2 = new Pen2("IT 공학전공");
        mypen2.write();

        Pen2 mypen3 = new Pen2("HSM", "BLUE", 1000);
        mypen3.write();

        Pen2 mypen4 = new Pen2(2022);
        mypen4.write();
    }
}
```

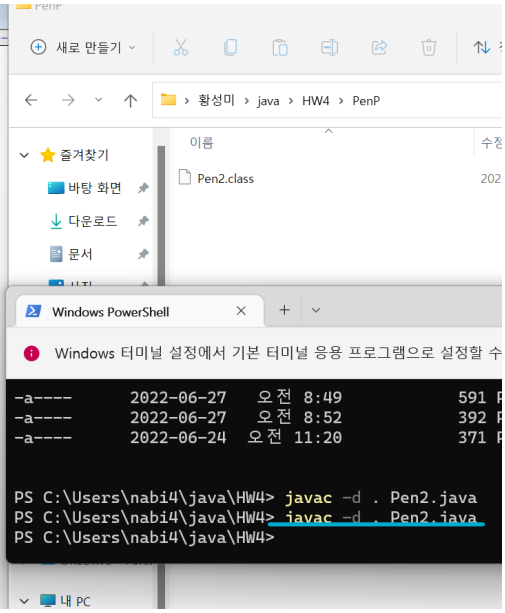
C:₩Windows₩system32₩cmd.exe

```
PenUser2.java:9: error: constructor Pen2 in class Pen2 cannot be applied to given types;
                Pen2 mypen1 = new Pen2();
                              ^
  required: no arguments
  found:    no arguments
  reason: Pen2() is not public in Pen2; cannot be accessed from outside package
PenUser2.java:12: error: no suitable constructor found for Pen2(String)
                Pen2 mypen2 = new Pen2("IT 공학전공");
                              ^
    constructor Pen2.Pen2(String) is not applicable
      (Pen2(String) is not public in Pen2; cannot be accessed from outside package)
    constructor Pen2.Pen2(int) is not applicable
      (argument mismatch; String cannot be converted to int)
Note: Some messages have been simplified; recompile with -Xdiags:verbose to get full output
2 errors
계속하려면 아무 키나 누르십시오 . . .
```

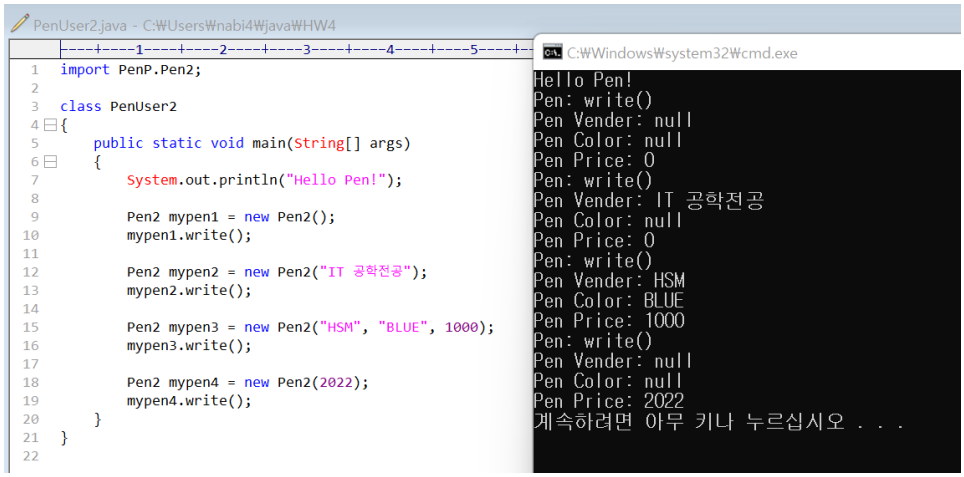
이후 PenUser2.java를 실행시키면 access visibility 문제 발생
이를 통해 패키지를 쓸 경우에는 public으로 선언해야함을 알 수 있음

Pen2.java - C:₩Users₩nabi4₩java₩HW4

```java
package PenP;

public class Pen2
{
    String vender;
    String color;
    int price;


    public Pen2(){

    }
    public Pen2(String name){
        vender = name;
    }


    public Pen2(int value){
        price = value;
    }
    public Pen2(String name, String col, int pp){
        vender = name;
        color = col;
        price = pp;
    }

    public void write()
    {
        System.out.println("Pen: write()");
        System.out.println("Pen Vender: " + vender);
        System.out.println("Pen Color: " + color);
        System.out.println("Pen Price: " + price);
    }
    public void erase(){
        System.out.println("Pen: erase()");
    }
}
```



Pen2.java로 돌아가 모두 다 public으로 바꿔준 다음 cmd창에서 javac를 한 번 더 해주고



PenUser2.java를 실행시키면 문제없이 실행되는 것을 볼 수 있음

▼ this and super

▼ this / this()

: 자기 자신을 가리킴(지칭) / 자기 자신의 객체를 또 다시 만듬(새로운 객체 생성하여 새롭게 사용하고 싶을때)

▼ super / super()

: 부모를 가리킴(지칭) / 부모 클래스 객체를 또 다시 만듬(디폴트가 부모 속성)

▼ instance of

: 클래스 타입을 체크하고 싶을 때

▼ HW2-1(ECA, IDL)

▼ ECA

▼ ECA Rule(Event Condition Action) ← Event-Driven 하는 모든 모델의 기본

Event(함수 호출 또는 GUI를 통한 클릭 등등),

Condition(event가 발생할 때 생기는 조건(=signature, message(함수 이름,변수 타입,변수의 수))을 체크하여 해당되는 method를 찾아줌),

Action(선택된 method를 실행)

▼ Message Passing

: ECA rule에 의해 method가 전달되면서 method invocation이 일어남

▼ IDL

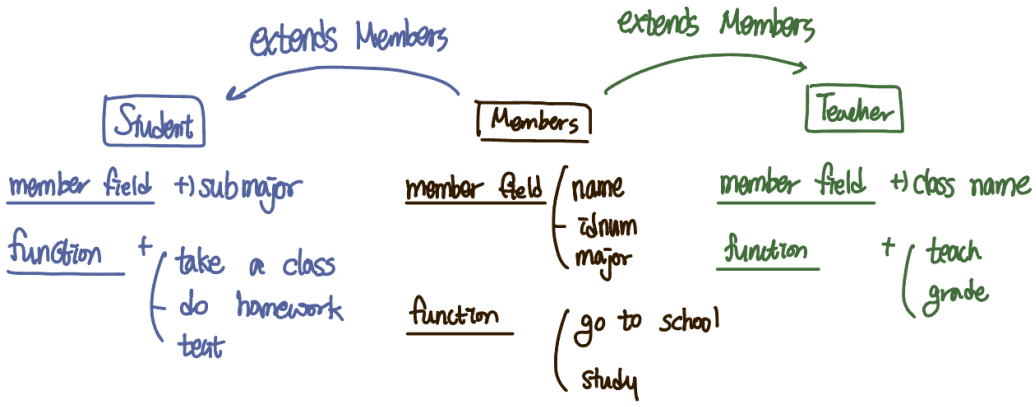
IDL : Interface Definition Language

= Class Definition(필요한 data와 method를 정의만 할 뿐 세부적인 구현은 아직 하지 않은 상태)

어느 한 언어에 국한되지 않은 언어중립적인 방법으로 인터페이스를 표현함으로써 같은 언어를 사용하지 않는 소프트웨어 컴포넌트 사이의 통신을 가능하게 함

쉽게 말해, 프로그램이나 객체가 알려지지 않은 언어로 작성된 다른 프로그램과 통신을 할 수 있도록 해주는 언어

▼ HW2-2(학습 관리 설계 student, teacher 구상해보기)



상위레벨로 Members 클래스가 있으며
이를 상속받는 하위 레벨로는 Student, Teacher가 있음

각각 Memeber에 있는 멤버 필드와 메서드를 가지되 추가로 각 클래스에 필요한 필드와 메서드들이 추가되어있음

▼ 3.3 Inheritance

▼ 상속이 필요한 이유

: 그냥 만드는 것이 아닌 이미 존재하는 클래스의 속성들을 받아 새로운 클래스의 속성을 “확장”하기 위함

예시) New_class extends Servlet , New_class extends Activity ,
New_class extends Frame, New_class extends JFrame : 이미 갖고 있는 창을 띄워서 실행시키겠다

▼ Is-a Relationship

: 의미, 분류 기준, 속성에 의해서 일반적인 개념의 상속을 정의

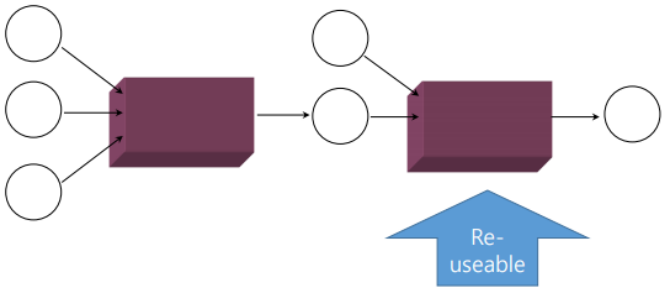
예시) 인간 → 여자 → 여대생 → 엔지니어링학과 여대생

각각의 속성, 기능

▼ CBD가 상속의 개념을 잘 정의해줌(잘못하면 속성이 많이 들어갈 수 있음)

▼ CBD(Component Based Design/Development)

: 기존의 시스템이나 소프트웨어를 구성하는 컴포넌트를 조립해서 하나의 새로운 응용 프로그램을 만드는 소프트웨어 개발방법론



*Re-useable(재사용 가능) 이 주요 특징

▼ 예시

: Student-Teacher는 동위 레벨 - 상위레벨로 Member - School가 있을 수 있음
각 상위레벨은 우리가 정의한 하위 레벨 외에 다른 하위 레벨을 가질 수 있음

▼ how?

: 상위레벨 -extends→ 하위레벨 (상위레벨의 속성을 상속받음, 하위레벨로 갈수록 속성이 member fields와 function(method)이 확장됨)

-final : 상속을 더이상 시키고 싶지 않을때

-static : 상속은 하지만 바꿀 수 없는 변수(고정된 값)

-overriding : 상속해서 method 변경 가능 - implements할 때 일어나게 됨(인터페이스 클래스 상속)

▼ java에서의 상속

-다른 클래스를 확장 : single inheritance

-다른 인터페이스를 implements(모든 사람들이 동일하게 쓰기 위한 양식 , body는 개발자가 목적에 따라서 구현) : multiple inheritance

▼ 이벤트 핸들링 시 사용

▼ super / super()

:부모 클래스를 지칭하고 부모 클래스의 속성을 디폴트값으로 하는 새로운 객체를 생성 → 상속에서 중요!

cf) this() : 현재 클래스의 속성을 초기화하고 다시 쓰고 싶을 때 사용하지만 잘 쓰진 않음

▼ Lab2

1. 클래스 정의
2. 클래스 디자인
3. 오버라이딩을 이용한 각 클래스의 메서드 구현

Members.java - C:₩Users₩nabi4₩java₩HW4

```java
class Members
{
    String name;
    String dept;
    String major;
    int id;

    public Members(String name, String dept)
    {
        this.name = name;
        this.dept = dept;
    }
    public void setId(int id){
        this.id = id;
    }
    public void setMafor(String major){
        this.major = major;
    }
    public void work(){
        System.out.println("\tMemebers : \"" + name + "\" does his best.");
    }
}
```

Teacher.java - C:₩Users₩nabi4₩java₩HW4

```java
class Teacher extends Members
{
    String dept;
    Members students[];

    public Teacher(String name, String idnum, String dept)
    {
        super(name, idnum);
        this.dept = dept;
    }
    public void setStudents(Members sub[]){
        students = sub;
    }

    //오버라이딩
    public void work(){
        System.out.println("Teacher : \"" + name +
            "\" studies hard with his students in " + dept +
            " dept.");
    }
}
```

Student.java - C:₩Users₩nabi4₩java₩HW4

```java
class Student extends Members
{
    String dept;

    public Student(String name, String idnum, String dept)
    {
        super(name, idnum);
        this.dept = dept;
    }

    //오버라이딩
    public void work(){
        System.out.println("Student : \"" + name +
            "\" studies hard with his teacher in " + dept +
            " dept.");
    }
}
```

상위 레벨 Member 를 상속받아 하위 레벨 Teacher와 Student 구현
이때 Members students[]; 는 선생님의 강의를 듣는 학생수를 알기 위한 setStudents를 작성하기 위해 구현했지만 Lab2에서 중점적으로 보려는 오버라이딩과는 무관!
Member의 work함수를 오버라이딩하여 Teacher, Student가 하는 work에 맞게 코드를 구현

School.java - C:₩Users₩nabi4₩java₩HW4

```
1  class School
2  {
3      Teacher yiyoon;
4      Student kim, song, choi, lee;
5      Members members[];
6
7      public School()
8      {
9          yiyoon = new Teacher("Yoon", "M103313", "IT공학전공");
10         kim = new Student("kim", "STAT1811943", "통계학과");
11         song = new Student("song", "IT201234", "IT공학전공");
12         choi = new Student("choi", "IT193834", "IT공학전공");
13
14         members = new Members[4];
15         members[0] = yiyoon;
16         members[1] = kim;
17         members[2] = song;
18         members[3] = choi;
19     }
20
21     public void makeWork(){
22         int n = members.length;
23
24         for (int i = 0; i < n ; i++ )
25         {
26             members[i].work();
27         }
28     }
29 }
```

SchoolMain.java - C:₩Users₩nabi4₩java₩HW4

```
1  class SchoolMain
2  {
3      public static void main(String[] args)
4      {
5          //만들어진 School생성자를 mycom
6          School mycom = new School();
7          mycom.makeWork();
8          //mycom의 makeWork 함수를 실행
9      }
10 }
```

Main에서 School을 새로운 생성자로 만든 (1) mycom의 makeWork 함수를 실행시키면, (2)School의 makeWork() 함수가 실행되고, (3) members의 work함수를 실행시키면, (4) 각각 상속받아 오버라이딩한 work의 함수가 실행됨!

C:₩Windows₩system32₩cmd.exe

```
Teacher : "Yoon" studies hard with his students in IT공학전공 dept.
Student : "kim" studies hard with his teacher in 통계학과 dept.
Student : "song" studies hard with his teacher in IT공학전공 dept.
Student : "choi" studies hard with his teacher in IT공학전공 dept.
계속하려면 아무 키나 누르십시오 . . .
```

위와 같이 오버라이딩한 Teacher 클래스의 work 함수와 Student 클래스의 work 함수가 잘 실행된 것을 볼 수 있음

▼ 3.4 super and super()

▼ super / super()

: 부모의 속성 사용 가능 / 부모 속성의 constructor (부모 속성이 디폴트로 다 적용되어있는 장점!)

▼ HW(inheritance 실습)

X.java - C:₩Users₩nabi4₩java₩HW4

```java
class X
{
    int xi = 10;
    String msg = "I am an X.";

    void print(){
        System.out.println(msg);
    }

    void play(){
        System.out.println("Play.." + msg);
    }
}
```

Y.java - C:₩Users₩nabi4₩java₩HW4

```java
class Y extends X
{
    int yi = 20;
    String msg = "I am an Y.";

    void print(){
        System.out.println(msg);
    }
}
```

Z.java - C:₩Users₩nabi4₩java₩HW4

```java
class Z extends Y
{
    int zi = 30;
    String msg = "I am an Z.";

    void print(){
        System.out.println(msg);
    }

    void play(){
        System.out.println("Play.." + msg);
    }

    void doZ(){
        System.out.println("do somthing in Z.");
    }

    void test(int ti){
        Z z = new Z();
        Y y = z;
        X x = z;
        z.print();
        y.print();
        super.print();
        play();
        super.play();

        System.out.println("\nti = " + ti);
        System.out.println("yi = " + yi);
        System.out.println("xi = " + xi);

        System.out.println("this.zi = " + this.zi);
        System.out.println("this.yi = " + this.yi);
        System.out.println("this.xi = " + this.xi);

        System.out.println("super.yi = " + super.yi);
        System.out.println("y.yi = " + y.yi);
        System.out.println("x.xi = " + x.xi);
        System.out.println("((Y))this).yi = " + ((Y)this).yi);
        System.out.println("((X))this).xi = " + ((X)this).xi);

    }
}
```

X 클래스(print(), play() 메서드),
X를 상속받은 Y 클래스(X의 print()메서드를 오버라이드),
Y를 상속받은 Z 클래스(Y의 print()메서드와 Y가 상속받은 X의 play() 메서드를 오버라이드하고
test() 메서드들 추가)

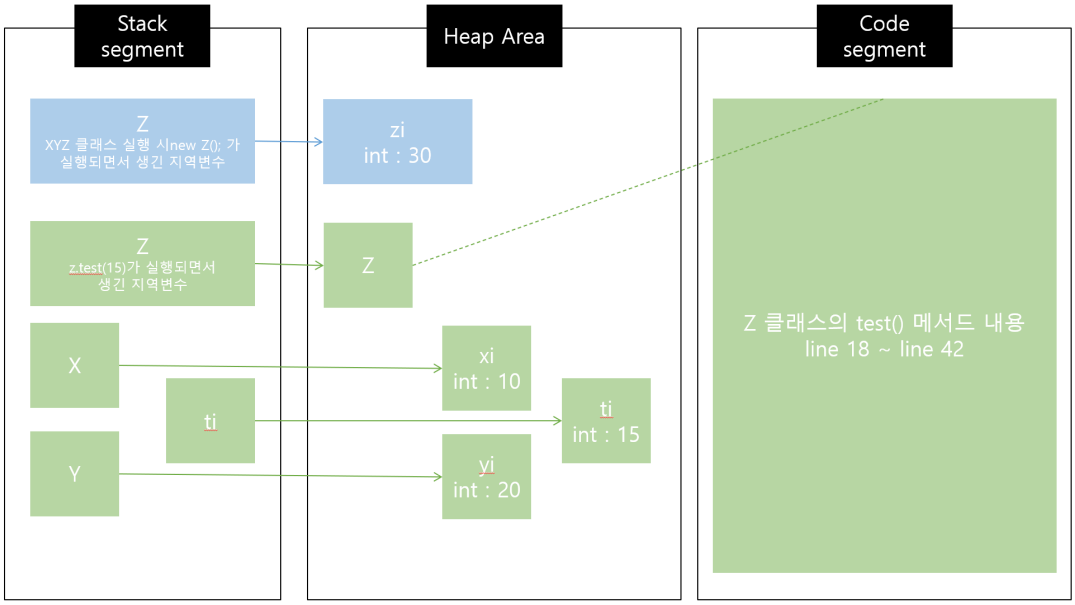
XYZ.java - C:₩Users₩nabi4₩java₩HW4

```java
class XYZ
{
    public static void main(String[] args)
    {
        Z z = new Z();
        z.test(15);
    }
}
```

C:₩Windows₩system32₩cmd.e...

```
I am an Z.
I am an Z.
I am an Y.
Play..I am an Z.
Play..I am an X.

ti = 15
yi = 20
xi = 10
this.zi = 30
this.yi = 20
this.xi = 10
super.yi = 20
y.yi = 20
x.xi = 10
((Y))this).yi = 20
((X))this).xi = 10
계속하려면 아무 키나 누르십시오 . . .
```

Z를 새로운 생성자로 만들고 입력인자를 15로 가지는 test 함수를 실행시킴
Z 클래스의 인스턴스인 z가 스택 영역에 생성되고
Z 클래스와 연관된 정보는 heap 영역에 올라감.
그 후 z.test(15)가 실행되면 ti = 15가 되면서 Z 클래스의 메서드 test()가 실행됨



XYZ클래스를 실행시키면서 new Z()를 통해 만들어진 지역변수가 생기고 이 변수가 heap area에 공간이 잡힘
이어서 z.test() 함수가 호출되면서 Z 영역이 다시 또 잡히고, 입력 인자로 준 ti도 값이 생기며, Z는 Y, Y는 X를 상속받은 변수이기 때문에 이들도 영역이 생김
z.test()를 실행시킬 때 스택과 힙 영역에 자리가 잡히면 코드 세그먼트에서 코드를 실행함.

*정확히 객체 생성시 스택 영역과 힙 영역, 코드 영역의 생성 과정에 대한 충분한 이해가 되지 않은 것 같다. 더 찾아보고 공부해봐야할 부분!!

# of line	출력	설명
line 22 z.print();	I am an Z.	Z 클래스의 print() 함수가 호출됨
line 23 y.print();	I am an Z.	print()함수가 Z 클래스에 의해 오버라이드되어 y.print()임에도 불구하고 Z 클래스의 print()함수가 호출됨
line 24 super.print();	I am an Y.	Z 클래스의 super 클래스인 Y 클래스의 print()함수가 호출됨
line 25 play();	Play.. I am an Z.	Z 클래스의 play() 함수가 호출됨
line 26 super.play();	Play..I am an X.	Z 클래스의 super class의 play() 함수가 호출되어야하는데 Y클래스에는 play()함수가 없으므로 상속받은 X 클래스의 play()함수가 호출됨
line 28 System.out.println("\nti = " + ti);	ti = 15	지역변수 출력됨

line 29 System.out.println("yi = " + yi);	yi = 20	Y 클래스의 yi 출력됨
line 30 System.out.println("xi = " + xi);	xi = 10	X 클래스의 xi 출력됨
line 32 System.out.println("this.zi = " + this.zi);	this.zi = 30	Z 클래스의 멤버 필드에 접근하기 위해 this를 사용했기 때문에 zi가 출력됨
line 33 System.out.println("this.yi = " + this.yi);	this.yi = 20	Y 클래스의 멤버 필드에 접근하기 위해 this를 사용했기 때문에 yi가 출력됨
line 34 System.out.println("this.xi = " + this.xi);	this.xi = 10	X 클래스의 멤버 필드에 접근하기 위해 this를 사용했기 때문에 xi가 출력됨
line 36 System.out.println("super.yi = " + super.yi);	super.yi = 20	Z의 super 클래스인 Y클래스의 멤버 필드에 접근하여 yi 출력
line 37 System.out.println("y.yi = " + y.yi);	y.yi = 20	Y 클래스의 멤버 필드에 접근하여 yi 출력
line 38 System.out.println("x.xi = " + x.xi);	x.xi = 10	X 클래스의 멤버 필드에 접근하여 xi 출력
line 39 System.out.println("((Y))this).yi = " + ((Y)this).yi);	((Y))this).yi = 20	자기 자신을 Y 클래스로 형변환 시키면 Y클래스의 레퍼런스가 되고 Y 클래스의 레퍼런스로는 Z 클래스의 멤버 필드에 접근할 수 없음(자신에게 해당하는 클래스에만 접근 가능). yi인 20 출력
line 40 System.out.println("((X))this).xi = " + ((X)this).xi);	((X))this).xi = 10	자기 자신을 X 클래스로 형변환 시키면 X클래스의 레퍼런스가 되고 X 클래스의 레퍼런스로는 Z 클래스의 멤버 필드에 접근할 수 없음. xi인 10 출력

▼ 컴파일 에러

```java
18    void test(int ti){
19        Z z = new Z();
20        Y y = z;
21        X x = z;
22        z.print();
23        y.print();
24        super.print();
25        play();
26        super.play();
27        y.doZ(); //컴파일에러
28        super.super.print(); //컴파일에러
29
30        System.out.println("\nti = " + ti);
31        System.out.println("yi = " + yi);
32        System.out.println("xi = " + xi);
33
34        System.out.println("this.zi = " + this.zi);
35        System.out.println("this.yi = " + this.yi);
36        System.out.println("this.xi = " + this.xi);
37
38        System.out.println("super.yi = " + super.yi);
39        System.out.println("y.yi = " + y.yi);
40        System.out.println("x.xi = " + x.xi);
41        System.out.println("((Y))this).yi = " + ((Y)this).yi);
42        System.out.println("((X))this).xi = " + ((X)this).xi);
43        super.super.xi = 10; //컴파일에러
44
45    }
```

```
C:₩Windows₩system32₩cmd.exe
Z.java:28: error: <identifier> expected
                super.super.print(); //컴파일에러
                      ^
Z.java:28: error: not a statement
                super.super.print(); //컴파일에러
                     ^
Z.java:43: error: <identifier> expected
                super.super.xi = 10; //컴파일에러
                      ^
Z.java:43: error: not a statement
                super.super.xi = 10; //컴파일에러
                     ^
4 errors
계속하려면 아무 키나 누르십시오 . . .
```

super를 연달아서 사용할 수 없음

```java
18    void test(int ti){
19        Z z = new Z();
20        Y y = z;
21        X x = z;
22        z.print();
23        y.print();
24        super.print();
25        play();
26        super.play();
27        y.doZ(); //컴파일에러
28        //super.super.print(); //컴파일에러
29
30        System.out.println("\nti = " + ti);
31        System.out.println("yi = " + yi);
32        System.out.println("xi = " + xi);
33
34        System.out.println("this.zi = " + this.zi);
35        System.out.println("this.yi = " + this.yi);
36        System.out.println("this.xi = " + this.xi);
37
38        System.out.println("super.yi = " + super.yi);
39        System.out.println("y.yi = " + y.yi);
40        System.out.println("x.xi = " + x.xi);
41        System.out.println("((Y))this).yi = " + ((Y)this).yi);
42        System.out.println("((X))this).xi = " + ((X)this).xi);
43        //super.super.xi = 10; //컴파일에러
44
45    }
```

```
C:₩Windows₩system32₩cmd.exe
Z.java:27: error: cannot find symbol
                y.doZ(); //컴파일에러
                  ^
  symbol:   method doZ()
  location: variable y of type Y
1 error
계속하려면 아무 키나 누르십시오 . . .
```

Y 클래스의 레퍼런스로는 Z 클래스의 멤버 필드나 메서드에 접근할 수 없음