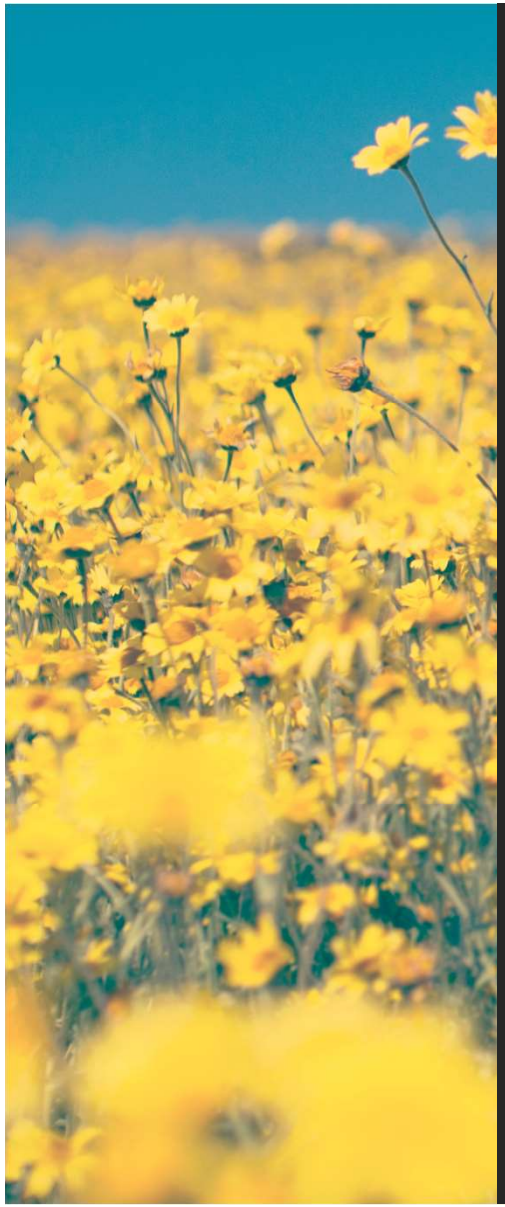


JAVA

GU, DA HAE


상 속
생 성 자
protected
오 버 라 이 딩
instanceof
abstract
interface
inner class
JAVA
객
체

JAVA

protected

키워드	클래스 내부	동일 패키지	상속받은 클래스	이외의 영역
private	O	X	X	X
default	O	O	X	X
protected	O	O	O	X
public	O	O	O	O

JAVA

예제

```
class A{
    int x;
    protected int y;
    public int z;
}
class B extends A{
    public B(){
        x = 10;
        // 같은 패키지로 묶여있기 때문에 접근 가능.
        y = 20;
        // 다른 패키지로 나뉘어있을 지라도 상속 관계라면 접근 가능.
        z = 30;
    }
}
```

```
public class Ex01{
    public static void main(String[] args){
        B obj = new B();
        System.out.println(obj.x);
        System.out.println(obj.y);
        System.out.println(obj.z);
    }
}
```

JAVA

예제

```
class A{
	private int x;
}
class B extends A{
	public B(){
		x = 10; // ERROR : private 멤버도 상속되지만, 직접 접근 불가능
	}
}

public class Ex01{
	public static void main(String[] args){
		B obj = new B();
		System.out.println(obj.x);
	}
}
```

JAVA

예제

```java
class A{
    private int x;
    int y;
    public A(int x, int y){
        this.x = x;
        this.y = y;
    }
    public int getX(){ return x; }
    public void setX(int x){ this.x = x;}
}
```

```java
class B extends A{
    public B(){
        super(10, 20);
        // priavte 멤버는 간접 접근
        System.out.println( getX() );
    }
}

public class Ex01{
    public static void main(String[] args){
        B obj = new B();
        System.out.println(obj.x);
    }
}
```

JAVA

오버라이딩 Overriding

하위 클래스에서 상위 클래스의 메서드를 다시 정의하는 것이다.

JAVA

예제

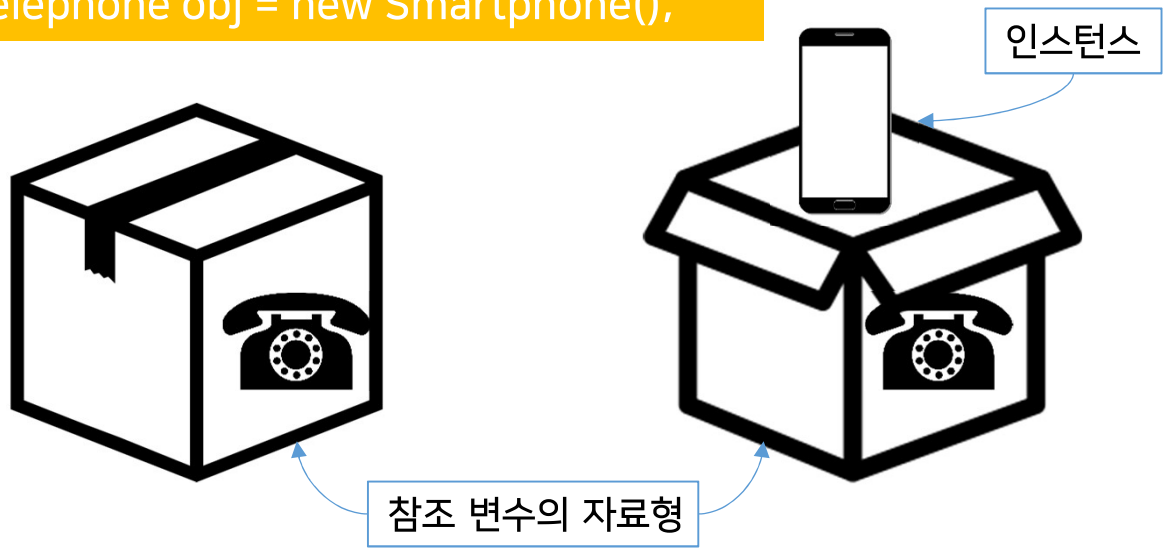
```
class Person{
    private String name;
    private int age;
    public Person(String name, int age){
        this.name = name;
        this.age = age;
    }
    public void showInfo(){
        System.out.println("이름 : " + name);
        System.out.println("나이 : " + age);
    }
}
class Student extends Person{
     private int year;
     private int group;

     public Student(String name, int age, int year,
     int group){
          super(name, age);
          this.year = year;
          this.group = group;
     }
     Public void showInfo(){
          super.showInfo();
          System.out.println(year + "학년 " + group +
          "반");
     }
}
public class Ex01{
    public static void main(String[] args){
        Person p = new Person("홍길동", 30);
        Student s = new Student("김진아", 12, 3, 1);
        p.showInfo();
        s.showInfo();
    }
}
```

JAVA

참조변수의 다형성

상위 클래스의 참조변수는 하위 클래스의 인스턴스 참조가 가능하다.

```
class Telephone{}
class Smartphone extends Telephone{}

Telephone obj = new Smartphone();
```



JAVA

예제

```
class A{
	public void override(){
		System.out.println("A override!");
	}
	public void lode(){
		System.out.println("lode()");
	}
}
class B extends A{
	public void override(){
		System.out.println("B override!");
	}
	public void lode(int x){
		System.out.println("lode(int)");
	}
}
```

```
class C extends B{
	public void override(){
		System.out.println("C override!");
	}
	public void lode(int x, int y){
		System.out.println("lode(int, int)");
	}
}
public class Ex01{
	public static void main(String[] args){
		A obj1 = new A();
		B obj2 = obj1; //ERROR
		C obj3 = obj1; //ERROR
	}
}
```

JAVA

예제

```
class A{
	public void override(){System.out.println("A override!"); }
	public void lode(){System.out.println("lode()");}
}
class B extends A{
	public void override(){System.out.println("B override!"); }
	public void lode(int x){System.out.println("lode(int)");}
}
class C extends B{
	public void override(){System.out.println("C override!"); }
	public void lode(int x, int y){System.out.println("lode(int, int)");}
}
```

```
public class Ex01{
	public static void main(String[] args){
		C ref1 = new C();
		B ref2 = ref1;
		A ref3 = ref1;

		ref1.override(); // C override!
		ref2.override(); // C override!
		ref3.override(); // C override!

		ref1.lode(); // lode()
		ref2.lode(); // lode()
		ref3.lode(); // lode()
	}
}
```

JAVA

예제

```
class A{
	public void override(){System.out.println("A override!"); }
	public void lode(){System.out.println("lode()");}
}
class B extends A{
	public void override(){System.out.println("B override!"); }
	public void lode(int x){System.out.println("lode(int)");}
}
class C extends B{
	public void override(){System.out.println("C override!"); }
	public void lode(int x, int y){System.out.println("lode(int, int)");}
}
```

```
public class Ex01{
	public static void main(String[] args){
		C ref1 = new C();
		B ref2 = ref1;
		A ref3 = ref1;

		// 어느 부분에서 ERROR가 발생할까?
		ref1.lode(1);
		ref2.lode(1);
		ref3.lode(1);

		ref1.lode(1, 2);
		ref2.lode(1, 2);
		ref3.lode(1, 2);
	}
}
```

JAVA

예제

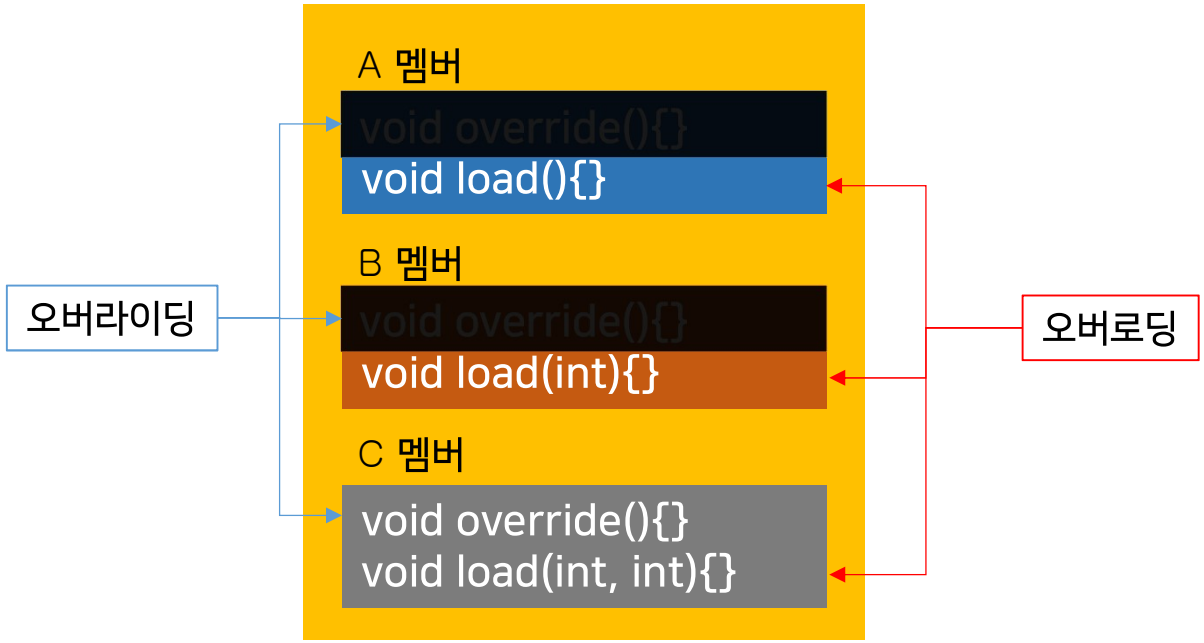
```
class A{
	public void override(){System.out.println("A override!"); }
	public void lode(){System.out.println("lode()");}
}
class B extends A{
	public void override(){System.out.println("B override!"); }
	public void lode(int x){System.out.println("lode(int)");}
}
class C extends B{
	public void override(){System.out.println("C override!"); }
	public void lode(int x, int y){System.out.println("lode(int, int)");}
}
```

```
public class Ex01{
	public static void main(String[] args){
		B ref2 = new B();
		A ref3 = ref2;

		ref2.override();
		ref3.override();

		ref2.lode();
		ref3.lode();
		ref2.lode(1);
		ref3.lode(1); // ERROR
	}
}
```

JAVA

예제

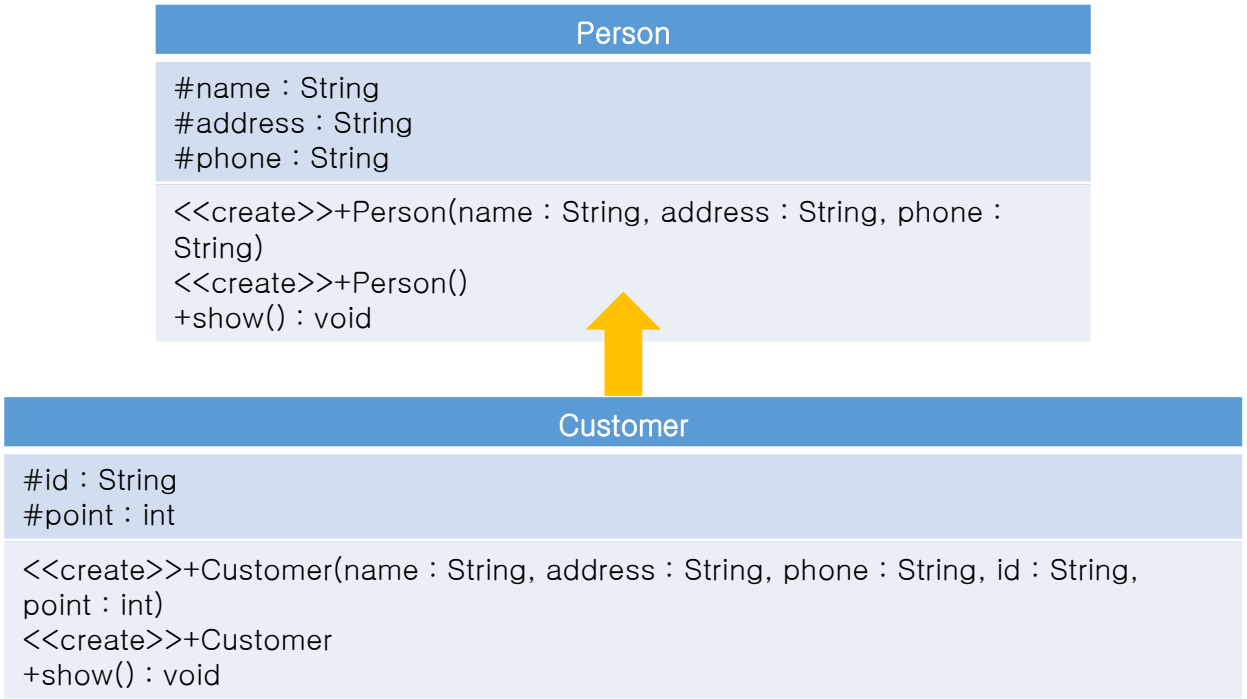
참조변수의 자료형에 상관 없이 마지막으로 오버라이딩한 메서드를 호출한다.



JAVA

해보기

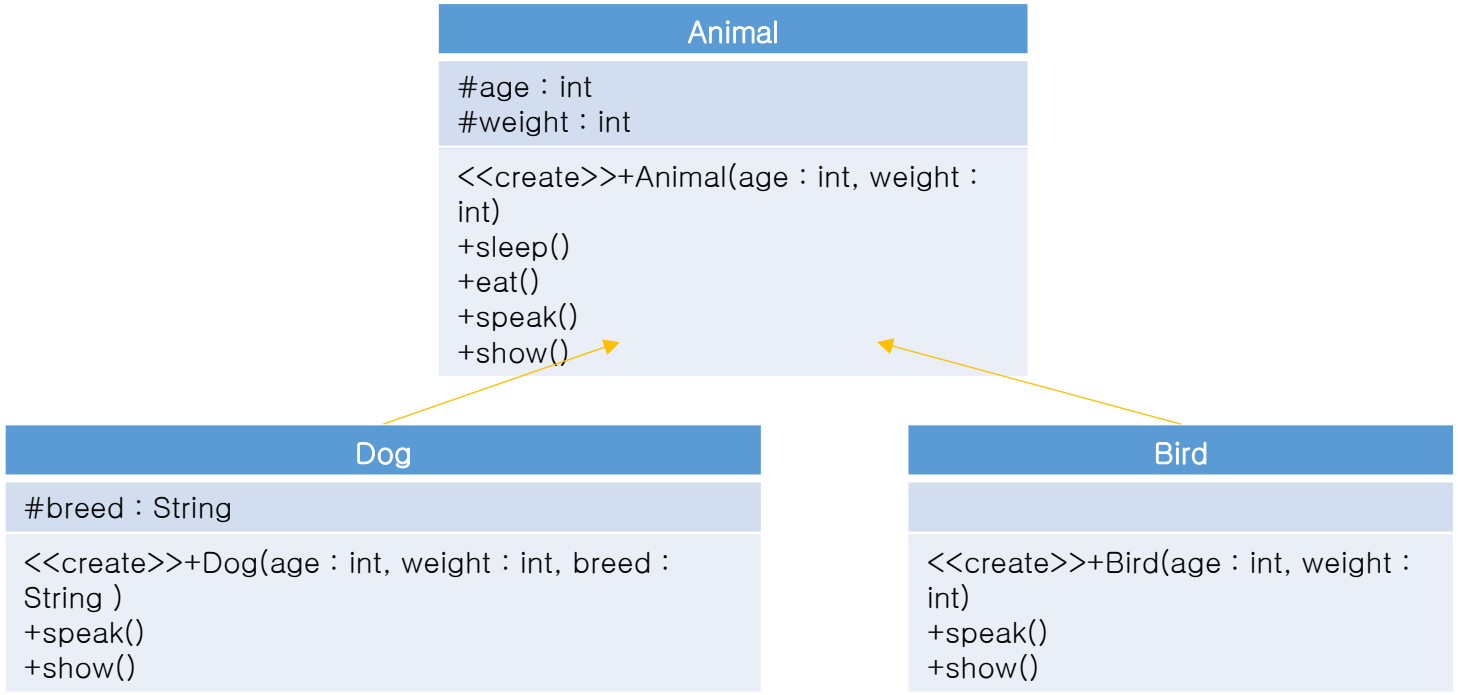
일반인과 달리 회원으로 가입된 고객에게는 포인트 점수를 누적하는 시스템을 운영하기 위해서 다음과 같은 상속 구조를 갖는 클래스를 정의해보자.



JAVA

해보기

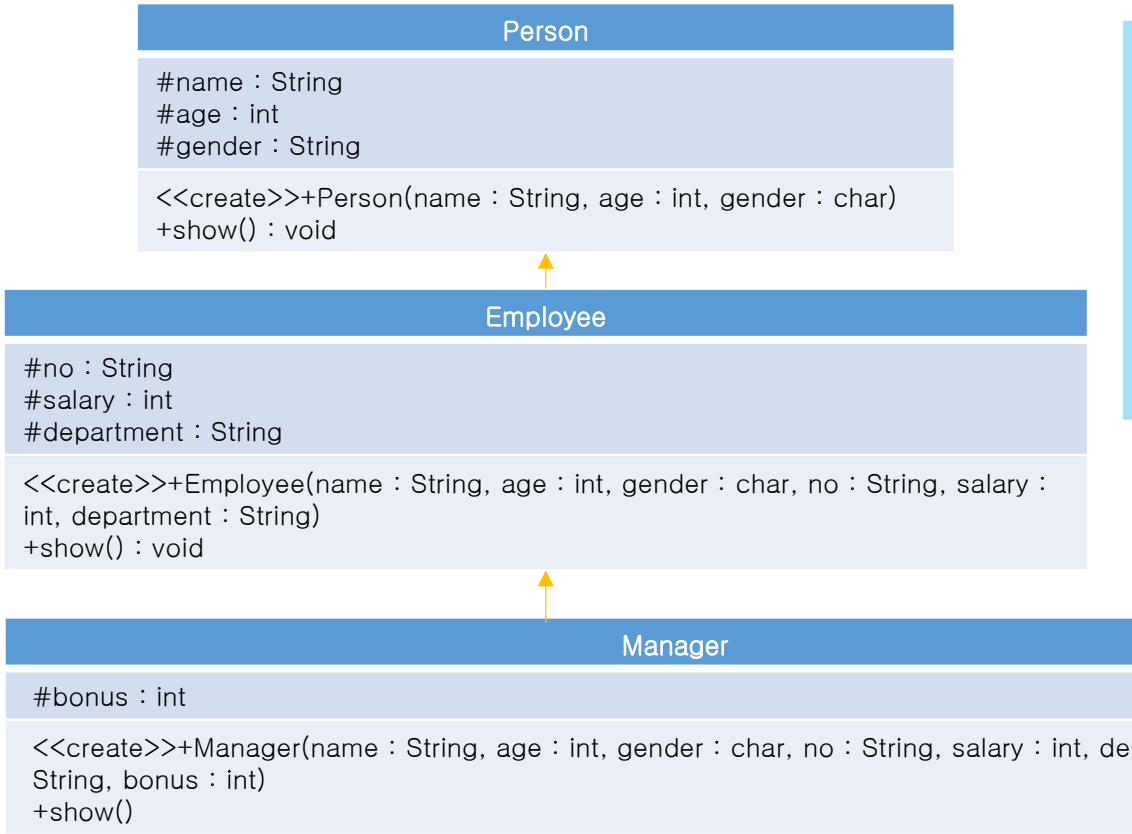
Animal 클래스의 speak() 멤버함수를 Dog, Bird 클래스에서 오버라이딩하여 각 동물 고유의 울음소리를 출력해보자.



JAVA

해보기

상속 관계의 클래스를 정의하고 다음과 같은 결과가 나오도록 프로그램을 작성해보자.

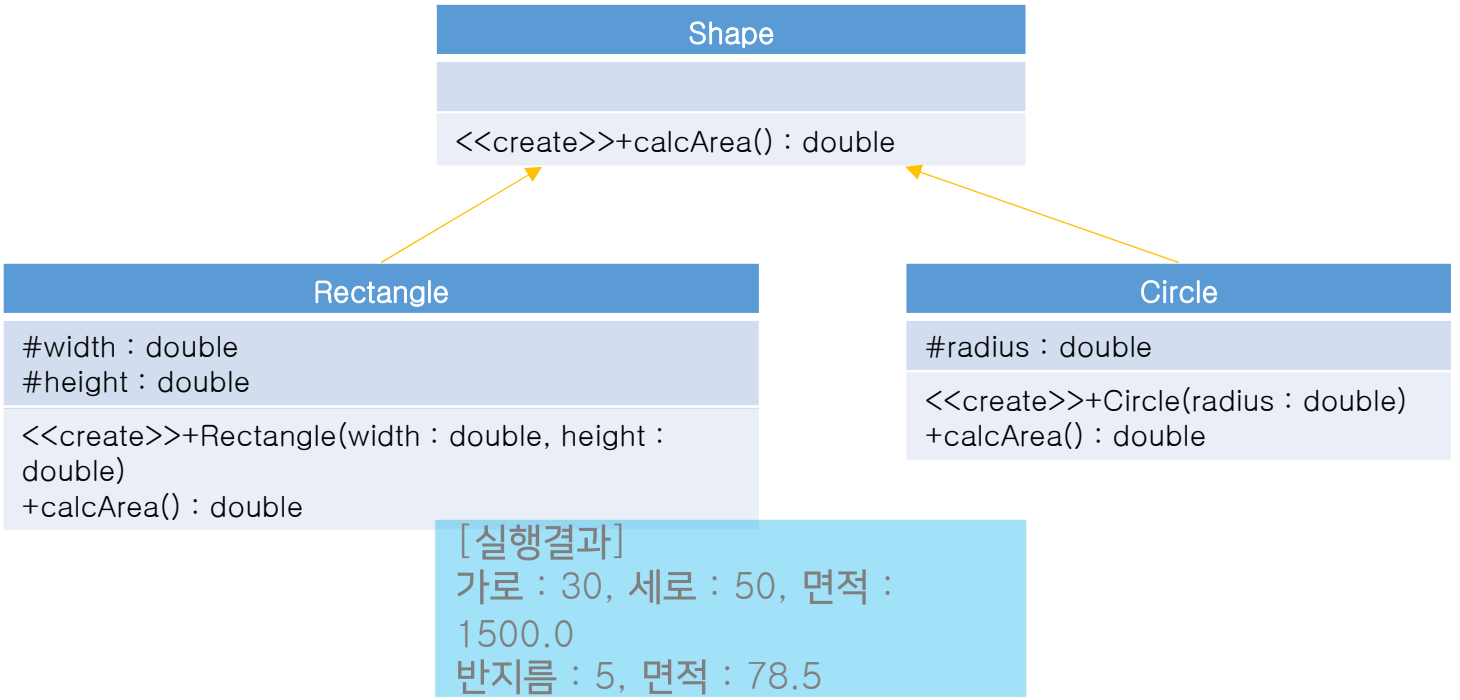


[실행 결과]
이름 : 홍길동
나이 : 30
성별 : M
사번 : 101
급여 : 5000
부서 : 인사과
보너스 : 2000

JAVA

해보기

사각형, 원 클래스를 설계하되 도형 클래스의 상속을 받는 자식 클래스로 설계하고 다음과 같은 결과가 나오도록 프로그램을 작성해보자.



JAVA

해보기

정사각형을 의미하는 Square 클래스와 직사각형을 의미하는 Rectangle 클래스를 정의하고자 한다. 그런데 정사각형은 직사각형의 일종이므로 상속이 가능하다. 아래의 main 메서드와 함께 실행이 가능하도록 클래스를 완성해보자.

```
class Rectangle{...}
class Square extends Rectangle{...}


public class Ex01 {
    public static void main(String[] args) {
        Rectangle rec(4, 3);
        rec.ShowAreaInfo();

        Square sqr(7);
        sqr.ShowAreaInfo();
    }
}
```