제 24 강

상속과 오버라이딩

교재:p108~117

목차

클래스와 객체 복습

클래스란?

객체를 만들기 위한 설계도

객체란?

독립적으로 존재할 수 있는 것

인스턴스란?

클래스로 실제 메모리에 생성된 객체(객체가 더 포괄적인 의미)

클래스와 객체 복습

클래스 생성 방법

```
class 클래스명{
    변수 혹은 메서드    }
```

객체(인스턴스) 생성 방법

```
클래스명 참조변수 = new 클래스명( );
```

객체 내 필드 접근 방법

```
참조변수.필드명
```

상속과 오버라이딩

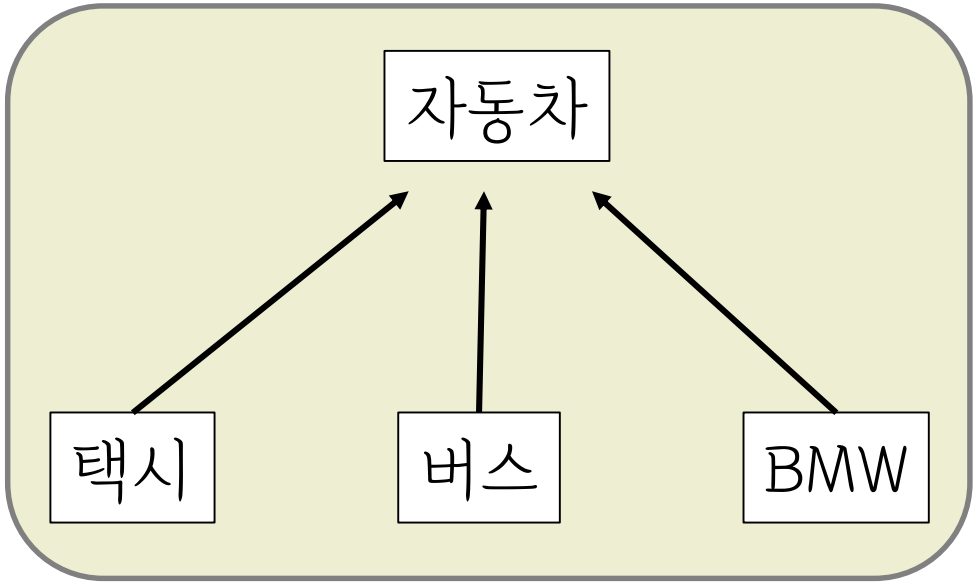
1. 상속

상속이란?

: 부모클래스의 기능을 자식클래스가 물려받는 것

사용목적?

: 부모클래스에서 작성한 기능의 재사용하기 위해



상속과 오버라이딩

상속 방법

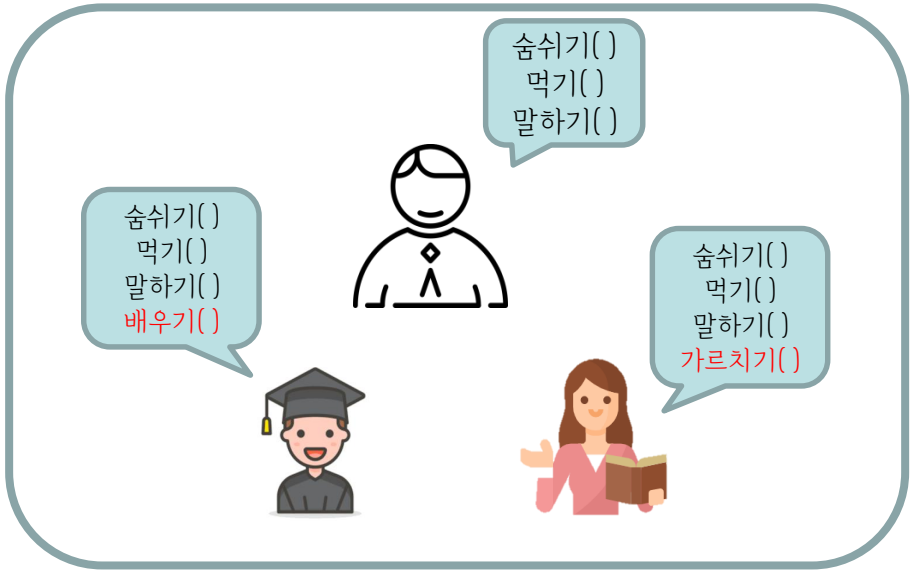
[구조]

```
class 클래스명 extends 부모클래스명{
        …
}
```

상속과 오버라이딩

<실습> Exam-40.java

사람 클래스를 생성하고, 사람의 기능을 상속받는
학생 클래스와 선생 클래스를 생성하기.



```
class Person{
    void breath() {
        System.out.println("숨 쉬기");
    }
    void eat() {
        System.out.println("밥먹기");
    }
    void say() {
        System.out.println("말하기");
    }
}
```

```
class Student extends Person{
    void learn() {
        System.out.println("배우기");
    }
}
```

```
class Teacher extends Person{
    void teach() {
        System.out.println("가르치기");
    }
}
```

상속과 오버라이딩

<실습> Exam-40.java

```
public class Inheritance1 {
    public static void main(String[] args) {
        Student s1=new Student();
        s1.breath();
        s1.learn();

        Teacher t1=new Teacher();
        t1.eat();
        t1.teach();
    }

}
```

Person 클래스에서 받아온 메서드

Person 클래스에서 받아온 메서드

<출력결과>

```
숨 쉬기
배우기
밥먹기
가르치기
```

상속과 오버라이딩

상속 시 주의할 점(p110)

- 다중상속 지원 x
- 클래스 앞 final 키드는 다른 클래스가 상속 불가

상속과 오버라이딩

2. 오버라이딩

오버라이딩?

: 자식클래스에서 부모클래스로부터 받아온 메서드를 재정의 하는 것

사용목적?

: 자식에 맞는 기능으로 맞춰 동작하기 위해

상속과 오버라이딩

<실습> Exam-41.java

리더 클래스를 생성하고 이전에 생성했던 학생 클래스를 상속받아, say( ) 메서드를 다음과 같이 재정의하기.

```
public class Inheritance2 {
    public static void main(String[] args) {
            Leader leader1= new Leader();
            leader1.eat();
            leader1.say();
    }

}
```

<출력결과>

```
밥 먹기
선생님께 인사
```

```
class Student{
    void learn() {
        System.out.println("배우기");
    }
    void eat() {
        System.out.println("밥 먹기");
    }
    void say() {
        System.out.println("선생님 안녕하세요~");
    }
}
```

```
class Leader extends Student{
    void lead() {}
    void say() {
        System.out.println("선생님께 인사"); //오버라이딩
    }
}
```

상속과 오버라이딩

<실습> Exam-41.java

리더 클래스를 생성하고 이전에 생성했던 학생 클래스를 상속받아, say( ) 메서드를 다음과 같이 재정의하기.

단, 리더클래스의 say( )에서 부모(학생)클래스의 say( ) 호출하기

부모 클래스의 필드 사용 방법?: super
자식 클래스 내 필드 사용 방법?: this

```
class Student{
    void learn() {
        System.out.println("배우기");
    }
    void eat() {
        System.out.println("밥 먹기");
    }
    void say() {
        System.out.println("선생님 안녕하세요~");
    }
}
```

```
class Leader extends Student{
    void lead() {}
    void say() {
        System.out.println("선생님께 인사"); //오버라이딩
        super.say();
    }
}
```

상속과 오버라이딩

3. 상속과 생성자

super( )

: 부모클래스의 생성자 호출

- 무조건 자식 클래스의 생성자 첫 줄에서 이뤄짐
- 작성하지 않을 시, 컴파일러가 자동 호출

상속과 오버라이딩

<실습> Exam-42.java

```
class Car2{
    int wheel;
    int speed;
    String color;

    Car2(String color){
        this.color=color;
    }
}
class SportsCar extends Car2{
    int speedLimit;
    SportsCar(String color,int speedLimit){
        this.color=color;
        this.speedLimit=speedLimit;
    }
}
```

오류 발생 이유?

자식클래스에서 생성자 호출 시,
부모클래스의 디폴트 생성자가 자동으로 호출되기 때문에
부모의 디폴트 생성자가 존재해야 한다

상속과 오버라이딩

<실습> Exam-42.java

오류 발생 이유?

자식클래스에서 생성자 호출 시,
부모클래스의 디폴트 생성자가 자동으로 호출되기 때문에
부모의 디폴트 생성자가 존재해야 한다

해결방법1

부모의 디폴트 생성자 만들기

```
class Car2{
    int wheel;
    int speed;
    String color;
    Car2(){}
    Car2(String color){
        this.color=color;
    }
}
class SportsCar extends Car2{
    int speedLimit;
    SportsCar(String color,int speedLimit){
        this.color=color;
        this.speedLimit=speedLimit;
    }
}
```

상속과 오버라이딩

<실습> Exam-42.java

오류 발생 이유?

자식클래스에서 생성자 호출 시,
부모클래스의 디폴트 생성자가 자동으로 호출되기 때문에
부모의 디폴트 생성자가 존재해야 한다

해결방법2

호출할 부모 생성자 지정하기

```
class Car2{
    int wheel;
    int speed;
    String color;
    Car2(String color){
        this.color=color;
    }
}
class SportsCar extends Car2{
    int speedLimit;
    SportsCar(String color,int speedLimit){
        super(color);
        this.speedLimit=speedLimit;
    }
}
```

상속과 오버라이딩

<실습> ObjectTest.java

```
public class ObjectTest {
    public static void main(String[] args) {
        Aclass a1 = new Aclass();
        Aclass a2 = new Aclass();

        System.out.println(a1.toString());
        System.out.println(a1.equals(a2));
        System.out.println(a1.getClass());
    }

}
class Aclass{
}
```

<출력결과>

```
day24.Aclass@28a418fc
false
class day24.Aclass
```

상속과 오버라이딩

<실습> ObjectTestOverride.java

```
public class ObjectTestOverride {
    public static void main(String[] args) {
        Bclass b1 = new Bclass();
        Bclass b2 = new Bclass();
        System.out.println(b1);
        System.out.println(b1.equals(b2));
    }
}
class Bclass{
    public String toString() {
        return "Bclass 객체";
    }
    public boolean equals(Object obj) {
        return true;
    }
}
```

<출력결과>

```
Bclass 객체
true
```