# 객체지향

클래스, 객체, 인스턴스, 인터페이스, 상속, 다형성, 추상 화

## 1. 클래스 - 가장 기본적.

```
public class Animal {

}
```

Animal cat = new Animal();

Animal 클래스의 인스턴스 cat 이 생성됨.

{객체 cat은 클래스 Animal의 인스턴스.}

- 메소드는 클래스 내에서 생성된 함수이다.
- 클래스 내에는 해당 클래스 고유의 객체 변수가 존재한다.

```
public class Animal {
    String name;
}
```

- 클래스 내의 메소드가 해당 클래스의 객체 변수에 접근할때 "this.name" 이런식으로 this를 사용할 수 있다.

## 2. 상속.

```
public class Animal {
    String name;

    public void setName(String name) {
        this.name = name;
    }
}
```

이런 Animal 클래스가 있다고 가정하자.

```
public class Dog extends Animal {

}
```

Animal 클래스를 Dog로 상속한다.

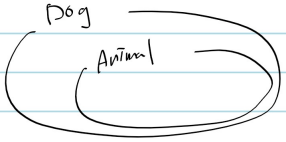
∴ Dog.setName("puppy") 가 가능해진다. (비록 Dog 에 setName 메소드를 설정하지는 않았지만, Animal을 상속받아 사용하기에 사용이 가능한 것)

# Dog is a Animal 성립.

∴ Animal abc = new Dog(); 이 가능함.

Dog abc = new Animal(); (X)

☆ 사용가능한 메소드의 종류로 변수에 값을 표시하면 다음과 같다.




Dog > Animal.

∴ Dog로 생성한 객체는 Animal이 되기위한 충분조건.

∴ Animal abc = new Dog(); 이 가능함.

반면 Animal로 생성한 객체는 Dog 자료형의 객체가 되기 부족하다. (필요조건)

∴ Dog abc = new Animal(); 이 불가능함

## 메소드 오버 라이딩 (메소드 덮어쓰기)

```
public class HouseDog extends Dog {
    public void sleep() {
        System.out.println(this.name+" zzz in house");
    }

    public static void main(String[] args) {
        HouseDog houseDog = new HouseDog();
        houseDog.setName("happy");
        houseDog.sleep();
    }
}
```

Dog 메소드에 "zzz"를 출력하는 sleep 메소드가 이미 구현됨! 상속에서

Dog 클래스를 상속받은 HouseDog 클래스에, 부모 메소드의 이름과같은 sleep 메소드를 새로 정의한 것.

→ 이때 자식클래스의 메소드가 우선 순위가 더 높다.

∴ "zzz in house" 가 출력된다.

## 메소드 오버로딩 (같은 이름, 다른 매개 변수)

```
public class HouseDog extends Dog {
    public void sleep() {//메소드 오버라이딩
        System.out.println(this.name+" zzz in house");
    }

    public void sleep(int hour) {//메소드 오버로딩
        System.out.println(this.name+" zzz in house for " + hour + " ho
urs");
    }
```

둘 다 사용 가능하다!!

☆자바는 다중 상속을 지원하지 않는다!!