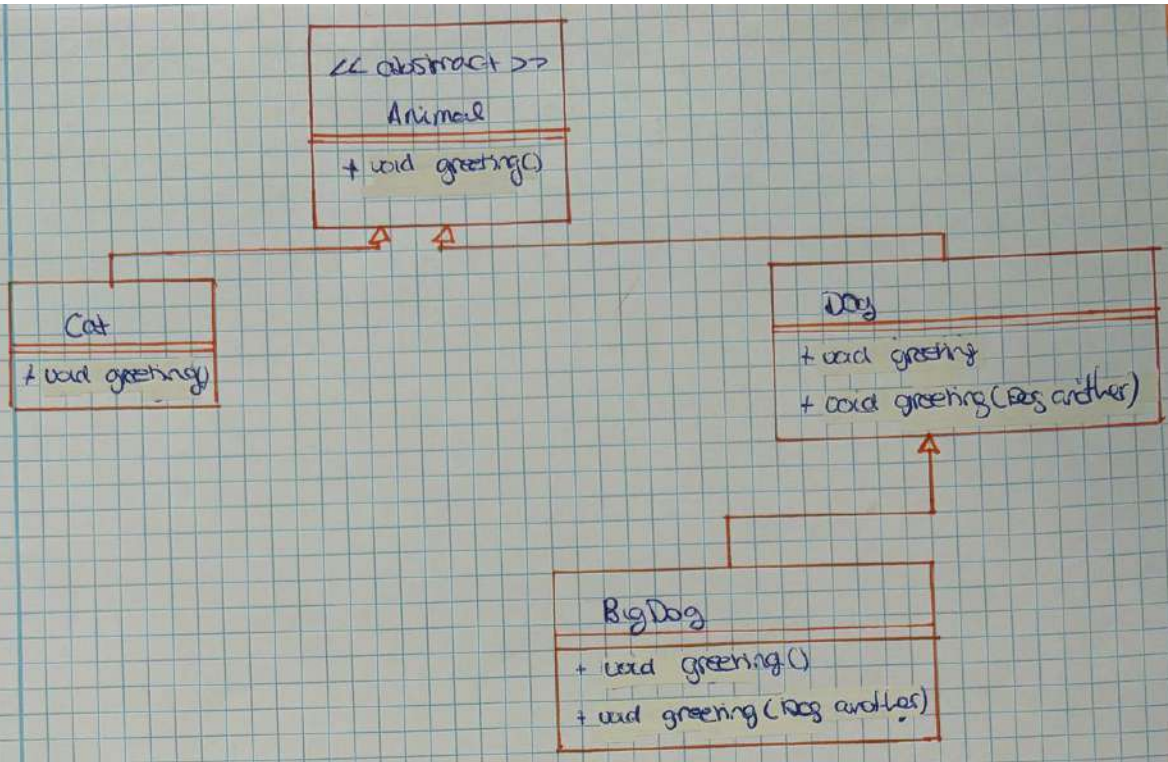


```
Cat cat1 = new Cat();
cat1.greeting();
Dog dog1 = new Dog();
dog1.greeting();
BigDog bigDog1 = new BigDog();
bigDog1.greeting();
```

Meow!
Woof!
Wooow!

Se crea un objeto y se llama al método greeting que imprime el sonido del animal.

```
Animal animal1 = new Cat();
animal1.greeting();
Animal animal2 = new Dog();
animal2.greeting();
Animal animal3 = new BigDog();
animal3.greeting();
Animal animal4 = new Animal();
```

Meow!
Woof!
Wooow!
Animal es abstracto no se instancia de ERROR

animal 1,2,3 es de tipo Animal, apunta a una clase.
Como llama a greeting() e importante. Cada llamada a su versión greeting().

```
Dog dog2 = (Dog)animal2;
BigDog bigDog2 = (BigDog)animal3;
Dog dog3 = (Dog)animal3;
Cat cat2 = (Cat)animal2;
```

animal 2 y 3 eran Dog y BigDog por lo que el "downcasting" funciona
Animal 2 apunta a Dog y un Dog no puede convertirse en Cat de ERROR

```
dog2.greeting(dog3);}
dog3.greeting(dog2);}
dog2.greeting(bigDog2);}
bigDog2.greeting(dog2);}
bigDog2.greeting(bigDog1);}
```

dog y dog ⇒ function Dog(anotherDog) ⇒ "woooof"

dog y dog ⇒ function Dog(anotherDog) ⇒ "woooof"

dog y bigdog (es un dog) ⇒ Dog(anotherDog) ⇒ "woooof"

BigDog, sobreescribe greeting(anotherDog) y da "WOOOOOW"

BigDog y BigDog ⇒ Bigdog.greeting(Dog another) ⇒ "WOOOOOW"