Inheritance

Inheritance is when a class inherits behavior from another class. The class that is inheriting behavior is called the subclass and the class it inherits from is called the superclass. We use inheritance as a way to extract common behaviors from classes that share that behavior, and move it to a superclass. This lets us keep logic in one place.

```
class Animal
  def speak
    "Hi I am an Animal"
  end
end

class Dog < Animal
end

class Cat < Animal
end

dog = Dog.new
cat = Cat.new
```

```
puts dog.speak

Hi I am an Animal

=> nil


puts cat.speak

Hi I am an Animal

=> nil
```

We use the < symbol to signify that the Dog and Cat class is inheriting from the Animal class.

Now let's say we wanted to override the behavior and use the method from the subclass.

```
class Animal

  def speak

    "Hi I am an Animal"

  end

End


class Dog < Animal

  def speak

    "Hi I am an Dog"
  end
end

class Cat < Animal
End

dog = Dog.new
cat = Cat.new
puts dog.speak
Hi I am an Dog
=> nil
puts cat.speak
Hi I am an Animal
=> nil
```

In the Dog class, we're overriding the speak method in the Animal class because Ruby checks the object's class first for the method before it looks in the superclass. So, that means when we wrote the code dog.speak, it first looked at dog's class, which is Dog. It found the speak method there and used it. When we wrote the code cat.speak, Ruby first looked at cat's class, which is Cat. It did not find a speak method there, so it continued to look in Cat's superclass, Animal. It found a speak method in Animal, and used it. This process is what called method lookup.