Class Design

Inheritance

Inheritance

- let's say we have class Dog which extends class Animal

```
public class Animal { }

public class Dog extends Animal { }
```

- here, Animal is referred to as **superclass (parent)**, and Dog as **subclass (child)**
- subclass can **inherit** members (fields and methods) from a superclass
  - this is called **inheritance**

Inheritance is transitive

```
public class Animal { }
public class Mammal extends Animal { }
public class Dog extends Mammal { }
```

- Dog inherits from `Animal`, but only through `Mammal`
- Java supports **single inheritance**
  - class can have only one direct superclass
  - (unlike in some other languages, like C++)
- but class can implement multiple interfaces

Class modifiers

- `final`
- `abstract`
- `static`
- `sealed`
- `non-sealed`

Java 17

Object class

- all Java classes implicitly inherit from `java.lang.Object` class
- `Object` is the only class which doesn't have a parent class

```
public class Dog { }

public class Dog extends java.lang.Object { }
```

- every class has access to methods defined in `Object` class
  - e.g. `toString()`, `equals()`, `hashCode()`, etc.