CYDEO

Inheritance

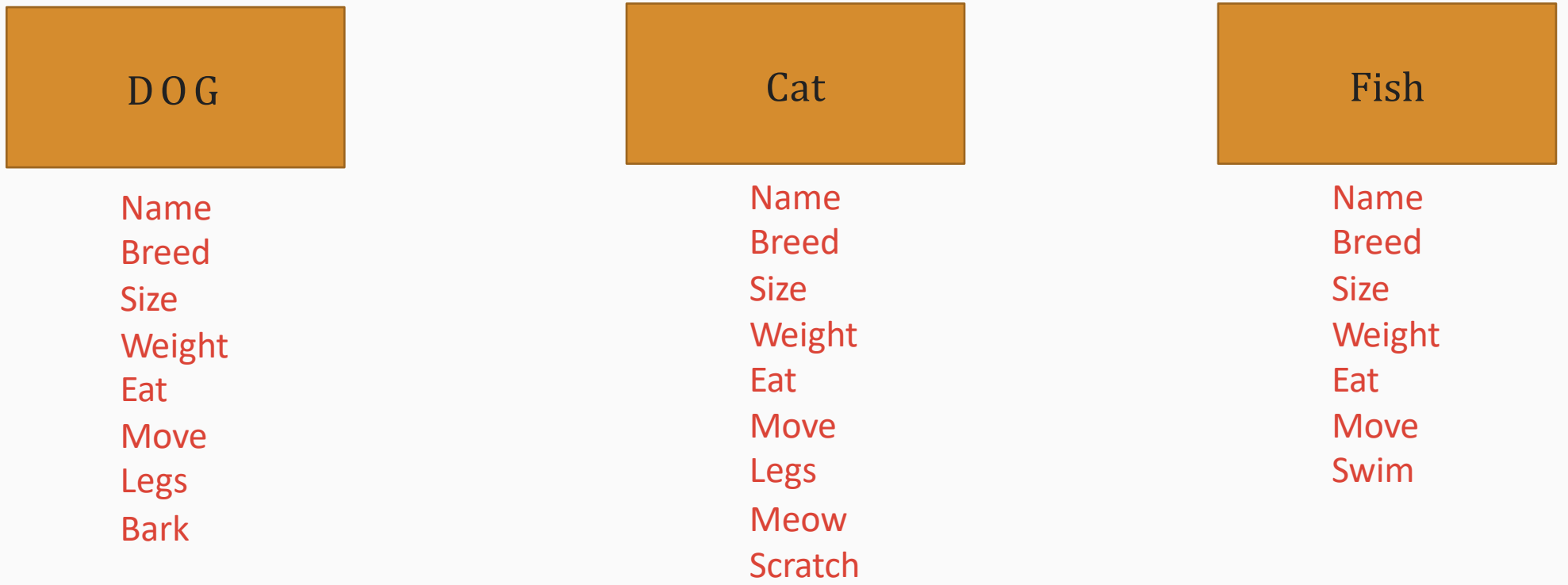
OOP Principles

- There are 4 Object Oriented Programming (OOP) principles:
  - Encapsulation
  - Inheritance
  - Abstraction
  - Polymorphism

Inheritance (Is A relation)

- Used for creating an “is a” relationship among the classes
- When an “is-a” relationship exists between objects, it means that the specialized object has all of the characteristics of the general object.
- It allows one class to inherit the features (variables & methods) of an another class

Inheritance



Inheritance

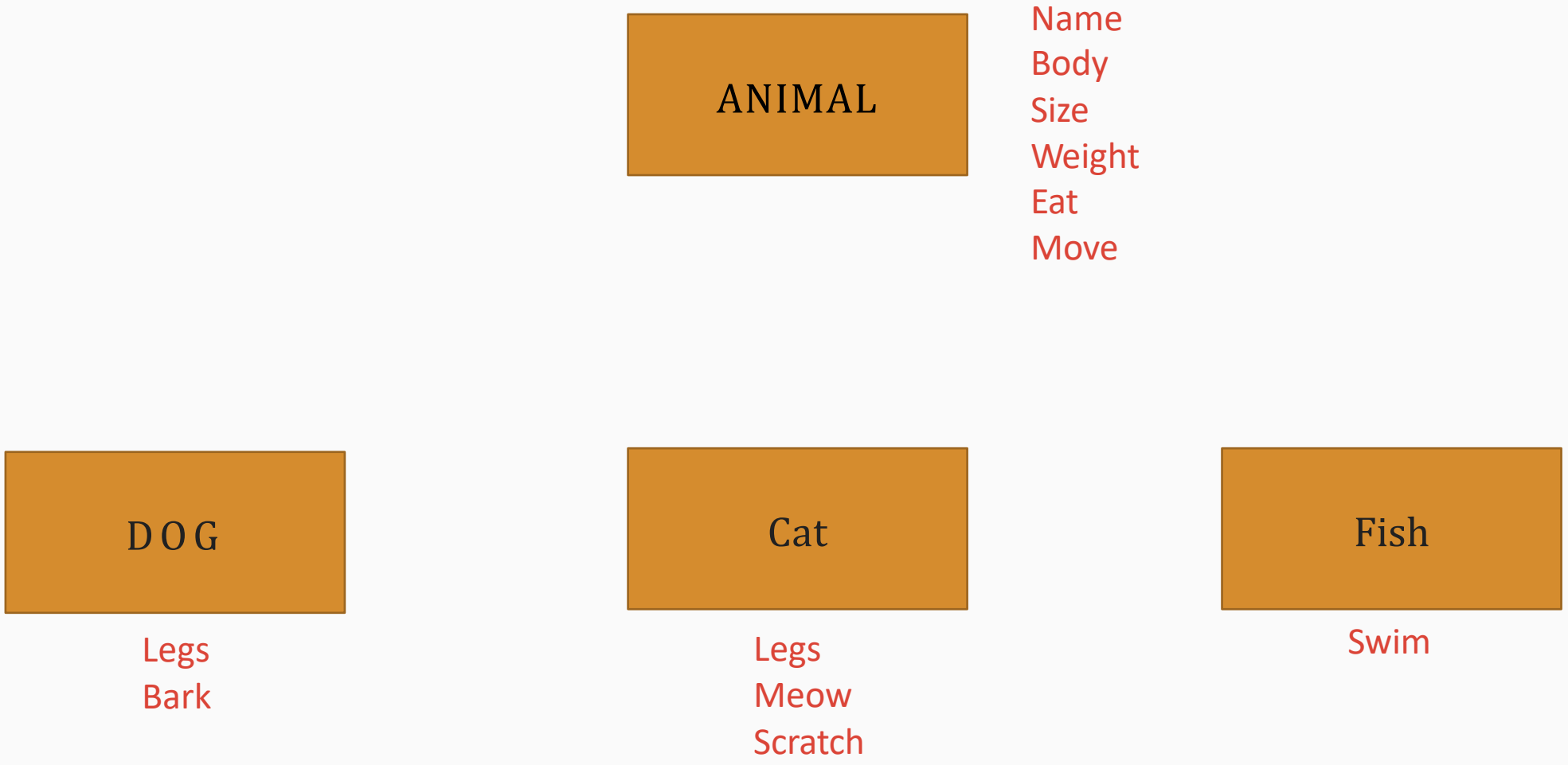
All animals have certain characteristics.

In addition to the common animal characteristics, the dog has its own unique characteristics.

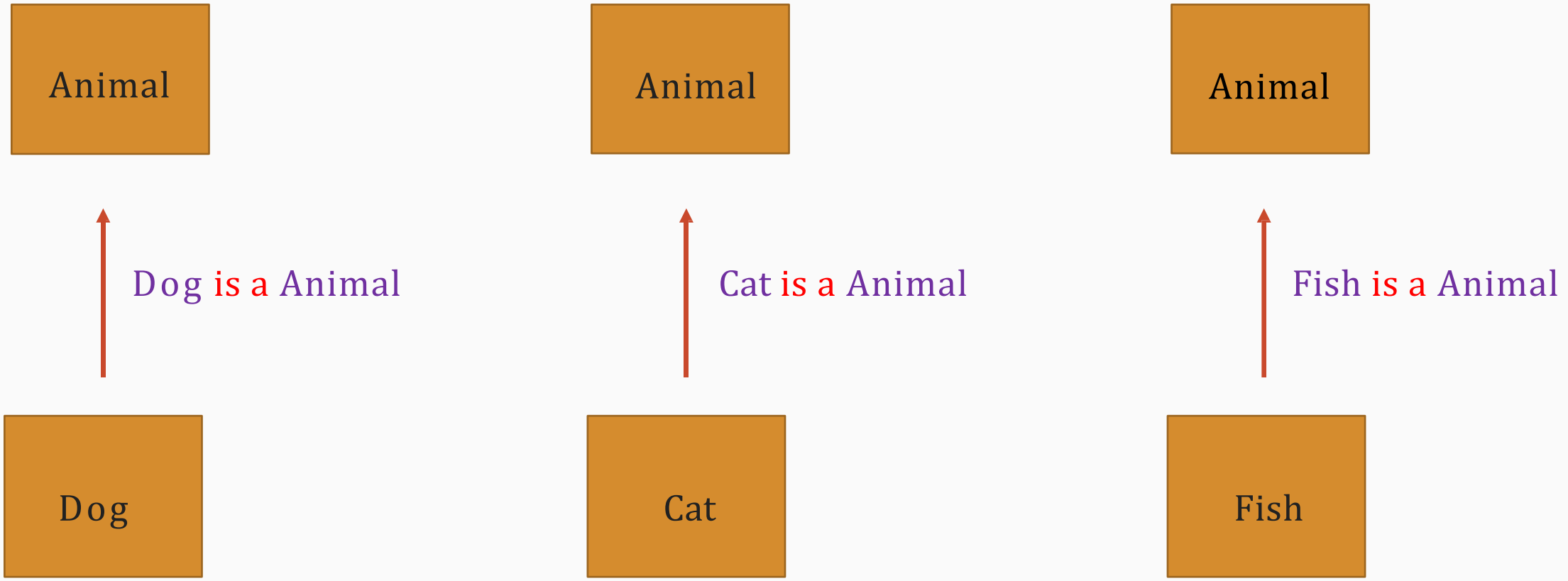
In addition to the common animal characteristics, the fish has its own unique characteristics.

In addition to the common animal characteristics, the bird has its own unique characteristics.

Inheritance

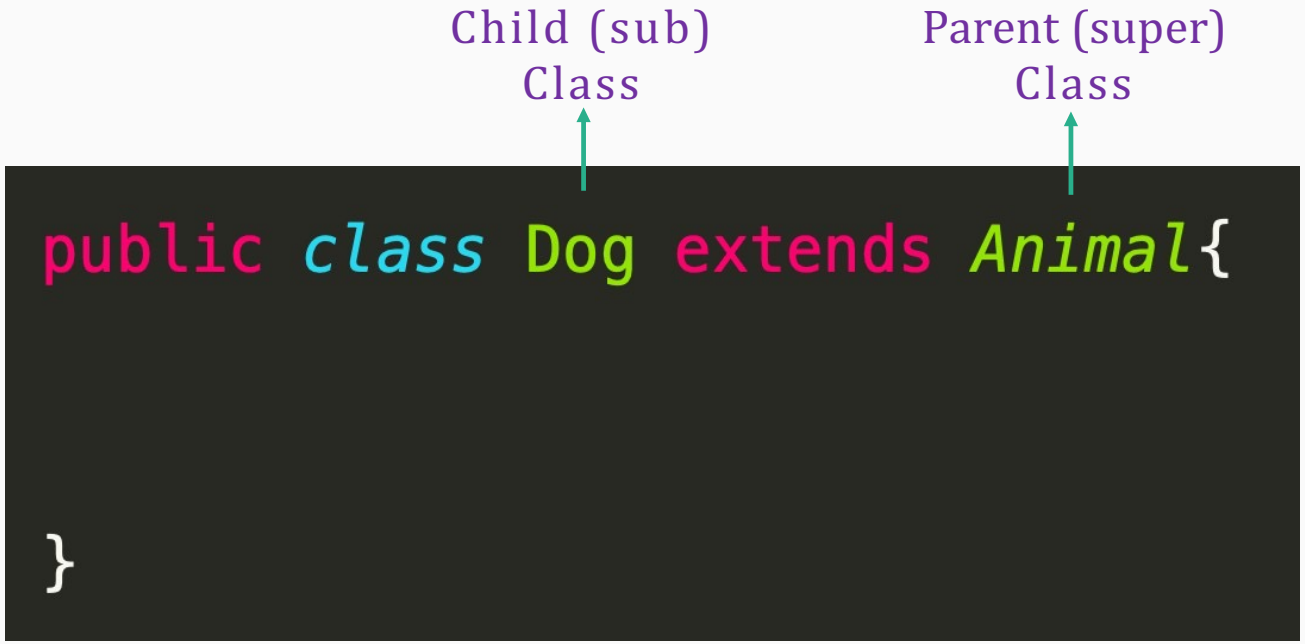


Inheritance

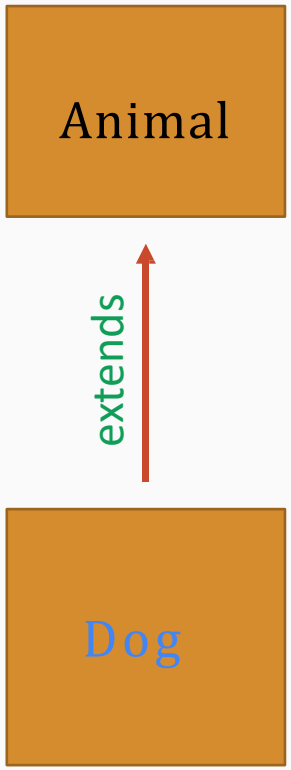


Inheritance

- The keyword used for inheritance is extends



Inheritance



Animal is called SUPER class and Dog is called SUB class
OR
Animal is called PARENT class and Dog is called CHILD class

Super & Sub Classes (Is A relation)

- Super Class: The class whose features are inherited is known as super class (or a base class or a parent class)
- Sub Class: The class that inherits the other class is known as sub class(derived class, or child class). The subclass can add its own fields and methods in addition to the superclass fields and methods.

What is inherited to sub class?

- All the accessible variables & methods (depending on the access modifier)
- Private variables and methods are not inherited.
- Constructors are not inherited, but child class must call one of parent class' constructor

Calling the Super Class Constructor

- If parent class has default constructor, sub class calls it implicitly, otherwise super class' constructor need to be called explicitly in the sub class
- The super keyword refers to an object's super class. We can use the super key word to call a superclass constructor

Super keyword

- super() is used to call Parent class constructor from Child class constructor
- If parent class has default (No-Argument) constructor, compiler will put super() automatically
- If parent class only has constructor with parameters, then child constructor MUST make a matching super(params) call

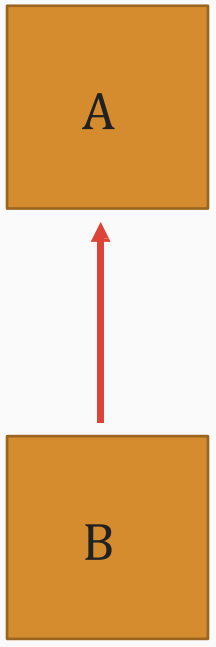
Super Keyword Example

```java
public class Animal{

    public String name;
    public String breed;
    public String size;

    public Animal(String name, String breed, String size){
        this.name = name;
        this.breed = breed;
        this.size = size;
    }

    public void eat(){
        System.out.println(name+" is eating");
    }

}
```

```java
public class Dog extends Animal{

    public Dog(String name, String breed, String size){
        super(name, breed, size);

    }


}
```

Types of Inheritance

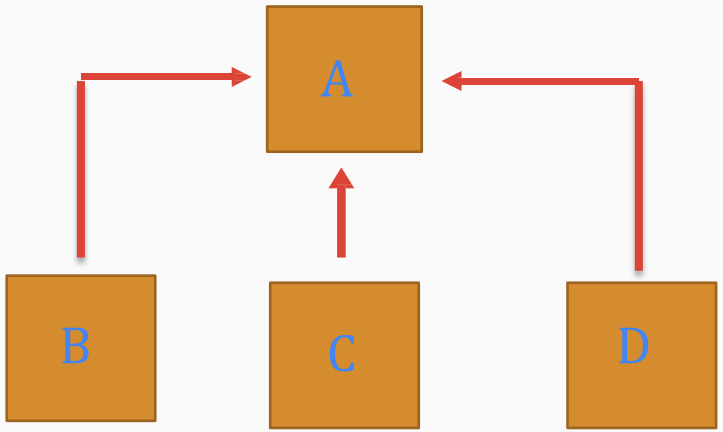
- Single Inheritance: Sub classes inherit the features of one super class




```
public class A{

}

public class B extends A{

}
```

Types of Inheritance

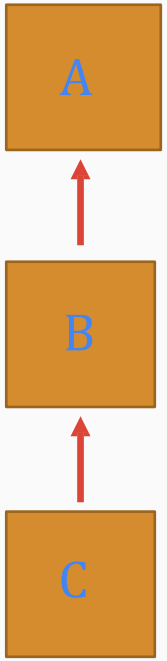
- Hierarchical Inheritance: Once class serves as superclass for more than one sub class




```
public class A{

}

public class B extends A{

}

public class C extends A{

}

public class D extends A{

}
```

Types of Inheritance

- Multi Level Inheritance: Subclass will be inheriting a Super Class and as well as the subclass also act as the Super Class to the other class




```
public class A{

}

public class B extends A{

}

public class C extends B{

}
```

Types of Inheritance

- Multiple Inheritance: Java DOES NOT support multiple inheritance with classes. One class can not have more than one superclass and inherit features from all parent class