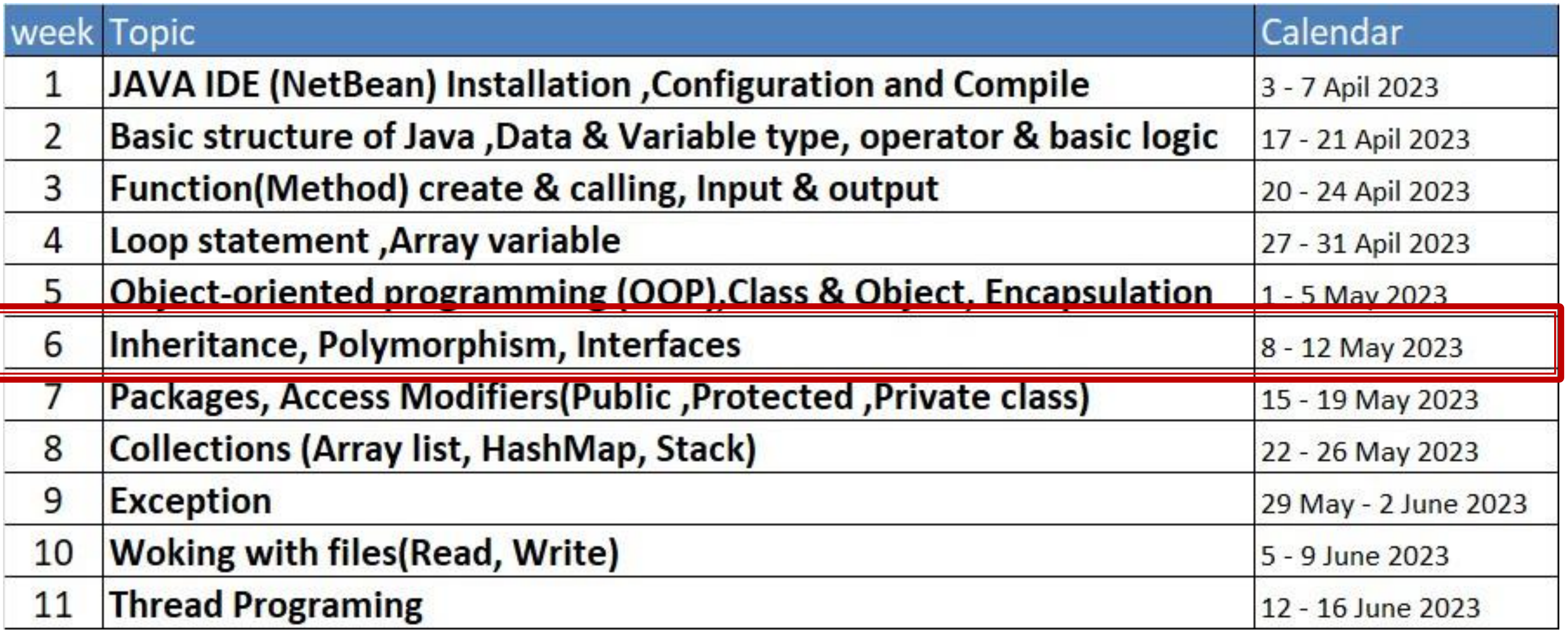
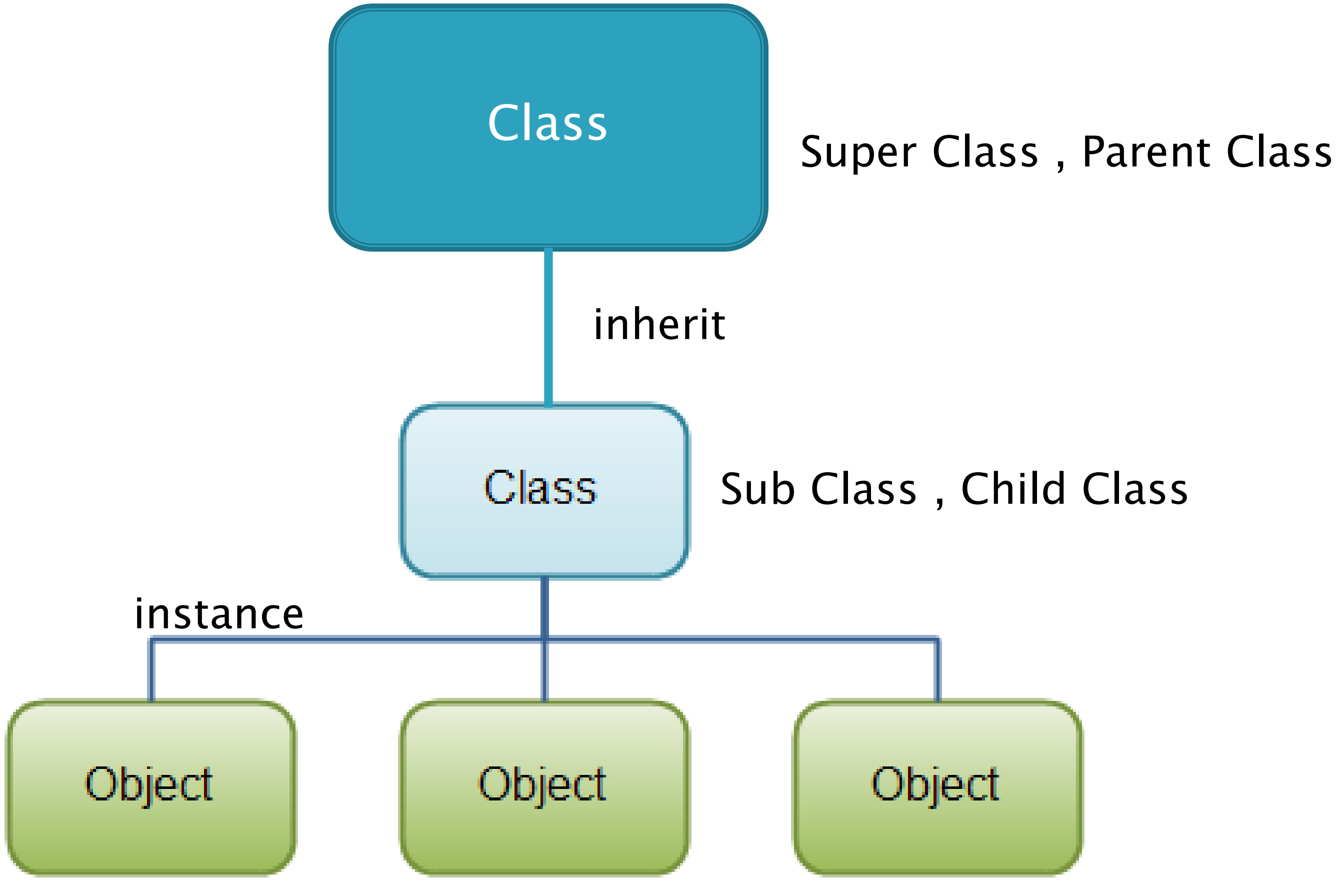
Introduction to Java week#6

20/06/2023



week	Topic	Calendar
1	JAVA IDE (NetBean) Installation ,Configuration and Compile	3 - 7 Apil 2023
2	Basic structure of Java ,Data & Variable type, operator & basic logic	17 - 21 Apil 2023
3	Function(Method) create & calling, Input & output	20 - 24 Apil 2023
4	Loop statement ,Array variable	27 - 31 Apil 2023
5	Object-oriented programming (OOP),Class & Object, Encapsulation	1 - 5 May 2023
6	Inheritance, Polymorphism, Interfaces	8 - 12 May 2023
7	Packages, Access Modifiers(Public ,Protected ,Private class)	15 - 19 May 2023
8	Collections (Array list, HashMap, Stack)	22 - 26 May 2023
9	Exception	29 May - 2 June 2023
10	Woking with files(Read, Write)	5 - 9 June 2023
11	Thread Programing	12 - 16 June 2023

Object-Oriented Programing(OOP)



Object-Oriented Programing(OOP)

4 pillars of object-oriented programming

1. Encapsulation:

containing information in an object, exposing only selected information

2. Inheritance:

child classes inherit data and behaviors from the parent class

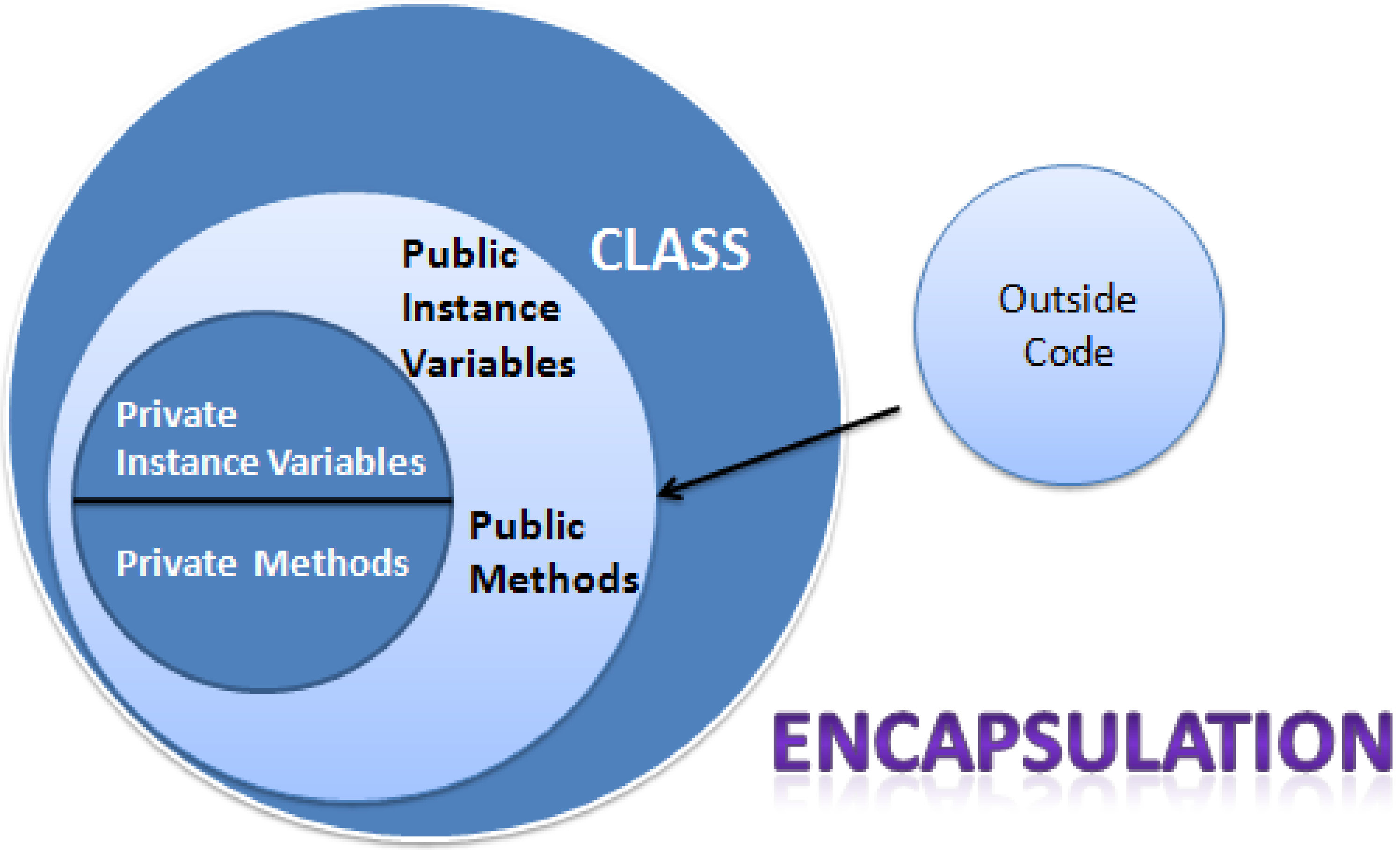
3. Abstraction:

only exposing high-level public methods for accessing an object

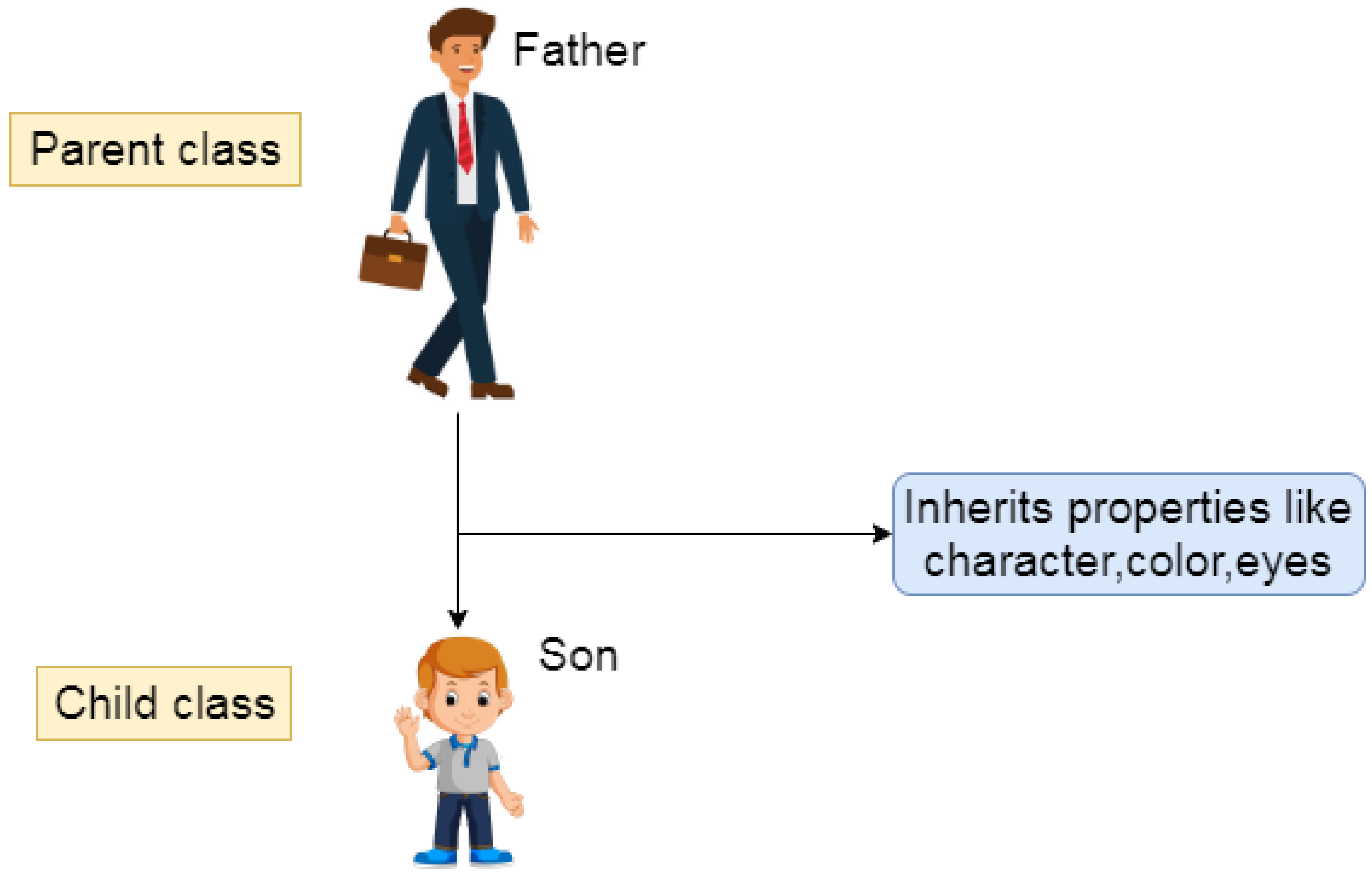
4. Polymorphism:

many methods can do the same task

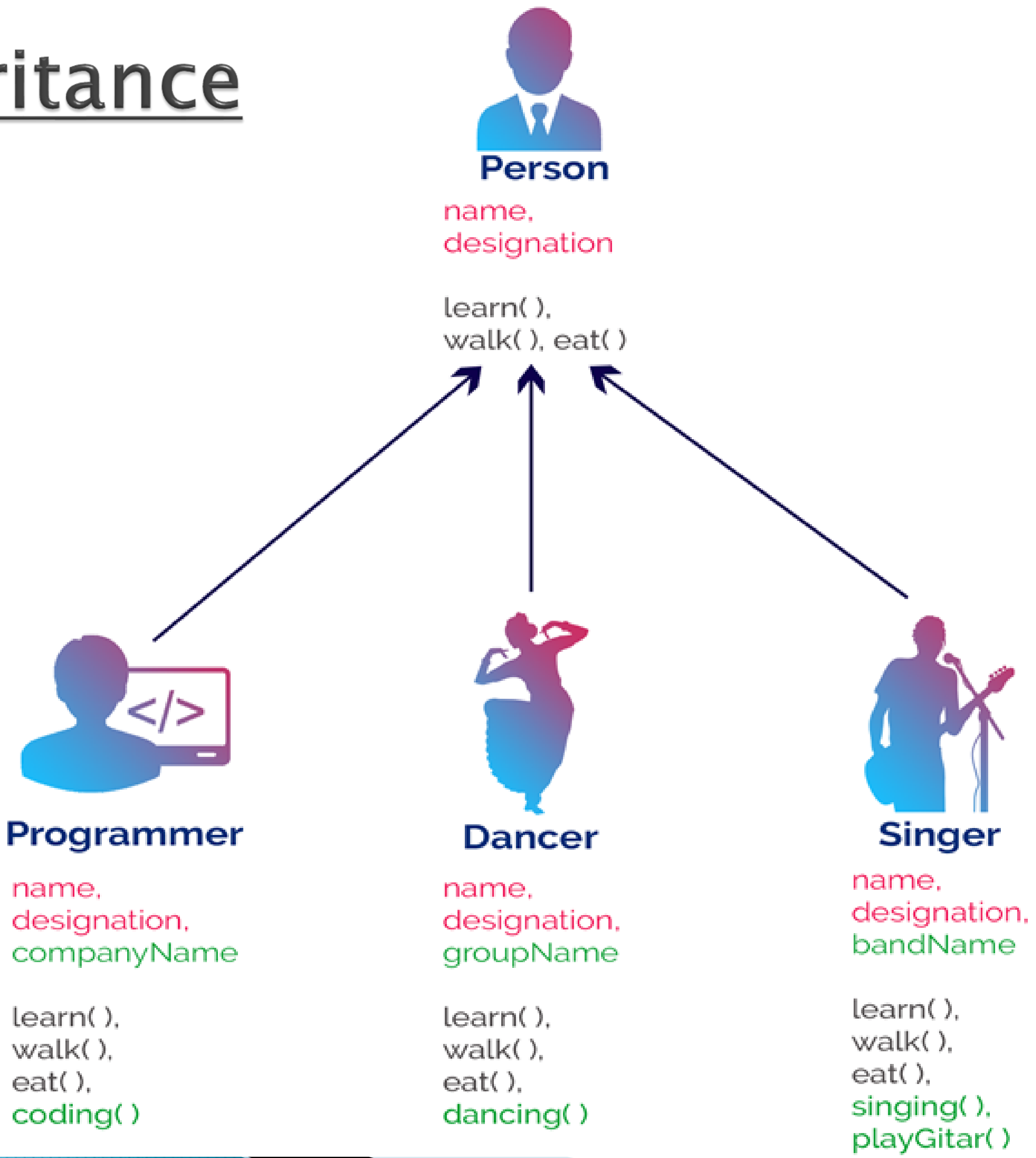
Encapsulation



Inheritance



Inheritance



Inheritance

```
class ClassName extends SuperClass
{


      ......


}
```

Inheritance

```
class Person {
    String name;
    int age;

    public Person() { }

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public void introduce() {
        System.out.println("My name is " + name);
    }
}
```

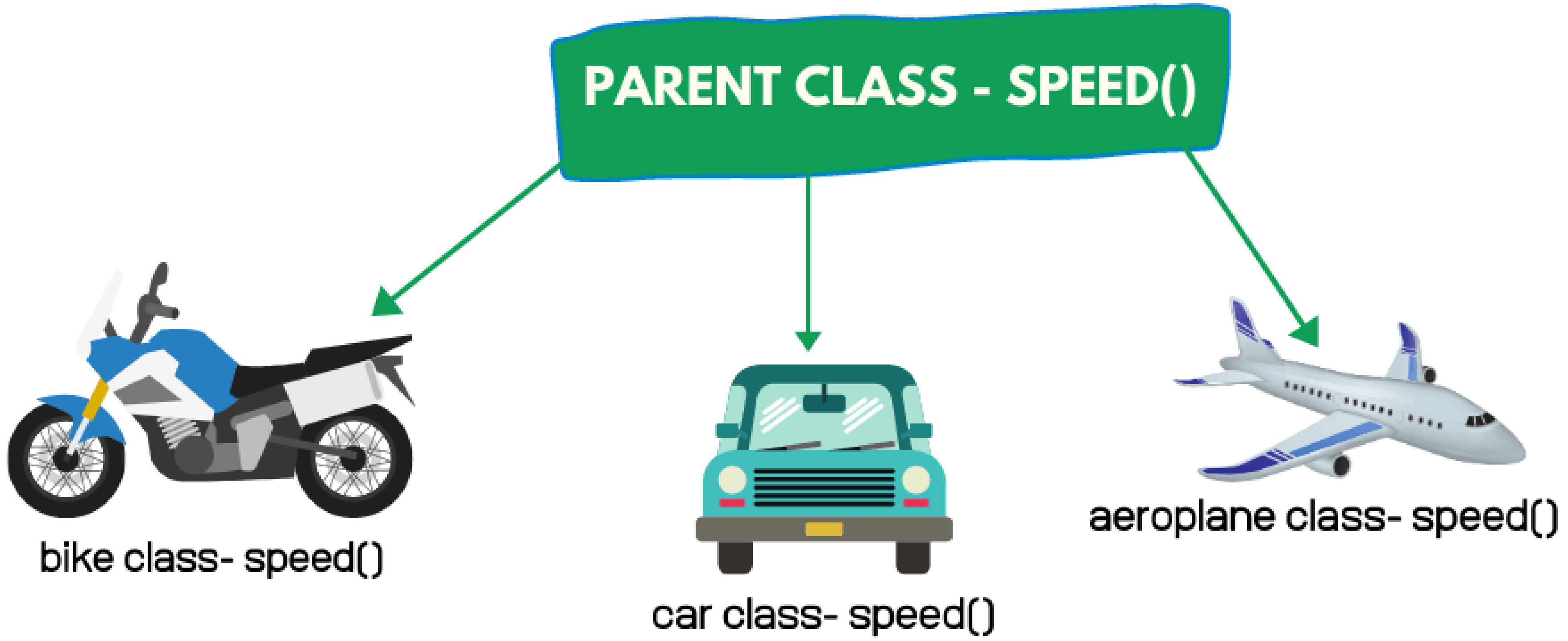
Inheritance

```
class Artist extends Person {
    String genre;
    public Artist(String name, int age){
        this.name = name;
        this.age = age;
    }
    public void playMusic() {
        System.out.println(name + " is playing " + genre + " music.");
    }
}

class Athlete extends Person {
        String sport;
        public Athlete(String name, int age){
                this.name = name;
                this.age = age;
        }
        public void playSport() {
                System.out.println(name + " is playing " + sport + ".");
        }
}
```

Inheritance

```
public static void main (String[] args) {

    Artist art = new Artist("Marcus", 20);
    Athlete ath = new Athlete("Danny", 25);

    art.genre = “Pop";
    ath.sport = "Football";

    art.introduce();
    art.playMusic();

    System.out.println();

    ath.introduce();
    ath.playSport();

}
```

Overriding Super Class

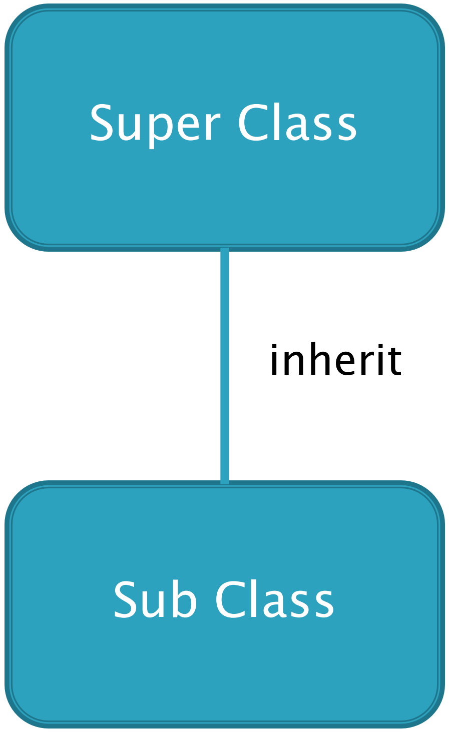


Overriding Super Class

```java
class Artist extends Person {
    String genre;
    public Artist(String name, int age){
        this.name = name;
        this.age = age;
    }

    @Override
    public void introduce() {
        System.out.println("My name is " + name);
        System.out.println("I'm an artist.");
        System.out.println("I'm " + age + " years old." );
    }

    public void playMusic() {
        System.out.println(name + " is playing " + genre + " music.");
    }
}
```

This & Super

Super Class , Parent Class



Sub Class , Child Class

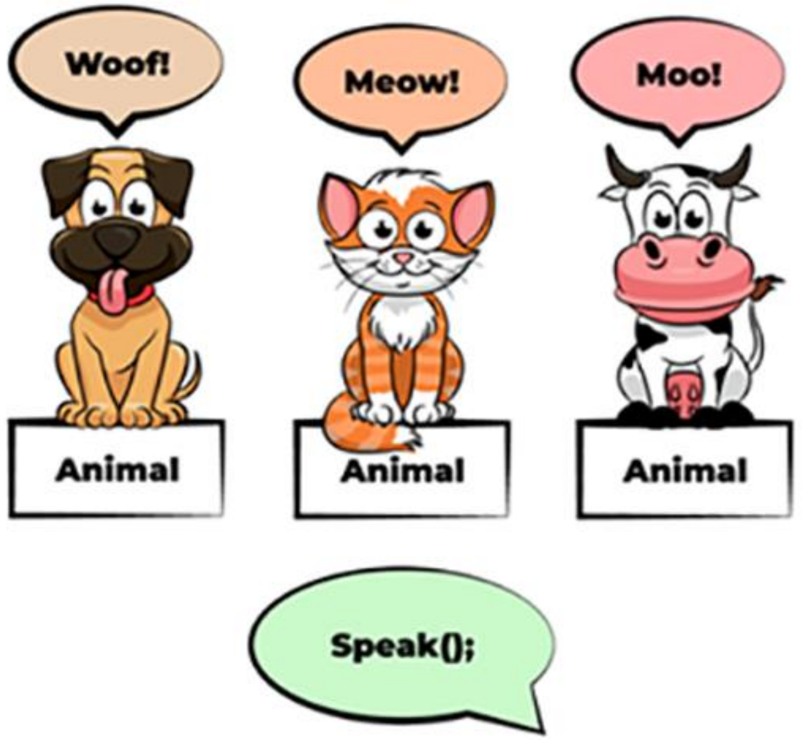
```
class Person {
    public Person()
    public Person(String name, int age)
    public void introduce()
```

```
class Artist extends Person {
    super()
    super(name, age)
    super.introduce()
```

This & Super

```
class Artist extends Person {
    String genre;
    public Artist (String name, int age){
        super(name, age);
    }
    @Override public void introduce () {
        super.introduce();
        System.out.println("I'm an artist.");
        System.out.println("I'm " + age + " years old." );
    }
    public void playMusic () {
        System.out.println(name + " is playing " + genre + " music.");
    }
}
```

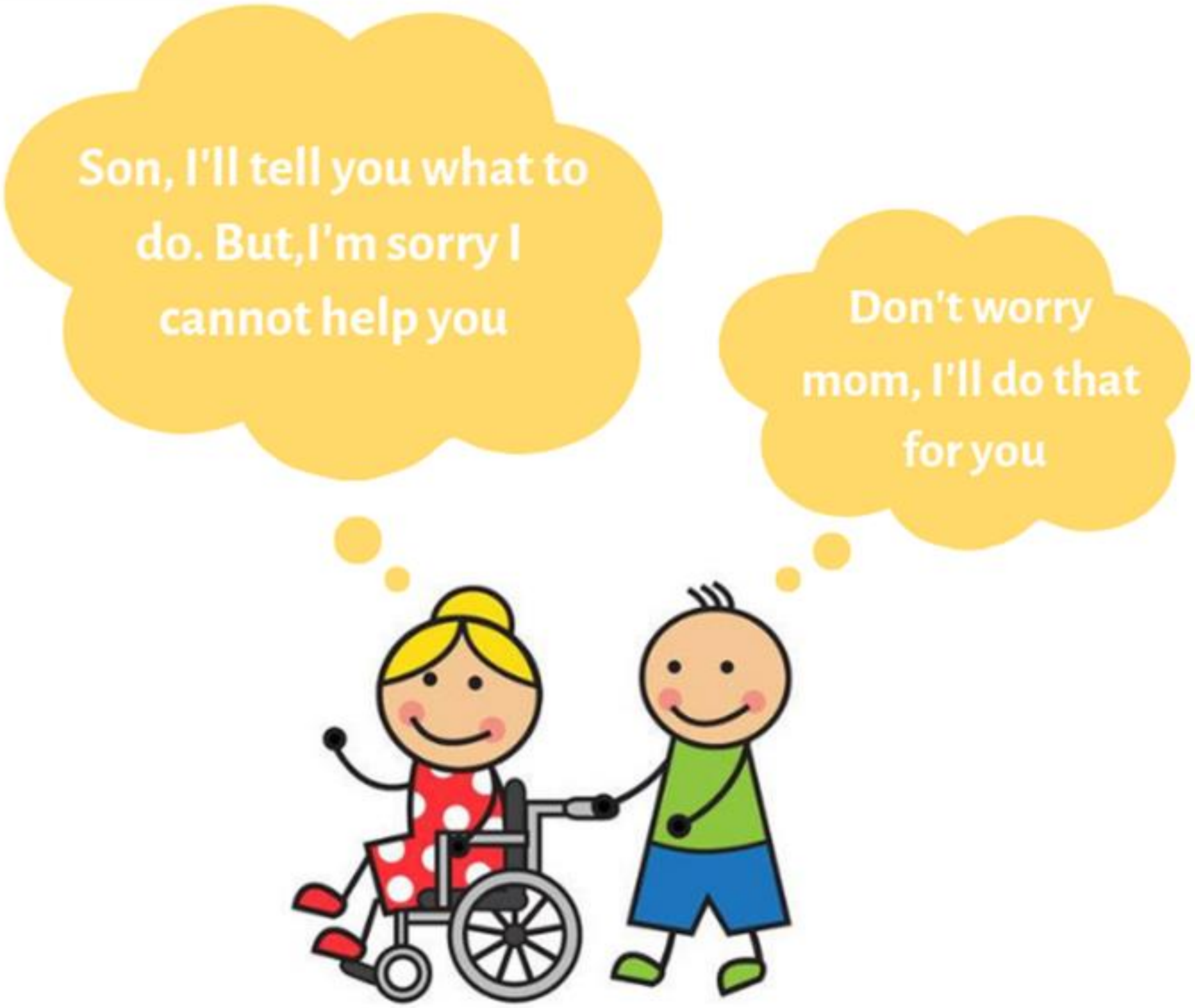
Polymorphism



Polymorphism

```java
public static void main (String[] args) {

   Person person1, person2, person3;

   person1 = new Person("Mark", 30);
   person2 = new Artist("Mateo", 19);
   person3 = new Athlete("Danny", 16);

   person1.introduce();
   person2.introduce();
   person3.introduce();

}
```

Interface


Son, I'll tell you what to do. But, I'm sorry I cannot help you
Don't worry mom, I'll do that for you

Interface

```
interface InterfaceName {

      ...

}
```

```
class CarName implements Ineterface1, Interface2, ... {
        ...
}

class CarName extends SuperClass implements Ineterface1, Interface2, ...
{
        ...
}
```

Interface

```
interface IVolume{

    public void increaseVolume();
    public void decreaseVolume();
}

interface IChannel {

    public void nextChannel();
    public void previousChannel();
}

interface INetwork {

    public void connectNetwork();
    public void disconnectNetwork();
}
```

Interface

```
class Radio implements IVolume, IChannel {

    public void increaseVolume() {
        System.out.println("volume up");
    }

    public void decreaseVolume() {
        System.out.println("volume down");
    }

    public void nextChannel() {
        System.out.println("next channel");
    }

    public void previousChannel() {
        System.out.println("previous channel");
    }
}
```

Interface

```java
class Computer implements IVolume, INetwork {

    public void increaseVolume() {
        System.out.println("volume up");
    }

    public void decreaseVolume() {
        System.out.println("volume down");
    }

    public void connectNetwork() {
        System.out.println("connected to network");
    }

    public void disconnectNetwork() {
        System.out.println("disconnected from network");
    }
}
```

Assignments

คลาส สำหรับ “คน” จาก assignment ที่แล้วมาสร้าง subclass เป็น class ของ

- Spiderman
- Ironman
- {up to you}

และทำการ สร้างฮีโร่จากคลาสที่สร้างเป็นของ 3 multiverse

Assignmentsที่แล้ว

ออกแบบ และ สร้างคลาส สำหรับ “คน” ให้มีคุณสมบัติ ชื่อ, อายุ , เพศ, อาชีพ และ มีเมธอด แนะนำตัว, กิน, นอน, เดิน

และทำการ สร้างออปเจ็คจากคลาสที่สร้างเป็น คน 3 คนที่แตกต่างกัน

Thank you