继承：

继承关系的理解

- 继承指的是类与类之间的关系
    - PS:
        - 人类
            - 学生类
                - 大学生类
                - 中学生类
    - 上级类是下级类的父类 SuperClass
    - 一个类只能有一个直接的父类，但是可以有多个直系父类
    - 一个父类可以有多个子类
    - Java中所有的类都默认Object 为父类
- 语法： extends
    - 子类 extends 父类名
    - PS: public class UNStudent extends Student{ }

继承了什么

- 子类到底继承了父类的什么?
    - 类的具体的结构：
        - 面向对象成员部分：
            - 属性： 继承
            - 方法： 继承
            - 构造方法： 连带调用
                - 子类创建对象时除了调用自己本身的构造方法，也会调用父类的构造方法
            - 构造代码块： 连带调用 除了自己的代码块 也会调用父类的代码块
        - 静态资源部分：static
            - 静态属性: 属于类的，在内存只有一份，子类可以与父类共用
            - 静态方法：属于类的 可以使用类名直接调用 子类名可以调用父类的静态方法
            - 静态代码块： 所属于加载时，静态代码块只执行一次

```java
package com.oop.zyf1020;

public class A {
    String name;

    {
        System.out.println("A类的代码块");
    }

    public A() {
        System.out.println("A类的构造方法");
    }

    public void testA() {
        System.out.println("A类的testA方法");
    }

    // 静态资源
    static int count;

    static {
        System.out.println("A类的静态代码块");
    }

    public static void staticMethod() {
        System.out.println("A类的静态方法");
    }
}

class B extends A {
    String pwd;

    {
        System.out.println("B类的代码块");
    }

    public B() {
        System.out.println("B类的构造方法");
    }

    public void testB() {
        System.out.println("B类的testB方法");
    }
}

class C extends B {
    int id;

    {
        System.out.println("C类的代码块");
```

```
    }

    public C() {
        System.out.println("C类的构造方法");
    }

    public void showInfo() {
        System.out.println("姓名: " + name);
        System.out.println("密码: " + pwd);
        System.out.println("学号: " + id);
    }
}

class Main {
    public static void main(String[] args) {
        C c = new C();
        c.name = "张三";
        c.pwd = "123";
        c.id = 1;
        c.showInfo();
        c.testA();
        c.testB();


        // 子类名调用父类的静态属性
        A.count = 20;
        C.count = 100;
        System.out.println(A.count);
        System.out.println(C.count);
        C.staticMethod();
        B.staticMethod();
        A.staticMethod();

    }
}
```

继承的用处

代码复用:

- 可以通过继承父类，避免一些代码的重复编写
- 比如，有多个子类它们的属性 和方法有一部分是完全相同的，此时我们可以采用创建一个父类
- 将相同的属性和方法都定义在父类中，子类继承父类，就不用再重复定义这些属性和方法了

- PS: 西游记PK游戏：
    - 孙悟空 猪八戒 唐僧 角色
    - 属性： 名字 攻击力 防御力 生命值 等级
    - 方法： 初始化 升级 攻击
        - 不同的方法： 各自有各自的技能
- PS: 学生类别：
    - 学生类：
        - 大学生类
        - 中学生类

继承的宗旨

- 继承一个类，如果不新增任何属性或者方法，或者不对父类的内容进行扩展 升级 改造，那么就不要继承
- 改造扩展1：新增属性 和方法
- 改造扩展2：重写父类的方法并修改方法的内容
- 方法的重写：子类通过重写父类的方法 改造/升级/重构 父类方法的内容和逻辑
    - 格式：
        - 子类中可以将父类中定义的方法进行重写一次
        - 并且要求方法的结构与父类方法保持一致
        - 重写时，方法体可以修改
    - 子类重写的方法会被子类优先调用，而不是父类方法
    - PS:
        - 动物类： 吃
            - 野猪
            - 狮子
            - 老虎： 吃 一周三顿
            - 兔子： 吃 一天吃三顿
- 如何确定重写的格式正确：
    - @Override// 重写的注解 检查重写的方法是否格式正确

类型转换：

- 子类创建的对象到底有几个类型呢?
    - UNStudent stuA = new UNStudent();
        - stuA.name="美佳";
        - 美佳是不是大学生? 是的 美佳属于大学生类
        - 美佳是不是学生? 是的 美佳属于学生类
        - 美佳是不是人类? 是的 美佳属于人类
- 自动转型:向上类型转型
    - 将子类对象赋给一个父类类型对象变量名 叫做自动转型 (向上转型)
- 强制转型:向下类型转型
    - UNStudent unstu1 = new UNStudent();
    - Student stu1 = unstu1;
    - UNStudent unstu2 = (UNStudent)stu1;// 强制转型
    - 先向上转型后 才可以向下转型 必须转为原本的类型 否则会抛出异常

```java
package com.oop.zyf1020;

public class Student {
    // 属性：  大学生 中学生 都有的属性
    String name;
    int age;
    String className;
    String stuID;

    public void read(String str) {
        System.out.println("学生正在阅读：" + str);
    }

    public void exam() {
        System.out.println("学生正在考试");
    }

    public void showInfo() {
        System.out.println("---学生个人信息：");
        System.out.println("姓名：" + name);
        System.out.println("年龄：" + age);
        System.out.println("班级：" + className);
        System.out.println("学号：" + stuID);
    }
}

class UNStudent extends Student {
    // 属性：  只属于大学生类有的属性
    int CET_4;
    int CET_6;

    public void exam() {
        System.out.println("大学生考试");
    }

    @Override// 重写的注解 检查重写的方法是否格式正确
    public void showInfo() {
        super.showInfo();// 调用父类原本的方法执行一次
        System.out.println("大学生独有的信息：");
        System.out.println("CET_4：" + CET_4);
        System.out.println("CET_6：" + CET_6);
    }


}

class MIDStudent extends Student {
    // 属性：  只属于中学生类有的属性 科目成绩
    int math;
    int chinese;
```

```
    int english;


}

class MainA {
    public static void main(String[] args) {
        UNStudent unStu1 = new UNStudent();
        MIDStudent midStu1 = new MIDStudent();

        // 继承的父类的属性
        unStu1.name = "张三";
        unStu1.age = 18;
        unStu1.className = "计算机科学与技术";
        unStu1.stuID = "2019001";
        // 自己独有的属性
        unStu1.CET_4 = 500;
        unStu1.CET_6 = 600;


        // 继承的父类的属性
        midStu1.name = "李四";
        midStu1.age = 17;
        midStu1.className = "高三3班";
        midStu1.stuID = "2019002";
        // 自己独有的属性
        midStu1.math = 90;
        midStu1.chinese = 80;
        midStu1.english = 70;

        // 子类继承父类的方法并且子类对象可以直接调用
        unStu1.read("《Java从入门到精通》");
        midStu1.read("《五年高考三年模拟》");

        // 方法重写
        unStu1.exam();// 调用的是 子类中重写的方法
        midStu1.exam();// 调用的是 继承的父类中的方法

        // 方法重写时。完整的保留父类方法的调用 增加自己的代码 - super
        // super 在子类指代父类 使用 super调用父类的属性 以及方法执行
        unStu1.showInfo();


        // 类型关系:
        // 将子类对象赋给一个父类类型对象变量名  叫做自动转型 （向上转型）
        UNStudent us1 = unStu1;
        Student s = unStu1;
        Object o1 = unStu1;


        Student s1 = unStu1;
```

```
        Student s2 = midStu1;

        // 有一个活动方法：  需要两名学生参加
        // 张三 和 李四能不能一起参加这个活动
        readActivity(s1);
        readActivity(s2);

        // 活动2：  需要一名中学生参加 只支持中学生类型对象入参
        midStuReadActivity(midStu1);

        // PS: 我们在存储对象时，也可以使用父类作为数组的类型声明，那么这个数组可以存储元素类型就包含所有子类对象
        Student[] list = new Student[2];
        list[0] = unStu1;
        list[1] = midStu1;

    }

    // 学生阅读活动
    // 使用父类类型作为参数类型时，可以在调用方法时使用子类对象入参 （自动转型）
    public static void readActivity(Student s1) {
        s1.read("《数学之美》");
    }

    public static void midStuReadActivity(MIDStudent s1) {
        s1.read("《五年高考三年模拟》");
    }
}
```

练习： 简单的完成这个继承的样例 - PK游戏：

- 定义一个父类： 角色类
    - 属性： 名字 攻击力 防御力 生命值 等级
    - 方法： 显示基本信息 攻击其他角色
- 定义好所有的子类： 孙悟空 猪八戒 唐僧 等
    - 子类中定义子类独有属性和方法 ：子类独有的属性 初始化 升级
    - 重写父类的方法，比如显示的信息 增加子类独有的属性输出