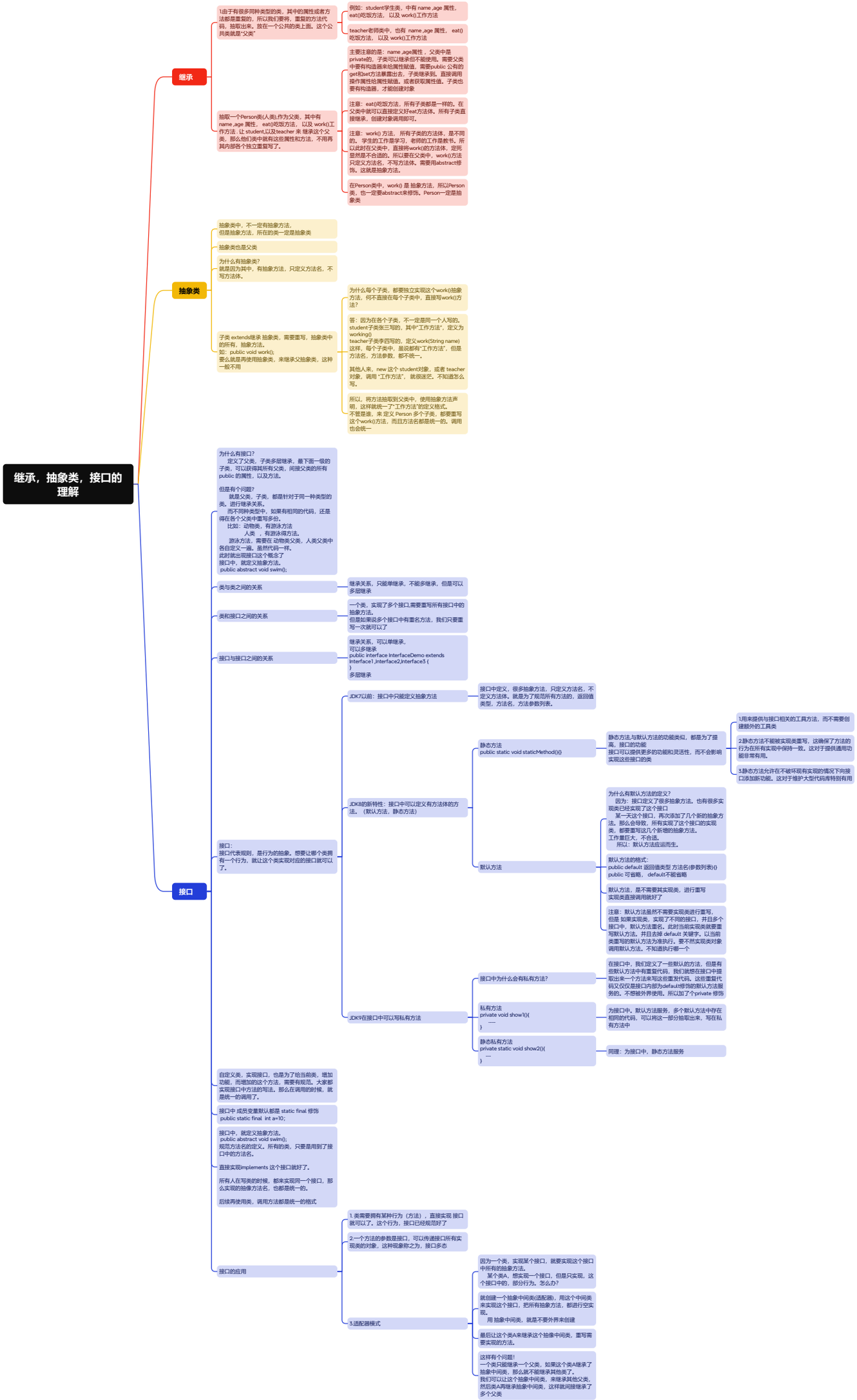
继承，抽象类，接口的理解
继承
1.由于有很多同种类型的类，其中的属性或者方法都是重复的。所以我们要将，重复的方法代码，抽取出来。放在一个公共的类上面。这个公共类就是"父类"
例如：student学生类，中有 name ,age 属性，eat()吃饭方法，以及 work()工作方法
teacher老师类中，也有 name ,age 属性， eat()吃饭方法， 以及 work()工作方法
抽取一个Person类(人类),作为父类，其中有 name ,age 属性，eat()吃饭方法，以及 work()工作方法 .让 student,以及teacher 来 继承这个父类，那么他们类中就有这些属性和方法，不用再其内部各个独立重复写了。
主要注意的是：name ,age属性 ，父类中是private的，子类可以继承但不能使用。需要父类中要有构造器来给属性赋值，需要public 公有的get和set方法暴露出去，子类继承到。直接调用操作属性给属性赋值。或者获取属性值。子类也要有构造器，才能创建对象
注意：eat()吃饭方法，所有子类都是一样的。在父类中就可以直接定义好eat方法体。所有子类直接继承，创建对象调用即可。
注意：work() 方法， 所有子类的方法体，是不同的。 学生的工作是学习，老师的工作是教书。所以此时在父类中，直接将work()的方法体，定死显然是不合适的。所以要在父类中，work()方法只定义方法名，不写方法体。需要用abstract修饰。这就是抽象方法。
在Person类中，work() 是 抽象方法，所以Person类，也一定要abstract来修饰。Person一定是抽象类
抽象类
抽象类中，不一定有抽象方法，
但是抽象方法，所在的类一定是抽象类
抽象类也是父类
为什么有抽象类?
就是因为其中，有抽象方法，只定义方法名，不写方法体。
子类 extends继承 抽象类，需要重写，抽象类中的所有，抽象方法。
如：public void work();
要么就是再使用抽象类，来继承父抽象类，这种一般不用
为什么每个子类，都要独立实现这个work()抽象方法，何不直接在每个子类中，直接写work()方法?
答：因为在各个子类，不一定是同一个人写的。
student子类张三写的，其中"工作方法"，定义为working()
teacher子类李四写的，定义work(String name)
这样，每个子类中，虽说都有"工作方法"，但是方法名，方法参数，都不统一。
其他人来，new 这个 student对象，或者 teacher对象，调用 "工作方法"， 就很迷茫。不知道怎么写。
所以，将方法抽取到父类中，使用抽象方法声明，这样就统一了"工作方法"的定义格式。
不管是谁，来 定义 Person 多个子类，都要重写这个work()方法，而且方法名都是统一的。调用也会统一
接口
为什么有接口?
定义了父类，子类多层继承，最下面一级的子类，可以获得其所有父类，间接父类的所有public 的属性，以及方法。
但是有个问题?
就是父类，子类，都是针对于同一种类型的类。进行继承关系。
而不同种类型中，如果有相同的代码，还是得在各个父类中重写多份。
比如：动物类，有游泳方法
人类 ，有游泳得方法。
游泳方法，需要在 动物类父类，人类父类中各自定义一遍。虽然代码一样。
此时就出现接口这个概念了
接口中，就定义抽象方法。
public abstract void swim();
类与类之间的关系
继承关系，只能单继承，不能多继承，但是可以多层继承
类和接口之间的关系
一个类，实现了多个接口,需要重写所有接口中的抽象方法。
但是如果说多个接口中有重名方法，我们只要重写一次就可以了
接口与接口之间的关系
继承关系，可以单继承，
可以多继承
public interface InterfaceDemo extends Interface1 ,Interface2,Interface3 {
}
多层继承
接口:
接口代表规则，是行为的抽象。想要让哪个类拥有一个行为，就让这个类实现对应的接口就可以了。
JDK7以前：接口中只能定义抽象方法
接口中定义，很多抽象方法，只定义方法名，不定义方法体。就是为了规范所有方法的，返回值类型，方法名，方法参数列表。
JDK8的新特性：接口中可以定义有方法体的方法。（默认方法，静态方法）
静态方法
public static void staticMethod(){}
静态方法与默认方法的功能类似，都是为了提高，接口的功能
接口可以提供更多的功能和灵活性，而不会影响实现这些接口的类
1.用来提供与接口相关的工具方法，而不需要创建额外的工具类
2.静态方法不能被实现类重写，这确保了方法的行为在所有实现中保持一致。这对于提供通用功能非常有用。
3.静态方法允许在不破坏现有实现的情况下向接口添加新功能。这对于维护大型代码库特别有用
默认方法
为什么有默认方法的定义?
因为：接口定义了很多抽象方法。也有很多实现类已经实现了这个接口
某一天这个接口，再次添加了几个新的抽象方法。那么会导致，所有实现了这个接口的实现类，都要重写这几个新增的抽象方法。
工作量巨大，不合适。
所以：默认方法应运而生。
默认方法的格式:
public default 返回值类型 方法名(参数列表){}
public 可省略， default不能省略
默认方法，是不需要其实现类，进行重写
实现类直接调用就好了
注意：默认方法虽然不需要实现类进行重写，但是 如果实现类，实现了不同的接口，并且多个接口中，默认方法重名，此时当前实现类就要重写默认方法，并且去掉 default 关键字。以当前类重写的默认方法为准执行。要不然实现类对象调用默认方法。不知道执行哪一个
JDK9在接口中可以写私有方法
接口中为什么会有私有方法?
在接口中，我们定义了一些默认的方法，但是有些默认方法中有重复代码，我们就想在接口中提取出来一个方法来写这些重复代码。这些重复代码又仅仅是接口内部为default修饰的默认方法服务的。不想被外界使用。所以加了个private 修饰
私有方法
private void show1(){
——
}
为接口中，默认方法服务，多个默认方法中存在相同的代码，可以将这一部分抽取出来，写在私有方法中
静态私有方法
private static void show2(){
——
}
同理：为接口中，静态方法服务
自定义类，实现接口，也是为了给当前类，增加功能，而增加的这个方法，需要有规范。大家都实现接口中方法的写法。那么在调用的时候，就是统一的调用了。
接口中 成员变量默认都是 static final 修饰
public static final int a=10;
接口中，就定义抽象方法。
public abstract void swim();
规范方法名的定义。所有的类，只要是用到了接口中的方法名。
直接实现implements 这个接口就好了。
所有人在写类的时候，都来实现同一个接口，那么实现的抽像方法名，也都是统一的。
后续再使用类，调用方法都是统一的格式
接口的应用
1.类需要拥有某种行为（方法），直接实现 接口就可以了。这个行为，接口已经规范好了
2.一个方法的参数是接口，可以传递接口所有实现类的对象，这种现象称之为，接口多态
3.适配器模式
因为一个类，实现某个接口，就要实现这个接口中所有的抽象方法。
某个类A，想实现一个接口，但是只实现，这个接口中的，部分行为。怎么办?
就创建一个抽象中间类(适配器)，用这个中间类来实现这个接口，把所有抽象方法，都进行空实现。
用 抽象中间类，就是不要外界来创建
最后让这个类A来继承这个抽像中间类，重写需要实现的方法。
这样有个问题!
一个类只能继承一个父类，如果这个类A继承了抽象中间类，那么就不能继承其他类了。
我们可以让这个抽象中间类，来继承其他父类，然后类A再继承抽象中间类，这样就间接继承了多个父类