Java中继承的学习

1. 继承（inheritence）：**Java是单继承的**，意味着从一个类只能从另一个类中继承。被继承的类叫做父类或者基类，继承的类叫做子类。Java中的继承用关键字extends。
2. 单继承：一个类只能有一个直接的父类；多继承：一个类可以有多个直接的父类。
3. C++ 是多继承的，java是单继承的，java是对C++的改良。单继承有很多好处的。多继承会产生不确定性，如果两个父类中有名字相同的方法，此时调用此方法时，不知道调用从哪个父类继承下来的方法。
4. **Java支持多层（多重）继承**。Java是通过“多实现“的方式进行对c++的多继承性质改良的。
5. **当使用一个继承体系时：**
6. **查看该体系的顶层类，了解该体系的基本功能；**
7. **创建该体系的最子类对象，完成功能的使用。**

**（这是一种学习方法）**

1. 什么时候使用继承？（不能胡乱继承）

当类与类之间存在所属关系时，定义继承。Xxx 是 Yyy的一种，则 Xxx extends Yyy

1. 在父子类中，成员主要是三个：成员变量，成员方法，构造方法。
2. Java中紧紧支持单继承。
3. 继承的关键词是extends ,因为不仅是继承还会对其进行拓展。继承中有种拓展的含义。
4. 继承的作用：减少了程序的重复代码。
5. 在定义一个类的时候，如果没有显示地指定该类的父类，那么该类就会继承于java.lang.Object类，这是JDK提供的一个类，Object类是java中所有类的直接或者间接父类。
6. 一般情况下，类中的属性都会定义为私有的private；私有的属性只能在本类中进行访问，所以一般是向外界提供两个非私有的方法setXxx 和 getXxx进行访问私有属性。同样的，如果父类中的属性是私有的，子类其实也是继承了，但是既然是父类已经私有了，子类就不能直接访问，子类要访问也只能通过继承的父类的setXXX和 getXxx方法进行访问。**子类不能直接访问父类中的私有内容。**
7. this 代表的是一个本类对象的引用；但是super代表的只是一个父类空间（父类加载进来的所属空间），因为可能此时并没有创建一个父类对象；或者说继承的是父类，不是继承的父类的看某个对象。
8. 在生成子类对象时，首先调用父类的构造函数，然后生成父类对象，（因为没有父亲就没有孩子），进而调用子类构造方法，生成子类对象。在调用父类构造方法时，默认情况下，是调用父类的不带参数的构造方法。但是，可以修改这种父类的构造方法的调用，利用关键字super（），super表示对父类构造方法的显式调用，调用与super（）一致的构造方法，不再调用不带参数的构造方法了；小括号的参数与所要调用的构造方法的参数一致。（要想生成子类对象，必须先生成父类对象，没有父类对象就没有子类对象）。继承会生成多个对象。（这句话我觉得不对）
9. Super和this一样，必须放在构造方法的第一行，即其之前不能有其他可以执行的代码。注意：super是放在子类的构造方法的第一行的，控制调用父类的哪个构造方法，目的是为了保证先生成父类对象，再生成子类对象。
10. 关于继承的三点结论：① 父类有的（属性），子类也有；②父类有的，子类可以改变；③父类没有的，子类可以增加。
11. 关于继承的注意事项：
12. 构造方法不可以被继承；
13. 方法和属性可以被继承；
14. 子类的构造方法隐式地调用父类的不带参数的构造方法；
15. 子类可以利用super显示地调用父类的构造方法，super指的是对父类构造方法的调用；
16. super关键字必须是子类构造方法的第一行；
17. **继承的好处：（1）提高了代码的复用性；（2）使类与类之间产生了关系，为第三个特征多态提供了前提。**
18. **为什么父类要优先于子类加载呢？**
19. **因为子类需要使用父类中的变量和方法，如果父类不进来，子类就用不了，所以需要父类先进来。**
20. **子父类中成员变量的特点：子类继承父类，子类创建对象时，子类的对象会把父类的成员变量也封装进来。如果子类和父类有同名同类型的变量，此时子类对象会对这两个同名变量同时封装，会有标志子类和父类的，利用this和super进行区分，不过这种情况很少见，（不过面试的时候经常见到），因为既然子类继承父类，就是说明有共性，那么子类就没有必要再定义一个自己变量，直接继承下来用就可以了呗。（一般情况下不需要考虑这种情况）注意变量是这种情况，而方法是存储在方法区中，考虑的方式不一样。只考虑变量即可。**
21. **子父类中成员方法特点：方法存在于方法区中。子类对象在调用方法时，先在子类加载进来的方法中进行查找，找不到后，再到父类加载区中寻找。**
22. **继承中的构造方法：**

**在子类创建对象时，会先执行父类的构造方法，为什么呢？**

**答：在子类的构造方法中的第一行默认情况下有一个隐式语句super（）；调用父类中空参数的构造方法。**

1. **构造方法不会发生重写，因为构造方法的方法名始终与所在类的类名一致，所以不会出现子父类的构造方法重名现象，更不可能发生重写了。**
2. **构造方法也不会被继承，而是利用super（\*\*\*\*\*）；进行调用父类的构造方法。**
3. **子类的实例化过程：子类中的所有的构造方法在默认情况下，都会访问父类中的空参数的构造方法。**
4. **为什么子类实例化过程中要调用父类的构造方法？**

**因为子类继承了父类中的内容（属性），在使用父类内容之前，要先看看父类是如何对自己的内容进行初始化的。所以子类实例化过程中（创建对象的过程中）必须访问父类的构造函数。因此默认情况下，在子类的构造方法的第一行添加了super（）；语句。如果父类中没有空参数的构造方法，必须在子类的构造方法中第一行显式地声明相应的构造方法。注意：super语句必须定义在子类构造方法中的第一行，因为父类的初始化动作要先完成。**

1. **在子类实例化过程中，根据子类的关键词extends，会先加载父类，再加载子类。**
2. **继承的弊端：例如一个类继承了底层系统的类，然后复写其中的方法，就会导致底层某些功能不能使用了，解决措施是让这些类或者方法或者属性禁止重写，即利用final进行修饰即可。**