Java中的abstract学习

1. 抽象类：
2. **抽象类不能够生成对象**；所以抽象类就是用来被继承的，否则就没有意义了，（活着就是当爹的），所以抽象类不能再用final 进行修饰了；final和abstract 不能共同修饰一个类；
3. 抽象方法只能存在于抽象类中，如果一个类中有抽象方法，那么此类必须为抽象类；但是抽象类中可以没有抽象方法（即只有具体方法，目的是为了禁止生成此类的对象，没有抽象方法反而定义为抽象类就是为了让其他类继承，禁止生成对象的），可以既有具体方法又有抽象方法，也可以只有抽象方法；
4. **抽象类中允许有构造方法**。为什么？虽然抽象类不能生成对象，貌似没有构造方法的必要了，但是在抽象类的子类生成对象的时候，调用子类的构造方法时必须调用父类（即抽象类）的构造方法，所以抽象类也得有构造方法，**抽象类的构造方法是为了子类生成对象而准备的。**
5. 为什么要使用抽象类？
6. 一个类继承一个抽象类，抽象类中抽象方法一定要被重写（覆盖），否则出错；
7. （1） 抽象方法只有声明，没有方法体，不能带大括号。

（2） 抽象方法必须定义在抽象类中，也就是如果一个类包括了至少一个抽象方法，则该类必须是抽象类。

1. 抽象类（abstract class）：使用了abstract关键字修饰的类称为抽象类。抽象类的特点：抽象类不能够实例化，即不能过new出一个对象或者实例。
2. 抽象方法：抽象方法必须定义在抽象类中。**只有声明，没有实现。**
3. 如果一个类中包含了抽象方法，则该类一定是抽象类，即该类必须利用abstract声明；一个抽象类中可以包含抽象类也可以包含具体类（既有声明又有实现）也可以只包含具体类或者只包含抽象类。
4. 无论在任何情况下，只要是抽象类，无论是具体方法还是抽象方法，都是无法实例化的。
5. 如果一个父类是抽象类，那么子类继承父类时，必须实现父类的所有抽象方法，否则必须定义为抽象类abstract class。
6. 抽象类为什么不能实例化？

因为调用抽象方法没有意义。

1. 一个类要想实现一个抽象类，就必须实现其中所有的抽象方法。否则，无法实现该抽象类，也就无法实例化，仍然是一个抽象类。
2. Question1 抽象类中有构造方法吗？

答：有。首先明确构造方法的作用，由学习继承可以知道，构造方法不仅可以给自己实例化初始化，还可以给其子类的实例化初始化。抽象类本身不可以实例化，但是实现它的子类需要实例化，抽象类为了给子类实例化时初始化，所以自己也包括了抽象方法。**故抽象类中也存在构造方法。**

1. Question2 抽象关键字abstract不可以和哪些关键字共存？

答：

1. **abstract与private不能共存。**因为abstract修饰目的就是被实现的，然而private是把此方法隐藏了，子类无法实现它了。
2. **abstract与static不能共存。**Static修饰的目的是直接利用类名进行访问的，然而方法再用abstract修饰，表示此方法没有方法体，那么利用类名调用没有方法体的方法没有意义。
3. **abstract与final不能共存。** 正好相反的两个，水火不相容。Abstract是让别人继承或者实现的，而final是禁止继承或禁止实现的。
4. Question3 抽象类中可以不定义抽象方法吗？

答：可以的。但是很少见，这样做的目的就是不让此类创建对象。AWT的适配器对象就是这种类。通常这种类中的方法有方法体，但是却没有内容，即大括号里面是空的。这种类也是没有具体意义的，所以也定义为抽象的。

1. Question4 抽象类和一般类的异同？

答：相同点：都是用来描述事物的，都定义了一些属性和方法；

不同点：

1. 一般类中有足够的信息描述事物，而抽象类中描述事物的信息不足；
2. 一般类中不能定义抽象方法，只能定义非静态方法，而抽象类中既可以定义抽象方法同时也可以定义非抽象方法。
3. 一般类可以实例化，抽象类不可以实例化。
4. Question 5 抽象一定是个父类吗？

答： 一定是。因为只有子类完全覆盖其中的抽象方法后，才可以进行实例化。