内部类：一个类可以定义在另外一个类中，外部的叫做外部类，内部的叫做内部类

内部定义在外部类中，通常只服务于外部类，对外部的其他类不可见，内部类可以直接调用外部的成员变量和方法（包括私有的）

匿名内部类---重点

何时用：

如果在一段程序中需要创建一类对象（一个对象），而且这个对象创建后，称之为匿名内部类

注意：创建对象的这个类一般需要继承某个类或者实现某个接口

Java中编辑器认为：父类大于子类，接口大于实现类强制类型转换的前提：存在继承或者实现关系

文件中有几个class编辑器就会生成几个class文件

抽象类：类中没有办法确定实际的逻辑业务，这个时候可以不写方法体，但是java中不建议空的方法体，因此引入一个关键字abstract来修饰方法，此时该方法不谢方法体，为不完整的方法即为抽象方法，既然方法不全，表示类也不全，所有用abstract修饰该类，整体为抽象类

1. 有抽象方法的类不一定是抽象类，抽象类中不一定有抽象方法
2. 抽象类不能创建对象（不能实例化）：抽象方法不全—抽象类不完整—不能创建对象（实例化）
3. 意义：可以用子类来继承这个类，去实现拓展里面的内容（需要重写）父类中的所有的抽象方法；可以提供一种公共的数据类型
4. 综合：
   1. 抽象方法：由abstract修饰，只有方法定义，没有方法体（abstract不能修饰变量）
   2. 抽象类： 由abstract修饰的类，可以有抽象方法，也可以没有抽象方法，同时可以包含普通方法
   3. 包含抽象方法的类，一定是抽象类，抽象类不一定有抽象方法
   4. 抽象类不能实例化（子类去实现方法体）
   5. 抽象类一般需要被继承（子类去实现方法体）
      1. 子类也可以定义为抽象类
      2. 子类（普通类）继承抽象类，需要重写抽象类中所有的抽象方法
   6. Abstract 和final不能同时修饰一个类，因为final不能被继承，而abstract需要被继承实现才有意义（功能有冲突），同理也不能同时修饰方法
   7. 一个继承抽象类之后，必须重写其全部的抽象方法，不同的子类可以有不同的实现
5. 抽象类的意义：封装子类的成员，为子类提供一种公共的数据类型
6. 定义抽象方法，由子类做不同的实现，但是入口（方法名）是一样的
7. 注意：和普通类的区别：
   1. 抽象类不能实例化
   2. 抽象类可以有抽象方法

接口：

1. java是面向对象的编程语言
2. 面向接口编程：是一面向对对象为基础的，面向接口编程
3. 接口就是一种标准，一个规范（目的是制定你要遵循的规范）关键字：interface表示这是一个接口
4. 接口中只能包含常量（较少） 和抽象方法（居多）
5. 接口不能实例化，实现接口的类，叫做实现类，引入一个关键字implements实现接口，同时接口支持多实现