多态（下）

接 口

- 接口定义了某一批类所需要遵守的规范
- 接口不关心这些类的内部数据，也不关心这些类里方法的实现细节，它只规定这些类里必须提供某些方法

接 口

- 语法：

  [修饰符] interface 接口名 [extends 父接口 1,父接口2...]

  {

  零个到多个常量定义...

  零个到多个抽象方法的定义...

  零个到多个默认方法的定义...（jdk1.8新增）

  零个到多个静态方法方法的定义...（jdk1.8新增）

  }

接口的语法规则

- 接口及接口成员默认的访问权限为：public 或 默认
- 常量默认添加 static final 关键字
- 抽象方法默认添加 abstract 关键字
- 只有default方法及static方法可以添加方法体

接口的语法规则

- 实现接口的类如果不能实现所有接口中待重写的方法，则必须设置为抽象类
- 接口可以实现多继承，即一个子接口可以同时继承多个父接口
- 一个类可以继承自一个父类，同时实现多个接口
- 当一个类同时实现多接口，且其中同时具有相同方法时，实现类需重写该方法，否则会编译报错

内部类

- 在Java中，可以将一个类定义在另一个类里面或者一个方法里面，这样的类称为内部类
- 与之对应，包含内部类的类被称为外部类。

内部类

```
//外部类： 人
public class Person {
    int age;//年龄

    public Heart getHeart(){
        return new Heart();
    }

    //内部类：心脏
    class Heart{
        public String beat(){
            return "心脏在跳动";
        }
    }
}
```

内部类

- **内部类提供了更好的封装，可以把内部类隐藏在外部类之内，不允许同一个包中的其他类访问该类，更好的实现了信息隐藏。**

内部类的分类

- 成员内部类
- 静态内部类
- 方法内部类
- 匿名内部类

成员内部类

- 内部类中最常见的就是成员内部类，也称为普通内部类

```
//外部类： 人
public class Person {
    int age;//年龄

    public Heart getHeart(){
        return new Heart();
    }

    //内部类：心脏
    class Heart{
        public String beat(){
            return "心脏在跳动";
        }
    }
}
```

成员内部类

- 内部类相当于外部类的一个成员变量，可以使用任意访问修饰符。
- 内部类中定义的方法可以直接访问外部类中的数据，而不受访问控制符的影响。
- 外部类不能直接使用内部类的成员和方法，需要借由内部类对象完成。
- 需要通过外部类对象来创建内部类实例。

成员内部类

- 如果外部类和内部类具有相同的成员，内部类默认优先访问自己的成员；可以通过“外部类.this.对象成员” 以及“外部类.静态成员” 的方式访问外部类成员。
- 编译后产生：外部类$内部类.class

静态内部类

- 静态内部类对象可以不依赖于外部类对象，直接创建。
- 静态内部类不能直接访问外部类的非静态成员，但可以通过“外部类对象.成员” 的方式访问 。
- 外部类中可以通过“类名.成员名”的方式直接访问内部类中静态成员

方法内部类

- 定义在外部类方法中的内部类，也称局部内部类。
- 方法内部类只在其定义所在的方法的内部可见，即只在该方法内可以使用。
- 方法内部类不能使用访问控制符和 static 修饰符，但可以使用final和abstract修饰。
- 编译后产生：外部类$数字.class

匿名内部类

- 如果某个类的实例只是用一次，则可以将类的定义与类的创建，放到一起完成，或者说在定义类的同时就创建一个类。以这种方法定义的没有名字的类称为匿名内部类。

匿名内部类

- 适用场景：
  - 只用到类的一个实例
  - 类在定义后马上用到
  - 给类命名并不会导致代码更容易被理解

匿名内部类

- 使用原则：
  - 不能有构造方法，可以通过构造代码块实现数据初始化。
  - 不能定义任何静态成员、静态方法。
  - 不能使用public、protected、private、static、abstract、final修饰。
  - 因匿名内部类也是局部内部类，所以局部内部类的所有限制都对其生效。

匿名内部类

- 使用原则：
  - 一个匿名内部类一定是在new的后面，用其隐含实现一个接口或继承一个类，但是两者不可兼得。
  - 只能创建匿名内部类的一个实例。
  - 匿名内部类在编译的时候由系统自动起名为Outter$1.class

匿名内部类

- **使用原则：**
  - **一般来说，匿名内部类用于继承其他类或是实现接口，并不需要增加额外的方法，只是对继承方法的实现或是重写。**
  - **通过匿名内部类返回的是一个对象的引用，所以可以直接使用或将其复制给一个对象变量。**