今日作业的目标

> 熟悉接口和内部类的特点和使用，掌握内部类的原理

完成作业后，需要将md文件转换成PDF格式，并命名为当天的课程名+下划线+自己的名字！压缩后提交！

- 可以通过查看共享目录下，课程资料中**dayXx_Xxx**就是课程名
- 下划线不要弄错了，不能是空格或者横杠
- 下划线后跟自己的名字，不要在名字后面添一些乱七八糟的东西，如pdf后缀名
- 必须压缩后提交，压缩格式不限，rar、7z等等都可以
- 以上格式满足后，就可以提交作业了

```
提交作业的网址（局域网内网网站）：
    http://192.168.2.100:8080/upload/java/..th
链接最后的“..th”表示班级的期数，比如你是Java28期学生，这里就填入28th

一般来说，打开这个网站对浏览器种类没有特别的要求，仅建议不要直接使用微信自带浏览器
需要注意的是，如果多次重复提交某一天的作业，必须保持名字不同
    建议在“课程名+下划线+自己的名字”的后面加上2，3...之类的数字以示区分
```

成员内部类为何不能有静态成员变量?

```
我是这样理解的:
成员内部类在成员位置  必须依赖于外围类对象创建  静态成员变量是类加载时期初始化的  不依赖于对象的创建是独立于对象是否被创建  有矛盾
```

```
百度答案:
静态的属性或者方法不允许声明在非静态的对象中.

内部类比如和外部类有联系,如果我们创建了static的方法或者字段就破坏了这种耦合,和java的设定背道而驰.如果一个非静态内部类有了静态成员，静态成员不依托于任何内部类实例，那结果也就是此内部类不需要外部类实例就初始化了变量，严重侵害了内部类的定向

内部类的对象  脱离了其外围类的对象  就不会存在，  静态变量  的作用就是  让该类的所有对象共享一个状态。  这个类的所有对象都可以获取和修改这个状态。如果仅仅是这个目的，就可以推出这个状态也是所有外部对象所共享的状态，因此这个定义就可以提升至  外围类中定义，没有必要在内部类中定义，因此在JAVA中不允许在内部类中声明  静态变量
```

操作题

> 操作题，无需表现在作业答案中，自己琢磨和练习即可

待补充！

非编程题

> 简答题，以下简答

- 总结一下各种内部类的成员特点和访问特点

- **成员内部类**
  成员特点:可以定义普通成员(成员变量 成员方法) 不能定义静态成员(静态方法 静态代码块 静态成员变量) 可以定义不会触发类加载的全局常量 构造方法也可以定义
  访问特点:
  1.成员内部类访问外围类成员
  因为外围类对象已经存在,所以可以直接访问
  同名成员:自身对象用this指向,外围类对象用"外围类类名.this"指向 局部变量同名直接输出就近原则
  2.外围类访问成员内部类成员
  直接创建内部类对象 直接用对象名点访问成员即可
  3.外部类中访问成员内部类成员
  首先要有外围类权限,其次还要有内部类权限
  需要先创建外围类对象,然后在外围类对象基础上创建成员内部类对象
  4.成员内部类访问外部类成员
  直接创建对象即可,但是受访问权限限制

  静态内部类
  成员特点:和普通类一样
  访问特点:
  1.静态内部类内部访问外围类
  不管是静态方法还是成员方法 都没有外围类对象,需要直接创建它的对象 创建好对象 用对象名访问即可
  2.外围类访问静态内部类成员
  不管是静态方法还是成员方法 都没有外围类对象,需要直接创建它的对象 创建好对象 用对象名访问即可
  3.外部类访问静态内部类成员
  在外部类中,创建静态内部类对象,需要明确指出该静态内部类属于哪个外围类
  4.静态内部类访问外部类成员
  创建对象即可,受访问权限限制

  局部内部类
  成员特点:和成员内部类是一样的,没有static声明,别的和普通类一致
  访问特点:
  1.局部内部类访问外围类成员
  (1)当局部内部类被定义在外围类的成员方法中时:
  由于已经有了外围类对象 可以直接访问
  出现同名 this指向自身对象
  外围类类名.this指向外围类对象
  全局常量同名就用类名点区分
  (2)当局部内部类被定义在外围类的静态成员方法中时:
  想要访问外部类对象 必须先直接创建对象 有了对象时用对象名点成员变量
  2.外围类访问局部内部类成员
  只有装着成员内部类的外围类成员方法才能够访问 在方法中创建局部内部类对象访问
  3.外部类中访问局部内部类成员
  访问不到
  4.局部内部类访问外部类成员
  创建对象 有访问权限限制

- 总结一下为什么在局部内部类当中访问局部变量必须是final修饰的

若在局部内部类中修改成员变量 方法局部变量不能同步更改 局部内部类和方法的生命周期有冲突 用final标记局部变量就解决了这个问题 这是语法糖

编程题

编程题的答题要求：

> 编程题，需要先编写代码，执行调试完毕后
> 将代码以代码块（CTRL+A贴入整个Java文件内容，而不是一个main方法）的格式贴入md文件
> 并附上执行结果图片

如何在Typora中插入代码块？

1. 可以直接从idea复制代码，然后粘贴进md文档，Typora会自动转换成代码块的格式
2. 可以在md文档空白处中右键，然后插入代码块，再把代码复制进来（熟练了可以使用快捷键）
3. 代码块右下角可以选择语言，建议直接填入Java（这样做会有颜色标记关键字）

如何在Typora中插入图片？

1. 可以使用微信/QQ/windows/Snipaste截图等截图工具截图到计算机粘贴板，然后直接粘贴到md文档中
2. 可以在md文档空白处中右键，然后插入图像，自己选择本地图片的路径（可以用，但不推荐）

敲一遍老师上课的代码

> 根据老师在每一个Demo类注释的头部写的问题，逐一敲一遍老师的代码
>
> 尤其是那些不知道该怎么下手做作业的同学，一定要认真敲一遍老师代码

- 手写一下各种内部类的成员和访问特点

填代码题

> 成员内部类，静态内部类的区别要理解

- 根据注释填写（1），（2），（3）处的代码

```
public class Test{
    public static void main(String[] args){
           //(1)创建并初始化Bean1类对象bean1
         Bean1 bean1 = new Test().new Bean1();
           //静态方法中外围类对象可能不存在
           bean1.i++;
           //(2)创建并初始化Bean2类对象bean2
         Bean2 bean2 = new Bean2();
           bean2.j++;
           //(3)创建并初始化Bean3类对象bean3
         Bean.Bean3 bean3 = new Bean().new Bean3();
           bean3.k++;
    }
    class Bean1{
           public int i = 0;
    }
    static class Bean2{
           public int j = 0;
    }
}
```

```
class Bean{
    class Bean3{
           public int k = 0;
    }
}
```

接口练习题

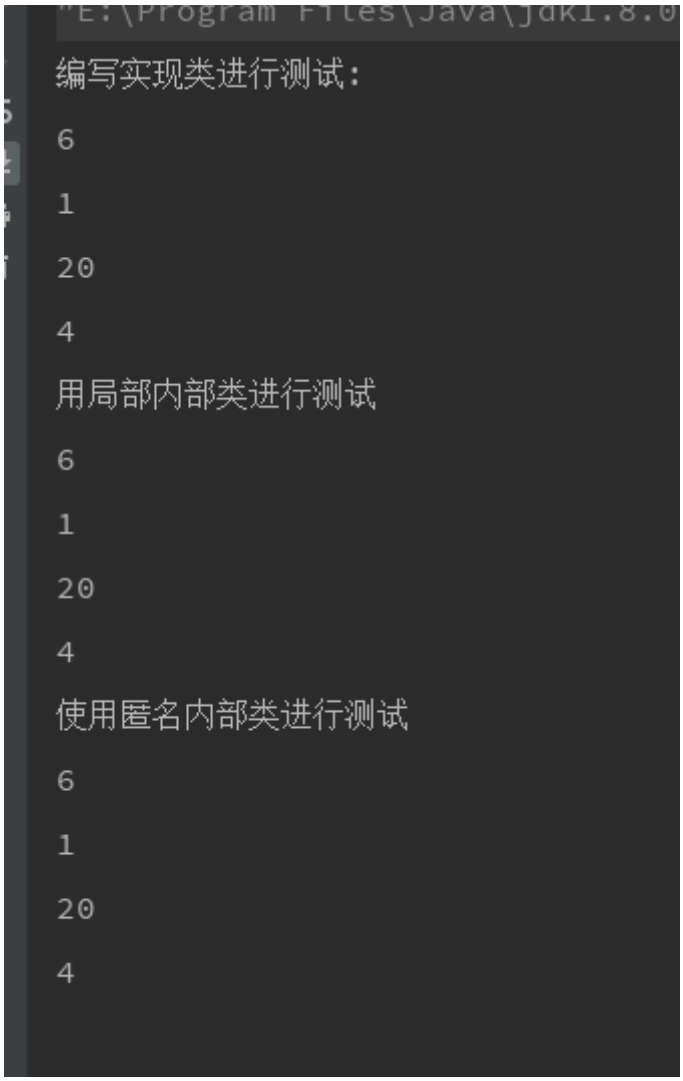
定义一个接口Compute，用来完成计算器的功能，比如最简单的加减乘除功能

请用以下两种方式测试：
    1，编写实现类进行测试
    2，用局部内部类进行测试
    3，使用匿名内部类进行测试

```
package homework.day11.work2;

/**
 * @author ycy
 * @Desc: 定义一个接口Compute，用来完成计算器的功能，比如最简单的加减乘除功能
 * 请用以下两种方式测试：
 * 1，编写实现类进行测试
 * 2，用局部内部类进行测试
 * 3，使用匿名内部类进行测试
 * @date 2021/10/15 0015 下午 21:21
 **/
public class Test2 {
    public static void main(String[] args) {
        System.out.println("编写实现类进行测试:");
        IComputerImpl iComputer = new IComputerImpl();
        System.out.println(iComputer.add(1, 5));
        System.out.println(iComputer.div(9, 6));
        System.out.println(iComputer.mul(4, 5));
        System.out.println(iComputer.div(8, 2));
        System.out.println("用局部内部类进行测试");
        class Inner implements IComputer {
            @Override
            public int add(int a, int b) {
                return a + b;
            }

            @Override
            public int sub(int a, int b) {
                return a - b;
            }

            @Override
            public int mul(int a, int b) {
                return a * b;
            }

            @Override
            public int div(int a, int b) {
                return a / b;
            }
        }
```

```
        Inner inner = new Inner();
        System.out.println(inner.add(1, 5));
        System.out.println(inner.div(9, 6));
        System.out.println(inner.mul(4, 5));
        System.out.println(inner.div(8, 2));
        System.out.println("使用匿名内部类进行测试");
        IComputer iC = new IComputer() {
            @Override
            public int add(int a, int b) {
                return a + b;
            }

            @Override
            public int sub(int a, int b) {
                return a - b;
            }

            @Override
            public int mul(int a, int b) {
                return a * b;
            }

            @Override
            public int div(int a, int b) {
                return a / b;
            }
        };
        System.out.println(iC.add(1, 5));
        System.out.println(iC.div(9, 6));
        System.out.println(iC.mul(4, 5));
        System.out.println(iC.div(8, 2));
    }
}

interface IComputer {
    int add(int a, int b);

    int sub(int a, int b);

    int mul(int a, int b);

    int div(int a, int b);
}

//编写实现类
class IComputerImpl implements IComputer {
    @Override
    public int add(int a, int b) {
        return a + b;
    }

    @Override
    public int sub(int a, int b) {
        return a - b;
    }

    @Override
    public int mul(int a, int b) {
```

```
        return a * b;
    }

    @Override
    public int div(int a, int b) {
        return a / b;
    }
}
```



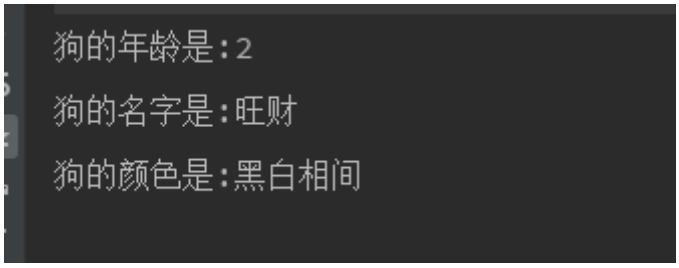
成员内部类练习

> 成员内部类依赖于它的外围类

定义一个类Dog
属性：age,name
除此之外,Dog类中需要定义一个成员内部类Body,Body类中有属性color
请私有化该成员内部类,然后在Dog类提供一个方法,展示Dog类的全部属性

```
package homework.day11.work3;

/**
 * @author ycy
 * @Desc: 成员内部类依赖于它的外围类
 * 定义一个类Dog
 * 属性：age,name
 * 除此之外,Dog类中需要定义一个成员内部类Body,Body类中有属性color
```

```
 * 请私有化该成员内部类,然后在Dog类提供一个方法,展示Dog类的全部属性
 * @date 2021/10/15 0015 下午 21:40
 **/
public class Test3 {
    public static void main(String[] args) {
        Dog dog = new Dog(2, "旺财", "黑白相间");
        dog.showDog();
    }
}

class Dog {
    private int age;
    private String name;
    //将成员内部类对象放入成员列表
    private Body body;

    private class Body {
        String color;

        public Body() {
        }

        public Body(String color) {
            this.color = color;
        }
    }

    public Dog(int age, String name, String color) {
        this.age = age;
        this.name = name;
        this.body = new Body(color);
    }

    public Dog() {
    }

    //提供一个能展示出狗所有属性的方法
    public void showDog() {
        System.out.println("狗的年龄是:" + age);
        System.out.println("狗的名字是:" + name);
        System.out.println("狗的颜色是:" + body.color);
    }
}
```



思考：

私有化成员内部类后，怎么初始化它的对象并赋值？可以思考多种方式

提供get/set方法

提供构造器 创建对象调用

以下为扩展题，需要一定的基础和思维能力，有时间就看看

扩展1：读程序题

> 子类对象初始化中，父类成员还未初始化完毕，不允许使用子类成员变量作为参数

- 以下代码会报错，思考怎么让代码正常运行。

```
class  Student{
    int age;
    String name;
    static int var;//变成静态全局
    public Student(){
    }
    public Student(int age){
        this.age = age;
    }
    public Student(String name){
        this(var);
        this.name = name;
    }
}
```

扩展2：读程序题

> 对象中的成员变量的赋值，初始化默认值是第一步，构造器是最后一步

- 读程序，然后分析过程和结果，思考结果为什么会如此，提供必要的文字说明

```
public class Test{
    public static void main(String[] args){
        Father f1 = new Son(1000);
        Father f2 = new Father();
        Son s = new Son(1000);
    }
}
class Father {
    int i = 10;
    public Father() {
        System.out.println(getI());
    }
    public int getI() {
        return i;
    }
}

class Son extends Father {
    int i = 100;
    public Son(int i) {
```

```
        this.i = i;
    }
    public int getI() {
        return i;
    }
}
```

0

10

0

四、预习问题

> 预习的题目仅为预习提供思路，不用表现在作业中

- 预习lambda表达式的简单使用，Object类