面向对象第8天：

潜艇游戏第一天：

1. 创建6个类，创建World类并测试

潜艇游戏第二天：

1. 给6个类添加构造方法，并测试

潜艇游戏第三天：

1. 创建侦察潜艇数组、鱼雷潜艇数组、水雷潜艇数组，水雷数组，炸弹数组，并测试
2. 设计SeaObject超类，6个类继承超类
3. 在SeaObject中设计两个构造方法，6个派生类分别调用

潜艇游戏第四天：

1. 将侦察潜艇数组、鱼雷潜艇数组、水雷潜艇数组统一组合为SeaObject超类数组，并测试
2. 在6个类中重写move()移动，并测试
3. 画窗口:------共3步

潜艇游戏第五天：

1. 给类中成员添加访问控制修饰符
2. 设计Images图片类

潜艇游戏第六天：

1. 设计窗口的宽和高为常量，适当地方做修改
2. 画对象：----------能够按照我的笔记步骤写出来，就OK了

```
1)想画对象需要获取对象的图片，每个对象都得获取图片，
  意味着获取图片行为为共有行为，所以设计在SeaObject超类中，
  每个对象获取图片的代码都是不一样的，所以设计为抽象方法
  ----在SeaObject中设计为抽象方法getImage()获取对象的图片
2)在派生类中重写getImage()获取对象图片
  ----在6个类中重写getImage()返回不同的图片
3)因为只有活着的对象才需要画到窗口中，所以需要设计对象的状态(活着还是死了)，
  每个对象都有状态，意味着状态为共有属性，所以设计在SeaObject超类中，
  状态一般都设计为常量，同时再设计state变量表示当前状态
  ----在SeaObject中设计LIVE、DEAD常量，state变量表示当前状态
  在后期的业务中经常需要判断对象的状态，每个对象都得判断，
  意味着判断状态的行为为共有行为，所以设计在SeaObject超类中，
  每个对象判断状态的代码都是一样的，所以设计为普通方法
  ----在SeaObject中设计isLive()、isDead()判断对象的状态
4)数据(状态、图片、x坐标、y坐标)都有了就可以开画了，每个对象都得画，
  意味着画对象行为为共有行为，所以设计在SeaObject超类中，
  每个对象画的代码都是一样的，所以设计为普通方法
  ----在SeaObject中设计paintImage()画对象
5)画对象的行为做好了，在窗口world类中调用即可
```

```
5.1)准备对象
5.2)重写paint()方法-----在paint()中调用paintImage()画对象即可
```

潜艇游戏第七天：-----能够按照我的步骤写出来就可以

1. 潜艇入场：
   - 潜艇是由窗口产生的，所以在窗口World类中设计nextSubmarine()生成潜艇对象
   - 潜艇入场为定时发生的，所以在run()中调用submarineEnterAction()实现潜艇入场

     在submarineEnterAction()中：

     每400毫秒，获取潜艇对象obj，submarines扩容，将obj添加到最后一个元素上

     > 注意：在run()中调用submarineEnterAction()之后，一定要调用repaint()来重画
2. 水雷入场：---------前半段(后半段下周二才讲)
   - 水雷是由水雷潜艇发射出来的，所以在MineSubmarine中设计shootMine()生成水雷对象
   - 水雷入场为定时发生的，所以在run()中调用mineEnterAction()实现水雷入场

     在mineEnterAction()中：

     每1000毫秒，.....----暂时搁置
3. 海洋对象移动：
   - 海洋对象移动为共有行为，所以在SeaObject中设计抽象方法move()实现移动，派生类中重写
   - 海洋对象移动为定时发生的，所以在run()中调用moveAction()实现海洋对象移动

     在moveAction()中：

     遍历所有潜艇--潜艇动，遍历所有水雷--水雷动，遍历所有炸弹--炸弹动

潜艇游戏第八天：-----能够按照我的步骤写出来就可以

1. 炸弹入场：
   - 炸弹是由战舰发射出来的，所以在Battleship中设计shootBomb()生成炸弹对象
   - 炸弹入场为事件触发的，所以在侦听器中重写keyReleased()按键抬起事件，在抬起事件中：
     - 判断若抬起的是空格键，则：

       获取炸弹对象obj，bombs扩容，将obj添加到bombs的最后一个元素上
2. 战舰移动：
   - 战舰移动为战舰的行为，所以在Battleship中设计moveLeft()左移、moveRight()右移
   - 战舰移动为事件触发的，所以在侦听器的重写keyReleased()按键抬起事件中：
     - 判断若抬起的是左箭头，则战舰左移
     - 判断若抬起的是右箭头，则战舰右移
3. 删除越界的海洋对象：
   - 在SeaObject中设计isOutOfBounds()检测潜艇是否越界，在Bomb和Mine中重写isOutOfBounds()检测炸弹和水雷是否越界
   - 删除越界海洋对象为定时发生的，所以在run()中调用outOfBoundsAction()删除越界海洋对象

     在outOfBoundsAction()中：

     遍历所有潜艇/水雷/炸弹数组，判断若越界了：

将越界元素替换为数组的最后一个元素，缩容

4. 设计EnemyScore得分接口，侦察潜艇和鱼雷潜艇实现得分接口

   设计EnemyLife得命接口，水雷潜艇实现得命接口

回顾：

1. 成员内部类：

   类中套类，外面的称为外部类，里面的称为内部类，内部类只能服务于外部类，对外不具备可见性。内部类对象也是需要在外部类中创建，内部类中可以直接访问外部类的成员，包括私有的，因为在内部类有个隐式的引用指向了创建它的外部类对象------外部类名.this----------API时会用

2. 匿名内部类：----------简化代码

   若想创建一个类(派生类)的对象，并且对象只被创建一次，此时可以设计为匿名内部类

   匿名内部类中不能修改外面局部变量的值，因为该值在此处默认为final的---------API时会用

精华笔记：

1. 接口：
   - 是一种数据类型(引用类型)
   - 由interface定义
   - 只能包含常量和抽象方法(所有数据默认都是常量，所有方法默认都是抽象的)
   - 接口不能被实例化
   - 接口是需要被实现/继承的，实现/派生类：必须重写所有抽象方法
   - 一个类可以实现多个接口，用逗号分隔。若又继承又实现时，应先继承后实现。
   - 接口可以继承接口

笔记：

1. 接口：
   - 是一种数据类型(引用类型)
   - 由interface定义
   - 只能包含常量和抽象方法(所有数据默认都是常量，所有方法默认都是抽象的)
   - 接口不能被实例化
   - 接口是需要被实现/继承的，实现/派生类：必须重写所有抽象方法
   - 一个类可以实现多个接口，用逗号分隔。若又继承又实现时，应先继承后实现。
   - 接口可以继承接口

```
//接口的演示
public class InterfaceDemo {
    public static void main(String[] args) {
        //Inter5 o = new Inter5(); //编译错误，接口不能被实例化
        Inter5 o1 = new Doo(); //向上造型
```

```
        Inter4 o2 = new Doo(); //向上造型
    }
}

//演示接口的语法
interface Inter{
    public static final int NUM = 5;
    public abstract void show();
    int COUNT = 5; //默认public static final
    void test();   //默认public abstract
    //int number; //编译错误，常量必须声明同时初始化
    //void say(){ } //编译错误，抽象方法不能有方法体
}

//演示接口的实现
interface Inter1{
    void show();
    void test();
}
class Aoo implements Inter1{
    public void show(){} //重写接口中的抽象方法时，必须加public权限
    public void test(){}
}

//演示接口的多实现
interface Inter2{
    void show();
}
interface Inter3{
    void test();
}
abstract class Boo{
    abstract void say();
}
class Coo extends Boo implements Inter2,Inter3{
    public void show(){}
    public void test(){}
    void say(){}
}

//演示接口继承接口
interface Inter4{
    void show();
}
interface Inter5 extends Inter4{
    void test();
}
class Doo implements Inter5{
    public void test(){}
    public void show(){}
}
```

补充：

1. 水雷入场、战舰移动的功能可以CV的代码：

```
KeyAdapter k = new KeyAdapter() {
    public void keyReleased(KeyEvent e) { //当按键抬起时会自动触发---不要求掌握
        if(e.getKeyCode()==KeyEvent.VK_SPACE){ //若抬起的是空格键---不要求掌握

        }
        if(e.getKeyCode()==KeyEvent.VK_LEFT){ //若抬起的是左箭头---不要求掌握

        }
        if(e.getKeyCode()==KeyEvent.VK_RIGHT) { //若抬起的是右箭头---不要求掌握

        }
    }
};
this.addKeyListener(k); //添加侦听---不要求掌握
```

2. 类中成员的默认访问权限---------默认的

   接口中成员的默认访问权限------public的

   重写接口中的抽象方法时，必须加public权限

3. 类和类-----------------------继承

   接口和接口-----------------继承

   类和接口---------------------实现

4. 能够造型成为的类型：超类+所实现的接口

5. 设计规则：

   - 将所有派生类所共有的属性和行为，抽到超类中----------抽共性
   - 若派生类的行为(实现代码)都一样，设计为普通方法

     若派生类的行为(实现代码)都不一样，设计为抽象方法
   - 将部分派生类所共有的属性和行为，抽到接口中

     接口是对继承的单根性的扩展----------实现多继承

     接口是一个标准、一种规范，实现了接口就能干某件事，不实现接口就干不了那个事

6. 如何调错：----------帮助我们找到问题位置所在

   - 快速锁定问题方法：
     - 将方法调用的代码全都注释掉，然后一个一个放开，

       放开哪个出错了，说明问题就出在那个方法上。
   - 打桩: System.out.println(数据);

7. 明日单词：

```
1)hit:撞
2)other:另一个
3)instanceof:实例
4)cut:剪
5)cast:类型
6)go:去
7)draw:画
```