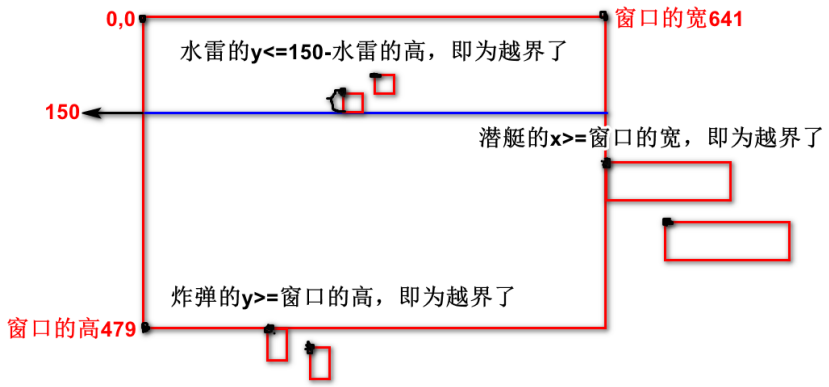
面向对象第8天：

潜艇游戏第八天：

0. 事件与事件处理：
    - 事件：发生了一件事
    - 事件处理：发生事之后所做的操作
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
    - 炸弹入场是事件触发的，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
    - 战舰移动是事件触发的，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：
    - 检测越界的标准：

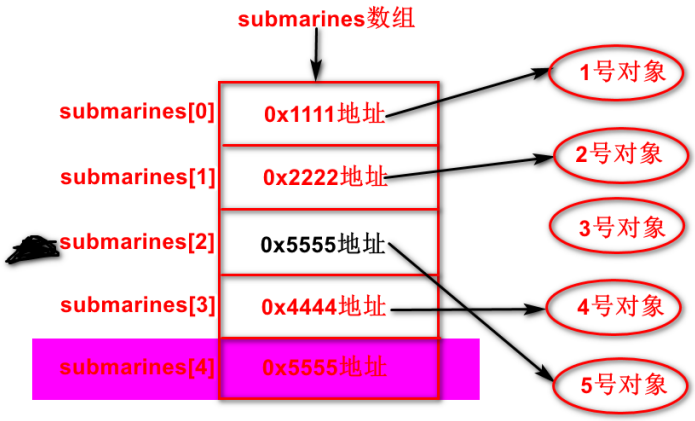


    - 在SeaObject中设计isOutOfBounds()检测潜艇是否越界，在Bomb和Mine中重写isOutOfBounds()检测炸弹和水雷是否越界
    - 删除越界海洋对象为定时发生的，所以在run()中调用outOfBoundsAction()删除越界海洋对象

      在outOfBoundsAction()中：

      遍历所有潜艇/水雷/炸弹，判断若越界了：

      将数组最后一个元素赋值给越界元素（实现删除越界元素），再缩容（实现删除数组最后一个元素）



4. 设计EnemyScore得分接口，ObserveSubmarine和TorpedoSubmarine实现得分接口
   设计EnemyLife得命接口，MineSubmarine实现得命接口

笔记：

1. 接口：
   - 是一种引用数据类型
   - 由interface定义
   - 只能包含常量和抽象方法
   - 不能被实例化
   - 接口是需要被实现/继承的，实现类/派生类：必须重写接口中的所有抽象方法
   - 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 o1 = new Inter5(); //编译错误，接口不能被实例化
        Inter5 o2 = new Doo(); //向上造型
        Inter4 o3 = new Doo(); //向上造型
    }
}

//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() {
    }
}
```

```
//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

    //接口的方法默认public abstract，所以类实现接口后，方法也要显式写出public
    public void test() {
    }
}

//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
}
```

补充：

1. 接口中常量默认public static final，抽象方法默认public abstract
2. 关系：

- 类和类：继承
- 接口和接口：继承
- 类和接口：实现

3. 设计规则：
   - 将所有派生类共有的属性和行为，抽到超类中：抽共性
   - 若派生类的行为都一样，设计为普通方法

     若派生类的行为都不一样，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，实现了接口就能干那个事，不实现接口就干不了那个事
4. 如何调错：
   - 快速锁定问题方法：将调用方法的语句全都注释掉，然后一个一个的放开，放开哪个方法出错，意味着问题就出在了那个方法中
   - 打桩：System.out.println(数据);
5. 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
```