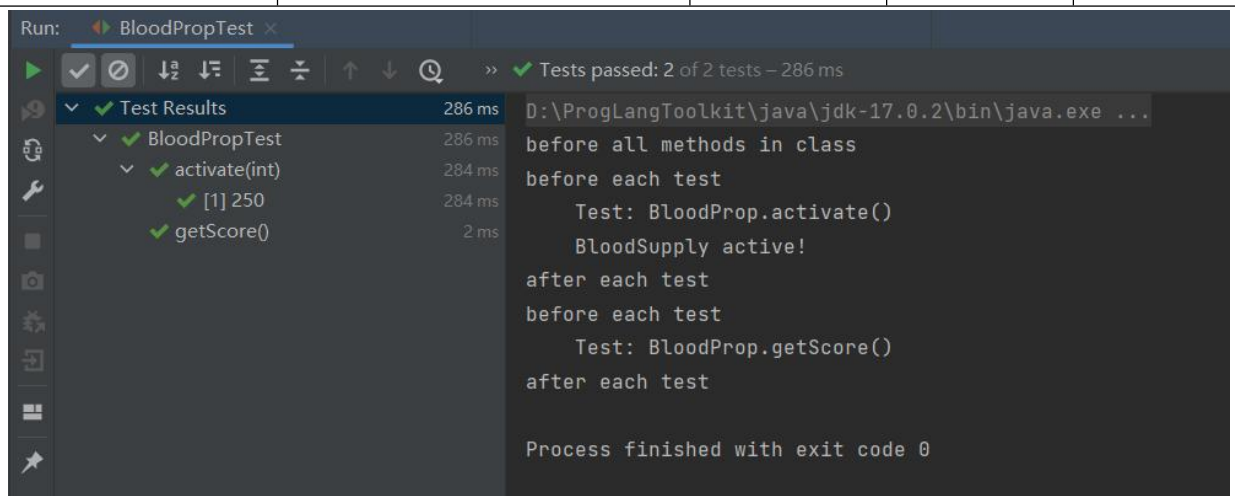


# 实验三报告

## 一、单元测试

### 1. 测试用例

用例编号	BloodPropTest1			
待测试类及方法	BloodProp.activate()			
测试类及方法	BloodPropTest.activate()			
前提条件（如有）	heroAircraft 等初始化，生成 bloodProp 道具			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试 测试加血道具功能 参数为英雄机血量	1.初始化各对象 2.生成加血道具 3.根据参数设置英雄机血量 4.激活道具并检查结果	300	300	通过
用例编号	BloodPropTest2			
待测试类及方法	BloodProp.getScore()			
测试类及方法	BloodPropTest.getScore()			
前提条件（如有）	生成 bloodProp 道具			
用例描述	测试步骤	期望结果	实际输出	测试结果
测试 测试加血道具分数	1.初始化道具工厂 2.生成加血道具 3.获取加血道具分数并核对正确数值	30	30	通过



```
Run: BloodPropTest x
Tests passed: 2 of 2 tests - 286 ms
Test Results
  BloodPropTest
    activate(int)
    getScore()
D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...
before all methods in class
before each test
Test: BloodProp.activate()
BloodSupply active!
after each test
before each test
Test: BloodProp.getScore()
after each test
Process finished with exit code 0
```

用例编号	BombPropTest1			
待测试类及方法	BombProp.activate()			
测试类及方法	BombPropTest.activate()			

前提条件（如有）	enemyAircrafts 等初始化，生成 bombProp 道具			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试 测试炸弹道具功能 参数为每种敌机的数量	1.初始化各对象 2.生成炸弹道具 3.由工厂制造各敌机 4.激活炸弹道具并检查结果（只能炸毁非 boss 机）	7	7	通过

```

Run: BombPropTest x
Tests passed: 2 of 2 tests - 304 ms
Test Results
  BombPropTest
    activate(int) [1] 7
    getScore()
D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...
before all methods in class
before each test
Test: BombProp.activate()
BombSupply active!
after each test
before each test
Test: BombProp.getScore()
after each test
Process finished with exit code 0

```

用例编号	BulletPropTest1			
待测试类及方法	BulletProp.activate()			
测试类及方法	BulletPropTest.activate()			
前提条件（如有）	heroAircraft 初始化，生成 bulletProp 道具			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试 测试火力道具功能 参数为激活火力道具的次数，改变子弹个数，该测试只测试子弹数	1.初始化各对象 2.生成火力道具 3.根据参数激活特定次数的火力道具 4.根据可装配子弹数目上限检查激活后的结果	5	5	通过

```

Run: BulletPropTest x
Tests passed: 2 of 2 tests - 265 ms
Test Results
  BulletPropTest
    activate(int) [1] 8
    getScore()
D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...
before all methods in class
before each test
Test: BulletProp.activate()
bulletProp: 8
activate number: 1 BulletSupply active!
activate number: 2 BulletSupply active!
activate number: 3 BulletSupply active!
activate number: 4 BulletSupply active!
activate number: 5 BulletSupply active!
activate number: 6 BulletSupply active!
activate number: 7 BulletSupply active!
activate number: 8 BulletSupply active!
after each test
before each test
Test: BulletProp.getScore()
after each test
Process finished with exit code 0

```

用例编号	HeroAircraftTest1			
待测试类及方法	HeroAircraft.shoot()			
测试类及方法	HeroAircraftTest.shoot()			
前提条件（如有）	heroAircraft, bulletProp 等对象初始化			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试 参数为火力道具 激活次数 测试火力道具对 shoot 的加成 测试 shoot 弹道	1. 初始化各对象 2. 使用火力道具 3. 分别测试 shoot 方法发射的子弹 xy 绝对坐标或均值（打印出的为便于观察的相对坐标） 注：因本实验要求只能测试一个测试数据，原参数化测试数据被注释	699 256	699 256	通过

```

Run: HeroAircraftTest
Tests passed: 8, ignored: 5 of 13 tests - 284 ms

Test Results
  HeroAircraftTest
    isShieldValid()
    getDirection()
    forward()
    shoot(int)
      setBulletValid()
      setDirection()
      getHeroAircraft()
      setPower()
      isBulletValid()
      setShieldValid()
      setShootNum()
      getShootNum()
      getPower()

void edu.hitsz.aircraft.HeroAircraftTest.isShieldValid() is @Disabled
before each test
after each test

void edu.hitsz.aircraft.HeroAircraftTest.forward() is @Disabled
before each test

Test: HeroAircraft.shoot()
activate times: 8
activate number: 1 BulletSupply active!
activate number: 2 BulletSupply active!
activate number: 3 BulletSupply active!
activate number: 4 BulletSupply active!
activate number: 5 BulletSupply active!
activate number: 6 BulletSupply active!
activate number: 7 BulletSupply active!
activate number: 8 BulletSupply active!
x_offset: -4 y_offset: 16
x_offset: -2 y_offset: 4
x_offset: 0 y_offset: 0
x_offset: 2 y_offset: 4
x_offset: 4 y_offset: 16
after each test
  
```

用例编号	BossEnemyTest1			
待测试类及方法	BossEnemy.getHp()			
测试类及方法	BossEnemyTest.getHp()			
前提条件（如有）	bossFactory 初始化，生成 BossEnemy 类的实例			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试 参数为难度 根据设定的难度 生成 boss 敌机， 获取 boss 敌机血 量值	1. bossFactory 初始化 2. 根据参数构造对应难度的 boss 敌机 3. 获取 boss 敌机血量 注：当前代码中，难度只调控了 boss 机血量	4000	4000	通过

用例编号	BossEnemyTest2			
待测试类及方法	BossEnemy.shoot()			
测试类及方法	BossEnemyTest.shoot()			
前提条件（如有）	bossFactory 初始化，生成 BossEnemy 类的实例			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试 参数为从最低难度变化到的最高难度 根据设定的难度生成 boss 敌机，检查 boss 在不同难度下的弹道	1.bossFactory 初始化	51	51	通过  注：由于 boss 敌机生成坐标有随机性，以及表格空间有限，此处仅列举示例的部分
	2. 根据参数构造对应难度的 boss 敌机	63	63	
	3. 遍历难度最小从 1 开始，检查 boss 敌机子弹 xy 坐标绝对值或均值（只检查了抛物线，散射未检查） 注：当前代码中，难度只调控了 boss 机血量	67	67	
		63	63	
		51	51	
		213	213	
		43	43	
		55	55	
		59	59	
		55	55	
		43	43	
		206	206	
		...	...	

```

Run: BossEnemyTest x
Tests passed: 6 of 6 tests - 294 ms

Test Results 294 ms
  BossEnemyTest 294 ms
    getHp(int) 281 ms
      [1] 8 281 ms
    shoot(int) 6 ms
      setPower() 3 ms
      setShootNum() 2 ms
      getShootNum() 1 ms
      getPower() 1 ms

D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...
before each test
  Test: BossEnemy.getHp()
  MaxHp: 4000
after each test
before each test
  Test: BossEnemy.shoot()
  Difficulty: 1
  x_offset: -4 y_offset: 16
  x_offset: -2 y_offset: 4
  x_offset: 0 y_offset: 0
  x_offset: 2 y_offset: 4
  x_offset: 4 y_offset: 16

  Difficulty: 1
  x_offset: -4 y_offset: 16
  x_offset: -2 y_offset: 4
  x_offset: 0 y_offset: 0
  x_offset: 2 y_offset: 4
  x_offset: 4 y_offset: 16

  Difficulty: 2
  x_offset: -4 y_offset: 16
  x_offset: -2 y_offset: 4
  x_offset: 0 y_offset: 0
  x_offset: 2 y_offset: 4
  x_offset: 4 y_offset: 16

  Difficulty: 3
  
```

用例编号	EliteEnemyTest1			
待测试类及方法	EliteEnemy.forward()			
测试类及方法	EliteEnemyTest.forward()			
前提条件（如有）	eliteFactory 初始化，生成 EliteEnemy 类的实例			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试 检测 forward 方法的范围限制 精英机的 forward 方法同时需要检测左右反弹和上下出界，故仅对其进行测试即可	1.初始化 eliteFactory 等对象 2.工厂模式生成 elite 实例 3.根据 elite 是否超出最下方边界对其状态进行检测	true true true true	true true true true	通过
用例编号	EliteEnemyTest2			
待测试类及方法	EliteEnemy.getHp()			
测试类及方法	EliteEnemyTest.getHp()			
前提条件（如有）	eliteFactory 初始化，生成 EliteEnemy 类的实例			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试，参数为难度 生成 elite 机，检查 getHp()方法	1.初始化 eliteFactory 等对象 2.工厂模式生成 elite 实例 3.测试 elite 血量值	540	540	通过
用例编号	EliteEnemyTest3			
待测试类及方法	EliteEnemy.shoot()			
测试类及方法	EliteEnemyTest.shoot()			
前提条件（如有）	eliteFactory 初始化，生成 EliteEnemy 类的实例			
用例描述	测试步骤	期望结果	实际输出	测试结果
参数化测试，参数为难度 检查 elite 子弹绝对位置	1.初始化 eliteFactory 等对象 2.工厂模式生成 elite 实例 3.测试 elite 子弹绝对坐标	12110 40742 57 60 227145 377106 141124	12110 40742 57 60 227145 377106 141124	通过 注：生成有随机性，仅举作示例

```

Run: EliteEnemyTest x
Tests passed: 3 of 3 tests - 287 ms

Test Results
  EliteEnemyTest
    forward(int)
      [1] 8
    getHp(int)
      6 ms
    shoot(int)
      [1] 6
      2 ms

before each test
Test: EliteEnemy.forward()
yloc: 0 <= 182 <= 768
xloc: -10 <= 106 <= 522
after each test
before each test
Test: EliteEnemy.getHp()
MaxHp: 540
after each test
before each test
Test: EliteEnemy.shoot()
after each test
  
```



## 2. JUnit 单元测试结果

Run: BloodPropTest x

Tests passed: 2 of 2 tests – 275 ms

Test Results	Time
BloodPropTest	275 ms
activate(int)	273 ms
[1] 250	273 ms
getScore()	2 ms

D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...

before all methods in class

before each test

Test: BloodProp.activate()

BloodSupply active!

after each test

before each test

Run: BombPropTest x

Tests passed: 2 of 2 tests – 271 ms

Test Results	Time
BombPropTest	271 ms
activate(int)	268 ms
[1] 7	268 ms
getScore()	3 ms

D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...

before all methods in class

before each test

Test: BombProp.activate()

BombSupply active!

after each test

before each test

Run: BulletPropTest x

Tests passed: 2 of 2 tests – 254 ms

Test Results	Time
BulletPropTest	254 ms
activate(int)	252 ms
[1] 8	252 ms
getScore()	2 ms

D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...

before all methods in class

before each test

Test: BulletProp.activate()

bulletProp: 8

activate number: 1 BulletSupply active!

activate number: 2 BulletSupply active!

Run: HeroAircraftTest x

Tests passed: 8, ignored: 5 of 13 tests – 291 ms

Test Results	Time
HeroAircraftTest	291 ms
isShieldValid()	
getDirection()	255 ms
forward()	
shoot(int)	27 ms
setBulletValid()	
setDirection()	1 ms
getHeroAircraft()	2 ms
setPower()	2 ms
isBulletValid()	
setShieldValid()	
setShootNum()	1 ms
getShootNum()	2 ms
getPower()	1 ms

D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...

before all methods in class

void edu.hitsz.aircraft.HeroAircraftTest.isShieldValid() is @Disabled

before each test

after each test

void edu.hitsz.aircraft.HeroAircraftTest.forward() is @Disabled

before each test

Test: HeroAircraft.shoot()

activate times: 8

activate number: 1 BulletSupply active!

activate number: 2 BulletSupply active!

activate number: 3 BulletSupply active!

Run: BossEnemyTest x

Tests passed: 6 of 6 tests – 278 ms

Test Results	Time
BossEnemyTest	278 ms
getHp(int)	267 ms
[1] 8	267 ms
shoot(int)	5 ms
setPower()	2 ms
setShootNum()	2 ms
getShootNum()	1 ms
getPower()	1 ms

D:\ProgLangToolkit\java\jdk-17.0.2\bin\java.exe ...

before each test

Test: BossEnemy.getHp()

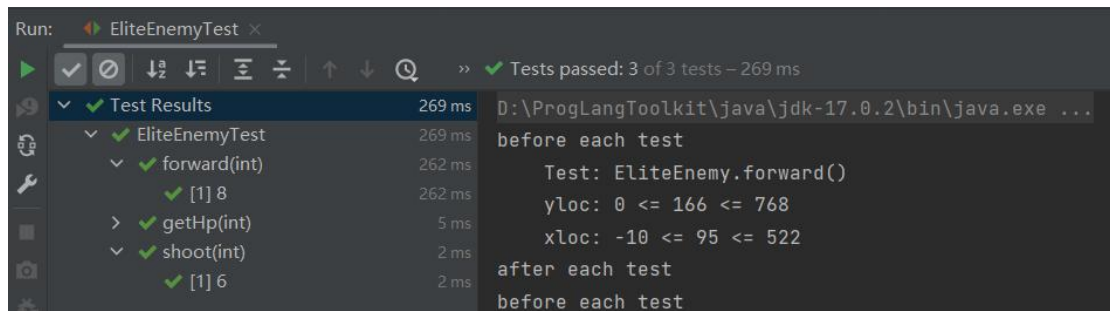
MaxHp: 4000

after each test

before each test

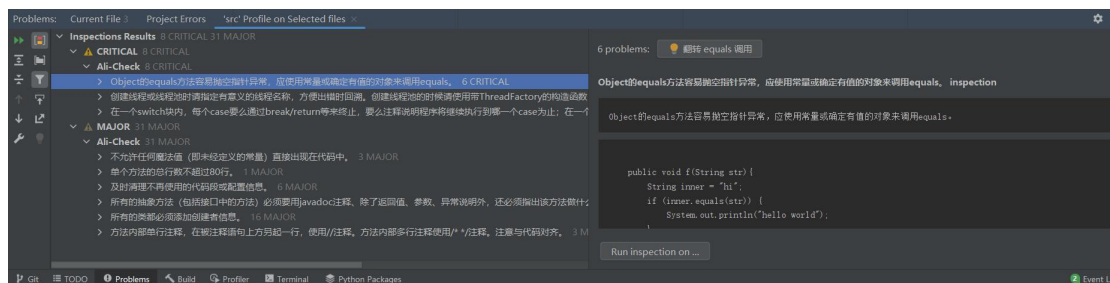
Test: BossEnemy.shoot()

Difficulty: 1



## 二、编码规约

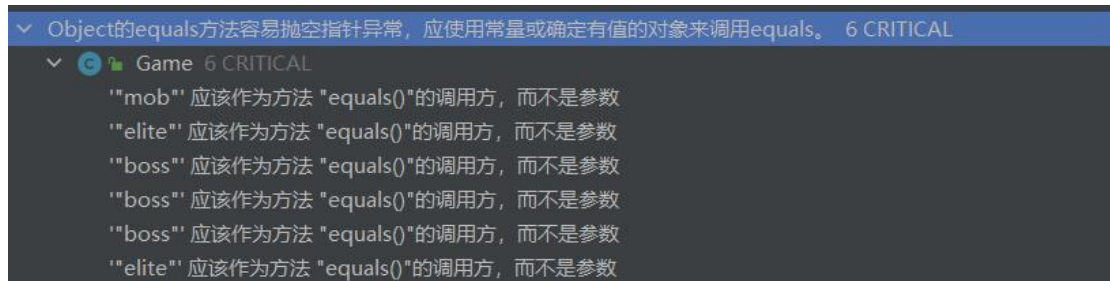
使用阿里编码规约插件能方便的找出代码中存在的问题，如下图所示：



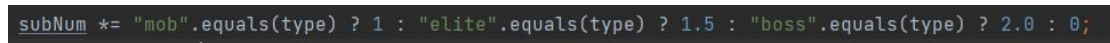
对于我写下的未特别加以注意的代码，插件扫描出了 8 CRITICAL，31 MAJOR 的问题。

下面对其中 2 类比较典型的问题进行修改：

### (1) Object 的 equals 方法调用的问题，应当使用确定有值的对象来调用



很容易能通过插件将原来代码中调用方进行调换，修复了这 6 个 CRITICAL，更正后的示例代码如下：



### (2) 使用 switch 时应当有 default 语句，即使什么也不做

```
switch (type) {
    case "blood": {
        props.add(bloodPropFactory.createProp(
            enemyAircraft.getLocationX(), enemyAircraft.getLocationY(), type));
        break;
    }
    case "bomb": {
        props.add(bombPropFactory.createProp(
            enemyAircraft.getLocationX(), enemyAircraft.getLocationY(), type));
        break;
    }
    case "bullet": {
        props.add(bulletPropFactory.createProp(
            enemyAircraft.getLocationX(), enemyAircraft.getLocationY(), type));
        break;
    }
    default:
```

最后，处理了剩余的绝大多数 MAJOR 问题，在代码中添加了完整的 javadoc 注释，最终的插件扫描结果如下：