**《软件测试》**

**实验报告六 ——综合测试1**

**姓 名： 汪成肸 学 号： 2020105002**   **院 系： 计算机与信息学院 专 业：计算机科学与技术**

**实 验 室： 实验日期： 2023/04/14**

**总评成绩： 审阅教师： 杨青**

### 一、实验目的：

1. 练习基本的编程能力；
2. 学习在一定规模实际项目中综合测试方法
3. 学习研读开源代码的技术
4. 熟悉项目构建工具gradle
5. 熟悉静态代码分析工具

### 二、实验环境：

工具：IntelliJ IDEA 2022.2.2

操作系统：Windows11

### 实验要求：

1. 从[https://gitee.com/jmhui/jpacman-ctgu](https://gitee.com/jmhui/jpacman-ctgu" \o "https://gitee.com/jmhui/jpacman-ctgu) 仓库克隆代码到自己仓库，导入到自己使用的IDE中,并能够构建运行；
2. 运行冒烟测试smoketest，在src/default-test/java/nl.tudelft.jpacman/下；
3. 项目有两个测试文件目录，default-test (src/default-test)包含一些已经给出的测试用例样例，test (src/test)目录作为作业编写检查的目录
4. 修改标题：在PacManUI类（nl.tudelft.tudelft.jpacman.ui 包），在构造函数的第一行，即 super(“JPacman”);修改为super(“JPacman <当前年份>”);
5. 完成下面题目1和题目2，提交到自己的代码仓库
6. 编写测试报告，提交到雨课堂“软件测试实验六”

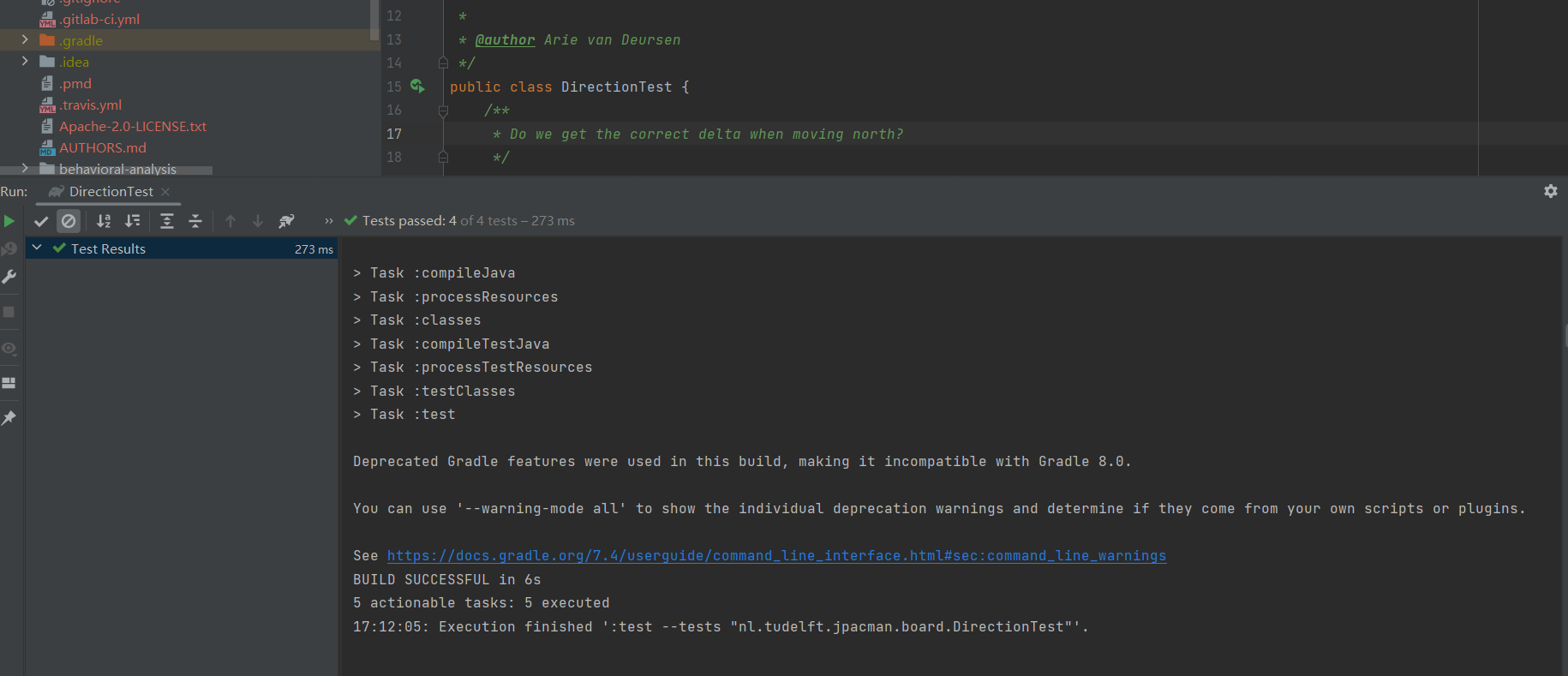
### 实验步骤与内容

**题目一：**在DirectionTest中创建测试方法，例如，南、东、西方向。运行测试，并确保测试通过。（注意在设置中配置测试运行器）。

DirectionTest代码：

public class DirectionTest {  
 */\*\*  
 \* Do we get the correct delta when moving north?  
 \*/* @Test  
 void testNorth() {  
 Direction north = Direction.*valueOf*("NORTH");  
   
 *assertThat*(north.getDeltaX()).isEqualTo(0);  
 *assertThat*(north.getDeltaY()).isEqualTo(-1);  
 }  
  
 @Test  
 void testEast() {  
 Direction east = Direction.*valueOf*("EAST");  
  
 *assertThat*(east.getDeltaX()).isEqualTo(1);  
 *assertThat*(east.getDeltaY()).isEqualTo(0);  
 }  
  
 @Test  
 void testWest() {  
 Direction east = Direction.*valueOf*("WEST");  
  
 *assertThat*(east.getDeltaX()).isEqualTo(-1);  
 *assertThat*(east.getDeltaY()).isEqualTo(0);  
 }  
  
 @Test  
 void testSouth() {  
 Direction east = Direction.*valueOf*("SOUTH");  
  
 *assertThat*(east.getDeltaX()).isEqualTo(0);  
 *assertThat*(east.getDeltaY()).isEqualTo(1);  
 }  
}

测试结果图片：



**题目2**

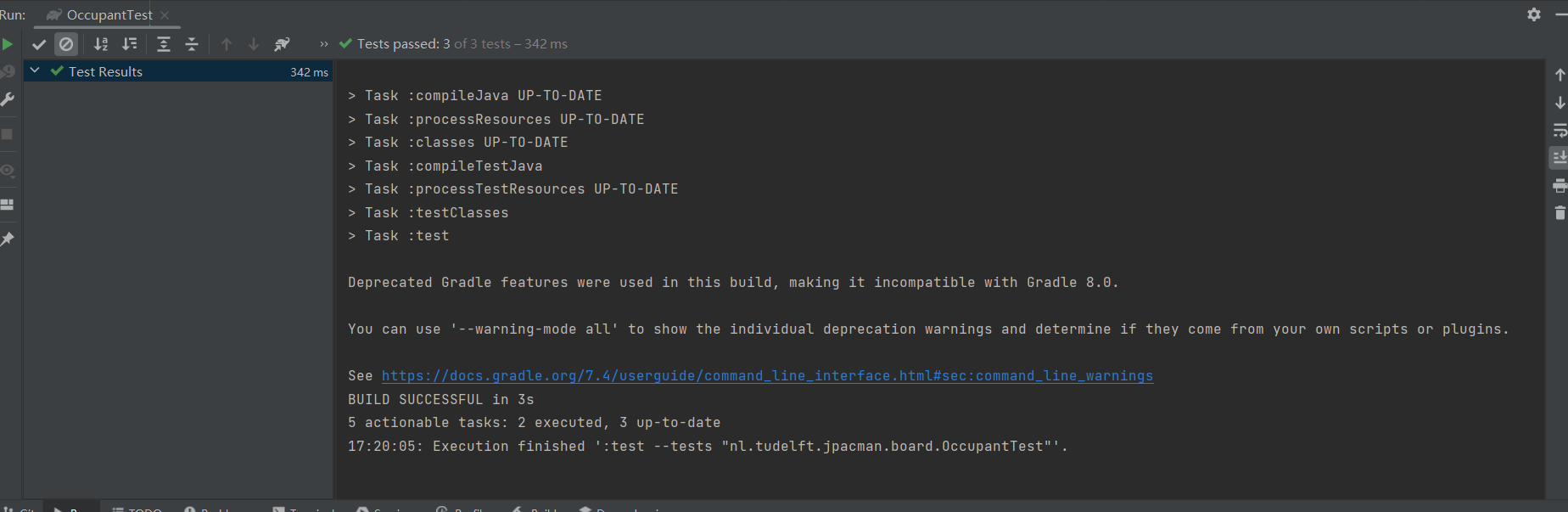
程序中留下了三个空的测试方法。你的目标是实现这些测试。

1. noStartSquare()，用来测试一个Unit在没有调用occupy来设置它的Square时，这个Unit不应该占据Square。程序中，一个Unit在最开始时，是没有Square的。
2. testOccupy()，用来验证Unit在占据一个Square后，它就拥有一个目标方格(target square)。换句话说，如果一个Unit占用了一个Square，那么这个Square应该容纳了(contains)一个Unit。
3. 最后，testReoccupy()是在Unit重复占用单元（连续调用两次occupy）后，验证2中的占用关系和容纳关系?  
   确保所有的测试通过，提交并push你的测试代码。

OccupantTest类代码:

class OccupantTest {  
  
 */\*\*  
 \* The unit under test.  
 \*/* private Unit unit;  
  
 */\*\*  
 \* Resets the unit under test.  
 \*/* @BeforeEach  
 void setUp() {  
 unit = new BasicUnit();  
 }  
  
 */\*\*  
 \* Asserts that a unit has no square to start with.  
 \*/* @Test  
 void noStartSquare() {  
 // Remove the following placeholder:  
 *assertThat*(unit).isNotNull();  
 *assertThat*(unit.hasSquare()).isFalse();  
   
 }  
  
 */\*\*  
 \* Tests that the unit indeed has the target square as its base after  
 \* occupation.  
 \*/* @Test  
 void testOccupy() {  
 Square square = new BasicSquare();  
 unit.occupy(square);  
   
 *assertThat*(unit.getSquare()).isEqualTo(square);  
 *assertThat*(square.getOccupants()).contains(unit);  
 }  
  
 @Test  
 void testReoccupy() {  
 Square square = new BasicSquare();  
 unit.occupy(square);  
 unit.occupy(square);  
  
 *assertThat*(unit.getSquare()).isEqualTo(square);  
 *assertThat*(square.getOccupants()).contains(unit);  
 }  
}

测试结果截图:



### 结论分析与体会

通过这次实验，我学会使用idea克隆gitee上的项目，复习了怎么编写单元测试，也进一步提高了我的研读源码的能力，阅读英文注释，精进了我的英文能力。

### 六、仓库地址

https://github.com/shuaishuaixi03/software-testing/tree/main/project-0414/jpacman-ctgu