**《软件测试》**

**实验报告六 ——综合测试1**

**姓 名： 黄文轩 学 号：2019176203**

**院 系： 计算机与信息学院 专 业： 数据科学与大数据技术**

**实 验 室： 实验日期：2022/5/2**

**总评成绩： 审阅教师：**

目录

[**《软件测试》** 1](#_Toc101792344)

[**实验报告六 ——综合测试1** 1](#_Toc101792345)

[一、实验目的： 3](#_Toc101792346)

[二、实验环境： 3](#_Toc101792347)

[三、实验要求： 3](#_Toc101792348)

[四、实验步骤与内容 3](#_Toc101792349)

[五、结论分析与体会 6](#_Toc101792350)

[六、仓库地址 6](#_Toc101792351)

### 一、实验目的：

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

### 二、实验环境：

Eclipse2020,JUnit

### 三、实验要求：

题目1

1. 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）

题目2

1. 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。
2. 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。
3. 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。
4. 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?
5. 确保所有的测试通过，提交并push你的测试代码。

### 四、实验步骤与内容

1. **从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”**

**测试用例：**

|  |  |  |
| --- | --- | --- |
| 输入方向 | 预测输出 | 实际输出 |
| SOUTH | 0,1 | 0,1 |
| WEST | -1,0 | -1,0 |
| EAST | 1,0 | 1,0 |
| NORTH | 0,-1 | 0,-1 |

1. **代码实现：**

|  |
| --- |
| **测试代码：**  **OccupantTest类：**  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 square1 = new BasicSquare();  Square square2 = new BasicSquare();  unit.occupy(square1);  assertThat(unit.getSquare()).isEqualTo(square1);  assertThat(square1.getOccupants()).contains(unit);  unit.occupy(square2);  assertThat(unit.getSquare()).isEqualTo(square2);  assertThat(square2.getOccupants()).contains(unit);  }  }  **DirectionTest类：**  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 testSouth() {  Direction south = Direction.valueOf("SOUTH");  assertThat(south.getDeltaX()).isEqualTo(0);  assertThat(south.getDeltaY()).isEqualTo(1);  }  @Test  void testWest() {  Direction west = Direction.valueOf("WEST");  assertThat(west.getDeltaX()).isEqualTo(-1);  assertThat(west.getDeltaY()).isEqualTo(0);  }  @Test  void testEast() {  Direction east = Direction.valueOf("EAST");  assertThat(east.getDeltaX()).isEqualTo(1);  assertThat(east.getDeltaY()).isEqualTo(0);  }    }  **过程：**    **测试：** |

### 五、结论分析与体会

学会了综合测试方法。

### 六、仓库地址

https://github.com/Osilly/Software-Testing-Experiment