**《软件测试》**

**实验报告七 ——综合测试2**

**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： 2021/5/15**

**总评成绩： 审阅教师：**

### 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：

Idea,JUnit,Chorm

### 三、实验要求：

题目1

1. 查看nl.tudelft.jpacman.npc.ghost包下的Clyde类，认真阅读该类的Javadoc描述，理解这类Ghost的行为方式。
2. 查看Clyde类的 Optional nextAiMove()”方法，考虑测试用例以确保Clyde按预期行为工作。
3. 实例化Clyde对象  
   在test模块java/nl/tudelft/jpacman包中的GhostMapParser类，它是提供快速生成地图的辅助工具类。比如，你提供下面这样三行字符串，将自动生成一个3X12大小，且含有一个玩家和一个Clyde幽灵的地图。玩家和Clyde之间间隔8个Square。 {“############”, “ #P C#”, “############”}, GhostMapParser的parseMap()方法接收上述字符串数组，并返回一个Level对象。而这个Level对象管理了Board、Player、Ghost等几乎游戏涉及到的所有模型对象。  
   比如，如果你的测试涉及到Player，需要利用Player的PlayerFactory生成Player对象。然后再调用Level的RegisterPlayer来注册这个新创建的Player对象，并调用Player的setDirection方法来设置一个初始的行走方向。 除了创建Player，通常还需要Ghost。四种Ghost对象在地图解析时，就已经创建好，可以通过调用Navigation的findUnitInBoard()方法来获取.
4. 实现这个ClydeTest测试类时，需要构建游戏运行的所有所需对象。这些对象包括PacManSprites（用于角色显示）、PlayerFactory（用于构造Player）、GhostFactory（提供给LevelFactory），BoardFactory（游戏场景）、GhostMapParser（地图解析）。  
   思考以上对象构造过程，建议利用JUnit5的[@BeforeEach](https://github.com/BeforeEach" \o "@BeforeEach)或[@BeforeAll](https://github.com/BeforeAll" \o "@BeforeAll)来为不同的测试方法创建统一的所需对象集。 在ClydeTest类中至少测试4个情况下的行为； 为你的每个函数和类增加javadoc注解。
5. 为了测试生成Clyde魔鬼，可以增加一个地图解析器，[Clyde魔鬼地图解析器](https://star.jmhui.com.cn/u/cms/www/202205/0915094561cy.rar)

### 实验步骤与内容

1. 在实验六的代码基础上， 完成下面题目1，提交到自己的代码仓库
2. 在完成题目1的基础上完成题目2，题目2选做（学有余力者做）
3. 编写测试报告，提交到雨课堂“软件测试实验七”

## 实验题目

1. **代码实现：**

|  |
| --- |
| **实现代码：**  package nl.tudelft.jpacman.npc.ghost;  import io.cucumber.java.nl.Stel;  import nl.tudelft.jpacman.board.BoardFactory;  import nl.tudelft.jpacman.board.Direction;  import nl.tudelft.jpacman.level.\*;  import nl.tudelft.jpacman.points.PointCalculator;  import nl.tudelft.jpacman.sprite.PacManSprites;  import org.assertj.core.util.Lists;  import org.junit.Test;  import org.junit.jupiter.api.\*;  import java.util.List;  import java.util.Optional;  import static org.assertj.core.api.AssertionsForClassTypes.assertThat;  import static org.mockito.Mockito.mock;  @TestMethodOrder(MethodOrderer.OrderAnnotation.class)  public class ClydeTest {  @Test  @Order(1)  public void departLessThanEight1(){  PacManSprites sprites =new PacManSprites();  LevelFactory levelFactory =new LevelFactory(  sprites,  new GhostFactory(sprites),  mock(PointCalculator.class));  BoardFactory boardFactory = new BoardFactory(sprites);  GhostFactory ghostFactor = new GhostFactory(sprites);  MapParser mapParser = new GhostMapParser(levelFactory, boardFactory, ghostFactor);  List<String> text = Lists.newArrayList(  "##############",  "#.#....C.....P",  "##############"  );  Level level = mapParser.parseMap(text);  Clyde clyde = Navigation.findUnitInBoard(Clyde.class,level.getBoard());  Player player = new PlayerFactory(new PacManSprites()).createPacMan();  player.setDirection(Direction.WEST);  level.registerPlayer(player);  Optional<Direction> opt = clyde.nextAiMove();  assertThat(opt.get()).isEqualTo(Direction.valueOf("WEST"));  }  @Test  @Order(2)  public void departLessThanEight2(){  PacManSprites sprites =new PacManSprites();  LevelFactory levelFactory =new LevelFactory(  sprites,  new GhostFactory(sprites),  mock(PointCalculator.class));  BoardFactory boardFactory = new BoardFactory(sprites);  GhostFactory ghostFactor = new GhostFactory(sprites);  MapParser mapParser = new GhostMapParser(levelFactory, boardFactory, ghostFactor);  List<String> text = Lists.newArrayList(  "#########.....",  "#.#..........C",  "#############P"  );  Level level = mapParser.parseMap(text);  Clyde clyde = Navigation.findUnitInBoard(Clyde.class,level.getBoard());  Player player = new PlayerFactory(new PacManSprites()).createPacMan();  player.setDirection(Direction.WEST);  level.registerPlayer(player);  Optional<Direction> opt = clyde.nextAiMove();  assertThat(opt.get()).isEqualTo(Direction.valueOf("NORTH"));  }  @Test  @Order(3)  public void departLessThanEight3(){  PacManSprites sprites =new PacManSprites();  LevelFactory levelFactory =new LevelFactory(  sprites,  new GhostFactory(sprites),  mock(PointCalculator.class));  BoardFactory boardFactory = new BoardFactory(sprites);  GhostFactory ghostFactor = new GhostFactory(sprites);  MapParser mapParser = new GhostMapParser(levelFactory, boardFactory, ghostFactor);  List<String> text = Lists.newArrayList(  "##############",  "#.#....P..C...",  "##############"  );  Level level = mapParser.parseMap(text);  Clyde clyde = Navigation.findUnitInBoard(Clyde.class,level.getBoard());  Player player = new PlayerFactory(new PacManSprites()).createPacMan();  player.setDirection(Direction.WEST);  level.registerPlayer(player);  Optional<Direction> opt = clyde.nextAiMove();  assertThat(opt.get()).isEqualTo(Direction.valueOf("EAST"));  }  @Test  @DisplayName("Clybe离Player距离小于8个方块")  @Order(4)  public void departLessThanEight4(){  PacManSprites sprites =new PacManSprites();  LevelFactory levelFactory =new LevelFactory(  sprites,  new GhostFactory(sprites),  mock(PointCalculator.class));  BoardFactory boardFactory = new BoardFactory(sprites);  GhostFactory ghostFactor = new GhostFactory(sprites);  MapParser mapParser = new GhostMapParser(levelFactory, boardFactory, ghostFactor);  List<String> text = Lists.newArrayList(  "###P....######",  "###C..........",  "###...########"  );  Level level = mapParser.parseMap(text);  Clyde clyde = Navigation.findUnitInBoard(Clyde.class,level.getBoard());  Player player = new PlayerFactory(new PacManSprites()).createPacMan();  player.setDirection(Direction.EAST);  level.registerPlayer(player);  Optional<Direction> opt = clyde.nextAiMove();  assertThat(opt.get()).isEqualTo(Direction.valueOf("SOUTH"));  }  @Test  @DisplayName("Clybe离Player距离大于8个方块")  @Order(5)  public void departLessThanEight5(){  PacManSprites sprites =new PacManSprites();  LevelFactory levelFactory =new LevelFactory(  sprites,  new GhostFactory(sprites),  mock(PointCalculator.class));  BoardFactory boardFactory = new BoardFactory(sprites);  GhostFactory ghostFactor = new GhostFactory(sprites);  MapParser mapParser = new GhostMapParser(levelFactory, boardFactory, ghostFactor);  List<String> text = Lists.newArrayList(  ".C................",  "#.....#......#..P.",  "##################"  );  Level level = mapParser.parseMap(text);  Clyde clyde = Navigation.findUnitInBoard(Clyde.class,level.getBoard());  Player player = new PlayerFactory(new PacManSprites()).createPacMan();  player.setDirection(Direction.WEST);  level.registerPlayer(player);  Optional<Direction> opt = clyde.nextAiMove();  assertThat(opt.get()).isEqualTo(Direction.valueOf("EAST"));  }  }  **测试结果：** |

### 结论分析与体会

学会了综合测试方法。

### 仓库地址

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