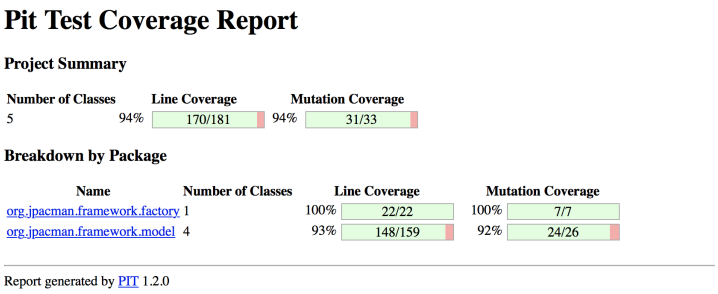
CPEN522 - Group22 - Assignment3

Yonghan Zhou 93506160

March 2020

Exercise 3.1

There are 33 mutations were generated by PIT in total. 31 were killed.



Pit Test Coverage Report

Project Summary

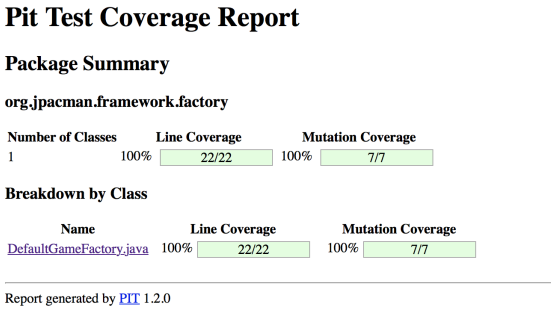
Number of Classes	Line Coverage		Mutation Coverage	
5	94%	170/181	94%	31/33

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.jpacman.framework.factory	1	100%	22/22	100%	7/7
org.jpacman.framework.model	4	93%	148/159	92%	24/26

Report generated by PIT 1.2.0

The mutation coverage score is 100% for `DefaultGameFactory` class.



Pit Test Coverage Report

Package Summary

org.jpacman.framework.factory

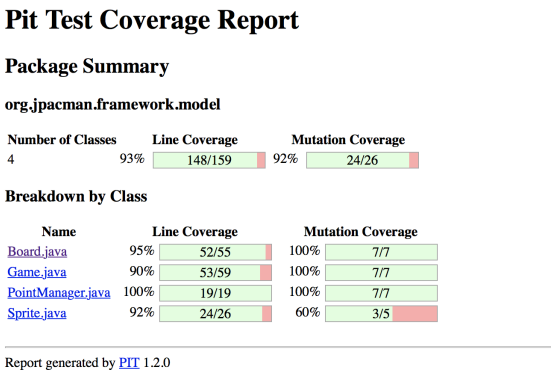
Number of Classes	Line Coverage		Mutation Coverage	
1	100%	22/22	100%	7/7

Breakdown by Class

Name	Line Coverage		Mutation Coverage	
DefaultGameFactory.java	100%	22/22	100%	7/7

Report generated by PIT 1.2.0

The mutation coverage scores for `Board`, `Game`, and `PointManager` are 100%. Class `sprite`'s score is 60%.



Pit Test Coverage Report

Package Summary

org.jpacman.framework.model

Number of Classes	Line Coverage		Mutation Coverage	
4	93%	148/159	92%	24/26

Breakdown by Class

Name	Line Coverage		Mutation Coverage	
Board.java	95%	52/55	100%	7/7
Game.java	90%	53/59	100%	7/7
PointManager.java	100%	19/19	100%	7/7
Sprite.java	92%	24/26	60%	3/5

Report generated by PIT 1.2.0

Exercise 3.2

There are two mutations in `Sprite` class not killed by the test suite. Both are due to NO_COVERAGE.

```
82  1. mutated return of Object value for org/jpacman/framework/model/Sprite::getSpriteType to ( if (x != null) null else throw new RuntimeException ) → NO_COVERAGE
87  1. mutated return of Object value for org/jpacman/framework/model/Sprite::toString to ( if (x != null) null else throw new RuntimeException ) → NO_COVERAGE
```

They are a weakness of test suite. NO_COVERAGE means there were no tests that exercised the line of code where the mutation was created. Actually, even without mutation, these lines in the original program code are not covered by executing test suites. The mutation score of this test suite is 31/33.

Exercise 3.3

In `Sprite` class, there are two method `getSpriteType` and `toString` not covered by the test suite. So two more test cases testing these two methods are needed.

```
128     /**
129      * Test when getting the type of a sprite
130      */
131
132     @Test
133     public void testGetSpriteType() {
134         assertEquals(SpriteType.OTHER, john.getSpriteType());
135     }
136
137     /**
138      * Test when getting the string of the sprite type
139      */
140
141     @Test
142     public void testToString() {
143         assertTrue(john.toString().contains(john.getSpriteType().toString()));
144     }
```

Then we can get a new mutation score of 33/33.

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
5	95%	172/181	100%	33/33

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.jpacman.framework.factory	1	100%	22/22	100%	7/7
org.jpacman.framework.model	4	94%	150/159	100%	26/26

Report generated by PIT 1.2.0

Exercise 3.4

Besides the 7 default mutators, 4 extra mutators iconstructor call, inline constant, remove conditionals, and remove increments) are added. Then there is one new mutation not killed. As a result, the mutation score is 49/50.

```
208                 <mutators>
209                     <mutator>CONDITIONALS_BOUNDARY</mutator>
210                     <mutator>INCREMENTS</mutator>
211                     <mutator>INVERT_NEGS</mutator>
212                     <mutator>MATH</mutator>
213                     <mutator>NEGATE_CONDITIONALS</mutator>
214                     <mutator>RETURN_VALS</mutator>
215                     <mutator>VOID_METHOD_CALLS</mutator>
216
217                     <mutator>CONSTRUCTOR_CALLS</mutator>
218                     <mutator>INLINE_CONSTS</mutator>
219                     <mutator>REMOVE_INCREMENTS</mutator>
220                     <mutator>REMOVE_CONDITIONALS_EQ_IF</mutator>
```

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
5	95%	172/181	98%	49/50

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.jpacman.framework.factory	1	100%	22/22	100%	11/11
org.jpacman.framework.model	4	94%	150/159	97%	38/39

Report generated by PIT 1.2.0

Look into the detailed info. In `Game` class, the return of function `getBoardInspector()` is not covered.

Mutations

55	1. negated conditional → KILLED
58	1. removed call to org/jpacman/framework/model/Game::dieIfGhost → KILLED
62	1. removed call to org/jpacman/framework/model/Game::notifyViewers → KILLED
76	1. removed call to org/jpacman/framework/model/Food::deoccupy → KILLED
95	1. removed conditional - replaced equality check with true → KILLED
98	1. removed call to org/jpacman/framework/model/Player::die → KILLED
114	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
116	1. Substituted 1 with 0 → KILLED
147	1. mutated return of Object value for org/jpacman/framework/model/Game::getBoard to (if (x != null) null else throw new RuntimeException) → KILLED
172	1. removed call to org/jpacman/framework/model/Game::notifyObservers → KILLED
194	1. mutated return of Object value for org/jpacman/framework/model/Game::getBoardInspector to (if (x != null) null else throw new RuntimeException) → NO_COVERAGE

```
192         @Override
193         public IBoardInspector getBoardInspector() {
194 1               return getBoard();
195         }
```

Extend the test suite by adding a new test case to test the `getBoardInspector()`, resulting in a 50/50 mutation score.

```
292     /**
293      * Test the method getBoardInspector()
294      * @throws FactoryException
295      */
296     @Test
297     public void testGetBoardInspector() throws FactoryException {
298         Game g = makePlay("PG#");
299         Board b = g.getBoard();
300         assertEquals(b, g.getBoardInspector());
301     }
```

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
5	96%	174/181	100%	50/50

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.jpacman.framework.factory	1	100%	22/22	100%	11/11
org.jpacman.framework.model	4	96%	152/159	100%	39/39

Report generated by PIT 1.2.0

Exercise 3.5

When running PIT on the whole test suit, the mutation coverage score is only 54%. Especially, the scores of package `controller`, `ui`, and `view` are so low. We extended the test suite, increasing the mutation score by 23% to 77%.

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
23	91%	730/803	54%	74/137

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.jpacman.framework.controller	2	98%	43/44	10%	1/10
org.jpacman.framework.factory	2	89%	74/83	93%	13/14
org.jpacman.framework.model	11	93%	228/245	91%	48/53
org.jpacman.framework.ui	5	86%	237/277	18%	7/39
org.jpacman.framework.view	3	96%	148/154	24%	5/21

Report generated by PIT 1.2.0

Pit Test Coverage Report

Project Summary

Number of Classes	Line Coverage		Mutation Coverage	
23	92%	739/803	77%	105/137

Breakdown by Package

Name	Number of Classes	Line Coverage		Mutation Coverage	
org.jpacman.framework.controller	2	100%	44/44	60%	6/10
org.jpacman.framework.factory	2	92%	76/83	100%	14/14
org.jpacman.framework.model	11	94%	231/245	100%	53/53
org.jpacman.framework.ui	5	87%	240/277	64%	25/39
org.jpacman.framework.view	3	96%	148/154	33%	7/21

Report generated by PIT 1.2.0

(1) For class `AbstractGhostMover`, a test case was added to trigger the `getRandomGhost` method, killing two mutations related to this method.

```
public class AbstractGhostMoverTest {

    //3.5

    @Test
    public void testAbstractGhostMover() throws FactoryException {
        MapParser p = new MapParser(new DefaultGameFactory());
        Game game = p.parseMap(new String[]{"P  ","   ","   "});

        AbstractGhostMover rgm = new RandomGhostMover(game);
        rgm.start();

        try {
            rgm.doTick();
        } catch (Exception e) {
            fail("Unexpected Exception");
        }

        ActionEvent ae = new ActionEvent("", 0, null);

        try {
            rgm.actionPerformed(ae);
        } catch (Exception e) {
            fail("Unexpected Exception");
        }
    }
}
```

Mutations

59	1. removed call to javax/swing/Timer::<init> → KILLED 2. Substituted 40 with 41 → SURVIVED
79	1. removed call to org/jpacman/framework/controller/AbstractGhostMover::doTick → SURVIVED
91	1. removed call to javax/swing/Timer::start → SURVIVED
100	1. removed call to javax/swing/Timer::stop → SURVIVED
111	1. negated conditional → SURVIVED 2. removed conditional - replaced equality check with true → SURVIVED

Mutations

59	1. removed call to javax/swing/Timer::<init> → KILLED 2. Substituted 40 with 41 → SURVIVED
79	1. removed call to org/jpacman/framework/controller/AbstractGhostMover::doTick → SURVIVED
91	1. removed call to javax/swing/Timer::start → SURVIVED
100	1. removed call to javax/swing/Timer::stop → SURVIVED
111	1. negated conditional → KILLED 2. removed conditional - replaced equality check with true → KILLED

(2) For class `RandomGhostMover`, we created a test to randomly move the ghost. It killed three mutations.

```
public class RandomGhostMoverTest {

    //3.5

    @Test
    public void testDotick() throws FactoryException {
        MapParser p = new MapParser(new DefaultGameFactory());
        Game game = p.parseMap(new String[]{"P  "," G ","   "});
        Ghost ghost = game.getGhosts().get(0);
        int x = ghost.getTile().getX();
        int y = ghost.getTile().getY();

        AbstractGhostMover rgm = new RandomGhostMover(game);
        rgm.start();
        rgm.doTick();
        assertTrue(ghost.getTile().getX() != x || ghost.getTile().getY() != y);
    }

}
```

Mutations

30	1. negated conditional → SURVIVED 2. removed conditional - replaced equality check with true → SURVIVED
35	1. removed call to org/jpacman/framework/model/IGameInteractor::moveGhost → SURVIVED

Mutations

30	1. negated conditional → KILLED 2. removed conditional - replaced equality check with true → KILLED
35	1. removed call to org/jpacman/framework/model/IGameInteractor::moveGhost → KILLED

(3) In class `MapParser`, there is no coverage on `invalidSprite`. By adding a test on this method, a NO_COVERAGE mutation was killed.

```java
//3.5

private final String[] invalidMap = new String[] {"A"};

/**
 * Test the invalidSprite() method
 */
@Test
public void testInvalidSprite() {
    try {
        parser.parseMap(invalidMap);
        fail("FactoryException expected");
    } catch (FactoryException e) {
        assertTrue(true);
    }
}
```

Mutations

66	1. negated conditional → KILLED
73	1. Changed increment from 1 to -1 → KILLED
78	1. changed conditional boundary → KILLED
95	1. removed conditional - replaced equality check with true → KILLED
136	1. removed call to org/jpacman/framework/factory/FactoryException::<init> → NO_COVERAGE
162	1. mutated return of Object value for org/jpacman/framework/factory/MapParser::parseFromFile to (if (x != null) null else throw new RuntimeException) → KILLED
190	1. negated conditional → KILLED

Mutations

66	1. negated conditional → KILLED
73	1. Changed increment from 1 to -1 → KILLED
78	1. changed conditional boundary → KILLED
95	1. removed conditional - replaced equality check with true → KILLED
136	1. removed call to org/jpacman/framework/factory/FactoryException::<init> → KILLED
162	1. mutated return of Object value for org/jpacman/framework/factory/MapParser::parseFromFile to (if (x != null) null else throw new RuntimeException) → KILLED
190	1. negated conditional → KILLED

(4) For class `Food`, adding a test case on getting the default points can kill two survived mutations.

```java
public class FootTest {

    //3.5

    private final int DEFAULT_POINTS = 10;

    /**
     * Test default points
     */
    @Test
    public void testGetPoints() {
        Food food = new Food();
        assertEquals(DEFAULT_POINTS, food.getPoints());
        food.setPoints(12);
        assertEquals(12, food.getPoints());
    }


    /**
     * Test the getSpriteType() method
     */
    @Test
    public void testGetSpriteType() {
        Food food = new Food();
        assertEquals(SpriteType.FOOD, food.getSpriteType());
    }
}
```

Mutations

17	1. Substituted 10 with 11 → SURVIVED
23	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → SURVIVED
38	1. mutated return of Object value for org/jpacman/framework/model/Food::getSpriteType to (if (x != null) null else throw new RuntimeException) → KILLED

Mutations

17	1. Substituted 10 with 11 → KILLED
23	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
38	1. mutated return of Object value for org/jpacman/framework/model/Food::getSpriteType to (if (x != null) null else throw new RuntimeException) → KILLED

(5) In class `Game`, a mutation related to `died()` method was survived before. Testing this method can kill this mutation.

```
//3.5


/**
 * Test the died() method
 * @throws FactoryException
 */
@Test
public void testDied() throws FactoryException {
    Game g = makePlay("P#");
    Player p = g.getPlayer();
    assertFalse(g.died());
    p.die();
    assertTrue(g.died());
}

/**
 * Test the getGhosts() method
 * @throws FactoryException
 */
@Test
public void testGhosts() throws FactoryException {
    Game g = makePlay("PG#");
    List<Ghost> ghosts = g.getGhosts();
    assertEquals(1, ghosts.size());
}
```

Mutations

57	1. removed call to org/jpacman/framework/model/Game::eatFood → KILLED
73	1. negated conditional → KILLED
95	1. negated conditional → KILLED
114	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
140	1. removed call to org/jpacman/framework/model/PointManager::addPointsToBoard → KILLED
187	1. removed call to java/util/ArrayList::<init> → KILLED
199	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → SURVIVED

Mutations

57	1. removed call to org/jpacman/framework/model/Game::eatFood → KILLED
73	1. negated conditional → KILLED
95	1. negated conditional → KILLED
114	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
140	1. removed call to org/jpacman/framework/model/PointManager::addPointsToBoard → KILLED
187	1. removed call to java/util/ArrayList::<init> → KILLED
199	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED

(6) By adding a test on `getGame()` in `Level` class, all seven mutations can be killed.

```
public class LevelTest {

    //3.5

    /**
     * Test parseMap() method
     * @throws FactoryException
     */
    @Test
    public void testParseMap() throws FactoryException {
        Level l = new Level();
        IGameInteractor g = l.parseMap();


        IGameFactory factory = new DefaultGameFactory();
        MapParser parser = new MapParser(factory);
        Game g2 = parser.parseFromFile("board.txt");
        assertEquals(g.getGhosts().size(), g2.getGhosts().size());

    }

    /**
     * Test getGame() method
     * @throws FactoryException
     */
    @Test
    public void testGetGame() throws FactoryException {
        Level l = new Level();
        IGameInteractor g = l.parseMap();
        assertEquals(g, l.getGame());
    }


}
```

Mutations

23	1. removed call to org/jpacman/framework/factory/DefaultGameFactory::<init> → KILLED
61	1. removed call to org/jpacman/framework/factory/MapParser::<init> → KILLED
64	1. mutated return of Object value for org/jpacman/framework/model/Level::parseMap to (if (x != null) null else throw new RuntimeException) → KILLED
71	1. mutated return of Object value for org/jpacman/framework/model/Level::getMapFile to (if (x != null) null else throw new RuntimeException) → KILLED
78	1. mutated return of Object value for org/jpacman/framework/model/Level::getGame to (if (x != null) null else throw new RuntimeException) → NO_COVERAGE

Mutations

23	1. removed call to org/jpacman/framework/factory/DefaultGameFactory::<init> → KILLED
61	1. removed call to org/jpacman/framework/factory/MapParser::<init> → KILLED
64	1. mutated return of Object value for org/jpacman/framework/model/Level::parseMap to (if (x != null) null else throw new RuntimeException) → KILLED
71	1. mutated return of Object value for org/jpacman/framework/model/Level::getMapFile to (if (x != null) null else throw new RuntimeException) → KILLED
78	1. mutated return of Object value for org/jpacman/framework/model/Level::getGame to (if (x != null) null else throw new RuntimeException) → KILLED

(7) In class `Player`, a survived mutation related to `getPoint()` can be killed by adding one more test.

```
public class PlayerTest {

    //3.5

    /**
     * Test the addPoints() method
     */
    @Test
    public void testAddPoints() {
        Player player = new Player();
        assertEquals(10, player.addPoints(10));
    }

}
```

Mutations

14	1. Substituted 0 with 1 → KILLED
15	1. Substituted 1 with 0 → KILLED
22	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
40	1. Replaced integer addition with subtraction → KILLED
42	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → SURVIVED
49	1. Substituted 0 with 1 → KILLED
56	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED

Mutations

14	1. Substituted 0 with 1 → KILLED
15	1. Substituted 1 with 0 → KILLED
22	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
40	1. Replaced integer addition with subtraction → KILLED
42	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED
49	1. Substituted 0 with 1 → KILLED
56	1. replaced return of integer sized value with (x == 0 ? 1 : 0) → KILLED

(8) For `ButtonPanel` class, three extra test cases were added, resulting in two more killed mutations.

```java
/**
 * Test the initialize() method
 */
@Test
public void testInitialize() {
    ButtonPanel bp = new ButtonPanel();
    bp.initialize();
    assertEquals(bp.getWidth(), 3 * BUTTON_WIDTH); //three buttons added in initialize()
}


private final int BUTTON_WIDTH = 80;
private final int BUTTON_HEIGHT = 45;
private int buttonCount = 0;
/**
 * Test the setPanelSize() method
 */
@Test
public void testPanelSize() {
    ButtonPanel bp = new ButtonPanel();
    bp.setSize(BUTTON_WIDTH * buttonCount, BUTTON_HEIGHT);
    assertEquals(bp.getWidth(), BUTTON_WIDTH * buttonCount);
    assertEquals(bp.getHeight(), BUTTON_HEIGHT);
}
```

```java
@Test
public void testButtonPanelActions() {
    MainUI ui = new MainUI();
    try {
        ui.initialize();
        ui.start();
        IPacmanInteraction pi = ui.eventHandler();
        pi.start();
        pi.stop();
        pi.start();
        pi.exit();
        assertTrue(true);
    } catch (FactoryException e) {
        fail("Unexpected FactoryException");
    }
}
```

Mutations

70	1. removed call to org/jpacman/framework/ui/ButtonPanel::initializeStartButton → SURVIVED
89	1. Substituted 1 with 0 → SURVIVED
105	1. removed call to org/jpacman/framework/ui/ButtonPanel$1::<init> → SURVIVED
108	1. removed call to org/jpacman/framework/ui/ButtonPanel::start → NO_COVERAGE
123	1. removed call to org/jpacman/framework/ui/ButtonPanel::pause → NO_COVERAGE
133	1. removed call to javax/swing/JButton::<init> → KILLED
137	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::exit → NO_COVERAGE
157	1. removed call to org/jpacman/framework/ui/ButtonPanel::enableStartStop → NO_COVERAGE
181	1. Substituted 1 with 0 → SURVIVED
184	1. Substituted 0 with 1 → SURVIVED

Mutations

70	1. removed call to org/jpacman/framework/ui/ButtonPanel::initializeStartButton → KILLED
89	1. Substituted 1 with 0 → KILLED
105	1. removed call to org/jpacman/framework/ui/ButtonPanel$1::<init> → SURVIVED
108	1. removed call to org/jpacman/framework/ui/ButtonPanel::start → NO_COVERAGE
123	1. removed call to org/jpacman/framework/ui/ButtonPanel::pause → NO_COVERAGE
133	1. removed call to javax/swing/JButton::<init> → KILLED
137	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::exit → NO_COVERAGE
157	1. removed call to org/jpacman/framework/ui/ButtonPanel::enableStartStop → NO_COVERAGE
181	1. Substituted 1 with 0 → SURVIVED
184	1. Substituted 0 with 1 → SURVIVED

(9) To kill more mutations, two test cases on `setGridSize` and `withModelInteraction` were added for `MainUI` class.

```java
/**
 * Test the method setGridSize()
 * @throws FactoryException
 */

@Test
public void testSetGridSize() throws FactoryException {
    MainUI ui = new MainUI();
    ui.initialize();
    ui.start();

    PointsPanel points = new PointsPanel();
    points.initialize(ui.getGame().getPointManager());
    ui.getGame().attach(points);

    BoardView boardView = new BoardView(ui.getGame().getBoardInspector());

    ButtonPanel buttonPanel = new ButtonPanel();
    buttonPanel.initialize();

    assertEquals(ui.getSize().height, points.getHeight() + boardView.windowHeight() + buttonPanel.getHeight());
    assertEquals(ui.getSize().width, Math.max(boardView.windowWidth(), buttonPanel.getWidth()));
}

/**
 * Test the withModelInteractor() method
 */
@Test
public void testWithModelInteractor() {
    MainUI ui = new MainUI();
    PacmanInteraction pi = new PacmanInteraction();
    ui = ui.withModelInteractor(pi);
    assertEquals(pi, ui.eventHandler());
}
```

Mutations

91	1. removed call to org/jpacman/framework/ui/ButtonPanel::initialize → KILLED
113	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::createButtonPanel to (if (x != null) null else throw new RuntimeException) → KILLED
142	1. Replaced integer addition with subtraction → SURVIVED
153	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::createStatusField to (if (x != null) null else throw new RuntimeException) → SURVIVED
193	1. removed call to org/jpacman/framework/controller/RandomGhostMover::<init> → KILLED
217	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::createModel to (if (x != null) null else throw new RuntimeException) → KILLED
287	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::withModelInteractor to (if (x != null) null else throw new RuntimeException) → NO_COVERAGE

Mutations

91	1. removed call to org/jpacman/framework/ui/ButtonPanel::initialize → KILLED
113	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::createButtonPanel to (if (x != null) null else throw new RuntimeException) → KILLED
142	1. Replaced integer addition with subtraction → KILLED
153	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::createStatusField to (if (x != null) null else throw new RuntimeException) → SURVIVED
193	1. removed call to org/jpacman/framework/controller/RandomGhostMover::<init> → KILLED
217	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::createModel to (if (x != null) null else throw new RuntimeException) → KILLED
287	1. mutated return of Object value for org/jpacman/framework/ui/MainUI::withModelInteractor to (if (x != null) null else throw new RuntimeException) → KILLED

(10) For `PacmanInteraction` class, adding a test about `MatchState` killed four more mutations.

```java
public class PacmanInteractionTest {

    //3.5
    @Test
    public void testMatchState() throws FactoryException, InterruptedException {
        MainUI ui = new MainUI();
        ui.initialize();
        ui.start();
        PacmanInteraction pi = ui.eventHandler();

        assertEquals(pi.getCurrentState().toString(),"PAUSING");
        pi.start();
        assertEquals(pi.getCurrentState().toString(),"PLAYING");
        pi.stop();
        assertEquals(pi.getCurrentState().toString(),"PAUSING");
        pi.start();

        Game g = (Game)ui.getGame();
        Player p = g.getPlayer();

        int x = p.getTile().getX();
        int y = p.getTile().getY();

        pi.right();
        pi.up();
        assertEquals(x+1, p.getTile().getX());
        assertEquals(y-1, p.getTile().getY());

        p.die();
        pi.updateState();
        assertEquals(pi.getCurrentState().toString(),"LOST");
        pi.exit();
        assertNotEquals(pi.getCurrentState().toString(),"PAUSING");
    }
}
```

Mutations

36	1. mutated return of Object value for org/jpacman/framework/ui/PacmanInteraction$MatchState::message to (if (x != null) null else throw new RuntimeException) → SURVIVED
73	1. removed conditional - replaced equality check with true → SURVIVED
87	1. removed call to org/jpacman/framework/ui/PacmanInteraction::movePlayer → SURVIVED
112	1. removed call to org/jpacman/framework/model/IGameInteractor::movePlayer → SURVIVED
157	1. removed conditional - replaced equality check with true → KILLED
169	1. mutated return of Object value for org/jpacman/framework/ui/PacmanInteraction::getGame to (if (x != null) null else throw new RuntimeException) → NO_COVERAGE
180	1. removed call to org/jpacman/framework/ui/PacmanInteraction::updateState → SURVIVED
192	1. removed call to org/jpacman/framework/ui/PacmanInteraction::setChanged → SURVIVED

Mutations

36	1. mutated return of Object value for org/jpacman/framework/ui/PacmanInteraction$MatchState::message to (if (x != null) null else throw new RuntimeException) → KILLED
73	1. removed conditional - replaced equality check with true → SURVIVED
87	1. removed call to org/jpacman/framework/ui/PacmanInteraction::movePlayer → KILLED
112	1. removed call to org/jpacman/framework/model/IGameInteractor::movePlayer → KILLED
157	1. removed conditional - replaced equality check with true → KILLED
169	1. mutated return of Object value for org/jpacman/framework/ui/PacmanInteraction::getGame to (if (x != null) null else throw new RuntimeException) → NO_COVERAGE
180	1. removed call to org/jpacman/framework/ui/PacmanInteraction::updateState → KILLED
192	1. removed call to org/jpacman/framework/ui/PacmanInteraction::setChanged → SURVIVED

(11) For class `PacmanKeyListener`, testing current state and position for each key event can kill most of mutations.

```java
@Test
public void testKeyPressed() throws FactoryException, AWTException, InterruptedException {

    MainUI ui = new MainUI();
    ui.initialize(); //call addKeyListener(new PacmanKeyListener(pi));
    ui.start();

    Robot robot = new Robot();


    int[] keycodes = {KeyEvent.VK_S, //start
                  KeyEvent.VK_LEFT, //left
                  KeyEvent.VK_RIGHT, //right
                  KeyEvent.VK_H, //left
                  KeyEvent.VK_UP, //up
                  KeyEvent.VK_DOWN, //down
                  KeyEvent.VK_K, //up
                  KeyEvent.VK_J, //down
                  KeyEvent.VK_L, //right
                  KeyEvent.VK_Q, //stop
                  KeyEvent.VK_S, //start
                  KeyEvent.VK_X //exit
                  };

    try {
        for (int keycode : keycodes) {
            robot.keyPress(keycode);
            robot.keyRelease(keycode);
            robot.delay(200);
        }
    } catch(Throwable e) {
        fail("Unexpected exception in keyListener");
    }



       // Test keyPressed() method in PacmanKeyListener

       Direction[] dirs =  {null,
                            Direction.LEFT,
                            Direction.RIGHT,
                            Direction.LEFT,
                            Direction.UP,
                            Direction.DOWN,
                            Direction.UP,
                            Direction.DOWN,
                            Direction.RIGHT,
                            null,
                            null,
                            null};
       Game g = (Game)ui.getGame();
       Player p = g.getPlayer();

       Button a = new Button("click");
       PacmanKeyListener pi = (PacmanKeyListener)ui.getKeyListeners()[0];


        for (int i = 0; i < keycodes.length; i++) {
            int keycode = keycodes[i];

            KeyEvent e = new KeyEvent(a, 1, 20, 1, keycode,'a');
            pi.keyPressed(e);
            pi.keyReleased(e);

            if (i == 0) { //start
                assertEquals(ui.eventHandler().getCurrentState().toString(),"PLAYING");
            }
            if (i == 9) { //stop
                assertEquals(ui.eventHandler().getCurrentState().toString(),"PAUSING");
            }
            if (i == 10) { //start
                assertEquals(ui.eventHandler().getCurrentState().toString(),"PLAYING");

            }
            if (i == 11) { //exit
                assertEquals(ui.eventHandler().getCurrentState().toString(),"PLAYING");
            }
            if (dirs[i] == null) continue;
            assertEquals(p.getDirection().getDx(), dirs[i].getDx());
            assertEquals(p.getDirection().getDy(), dirs[i].getDy());

            Thread.sleep(200);
        }
    }
```

Mutations

42	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::up → SURVIVED
46	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::down → SURVIVED
50	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::left → SURVIVED
54	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::right → SURVIVED
57	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::stop → SURVIVED
60	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::exit → SURVIVED
63	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::start → SURVIVED

Mutations

42	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::up → KILLED
46	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::down → KILLED
50	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::left → KILLED
54	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::right → KILLED
57	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::stop → KILLED
60	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::exit → SURVIVED
63	1. removed call to org/jpacman/framework/ui/IPacmanInteraction::start → KILLED

(12) A test case on `initialize()` method was added for class `PointsPanel`, killing four more mutations.

```
public class PointsPanelTest {

    //3.5
    private final int PANEL_HEIGHT = 45;
    private final int PANEL_WIDTH = 100;
    private final int EATEN_WIDTH = 7;

    @Test
    public void testInitialize() {

        PointsPanel p = new PointsPanel();
        p.initialize(new PointManager());


        JTextField field = (JTextField) p.getComponents()[1];

        assertEquals(field.getColumns(), EATEN_WIDTH);
        assertEquals(field.getHorizontalAlignment(), SwingConstants.RIGHT);
        assertFalse(field.isEditable());


        assertEquals(p.getHeight(), PANEL_HEIGHT);
        assertEquals(p.getWidth(), PANEL_WIDTH);

        assertEquals(p.getName(),"jpacman.points.panel");

    }

}
```

Mutations

40	1. Substituted 7 with 8 → SURVIVED
41	1. removed call to javax/swing/JTextField::<init> → KILLED
42	1. removed call to javax/swing/JTextField::setEditable → SURVIVED
43	1. removed call to javax/swing/JTextField::setHorizontalAlignment → SURVIVED
49	1. removed call to org/jpacman/framework/ui/PointsPanel::setName → SURVIVED
50	1. Substituted 45 with 46 → SURVIVED
51	1. removed call to org/jpacman/framework/ui/PointsPanel::displayPoints → SURVIVED

Mutations

40	1. Substituted 7 with 8 → SURVIVED
41	1. removed call to javax/swing/JTextField::<init> → KILLED
42	1. removed call to javax/swing/JTextField::setEditable → KILLED
43	1. removed call to javax/swing/JTextField::setHorizontalAlignment → KILLED
49	1. removed call to org/jpacman/framework/ui/PointsPanel::setName → KILLED
50	1. Substituted 45 with 46 → KILLED
51	1. removed call to org/jpacman/framework/ui/PointsPanel::displayPoints → SURVIVED

(13) For class `BoardView`, we added a test case on `getWindowHeight()` to kill one more mutation.

```
public class BoardViewTest {

    //3.5

    private static final int CELL_HEIGHT = 20;
    public static final int CELL_VGAP = 1;

    @Test
    public void testGetWindowHeight() throws FactoryException {
        Board board = new Board(10,10);
        BoardView bv = new BoardView(board);

        assertEquals((CELL_HEIGHT + CELL_VGAP) * (board.getHeight() + 1),bv.windowHeight());
    }
}
```

Mutations

115	1. Replaced integer addition with subtraction → SURVIVED
152	1. Substituted 5.0 with 1.0 → SURVIVED
165	1. Substituted 2 with 3 → SURVIVED
176	1. removed call to java/awt/Graphics2D::setColor → SURVIVED
196	1. Replaced integer subtraction with addition → SURVIVED
229	1. mutated return of Object value for org/jpacman/framework/view/BoardView::spriteColor to (if (x != null) null else throw new RuntimeException) → SURVIVED
259	1. Replaced integer modulus with multiplication → SURVIVED

Mutations

115	1. Replaced integer addition with subtraction → KILLED
152	1. Substituted 5.0 with 1.0 → SURVIVED
165	1. Substituted 2 with 3 → SURVIVED
176	1. removed call to java/awt/Graphics2D::setColor → SURVIVED
196	1. Replaced integer subtraction with addition → SURVIVED
229	1. mutated return of Object value for org/jpacman/framework/view/BoardView::spriteColor to (if (x != null) null else throw new RuntimeException) → SURVIVED
259	1. Replaced integer modulus with multiplication → SURVIVED