Sam Lee

CS 472

Task 2

GhostFactoryTest.java

The class coverage before any changes with just the isAlive() test added was at a total 16%. The method coverage of the jpacman was at a 9%. The first testing method I added was creating the ghost factory and having it create all of the ghosts.

```
public class GhostFactoryTest {
    private static final PacManSprites SPRITE_STORE = new PacManSprites();
    private GhostFactory GF = new GhostFactory(SPRITE_STORE);

    @Test
    void creatingGhost() {
        GF.createBlinky();
        GF.createPinky();
        GF.createInky();
        GF.createClyde();
    }
}
```

This increased the class coverage from 16% to 29%, while method coverage went from 9% to 14%.

LevelFactorytest.java

The next unit testing I added was the level factory test. This test would generate a level and start the level. With the addition of this unit test the class coverage went from 29% to 36%. The method coverage also went up from 14% to 18%.

```
public class LevelFactorytest {

    private Level lv = new Level(mock(Board.class), Lists.newArrayList(mock(Ghost.class)), Lists.newArrayList(mock(Square.class), mock(Square.class)), mock(CollisionMap.class));

    @Test
    void LVFTest() {
        lv.start();
    }
}
```

Gameisrunning.java

The final unit test I added was a test that would test the launcher and see if the isInProgress function. This unit test raised the class coverage from 36% to 74%, and the method coverage from 18% to 61%.

```
public class gameisrunning {

    private Launcher launcher = new Launcher();
    private Game getGame(){
        return launcher.getGame();
    }
    @Test
    void Running() {
        launcher.launch();
        getGame().start();
        assertThat(getGame().isInProgress()).isEqualTo(true);
    }
    }
```

Task 3

Are the coverage results from JaCoCo similar to the ones you got from IntelliJ in the last task? Why so or why not? It not quite the same compared to what I have in IntelliJ. In JaCoCo report he has a much higher and better coverage compared to what I have.

Did you find helpful the source code visualization from JaCoCo on uncovered branches?

Yea, I find it quite helpful after looking at it more than when I first saw it. It was a bit confusing to see the red on the left hand side of bar instead of the right hand side. It made it seem like the red bar is the coverage percentage when it is actually the green bar.

Which visualization did you prefer and why? IntelliJ’s coverage window or JaCoCo’s report?

I would prefer the visualization of the IntelliJ’s coverage window more compared to JaCoCo’s report. The main reasoning being that it show the percentage for class, method, and line, while on JaCoCo’s report it only show the percentage for instructions and branches. I would say that if IntelliJ had a way to show branch coverage it would be a lot better then what it is right now. url to my repository https://github.com/drkskies/jpacman