

CSC410, Fall 2018-Assignment 1

Name: Yuxin Li

Student Number:1002639567

Lecture: Thursday

Name: Xiawei Zhang

Student Number:1002137957

Lecture: Thursday

We are the sole authors of this homework.

Signatures: Yuxin Li, Xiawei Zhang

Question 1:

Scenario 1:

jpacman-framework

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed
nl.tudelft.jpacman.level	<div><div></div><div></div></div>	68%	<div><div></div><div></div></div>	54%	67	149	86	321	19	66	4
nl.tudelft.jpacman.board	<div><div></div><div></div></div>	65%	<div><div></div><div></div></div>	29%	52	93	12	110	4	40	0
nl.tudelft.jpacman.ui	<div><div></div><div></div></div>	78%	<div><div></div><div></div></div>	37%	50	86	12	144	4	31	0
nl.tudelft.jpacman.npc.ghost	<div><div></div><div></div></div>	82%	<div><div></div><div></div></div>	55%	41	89	14	145	2	30	0
nl.tudelft.jpacman.sprite	<div><div></div><div></div></div>	86%	<div><div></div><div></div></div>	56%	25	70	7	113	4	38	0
nl.tudelft.jpacman	<div><div></div><div></div></div>	77%	<div><div></div><div></div></div>	33%	9	29	10	51	3	23	1
nl.tudelft.jpacman.game	<div><div></div><div></div></div>	84%	<div><div></div><div></div></div>	50%	10	24	4	43	2	14	0
nl.tudelft.jpacman.npc	<div><div></div><div></div></div>	97%	<div><div></div><div></div></div>	83%	1	8	1	17	0	5	0
Total	1,029 of 4,313	76%	315 of 595	47%	255	548	146	944	38	247	5

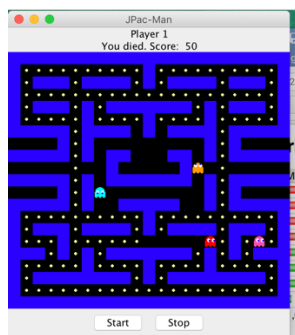
Created with JaCoCo 0.8.1.201803210924

nl.tudelft.jpacman.level

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed
CollisionInteractionMap	<div><div></div><div></div></div>	0%	<div><div></div><div></div></div>	0%	19	19	46	46	7
Level	<div><div></div><div></div></div>	82%	<div><div></div><div></div></div>	57%	25	55	6	105	1
MapParser	<div><div></div><div></div></div>	83%	<div><div></div><div></div></div>	78%	7	26	7	69	1
DefaultPlayerInteractionMap	<div><div></div><div></div></div>	0%	<div><div></div><div></div></div>	n/a	5	5	12	12	5
PlayerCollisions	<div><div></div><div></div></div>	77%	<div><div></div><div></div></div>	64%	4	14	5	24	1
CollisionInteractionMap.InverseCollisionHandler	<div><div></div><div></div></div>	0%	<div><div></div><div></div></div>	n/a	2	2	5	5	2
LevelFactory	<div><div></div><div></div></div>	88%	<div><div></div><div></div></div>	80%	1	8	1	16	0
LevelFactory.RandomGhost	<div><div></div><div></div></div>	0%	<div><div></div><div></div></div>	n/a	2	2	3	3	2
Player	<div><div></div><div></div></div>	93%	<div><div></div><div></div></div>	66%	2	9	1	20	0
Level.NpcMoveTask	<div><div></div><div></div></div>	100%	<div><div></div><div></div></div>	100%	0	3	0	10	0
PlayerFactory	<div><div></div><div></div></div>	100%	<div><div></div><div></div></div>	n/a	0	3	0	5	0
Pellet	<div><div></div><div></div></div>	100%	<div><div></div><div></div></div>	n/a	0	3	0	6	0
Total	402 of 1,295	68%	72 of 159	54%	67	149	86	321	19

nl.tudelft.jpacman.npc.ghost

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
Navigation	<div><div></div><div></div></div>	81%	<div><div></div><div></div></div>	70%	9	28	6	49	1	6	0	1
Inky	<div><div></div><div></div></div>	81%	<div><div></div><div></div></div>	50%	9	15	2	23	0	4	0	1
Clyde	<div><div></div><div></div></div>	80%	<div><div></div><div></div></div>	44%	8	12	2	22	0	3	0	1
Blinky	<div><div></div><div></div></div>	69%	<div><div></div><div></div></div>	37%	8	11	2	13	0	3	0	1
Pinky	<div><div></div><div></div></div>	73%	<div><div></div><div></div></div>	50%	6	11	1	13	0	3	0	1
Navigation.Node	<div><div></div><div></div></div>	92%	<div><div></div><div></div></div>	100%	1	6	1	13	1	5	0	1
GhostColor	<div><div></div><div></div></div>	100%	<div><div></div><div></div></div>	n/a	0	1	0	5	0	1	0	1
GhostFactory	<div><div></div><div></div></div>	100%	<div><div></div><div></div></div>	n/a	0	5	0	7	0	5	0	1
Total	121 of 687	82%	52 of 118	55%	41	89	14	145	2	30	0	8



















Scenario 1: This test scenario is achieved by going right and stopped until eating 5 dots. And waited to be killed by the ghost.

Under these jpacman packages, the package of npc ,game,sprite and npc.ghost have relative high Instruction cov rate. They are 97%,84%,86%,82% correspondingly. However, the three package that has lowest instruction cov rate are jpacman,board,level. Their instruction cov rate are 77%,65%,68% respectively. We notice that under the level package, the class of CollisionInteractionMap, DefaultPlayerInteraction Map and LevelFactory.RandomGhost all has 0% Instruction cov rate. These classes do not get cover by Junit test at all.Additionlly, CollisionInteractionMap,Level,PlayerCollisions has pretty low branches cov rate, which are 0%,57%,64%. In npc.ghost package, the class of Navigation,Inky,Clyde,Navigation.Node and GhostColor,GhostFactory has pretty high instruction cov rate. They are 81%,81%,80%,92%,100%,100% correspondingly. Especially GhostColor and GhostFatcory class both have 100% of the coverage rate since the all the cases has been covered.

Scenario 2:

Scenario that Pacman eating no dots and waiting for being killed by ghost

jpacman-framework

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed Cxty	Missed Lines	Missed Methods	Missed Classes
nl.tudelft.jpacman.level		68%		54%	67 149	86 321	19 66	4 12
nl.tudelft.jpacman.board		65%		29%	52 93	12 110	4 40	0 7
nl.tudelft.jpacman.ui		76%		34%	53 86	18 144	6 31	0 6
nl.tudelft.jpacman.npc.ghost		82%		55%	42 89	14 145	2 30	0 8
nl.tudelft.jpacman.sprite		85%		51%	29 70	10 113	5 38	0 5
nl.tudelft.jpacman		67%		16%	11 29	16 51	5 23	1 2
nl.tudelft.jpacman.game		84%		50%	10 24	4 43	2 14	0 3
nl.tudelft.jpacman.npc		97%		83%	1 8	1 17	0 5	0 1
Total	1,075 of 4,313	75%	324 of 595	45%	265 548	161 944	43 247	5 44

In this Scenario, the lowest three coverage rate packages are level; board and jpacman individually. Jpacman.level package only has 68% instructions Cov and 54%

branches Cov. In `jpacman.level` package, `LevelFactory.RamdomGhost`; `CollisionInteractionMap.InverseCollisionHandler`; `DefaultPlayerInteractionMap` and `CollisionInteractionMap` classes only have 0% instruction Cov since those classes don't have Junit tests. Besides, `jpacman` package only has 67% instructions Cov and 16% branches Cov. In `jpacman` package, `PacmanConfigurationException` class contains no test cases so it has Coverage of zero. `Launcher` class handles only 70% cases for instructions and 16% for the branches. In addition, `Jpacman.board` has lowest instructions Cov rate which is only 65%. In `Jpacman.board` package, there are four classes, which do not contain full instructions coverage, such as `Board`; `Square`; `Unit` and `BoardFactory`. Their instruction coverage rates are 31%; 50% 62% and 93% respectively. On the other hand, `npc.ghost`; `sprite`; `game` and `npc` packages contain very high coverage rate which is greater than 80% for instructions.

Scenario 3:

jpacman-framework

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
nl.tudelft.jpacman.level		68%		53%	69	149	88	321	19	66	4	12
nl.tudelft.jpacman.board		65%		29%	52	93	12	110	4	40	0	7
nl.tudelft.jpacman.ui		75%		34%	54	86	21	144	7	31	0	6
nl.tudelft.jpacman.npc.ghost		82%		54%	43	89	15	145	2	30	0	8
nl.tudelft.jpacman.sprite		84%		48%	30	70	11	113	5	38	0	5
nl.tudelft.jpacman		65%		16%	12	29	18	51	6	23	1	2
nl.tudelft.jpacman.game		84%		50%	10	24	4	43	2	14	0	3
nl.tudelft.jpacman.npc		97%		83%	1	8	1	17	0	5	0	1
Total	1,105 of 4,313	74%	329 of 595	44%	271	548	170	944	45	247	5	44

nl.tudelft.jpacman

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
Launcher		68%		16%	10	27	14	47	4	21	0	1
PacmanConfigurationException		0%		n/a	2	2	4	4	2	2	1	1
Total	76 of 219	65%	10 of 12	16%	12	29	18	51	6	23	1	2

In this scenario 3, We get this scenario by going left and eating 4 dots and hitting the stop button. Then restarts the game and wait the pacman to be killed by the ghost.

The lowest three branches cov rate in this scenario are ui,jpacman,sprite package, they have 16%,34%,48% branches cov rate correspondingly.The scenario 3 has similar instruction cov rate with scenario 1, the difference between them is one uses stop button. Under package of jpacman, the class Launch has the branch cov rate 16%, instruction cov rate 68% in scenario 3 and have 33% branch cov rate,instruction cov rate 80% in scenario 1. It means the more test case gets uncovered after using the keystrokes in the Laucher class. Additionally, PacmanConfigurationException class contains no test cases so it has Coverage of zero.

(2) In Jpacman.level package, LevelFactory.RamdomGhost; CollisionInteractionMap.InverseCollisionHandler and DefaultPlayerInteractionMap classes only have 0% instruction Cov since those classes don't have Junit tests. For improvement, we can add more test cases for those classes. For instance; we could use Assertion in those classes to check if the output for methods is correct.

(3) After changing Assertion to true, jpacman.level Cov. increases to 70% and jacman.ui; ghost and game increase to 81%; 87% and 88% correspondingly. We noticed that most packages instructions Cov rate increase. The reason is because we set assertion to be true. It allows more test cases to be checked, we can get more test coverage by using assertion.

Question2:

(1): We choose Random Testing Strategy and Partition Strategy for testing the Class of DefaultPlayerInteractionMap and the Class of CollisionInteractionMap.

(2): The Strategy used in DefaultPlayerInteractionMap is random Testing; for instance, we test when the Pacman ate a pellet, it increased by 10 points. We tested this increasing points behavior by using random Testing method called testCollideWithPelletWithPoints() in

DefaultPlayerInteractionMapTest Class. Adding this test case increases the instructions Coverage because it checks the collision of Pacman and a pellet. Therefore, the instructions coverage in DefaultPlayerInteractionMapTest increases to 100% and it had only 0 % before.

(3): The Strategy used in CollisionInteractionMap is Partition Testing; for instance, we test the situation that the Pacman had a collision with ghost by the method called testGhostCollideWithGhost(), which is the test for invalid Partition. Besides, we test the situation that the Ghost had a collision with Pellet by the method called testPelletCollideWithGhost(), which is also the test for invalid Partition. etc. By testing those, we increase instruction coverage and branch coverage because the original test cases did not contain those test cases. After we handle those different cases, the instructions coverage increases from 0 % to 96% in total.

Question3:

(1): Firstly, freeze/unfreeze button is created by adding a method called addFreezeButton which will call makeFreeze method in Game Class. Then we will create makeFreeze method in Game Class, which will call freeze() method in level class. Inside freeze(), we call stopNPCs() and set freezeStatus to True if we have false

for freezeStatus. If we call freeze() method with True for freezeStatus, then we will call startNPCs() and set isfreezeStatus to False. There are few test cases made inside LevelTest Class by using White Box testing strategy.

(2): The test case we made was able to capture a bug for our implementation. At beginning of our implementation, we made the isInProgress, this Boolean, to false after we press Freeze/UnFreeze button. However, everything was stoped including score board, npcs etc. Then the test case we made just checked this bug and we fix this bug by adding another Boolean value called isFreezeStatus. When we freeze the game, we made isFreezeStatus to ture and all npcs stops but player is still able to move.

(3)

A Standard Organization of an Analysis and Test Plan

Analysis and test items:

Test whether the Freeze/Unfreeze button works for all the Non-player Characters (NPCs). Specifically, test whether pacman still can move and ghost cannot after clicking the freeze button

Features to be tested:

Test whether Freeze button can make all NPCs in freeze status and test whether freeze/unfreeze button will not freeze pacman.

Features not to be tested:

Whether the speed of ghosts change after freeze and unfreeze action.

Approach:

Using branching coverage of White box strategy.

In our test case, we called freeze function to test the functionality.

We have two branches in function freeze, these two branches depend the Boolean value of freezeStatus, either true or false. It satisfies the condition of the branching coverage.

Pass/Fail criteria:

All the test cases in level.test have passed, the branch coverage rate and instruction coverage rate increase after adding these new tests.