COMP 3111 G55 Final Report

GitHub Link: https://github.com/timlaw0919/Comp3111F23G55

YouTube Link: https://www.youtube.com/watch?v=8G_UUu0ZvGs

Student Information:

Law Hui Nok (20860535)

Lo Wing Yan (20855700)

Lam Hoi Yi (20858647)

Readme

1. All the documentation files are available on GitHub too.

211 First Meeting Minutes

Meeting Minutes #1

COMP3111: Introduction to Software Engineering

Minutes of the 1st Project Meeting

Date: 2023/10/06 (Friday)

Time: 16:00

Venue: Group Study Rooms/ LG1-327

Attending: Law Hui Nok (Tim) , Lam Hoi Yi (Kelly) , Lo Wing Yan (Kelly)

Absent: N/A

Recorder: Lo Wing Yan (Kelly)

1. **Discussion of building Data Modeling**
   1.1. We clarify and unify the understanding on Game rule of "Tom and Jerry in Maze Game" and the related Class Diagram (Details please refer to Appendix).
   1.2. We do not have a clear understanding about some of the details of Class Diagram and Use Case Specification. We will make an appointment with the TA in the comping week to address the confusion.

2. **Goals for the coming week**

Name	Tasks that will be worked on in the coming week
Law Hui Nok	Complete the Class Diagram and Use Case specification for Function B
Lam Hoi Yi	Complete the Class Diagram and Use Case specification for Function C
Lo Wing Yan	Complete the Class Diagram and Use Case specification for Function A

3. **Meeting adjournment and next meeting**

The meeting was adjourned at 19:00. The next meeting is not scheduled.

211 First Meeting Minutes

Meeting Minutes #1

Appendix 1 [Game Rule]

Game Rule:

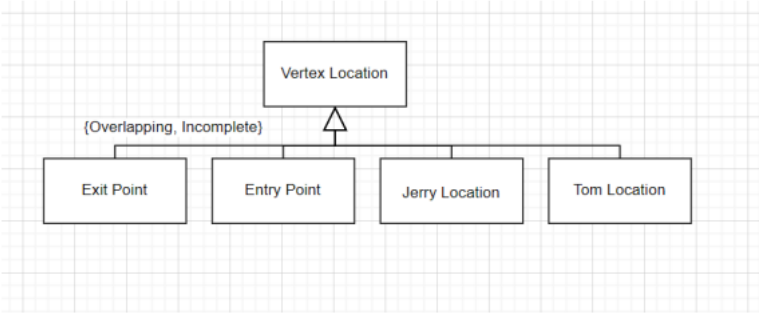
Player Perspective:	[Mouse: Jerry]
Wins if reach the Exit; loses if being caught by Computer before reaching the Exit	
Computer Perspective:	[Cat: Tom]
Always calculating the SHORTEST PATH between Tom and Jerry so as to catch Player	

211 First Meeting Minutes

Meeting Minutes #1

Appendix 2 [Class Diagram]

Class Diagram:



<table>
<tr><td>[Generalization]

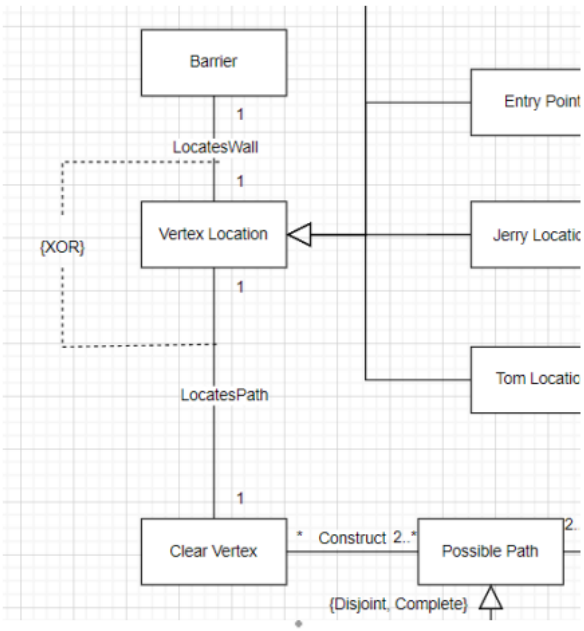
<Vertex Location>
<Exit Point>
<Entry Point>
<Tom Location>
<Jerry Location></td><td><Vertex Location> = [Row, Column]:
<Exit Point>, <Entry Point>, <Tom Location>, <Jerry Location>

<u>? Why not part of</u>
- <Vertex Location> = <Exit Point> || <Vertex Location> = <Tom Location> etc.
- But NOT <Vertex Location> (MUST BE) the CONTAINER of those Location</td></tr>
<tr><td>{overlapping}</td><td>Overlapping = representing different status/progress of the game

<u>Overlap sample cases</u>
<table>
<tr><td><Tom Location> + <Exit Point>
<Jerry Location> + <Entry Point></td><td></td><td>Game START status</td></tr>
<tr><td><Jerry Location> + <Exit Point></td><td>Player WIN the game</td><td rowspan="2">Game END status</td></tr>
<tr><td><Tom Location> + <Jerry Location></td><td>Player LOSE the game
I.e. Tom catches Jerry</td></tr>
<tr><td><Exit Point> + <Entry Point></td><td></td><td>Game START
= Game END
p.s. player can customize the maze = possible</td></tr>
</table></td></tr>
<tr><td>{incomplete}</td><td>There are other vertex locations for the path or barrier</td></tr>
</table>

211 First Meeting Minutes

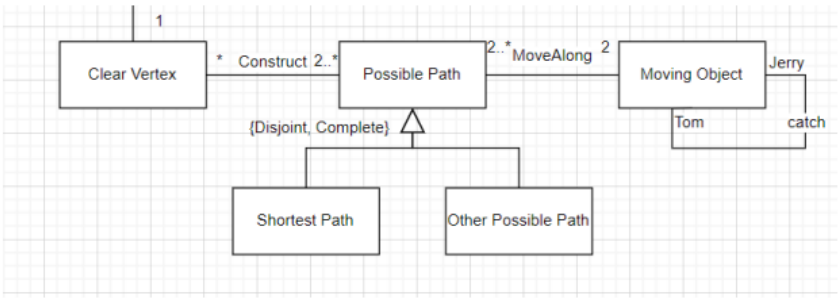
Meeting Minutes #1



[Association] <Vertex Location> <Clear Vertex> <Barrier> <Possible Path>	Relationship between Classes "The <Possible Path> (line) is *CONSTRUCTED* by <Clear Vertex> (white box) which is *LOCATED* by the <Vertex Location>(address)" Or "The <Barrier> (grey box) which is *LOCATED* by the <Vertex Location>(address)"
{XOR}	<Vertex Location> can either *LOCATE <Barrier> or <Clear Vertex>*
[multiplicity] <Vertex Location> <Clear Vertex> <Barrier>	1 <Vertex Location> can either locate 1* <Clear Vertex> or 1* <Barrier> (*marked XOR) Or 1 <Barrier> can only have 1 <Vertex Location> 1 <Clear Vertex> can only have 1 <Vertex Location>

211 First Meeting Minutes

Meeting Minutes #1



[multiplicity] <Clear Vertex> <Possible Path> <Moving Object>	(Requirement) (There must be at least 2 <Possible Path>) with constructed by * <Clear Vertex> Or (2 <Moving Object> move along 2 or * <Possible Path>)

? Relationship between <Moving Object> and <Vertex Location>

Assumption: <Moving Object> moving along the <Possible Path> = they are part of the Path

e.g. Tom/Jerry = the 2 ends of the path

- <Possible Path> is constructed by <Clear Vertex> which location is marked by <Location Vertex>
- = <Moving Object> location is marked by <Location Vertex>

211 Second Meeting Minutes

Meetings Minutes #2

COMP3111: Introduction to Software Engineering

Minutes of the 2nd Project Meeting

Date: 2023/10/13 (Friday)

Time: 17:00

Venue: Learning Commons/ LG1

Attending: Law Hui Nok (Tim) , Lam Hoi Yi (Kelly) , Lo Wing Yan (Kelly)

Absent: N/A

Recorder: Lo Wing Yan (Kelly), Law Hui Nok (Tim)

1. **Report on process during the previous week**

Name	Tasks worked on in the previous week
Law Hui Nok	Completed the Class Diagram and Use Case specification for Function B (1st Draft)
Lam Hoi Yi	Completed the Class Diagram and Use Case specification for Function C (1st Draft)
Lo Wing Yan	Completed the Class Diagram and Use Case specification for Function A (1st Draft)

2. **Discussion of building Data Modeling**
   2.1. Tim clarified the confusion regarding building Data Modeling after the discussion with TA on 2023/10/12 (Thursday) (Details please refer to Appendix).

3. **Goals for the coming weekend**

Name	Tasks that will be worked on in the coming weekend
Law Hui Nok	Modify the Class Diagram and Use Case specification for Function B
Lam Hoi Yi	Modify the Use Case specification for Function C
Lo Wing Yan	Modify the Use Case specification for Function A

4. **Meeting adjournment and next meeting**

   The meeting was adjourned at 19:00. The next meeting is not scheduled.

211 Second Meeting Minutes

Meetings Minutes #2

Appendix [Note for discussion with TA]

Note for Class Diagram:

The changes & explanations

Q: Tom/Jerry/Crystal Location is an object or coordinate?

A: Location = object --> belong to moving object

Q: Entry/Exit Point is a kind of vertex location?

A: No, it can only be on top of clear vertex --> associate + XOR

Q: The definition of Possible Path?

A: One clear vertex can also be treated as possible path even if one clear vertex which is surrounded and trapped by barriers will also be treated as a path. --> Multiplicity changes to 1 .. *

About the multiplicity between possible path and moving object:

1. One moving object can have at least one possible path since it must stand on one clear vertex(can belong to one path or more that one path)

2. One possible path have one moving object. {(0,0), (0,1), (0,2)} is a path starts from (0,0), {(0,2), (0,1), (0,0)} is treated as different since the path is started at (0,2). So, one possible path can only have one moving object, otherwise the game will terminate.

211 Second Meeting Minutes

Meetings Minutes #2

Notes for use case specification:

1. It is better for us to add the diagram of each part of the use case (copy the diagram from requirement note)

2. The different between assumption, precondition and postcondition:

Assumption: Some situations that need to be considered during the entire function call (conditions that should be existed before & during & after the function)

Precondition: The conditions require so that the function can be called. After the function called, the condition can be changes of does not exist anymore.

Postcondition: The conditions require when the function return

--> Some assumption & pre & post condition need to be changed belong to the precise definition.

3. For use case 1, Option 1 or Option 2 is decided by ourselves. The player is not able to choose.

4. Use case 1, CSV file generation is required in specification.

5. For se case 2, Actor = player, the basic flow structure is liked

```
void funB(){
    if (game start){
        Use BFS algo;

        explore possible path;

        Tom follow the shortest path to catch Jerry;
    }
}
```

-->The flow need to expand more based on the above

6. The use case is how the system behave --> in use case = need to be implemented. The features and behaviours can be decided freely

211 Third Meeting Minutes

Meetings Minutes #3

COMP3111: Introduction to Software Engineering

Minutes of the 3rd Project Meeting

Date: 2023/11/13 (Monday)

Time: 16:00

Venue: Café de Coral (Choi Hung)

Attending: Law Hui Nok (Tim) , Lam Hoi Yi (Kelly) , Lo Wing Yan (Kelly)

Absent: N/A

Recorder: Lo Wing Yan (Kelly)

1. **Report on process during the previous week**

Name	Tasks worked on in the previous week
Law Hui Nok	Completed implementation of Function B and Big Main (Menu for FunctionA/B/C), Burndown Chart
Lam Hoi Yi	Completed implementation of Function C
Lo Wing Yan	Completed implementation of Function A, Gantt Chart

2. **Discussion on finalizing the Project**
   - 2.1. Discussion on new features for optimize player experience
     - 2.1.1. Kelly Lam suggested allowing customization: player to adjust the maze size and speed of Tom.
     - 2.1.2. Kelly Lo suggested enhancing the "Tom and Jerry" theme elements: add Tom and Jerry Theme Song when the game starts and utilize the official theme color for UI.
     - 2.1.3. Kelly Lam agreed on that and suggested changing the appearance of the cells on maze to be more related to the theme.

3. **Discussion on Unit Testing**
   - 3.1. Tim pointed out that we need one test per method and the maximum number of methods is 80. Our project exceeded the limit so refactor is needed before start writing the Test case.
   - 3.2. We are confused with doing Unit Testing on GUI and CSV output. We signed up for the consultation with TA on 2023/11/15 (Wednesday).

4. **Discussion on Video format**
   - 4.1. The graphical elements of the video will be created by PowerPoint (PPT).

211 Third Meeting Minutes

Meetings Minutes #3

Name	Responsible Section of the PPT/video
Law Hui Nok	Explanation of A-star algorithm in Function B
Lam Hoi Yi	Innovative Idea and customized Features, Demo of the Game
Lo Wing Yan	Explanation of Depth-First-Search algorithm in Function A

5. Goals for coming week

Name	Tasks that will be worked on in the coming week
Law Hui Nok	JavaDoc comment and Unit Testing for Function B, Unit Testing for Big Main, Screenshot of Application Software, Git commit log, PPT for video
Lam Hoi Yi	New features [2], JavaDoc comment and Unit Testing for Function C
Lo Wing Yan	JavaDoc comment and Unit Testing for Function A, Documentation with JavaDoc, PPT for video

6. Meeting adjournment and next meeting

The meeting was adjourned at 18:30. The next meeting is not scheduled.

211 Forth Meeting Minutes

Meetings Minutes #4

COMP3111: Introduction to Software Engineering

Minutes of the 4th Project Mini-Meeting

Date: 2023/11/15 (Wednesday)

Time: 16:00

Venue: Classroom/ Rm2407

Attending: Law Hui Nok (Tim) , Lam Hoi Yi (Kelly) , Lo Wing Yan (Kelly)

Absent: N/A

Recorder: Lo Wing Yan (Kelly)

1. **Update on Big Main (Menu for FunctionA/B/C)**
   1.1. Tim suggested changing the implementation of Big Main form using terminal to using GUI due to inconvenience.

2. **Update on Unit Testing with note from TA**
   2.1. CSV code with its output should be tested
   2.2. All GUI output should be tested (can use external library "Fest")

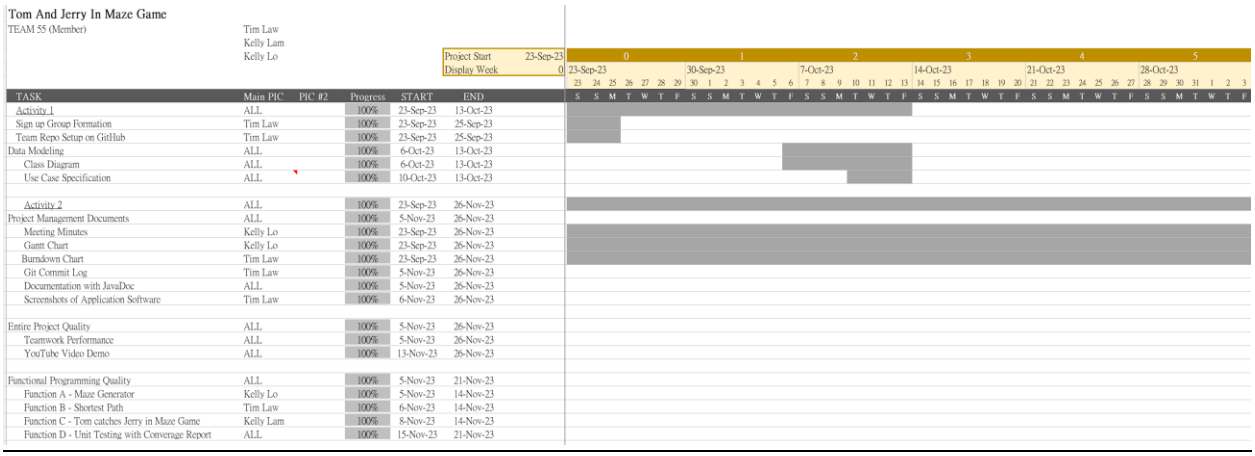
3. **Update goals for coming week from Minute 3**

Name	Tasks that will be worked on in the coming week
Law Hui Nok	GUI of Big Main, JavaDoc comment and Unit Testing for Function B, Unit Testing for Big Main, Screenshot of Application Software, Git commit log, PPT for video
Lam Hoi Yi	New features [2], JavaDoc comment and Unit Testing for Function C
Lo Wing Yan	JavaDoc comment and Unit Testing for Function A, Documentation with JavaDoc, PPT for video

4. **Meeting adjournment and next meeting**

   The meeting was adjourned at 16:20. The next meeting is not scheduled.

212 Gantt Chart

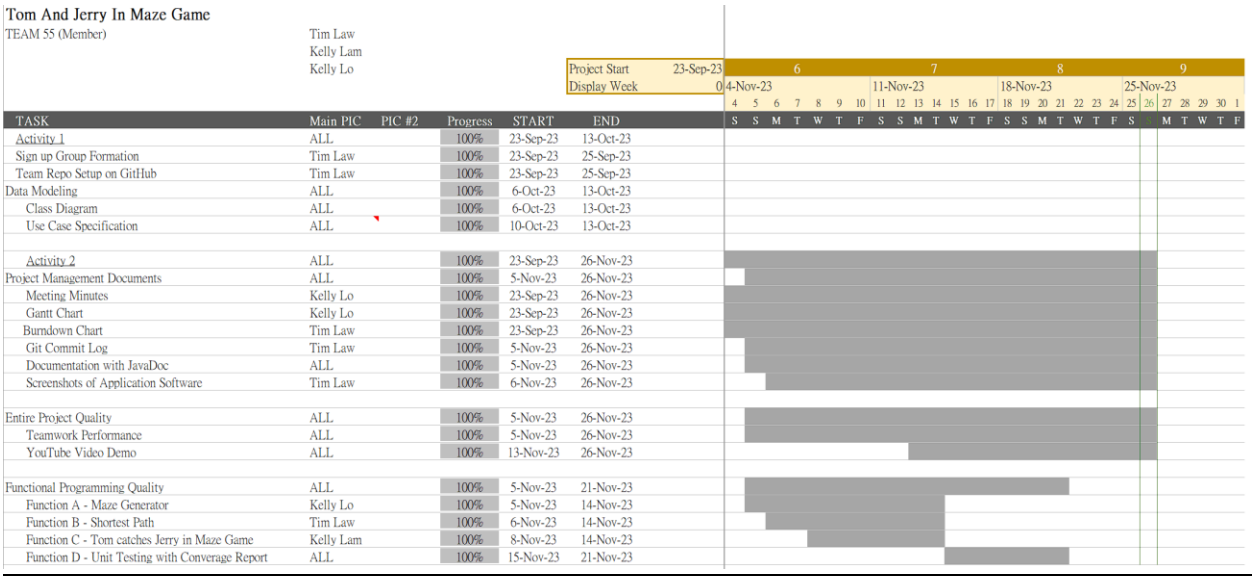


Tom And Jerry In Maze Game

TEAM 55 (Member): Tim Law, Kelly Lam, Kelly Lo

Project Start 23-Sep-23 | Display Week 0

TASK	Main PIC	PIC #2	Progress	START	END
Activity 1	ALL		100%	23-Sep-23	13-Oct-23
Sign up Group Formation	Tim Law		100%	23-Sep-23	25-Sep-23
Team Repo Setup on GitHub	Tim Law		100%	23-Sep-23	25-Sep-23
Data Modeling	ALL		100%	6-Oct-23	13-Oct-23
Class Diagram	ALL		100%	6-Oct-23	13-Oct-23
Use Case Specification	ALL		100%	10-Oct-23	13-Oct-23
Activity 2	ALL		100%	23-Sep-23	26-Nov-23
Project Management Documents	ALL		100%	5-Nov-23	26-Nov-23
Meeting Minutes	Kelly Lo		100%	23-Sep-23	26-Nov-23
Gantt Chart	Kelly Lo		100%	23-Sep-23	26-Nov-23
Burndown Chart	Tim Law		100%	23-Sep-23	26-Nov-23
Git Commit Log	Tim Law		100%	5-Nov-23	26-Nov-23
Documentation with JavaDoc	ALL		100%	5-Nov-23	26-Nov-23
Screenshots of Application Software	Tim Law		100%	6-Nov-23	26-Nov-23
Entire Project Quality	ALL		100%	5-Nov-23	26-Nov-23
Teamwork Performance	ALL		100%	5-Nov-23	26-Nov-23
YouTube Video Demo	ALL		100%	13-Nov-23	26-Nov-23
Functional Programming Quality	ALL		100%	5-Nov-23	21-Nov-23
Function A - Maze Generator	Kelly Lo		100%	5-Nov-23	14-Nov-23
Function B - Shortest Path	Tim Law		100%	6-Nov-23	14-Nov-23
Function C - Tom catches Jerry in Maze Game	Kelly Lam		100%	8-Nov-23	14-Nov-23
Function D - Unit Testing with Converage Report	ALL		100%	15-Nov-23	21-Nov-23



Tom And Jerry In Maze Game

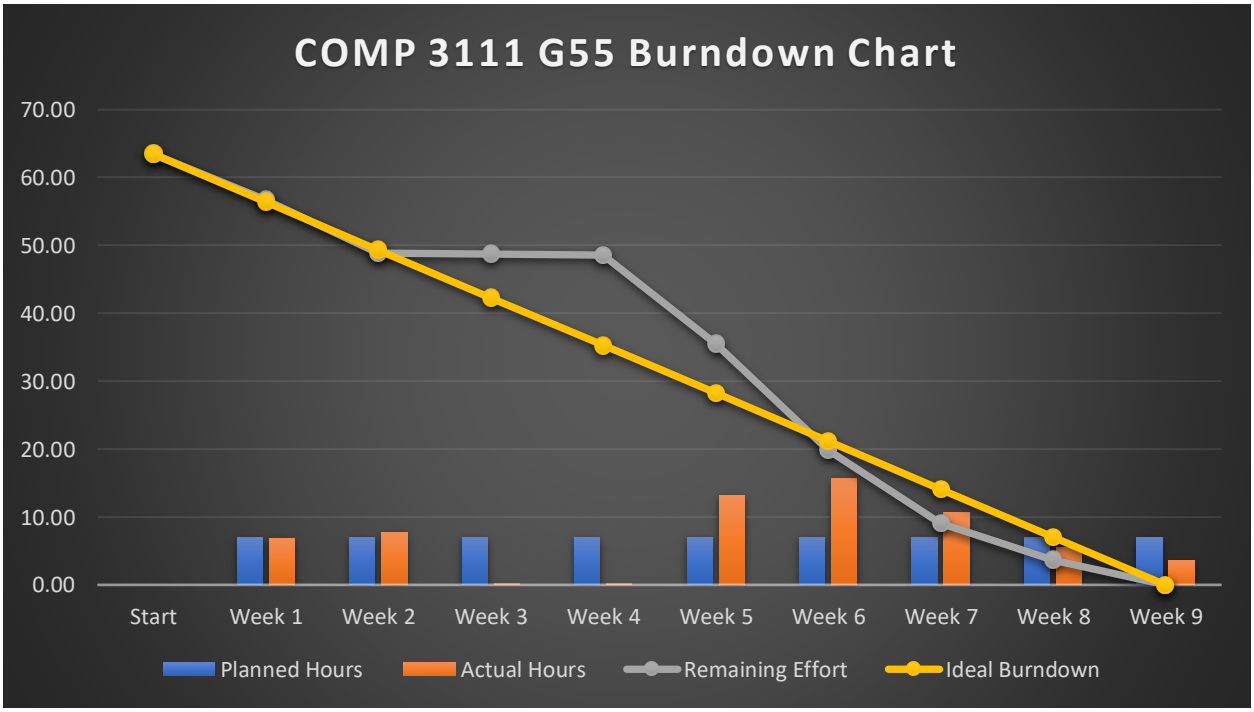
TEAM 55 (Member): Tim Law, Kelly Lam, Kelly Lo

Project Start 23-Sep-23 | Display Week 0

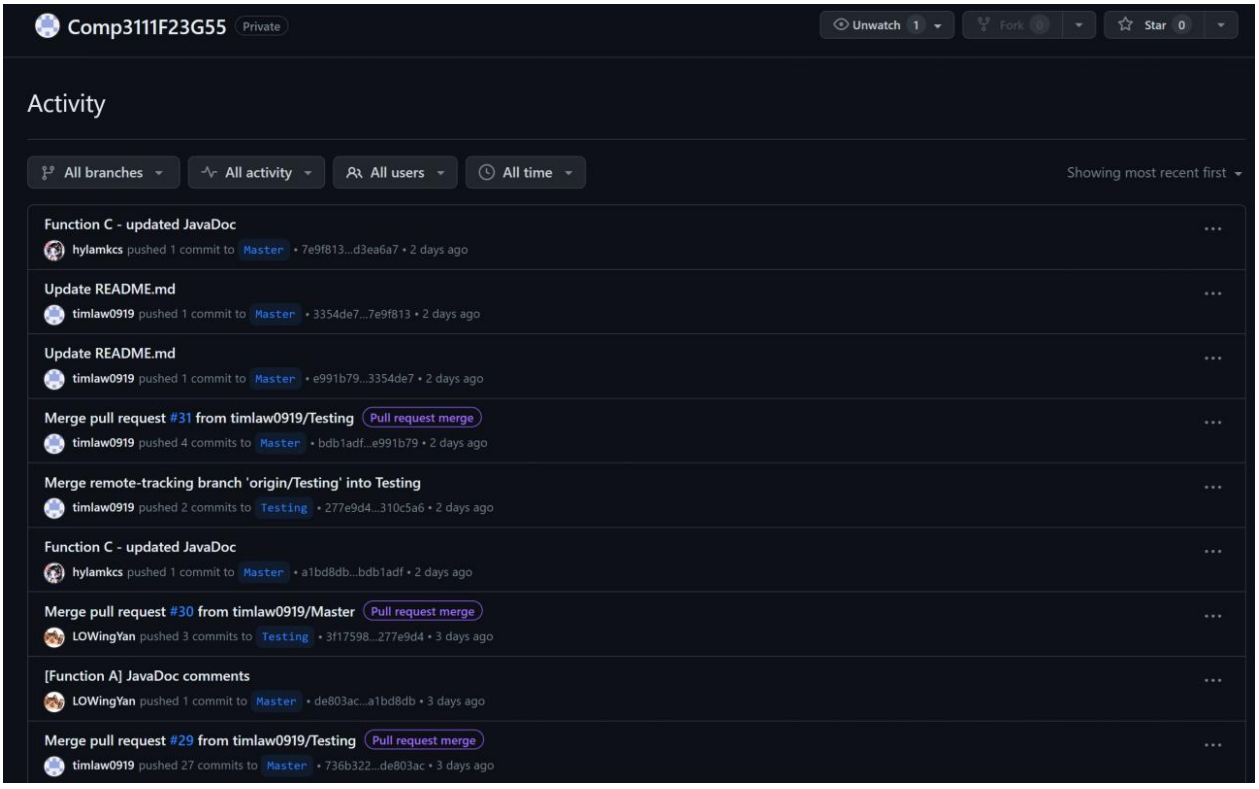
TASK	Main PIC	PIC #2	Progress	START	END
Activity 1	ALL		100%	23-Sep-23	13-Oct-23
Sign up Group Formation	Tim Law		100%	23-Sep-23	25-Sep-23
Team Repo Setup on GitHub	Tim Law		100%	23-Sep-23	25-Sep-23
Data Modeling	ALL		100%	6-Oct-23	13-Oct-23
Class Diagram	ALL		100%	6-Oct-23	13-Oct-23
Use Case Specification	ALL		100%	10-Oct-23	13-Oct-23
Activity 2	ALL		100%	23-Sep-23	26-Nov-23
Project Management Documents	ALL		100%	5-Nov-23	26-Nov-23
Meeting Minutes	Kelly Lo		100%	23-Sep-23	26-Nov-23
Gantt Chart	Kelly Lo		100%	23-Sep-23	26-Nov-23
Burndown Chart	Tim Law		100%	23-Sep-23	26-Nov-23
Git Commit Log	Tim Law		100%	5-Nov-23	26-Nov-23
Documentation with JavaDoc	ALL		100%	5-Nov-23	26-Nov-23
Screenshots of Application Software	Tim Law		100%	6-Nov-23	26-Nov-23
Entire Project Quality	ALL		100%	5-Nov-23	26-Nov-23
Teamwork Performance	ALL		100%	5-Nov-23	26-Nov-23
YouTube Video Demo	ALL		100%	13-Nov-23	26-Nov-23
Functional Programming Quality	ALL		100%	5-Nov-23	21-Nov-23
Function A - Maze Generator	Kelly Lo		100%	5-Nov-23	14-Nov-23
Function B - Shortest Path	Tim Law		100%	6-Nov-23	14-Nov-23
Function C - Tom catches Jerry in Maze Game	Kelly Lam		100%	8-Nov-23	14-Nov-23
Function D - Unit Testing with Converage Report	ALL		100%	15-Nov-23	21-Nov-23

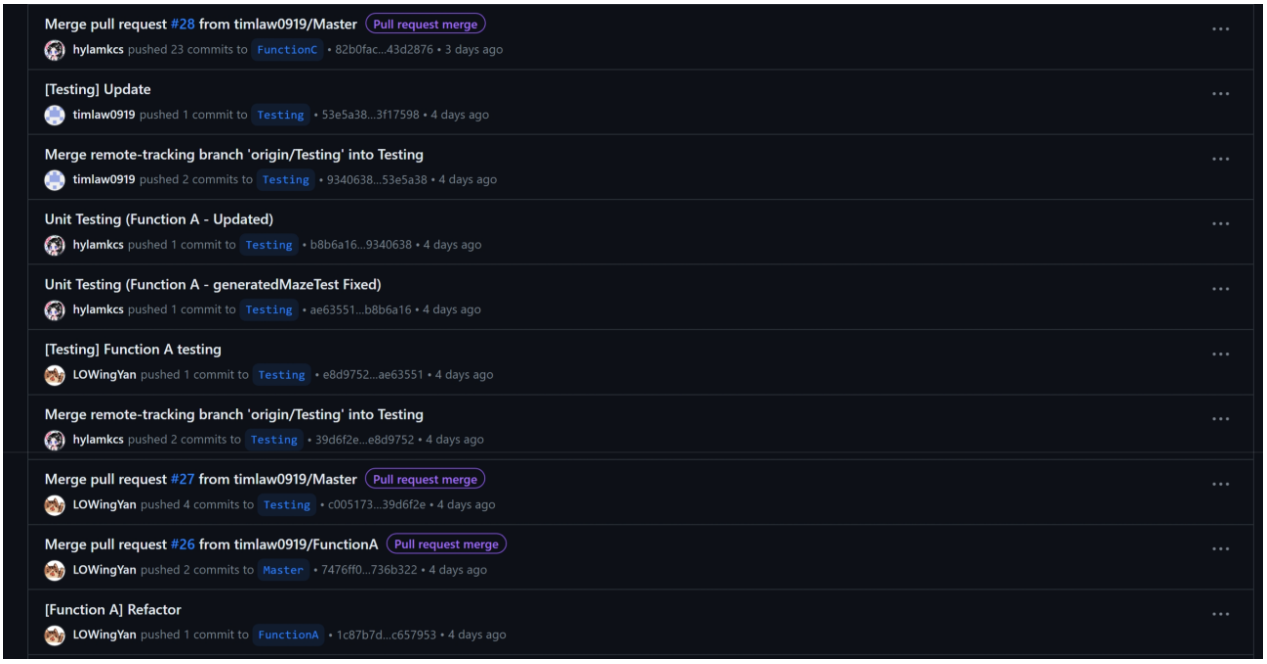
212 Burndown Chart

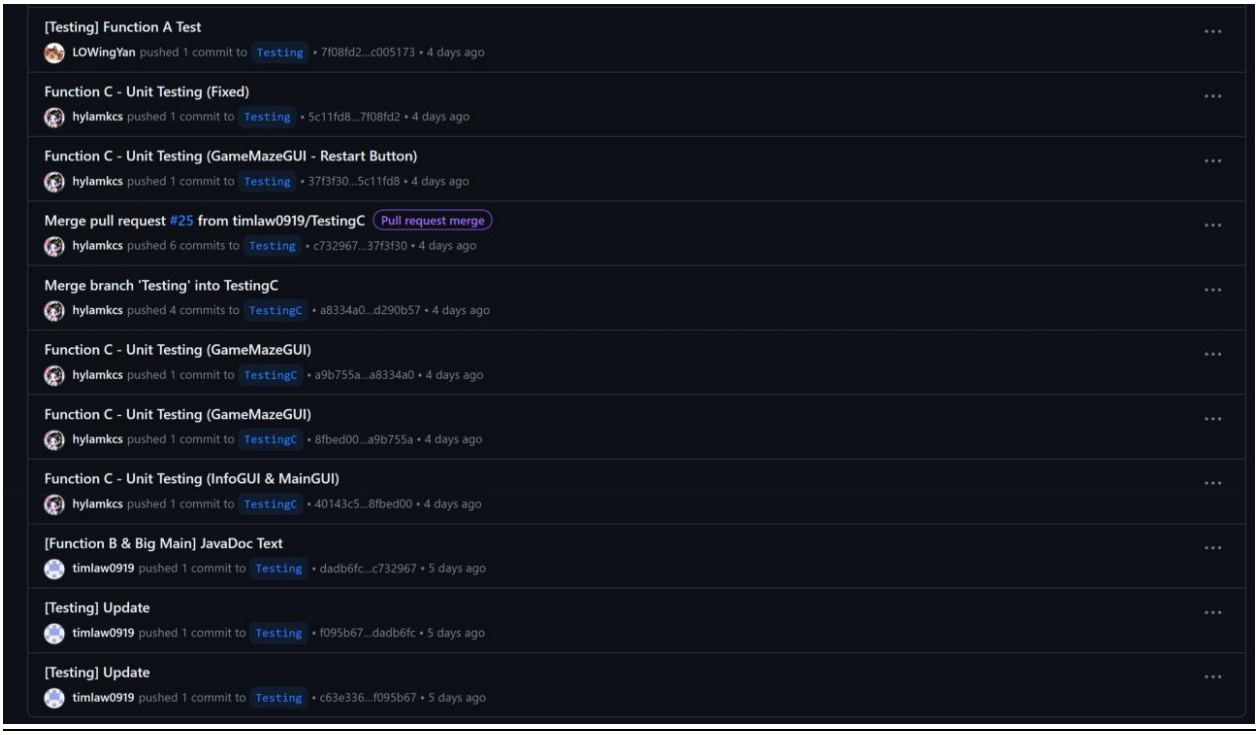
	Job Catalogs	Job ID	Job Details	Initial Estimate (in Hours)	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Hours Left
Activity 1	Project Initiation Setup (110)	111	Sign up Group Formation	0.25	0.25	0	0	0	0	0	0	0	0	0
		112	Team Repo Setup on GitHub	0.25	0.25	0	0	0	0	0	0	0	0	0
	Data Modeling (120)	121	Class Diagram	5	4	1	0	0	0	0	0	0	0	0
		122	Use Case Specification	7	1	6	0	0	0	0	0	0	0	0
Activity 2	Project Management Documents (210)	211	Meeting Minutes	2	0.5	0.5	0	0	0	0.5	0.5	0	0	0
		212	Gantt Chart	1	0.4	0.2	0.06	0.06	0.06	0.06	0.06	0.04	0.06	0
		213	Burndown Chart	1	0.4	0.08	0.08	0.08	0.08	0.08	0.08	0.08	0.04	0
		214	Git Commit Log	0.2	0	0	0	0	0	0	0	0.2	0	0
		215	Documentation with JavaDoc	0.5	0	0	0	0	0	0	0	0	0.5	0
		216	Screenshots of Application Software	0.25	0	0	0	0	0	0	0	0.25	0	0
	Entire Project Quality (220)	221	Teamwork Performance (Group Evaluation)	3	0	0	0	0	0	0	2	1	0	0
		222	YouTube Video Demo	3	0	0	0	0	0	0	0	0	3	0
	Functional Programming Quality (230)	231	Function A - Maze Generator	10	0	0	0	0	7	3	0	0	0	0
		232	Function B - Shortest Path	10	0	0	0	0	4	6	0	0	0	0
		233	Function C - Tom catches Jerry in Maze Game	10	0	0	0	0	2	6	2	0	0	0
		234	Function D - Unit Testing with Converage Report	10	0	0	0	0	0	0	6	4	0	0

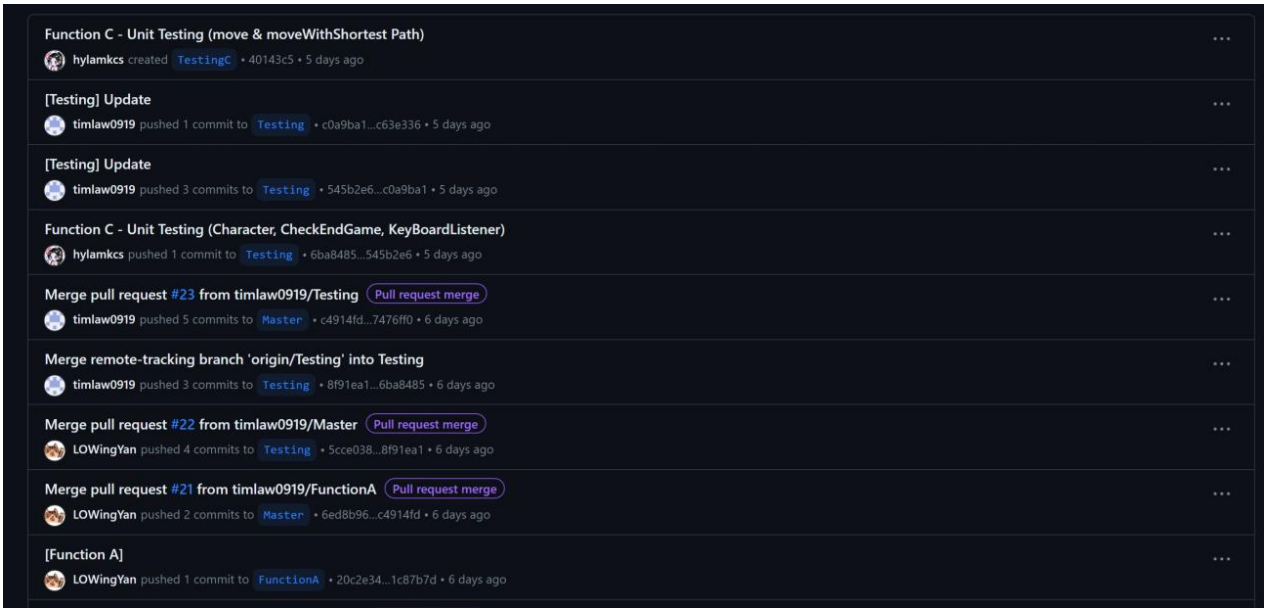


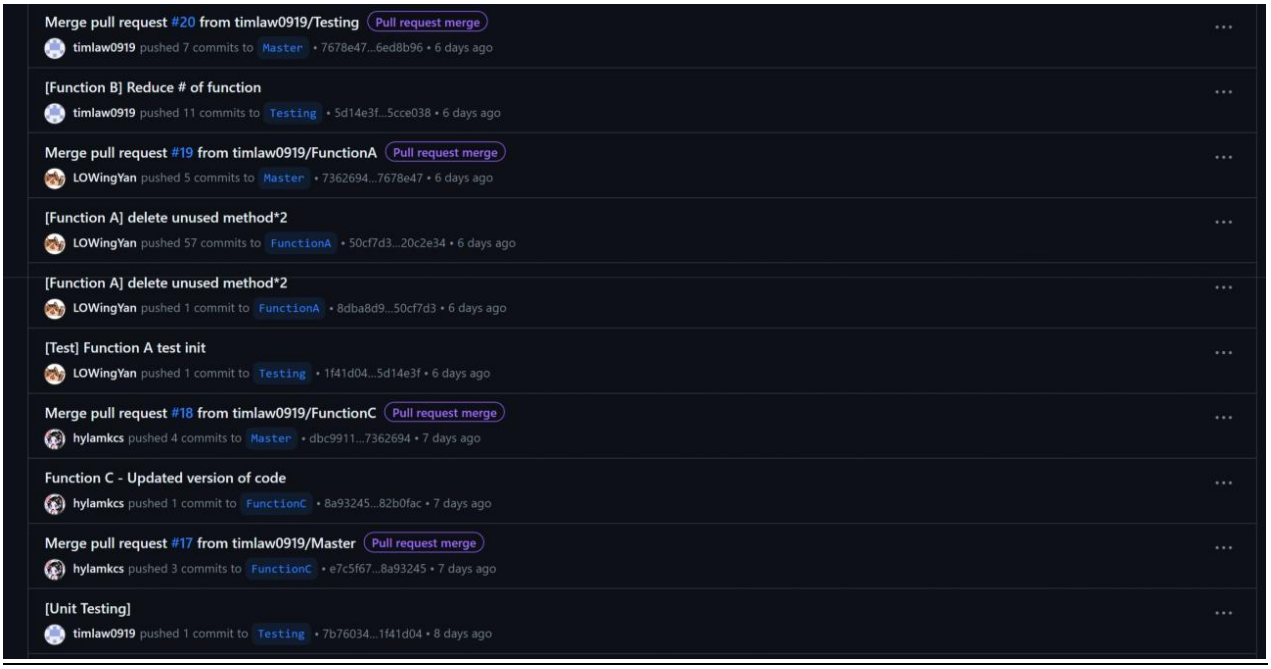
213 Git Commit Log

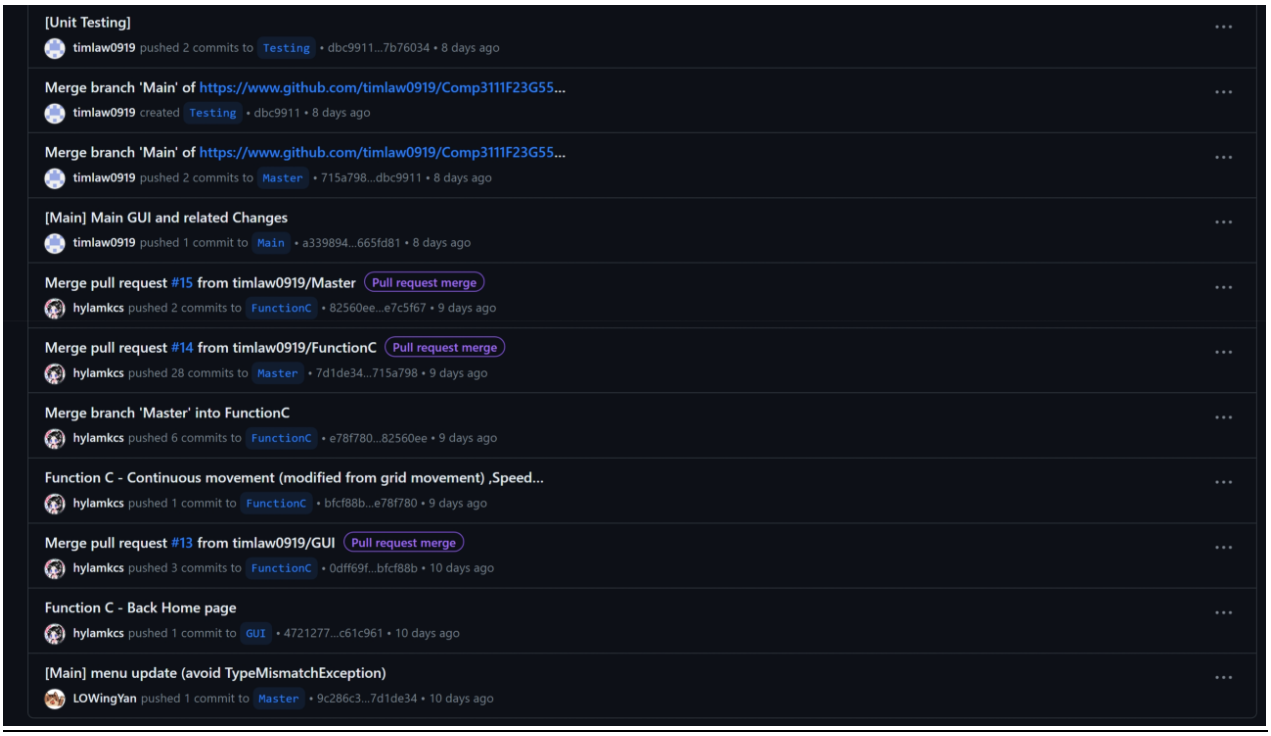


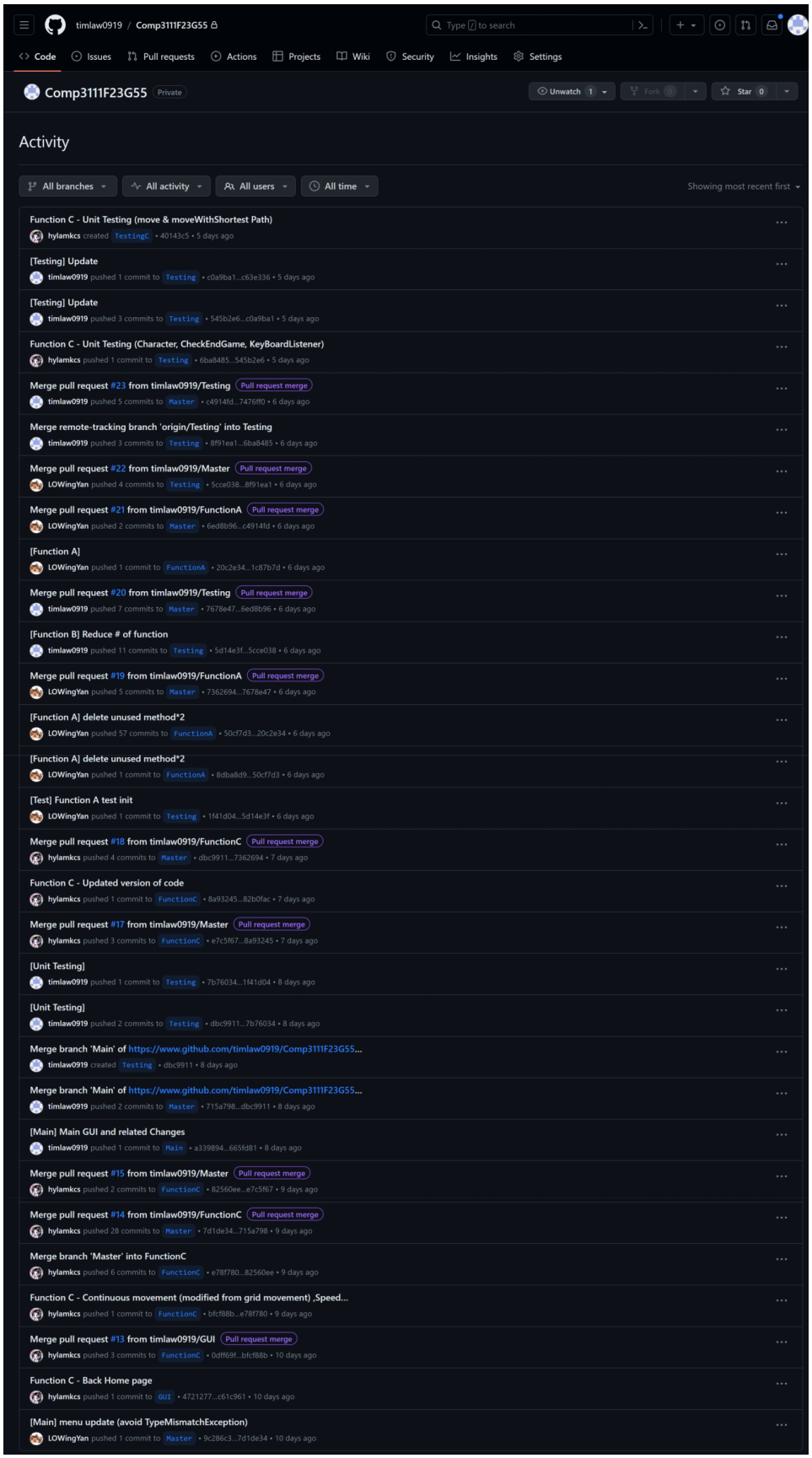


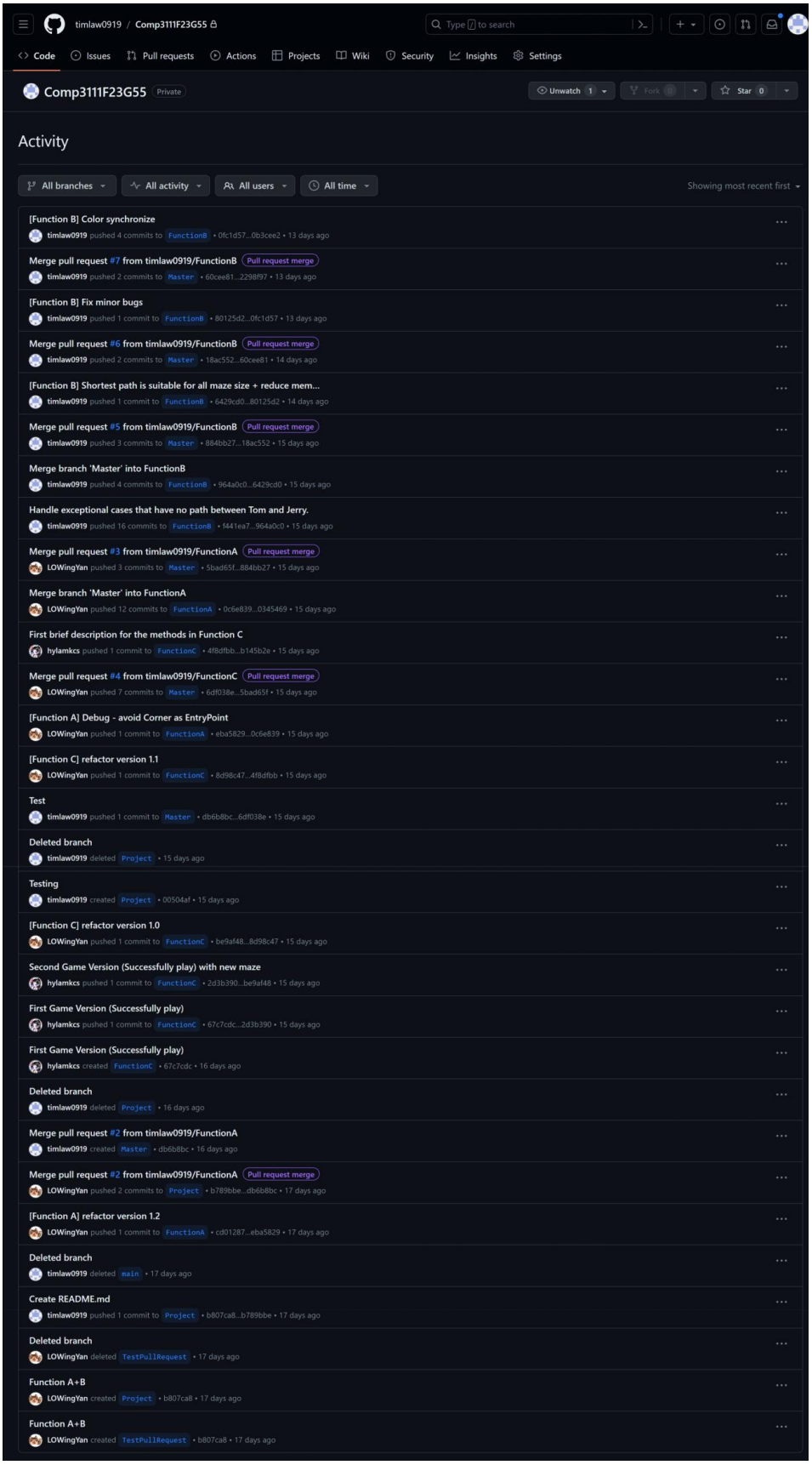

[Testing] Function A Test
LOWingYan pushed 1 commit to Testing • 7f08fd2...c005173 • 4 days ago
Function C - Unit Testing (Fixed)
hylamkcs pushed 1 commit to Testing • 5c11fd8...7f08fd2 • 4 days ago
Function C - Unit Testing (GameMazeGUI - Restart Button)
hylamkcs pushed 1 commit to Testing • 37f3f30...5c11fd8 • 4 days ago
Merge pull request #25 from timlaw0919/TestingC
Pull request merge
hylamkcs pushed 6 commits to Testing • c732967...37f3f30 • 4 days ago
Merge branch 'Testing' into TestingC
hylamkcs pushed 4 commits to TestingC • a8334a0...d290b57 • 4 days ago
Function C - Unit Testing (GameMazeGUI)
hylamkcs pushed 1 commit to TestingC • a9b755a...a8334a0 • 4 days ago
Function C - Unit Testing (GameMazeGUI)
hylamkcs pushed 1 commit to TestingC • 8fbed00...a9b755a • 4 days ago
Function C - Unit Testing (InfoGUI & MainGUI)
hylamkcs pushed 1 commit to TestingC • 40143c5...8fbed00 • 4 days ago
[Function B & Big Main] JavaDoc Text
timlaw0919 pushed 1 commit to Testing • dadb6fc...c732967 • 5 days ago
[Testing] Update
timlaw0919 pushed 1 commit to Testing • f095b67...dadb6fc • 5 days ago
[Testing] Update
timlaw0919 pushed 1 commit to Testing • c63e336...f095b67 • 5 days ago

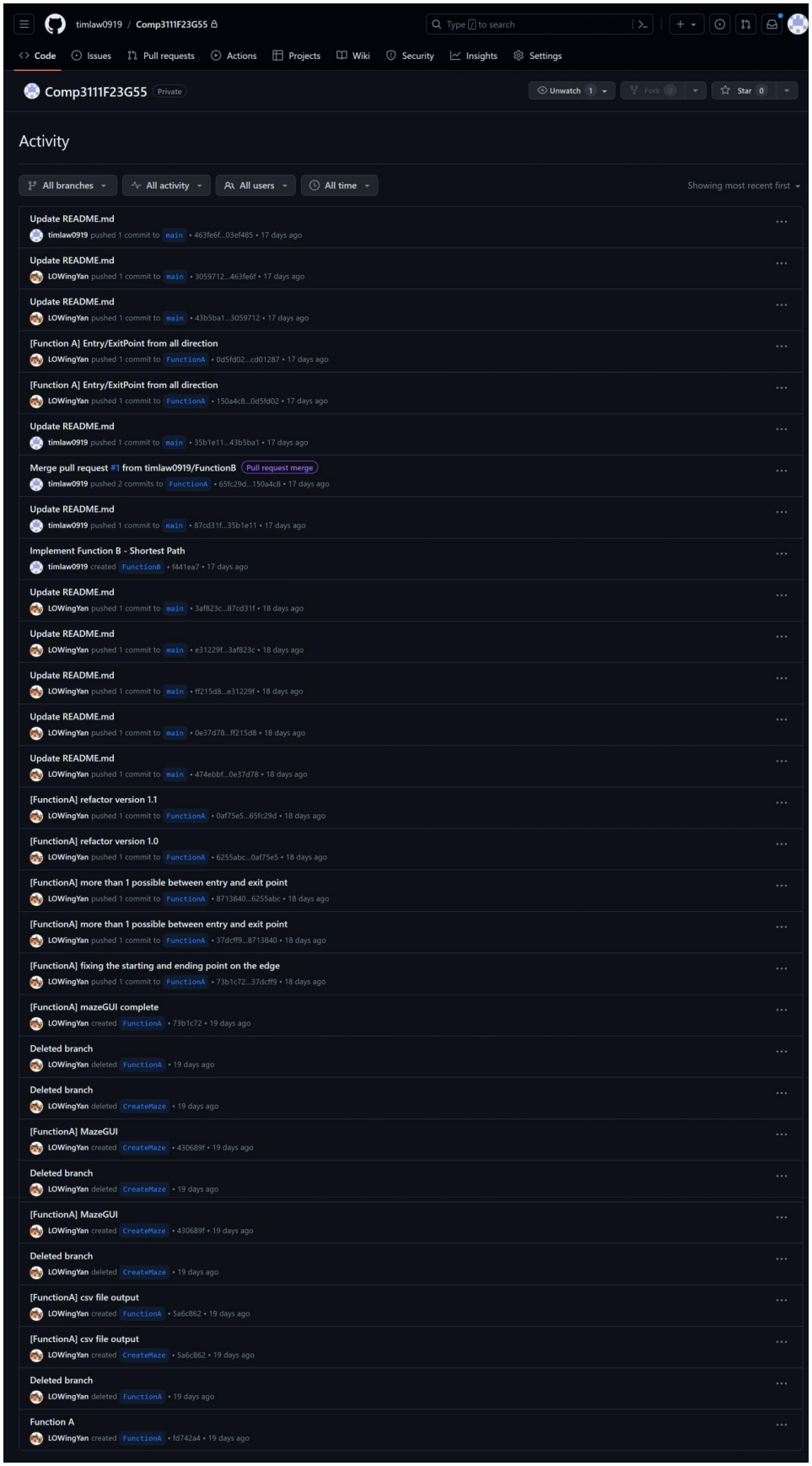

Function C - Unit Testing (move & moveWithShortest Path)
hylamkcs created TestingC • 40143c5 • 5 days ago
[Testing] Update
timlaw0919 pushed 1 commit to Testing • c0a9ba1...c63e336 • 5 days ago
[Testing] Update
timlaw0919 pushed 3 commits to Testing • 545b2e6...c0a9ba1 • 5 days ago
Function C - Unit Testing (Character, CheckEndGame, KeyBoardListener)
hylamkcs pushed 1 commit to Testing • 6ba8485...545b2e6 • 5 days ago
Merge pull request #23 from timlaw0919/Testing
Pull request merge
timlaw0919 pushed 5 commits to Master • c4914fd...7476ff0 • 6 days ago
Merge remote-tracking branch 'origin/Testing' into Testing
timlaw0919 pushed 3 commits to Testing • 8f91ea1...6ba8485 • 6 days ago
Merge pull request #22 from timlaw0919/Master
Pull request merge
LOWingYan pushed 4 commits to Testing • 5cce038...8f91ea1 • 6 days ago
Merge pull request #21 from timlaw0919/FunctionA
Pull request merge
LOWingYan pushed 2 commits to Master • 6ed8b96...c4914fd • 6 days ago
[Function A]
LOWingYan pushed 1 commit to FunctionA • 20c2e34...1c87b7d • 6 days ago

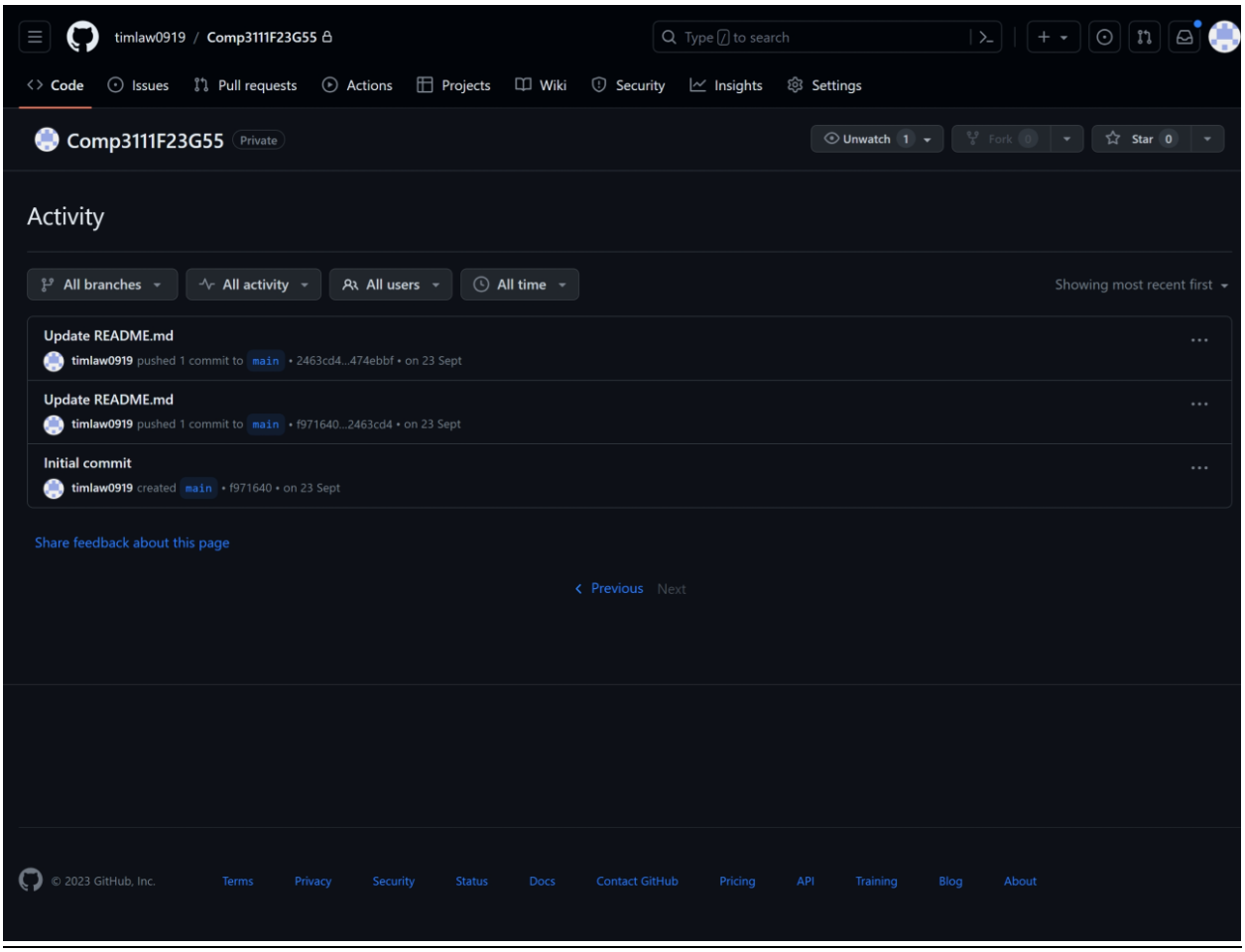

Merge pull request #20 from timlaw0919/Testing Pull request merge
timlaw0919 pushed 7 commits to Master • 7678e47...6ed8b96 • 6 days ago
[Function B] Reduce # of function
timlaw0919 pushed 11 commits to Testing • 5d14e3f...5cce038 • 6 days ago
Merge pull request #19 from timlaw0919/FunctionA Pull request merge
LOWingYan pushed 5 commits to Master • 7362694...7678e47 • 6 days ago
[Function A] delete unused method*2
LOWingYan pushed 57 commits to FunctionA • 50cf7d3...20c2e34 • 6 days ago
[Function A] delete unused method*2
LOWingYan pushed 1 commit to FunctionA • 8dba8d9...50cf7d3 • 6 days ago
[Test] Function A test init
LOWingYan pushed 1 commit to Testing • 1f41d04...5d14e3f • 6 days ago
Merge pull request #18 from timlaw0919/FunctionC Pull request merge
hylamkcs pushed 4 commits to Master • dbc9911...7362694 • 7 days ago
Function C - Updated version of code
hylamkcs pushed 1 commit to FunctionC • 8a93245...82b0fac • 7 days ago
Merge pull request #17 from timlaw0919/Master Pull request merge
hylamkcs pushed 3 commits to FunctionC • e7c5f67...8a93245 • 7 days ago
[Unit Testing]
timlaw0919 pushed 1 commit to Testing • 7b76034...1f41d04 • 8 days ago


[Unit Testing]
timlaw0919 pushed 2 commits to Testing • dbc9911...7b76034 • 8 days ago
Merge branch 'Main' of https://www.github.com/timlaw0919/Comp3111F23G55...
timlaw0919 created Testing • dbc9911 • 8 days ago
Merge branch 'Main' of https://www.github.com/timlaw0919/Comp3111F23G55...
timlaw0919 pushed 2 commits to Master • 715a798...dbc9911 • 8 days ago
[Main] Main GUI and related Changes
timlaw0919 pushed 1 commit to Main • a339894...665fd81 • 8 days ago
Merge pull request #15 from timlaw0919/Master Pull request merge
hylamkcs pushed 2 commits to FunctionC • 82560ee...e7c5f67 • 9 days ago
Merge pull request #14 from timlaw0919/FunctionC Pull request merge
hylamkcs pushed 28 commits to Master • 7d1de34...715a798 • 9 days ago
Merge branch 'Master' into FunctionC
hylamkcs pushed 6 commits to FunctionC • e78f780...82560ee • 9 days ago
Function C - Continuous movement (modified from grid movement) ,Speed...
hylamkcs pushed 1 commit to FunctionC • bfcf88b...e78f780 • 9 days ago
Merge pull request #13 from timlaw0919/GUI Pull request merge
hylamkcs pushed 3 commits to FunctionC • 0dff69f...bfcf88b • 10 days ago
Function C - Back Home page
hylamkcs pushed 1 commit to GUI • 4721277...c61c961 • 10 days ago
[Main] menu update (avoid TypeMismatchException)
LOWingYan pushed 1 commit to Master • 9c286c3...7d1de34 • 10 days ago


timlaw0919 / Comp3111F23G55
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Comp3111F23G55
Private
Unwatch 1
Fork 0
Star 0
Activity
All branches
All activity
All users
All time
Showing most recent first
Function C - Unit Testing (move & moveWithShortest Path)
hylamkcs created TestingC • 40143c5 • 5 days ago
[Testing] Update
timlaw0919 pushed 1 commit to Testing • c0a9ba1...c63e336 • 5 days ago
[Testing] Update
timlaw0919 pushed 3 commits to Testing • 545b2e6...c0a9ba1 • 5 days ago
Function C - Unit Testing (Character, CheckEndGame, KeyBoardListener)
hylamkcs pushed 1 commit to Testing • 6ba8485...545b2e6 • 5 days ago
Merge pull request #23 from timlaw0919/Testing
Pull request merge
timlaw0919 pushed 5 commits to Master • c4914fd...7476ff0 • 6 days ago
Merge remote-tracking branch 'origin/Testing' into Testing
timlaw0919 pushed 3 commits to Testing • 8f91ea1...6ba8485 • 6 days ago
Merge pull request #22 from timlaw0919/Master
Pull request merge
LOWingYan pushed 4 commits to Testing • 5cce038...8f91ea1 • 6 days ago
Merge pull request #21 from timlaw0919/FunctionA
Pull request merge
LOWingYan pushed 2 commits to Master • 6ed8b96...c4914fd • 6 days ago
[Function A]
LOWingYan pushed 1 commit to FunctionA • 20c2e34...1c87b7d • 6 days ago
Merge pull request #20 from timlaw0919/Testing
Pull request merge
timlaw0919 pushed 7 commits to Master • 7678e47...6ed8b96 • 6 days ago
[Function B] Reduce # of function
timlaw0919 pushed 11 commits to Testing • 5d14e3f...5cce038 • 6 days ago
Merge pull request #19 from timlaw0919/FunctionA
Pull request merge
LOWingYan pushed 5 commits to Master • 7362694...7678e47 • 6 days ago
[Function A] delete unused method*2
LOWingYan pushed 57 commits to FunctionA • 50cf7d3...20c2e34 • 6 days ago
[Function A] delete unused method*2
LOWingYan pushed 1 commit to FunctionA • 8dba8d9...50cf7d3 • 6 days ago
[Test] Function A test init
LOWingYan pushed 1 commit to Testing • 1f41d04...5d14e3f • 6 days ago
Merge pull request #18 from timlaw0919/FunctionC
Pull request merge
hylamkcs pushed 4 commits to Master • dbc9911...7362694 • 7 days ago
Function C - Updated version of code
hylamkcs pushed 1 commit to FunctionC • 8a93245...82b0fac • 7 days ago
Merge pull request #17 from timlaw0919/Master
Pull request merge
hylamkcs pushed 3 commits to FunctionC • e7c5f67...8a93245 • 7 days ago
[Unit Testing]
timlaw0919 pushed 1 commit to Testing • 7b76034...1f41d04 • 8 days ago
[Unit Testing]
timlaw0919 pushed 2 commits to Testing • dbc9911...7b76034 • 8 days ago
Merge branch 'Main' of https://www.github.com/timlaw0919/Comp3111F23G55...
timlaw0919 created Testing • dbc9911 • 8 days ago
Merge branch 'Main' of https://www.github.com/timlaw0919/Comp3111F23G55...
timlaw0919 pushed 2 commits to Master • 715a798...dbc9911 • 8 days ago
[Main] Main GUI and related Changes
timlaw0919 pushed 1 commit to Main • a339894...665fd81 • 8 days ago
Merge pull request #15 from timlaw0919/Master
Pull request merge
hylamkcs pushed 2 commits to FunctionC • 82560ee...e7c5f67 • 9 days ago
Merge pull request #14 from timlaw0919/FunctionC
Pull request merge
hylamkcs pushed 28 commits to Master • 7d1de34...715a798 • 9 days ago
Merge branch 'Master' into FunctionC
hylamkcs pushed 6 commits to FunctionC • e78f780...82560ee • 9 days ago
Function C - Continuous movement (modified from grid movement) ,Speed...
hylamkcs pushed 1 commit to FunctionC • bfcf88b...e78f780 • 9 days ago
Merge pull request #13 from timlaw0919/GUI
Pull request merge
hylamkcs pushed 3 commits to FunctionC • 0dff69f...bfcf88b • 10 days ago
Function C - Back Home page
hylamkcs pushed 1 commit to GUI • 4721277...c61c961 • 10 days ago
[Main] menu update (avoid TypeMismatchException)
LOWingYan pushed 1 commit to Master • 9c286c3...7d1de34 • 10 days ago


timlaw0919 / Comp3111F23G55
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Comp3111F23G55
Private
Unwatch 1
Fork 0
Star 0
Activity
All branches
All activity
All users
All time
Showing most recent first
[Function B] Color synchronize
timlaw0919 pushed 4 commits to FunctionB • 0fc1d57...0b3cee2 • 13 days ago
Merge pull request #7 from timlaw0919/FunctionB
Pull request merge
timlaw0919 pushed 2 commits to Master • 60cee81...2298f97 • 13 days ago
[Function B] Fix minor bugs
timlaw0919 pushed 1 commit to FunctionB • 80125d2...0fc1d57 • 13 days ago
Merge pull request #6 from timlaw0919/FunctionB
Pull request merge
timlaw0919 pushed 2 commits to Master • 18ac552...60cee81 • 14 days ago
[Function B] Shortest path is suitable for all maze size + reduce mem...
timlaw0919 pushed 1 commit to FunctionB • 6429cd0...80125d2 • 14 days ago
Merge pull request #5 from timlaw0919/FunctionB
Pull request merge
timlaw0919 pushed 3 commits to Master • 884bb27...18ac552 • 15 days ago
Merge branch 'Master' into FunctionB
timlaw0919 pushed 4 commits to FunctionB • 964a0c0...6429cd0 • 15 days ago
Handle exceptional cases that have no path between Tom and Jerry.
timlaw0919 pushed 16 commits to FunctionB • f441ea7...964a0c0 • 15 days ago
Merge pull request #3 from timlaw0919/FunctionA
Pull request merge
LOWingYan pushed 3 commits to Master • 5bad65f...884bb27 • 15 days ago
Merge branch 'Master' into FunctionA
LOWingYan pushed 12 commits to FunctionA • 0c6e839...0345469 • 15 days ago
First brief description for the methods in Function C
hylamkcs pushed 1 commit to FunctionC • 4f8dfbb...b145b2e • 15 days ago
Merge pull request #4 from timlaw0919/FunctionC
Pull request merge
LOWingYan pushed 7 commits to Master • 6df038e...5bad65f • 15 days ago
[Function A] Debug - avoid Corner as EntryPoint
LOWingYan pushed 1 commit to FunctionA • eba5829...0c6e839 • 15 days ago
[Function C] refactor version 1.1
LOWingYan pushed 1 commit to FunctionC • 8d98c47...4f8dfbb • 15 days ago
Test
timlaw0919 pushed 1 commit to Master • db6b8bc...6df038e • 15 days ago
Deleted branch
timlaw0919 deleted Project • 15 days ago
Testing
timlaw0919 created Project • 00504af • 15 days ago
[Function C] refactor version 1.0
LOWingYan pushed 1 commit to FunctionC • be9af48...8d98c47 • 15 days ago
Second Game Version (Successfully play) with new maze
hylamkcs pushed 1 commit to FunctionC • 2d3b390...be9af48 • 15 days ago
First Game Version (Successfully play)
hylamkcs pushed 1 commit to FunctionC • 67c7cdc...2d3b390 • 15 days ago
First Game Version (Successfully play)
hylamkcs created FunctionC • 67c7cdc • 16 days ago
Deleted branch
timlaw0919 deleted Project • 16 days ago
Merge pull request #2 from timlaw0919/FunctionA
timlaw0919 created Master • db6b8bc • 16 days ago
Merge pull request #2 from timlaw0919/FunctionA
Pull request merge
LOWingYan pushed 2 commits to Project • b789bbe...db6b8bc • 17 days ago
[Function A] refactor version 1.2
LOWingYan pushed 1 commit to FunctionA • cd01287...eba5829 • 17 days ago
Deleted branch
timlaw0919 deleted main • 17 days ago
Create README.md
timlaw0919 pushed 1 commit to Project • b807ca8...b789bbe • 17 days ago
Deleted branch
LOWingYan deleted TestPullRequest • 17 days ago
Function A+B
LOWingYan created Project • b807ca8 • 17 days ago
Function A+B
LOWingYan created TestPullRequest • b807ca8 • 17 days ago


timlaw0919 / Comp3111F23G55
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Comp3111F23G55
Private
Unwatch 1
Fork 0
Star 0
Activity
All branches
All activity
All users
All time
Showing most recent first
Update README.md
timlaw0919 pushed 1 commit to main • 463fe6f...03ef485 • 17 days ago
Update README.md
LOWingYan pushed 1 commit to main • 3059712...463fe6f • 17 days ago
Update README.md
LOWingYan pushed 1 commit to main • 43b5ba1...3059712 • 17 days ago
[Function A] Entry/ExitPoint from all direction
LOWingYan pushed 1 commit to FunctionA • 0d5fd02...cd01287 • 17 days ago
[Function A] Entry/ExitPoint from all direction
LOWingYan pushed 1 commit to FunctionA • 150a4c8...0d5fd02 • 17 days ago
Update README.md
timlaw0919 pushed 1 commit to main • 35b1e11...43b5ba1 • 17 days ago
Merge pull request #1 from timlaw0919/FunctionB
Pull request merge
timlaw0919 pushed 2 commits to FunctionA • 65fc29d...150a4c8 • 17 days ago
Update README.md
timlaw0919 pushed 1 commit to main • 87cd31f...35b1e11 • 17 days ago
Implement Function B - Shortest Path
timlaw0919 created FunctionB • f441ea7 • 17 days ago
Update README.md
LOWingYan pushed 1 commit to main • 3af823c...87cd31f • 18 days ago
Update README.md
LOWingYan pushed 1 commit to main • e31229f...3af823c • 18 days ago
Update README.md
LOWingYan pushed 1 commit to main • ff215d8...e31229f • 18 days ago
Update README.md
LOWingYan pushed 1 commit to main • 0e37d78...ff215d8 • 18 days ago
Update README.md
LOWingYan pushed 1 commit to main • 474ebbf...0e37d78 • 18 days ago
[FunctionA] refactor version 1.1
LOWingYan pushed 1 commit to FunctionA • 0af75e5...65fc29d • 18 days ago
[FunctionA] refactor version 1.0
LOWingYan pushed 1 commit to FunctionA • 6255abc...0af75e5 • 18 days ago
[FunctionA] more than 1 possible between entry and exit point
LOWingYan pushed 1 commit to FunctionA • 8713840...6255abc • 18 days ago
[FunctionA] more than 1 possible between entry and exit point
LOWingYan pushed 1 commit to FunctionA • 37dcff9...8713840 • 18 days ago
[FunctionA] fixing the starting and ending point on the edge
LOWingYan pushed 1 commit to FunctionA • 73b1c72...37dcff9 • 18 days ago
[FunctionA] mazeGUI complete
LOWingYan created FunctionA • 73b1c72 • 19 days ago
Deleted branch
LOWingYan deleted FunctionA • 19 days ago
Deleted branch
LOWingYan deleted CreateMaze • 19 days ago
[FunctionA] MazeGUI
LOWingYan created CreateMaze • 430689f • 19 days ago
Deleted branch
LOWingYan deleted CreateMaze • 19 days ago
[FunctionA] MazeGUI
LOWingYan created CreateMaze • 430689f • 19 days ago
Deleted branch
LOWingYan deleted CreateMaze • 19 days ago
[FunctionA] csv file output
LOWingYan created FunctionA • 5a6c862 • 19 days ago
[FunctionA] csv file output
LOWingYan created CreateMaze • 5a6c862 • 19 days ago
Deleted branch
LOWingYan deleted FunctionA • 19 days ago
Function A
LOWingYan created FunctionA • fd742a4 • 19 days ago


timlaw0919 / Comp3111F23G55
Type / to search
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Comp3111F23G55
Private
Unwatch 1
Fork 0
Star 0
Activity
All branches
All activity
All users
All time
Showing most recent first
Update README.md
timlaw0919 pushed 1 commit to main • 2463cd4...474ebbf • on 23 Sept
Update README.md
timlaw0919 pushed 1 commit to main • f971640...2463cd4 • on 23 Sept
Initial commit
timlaw0919 created main • f971640 • on 23 Sept
Share feedback about this page
Previous
Next
© 2023 GitHub, Inc.
Terms
Privacy
Security
Status
Docs
Contact GitHub
Pricing
API
Training
Blog
About

214 Documentation with JavaDoc

Packages

Package	Description
FunctionA_CreateMaze	
FunctionA_CreateMaze.constant	
FunctionB_ShortestPath	
FunctionC_TomCatchJerry	
Main	

Package Main

```
package Main
```

Classes

Class	Description
BigMainGUI	The BigMainGUI class is the main entry point for the G55 Tom and Jerry Maze Game Testing Menu.

Package Main

Class BigMainGUI

java.lang.Object
    javafx.application.Application
        Main.BigMainGUI

```
public class BigMainGUI
extends javafx.application.Application
```

The BigMainGUI class is the main entry point for the G55 Tom and Jerry Maze Game Testing Menu. It extends the Application class from JavaFX and provides the GUI functionality for the testing menu. The testing menu allows users to generate a new maze, display the shortest path in the maze, play the game, and exit the application by clicking the corresponding button. Users can test different functions related to maze generation, shortest path finding, and playing the game.

Nested Class Summary

Nested classes/interfaces inherited from class javafx.application.Application

javafx.application.Application.Parameters

Field Summary

Fields inherited from class javafx.application.Application

STYLESHEET_CASPIAN, STYLESHEET_MODENA

Constructor Summary

Constructors

Constructor	Description
BigMainGUI()	

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method	Description
void	start(javafx.stage.Stage stage)	

Methods inherited from class javafx.application.Application

getHostServices, getParameters, getUserAgentStylesheet, init, launch, launch, notifyPreloader, setUserAgentStylesheet, stop

Methods inherited from class java.lang.Object

clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait , wait , wait

Constructor Details

BigMainGUI

```
public BigMainGUI()
```

Method Details

start

```
public void start(javafx.stage.Stage stage)
          throws Exception
```

Specified by:

`start` in class `javafx.application.Application`

Throws:

`Exception`

Package FunctionA_CreateMaze

```
package FunctionA_CreateMaze
```

Related Packages

Package	Description
FunctionA_CreateMaze.constant	

Classes

Class	Description
Cell	The Cell class represents the building elements of the maze.
CSVOutput	The CSVOutput class is to output a CSV file representing the maze.
MazeGenerator	The MazeGenerator class is to generate a maze with multiple paths, one ENTRY and one EXIT on the opposite edge randomly The algorithm for generating the maze is Depth-First-Search Algorithm (DFS) DFS starts with the randomly created ENTRY on edge (initial current Cell) while all the remaining Cells are BLOCKS Then it expands the PATHs on maze by randomly selecting one of the valid neighbouring Cells of the current Cell which is also the next current cells Also, it finds the EXIT during the expansion process which is the first neighbouring cell of the current cell on the opposite edge of ENTRY
MazeGUI	The MazeGUI class represents a graphical user interface for displaying a maze and return button to Testing Menu.
MazeLoader	The MazeLoader class output the int array version of the maze which is loaded from the CSV file created by CSVOutput class

Package FunctionA_CreateMaze

Class MazeGenerator

java.lang.Object
FunctionA_CreateMaze.MazeGenerator

```
public class MazeGenerator
extends Object
```

The MazeGenerator class is to generate a maze with multiple paths, one ENTRY and one EXIT on the opposite edge randomly The algorithm for generating the maze is Depth-First-Search Algorithm (DFS) DFS starts with the randomly created ENTRY on edge (initial current Cell) while all the remaining Cells are BLOCKS Then it expands the PATHs on maze by randomly selecting one of the valid neighbouring Cells of the current Cell which is also the next current cells Also, it finds the EXIT during the expansion process which is the first neighbouring cell of the current cell on the opposite edge of ENTRY

Field Summary

Fields

Modifier and Type	Field	Description
`Cell`	`EntryPoint`	
`Cell`	`ExitPoint`	

Constructor Summary

Constructors

Constructor	Description
`MazeGenerator(int rows, int cols)`	Constructs a MazeGenerator object with the specified number of rows and columns.

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method	Description
`Boolean`	`cellOnCorner(Cell cell)`	Checks if a cell is on the corner of the maze.
`Boolean`	`cellOnEdge(Cell cell)`	Checks if a cell is on the edge of the maze.
`Boolean`	`cellOnGrid(int row, int col)`	Checks if a cell is on the grid.
`Boolean`	`CellOnOppositeEdge(Cell cell_1, Cell cell_2)`	Checks if two cells are on the opposite edges of the maze.
`Boolean`	`checkIfEntryPoint(Cell cell)`	Checks if a cell is the entry point of the maze.
`Boolean`	`checkIfExitPoint(Cell cell)`	Checks if a cell is the exit point of the maze.
`boolean`	`checkValidNeighbors(Cell cell)`	Checks if a cell is a valid neighboring cell.
`void`	`EntryPointGenerator()`	Generates the entry point of the maze.
`void`	`generateMaze()`	Generates the maze by DFS algorithm.
`Cell[][]`	`getMaze()`	Retrieves the maze array.
`List <Cell>`	`getValidNeighbors(Cell cell)`	Retrieves the valid neighboring cells of a given cell and set information for EXIT if found.
`void`	`initializeMaze()`	Initializes the maze grid with cells.

Methods inherited from class java.lang.Object

clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait , wait , wait

Field Details

EntryPoint

```
public Cell EntryPoint
```

ExitPoint

```
public Cell ExitPoint
```

Constructor Details

MazeGenerator

```
public MazeGenerator(int rows,
                     int cols)
```

Constructs a MazeGenerator object with the specified number of rows and columns.

Parameters:

`rows` - The number of rows in the maze.

`cols` - The number of columns in the maze.

Method Details

EntryPointGenerator

```
public void EntryPointGenerator()
```

Generates the entry point of the maze.

initializeMaze

```
public void initializeMaze()
```

Initializes the maze grid with cells.

generateMaze

```
public void generateMaze()
```

Generates the maze by DFS algorithm.

checkValidNeighbors

```
public boolean checkValidNeighbors(Cell cell)
```

Checks if a cell is a valid neighboring cell. Criteria for a valid neighboring cell if the Exit of the maze is not yet found The less than 4 of the 8 connected cells of the cell is PATH, the cell is not yet visited by the DFS algorithm, the cell is not on the edge (excluding the

opposite edge of ENTRY) Criteria for a valid neighbouring cell if the Exit of the maze is found The less than 4 of the 8 connected cells of the cell is PATH, the cell is not yet visited by the DFS algorithm, the cell is not on the edges (all)

Parameters:

`cell` - The cell to check.

Returns:

True if the cell is a valid neighboring cell, false otherwise.

getValidNeighbors

```
public List <Cell> getValidNeighbors(Cell cell)
```

Retrieves the valid neighboring cells of a given cell and set information for EXIT if found.

Parameters:

`cell` - The cell to retrieve the neighbors for.

Returns:

A list of valid neighboring cells.

cellOnGrid

```
public Boolean  cellOnGrid(int row,
                           int col)
```

Checks if a cell is on the grid.

Parameters:

`row` - The row index of the cell.

`col` - The column index of the cell.

Returns:

True if the cell is on the grid, false otherwise.

cellOnEdge

```
public Boolean  cellOnEdge(Cell cell)
```

Checks if a cell is on the edge of the maze.

Parameters:

`cell` - The cell to check.

Returns:

True if the cell is on the edge, false otherwise.

cellOnCorner

```
public Boolean  cellOnCorner(Cell cell)
```

Checks if a cell is on the corner of the maze.

Parameters:

`cell` - The cell to check.

Returns:

True if the cell is on the corner, false otherwise.

checkIfEntryPoint

```
public Boolean  checkIfEntryPoint(Cell cell)
```

Checks if a cell is the entry point of the maze.

Parameters:

`cell` - The cell to check.

Returns:

True if the cell is the entry point, false otherwise.

checkIfExitPoint

```
public Boolean  checkIfExitPoint(Cell cell)
```

Checks if a cell is the exit point of the maze.

Parameters:

`cell` - The cell to check.

Returns:

True if the cell is the exit point, false otherwise.

CellOnOppositeEdge

```
public Boolean  CellOnOppositeEdge(Cell cell_1,
                                   Cell cell_2)
```

Checks if two cells are on the opposite edges of the maze.

Parameters:

`cell_1` - The first cell.

`cell_2` - The second cell.

Returns:

True if the cells are on opposite edges, false otherwise.

getMaze

```
public Cell[][] getMaze()
```

Retrieves the maze array.

Returns:

The maze array.

Package FunctionB_ShortestPath

```
package FunctionB_ShortestPath
```

Classes

Class	Description
AStarAlgorithm	The AStarAlgorithm class implements the A Star algorithm for finding the shortest path between Tom and Jerry in a maze.
CSVOutput	The CSVOutput class is to output a CSV file containing the shortest path coordinates.
CSVOutputForGUI	The CSVOutputForGUI class provides functionality to output a maze with a path as a CSV file.
MazeWithShortestPathGUI	The MazeWithShortestPathGUI class is a JavaFX application that displays a maze with the shortest path highlighted.
Node	The Node class represents a node in the context of A star algorithm.

Package FunctionB_ShortestPath

Class AStarAlgorithm

java.lang.Object
FunctionB_ShortestPath.AStarAlgorithm

```
public class AStarAlgorithm
extends Object
```

The AStarAlgorithm class implements the A Star algorithm for finding the shortest path between Tom and Jerry in a maze. It takes the locations of Tom and Jerry, as well as the maze configuration (read by Function A's MazeLoader), and calculates the shortest path using the A Star algorithm. A Star algorithm's heuristic function is using Manhattan Distance which is admissible and will lead to an optimal solution. This class provides methods to change the locations of Tom and Jerry, check if a node has been explored, check if a node is valid, find the neighbors of a given node, generate the shortest path using A Star Algorithm, and determine Tom's next movement based on the calculated path.

See Also:

Node, MazeLoader

Field Summary

Fields

Modifier and Type	Field	Description
int[]	jerryLocation	
final int[][]	maze	
int[]	tomLocation	

Constructor Summary

Constructors

Constructor	Description
AStarAlgorithm(int[] tomLocation, int[] jerryLocation, String maze)	AStarAlgorithm's constructor for constructing the object.

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method	Description
int[]	changeLocation(int[] location, int who)	Changing the location of Tom or Jerry
boolean	checkExplored(List <Node> listOfNode, int[] temp)	Check the node is already explored or not.
boolean	checkValidNode(List <Node> expandedNode, List <Node> frontier, int[] temp)	Check the location is valid to be a neighbor or not
List <Node>	findNeighbor(Node currentNode, List <Node> expandedNode, List <Node> frontier)	Find all neighbor near the current node
List <int[]>	pathGeneratorByAStar()	Generate the shortest path between Tom and Jerry
int[]	tomNextMovement()	Tom's next action

Methods inherited from class java.lang.Object

clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait , wait , wait

Field Details

tomLocation

```
public int[] tomLocation
```

jerryLocation

```
public int[] jerryLocation
```

maze

```
public final int[][] maze
```

Constructor Details

AStarAlgorithm

```
public AStarAlgorithm(int[] tomLocation,
                      int[] jerryLocation,
                      String  maze)
```

AStarAlgorithm's constructor for constructing the object.

Parameters:

`tomLocation` - Tom's current location

`jerryLocation` - Jerry's current location

`maze` - CSV file name of the maze

Method Details

changeLocation

```
public int[] changeLocation(int[] location,
                            int who)
```

Changing the location of Tom or Jerry

Parameters:

`location` - The latest location

`who` - 0 for changing Tom's location, other for changing Jerry's location

Returns:

The latest location

checkExplored

```
public boolean checkExplored(List <Node> listOfNode,
                             int[] temp)
```

Check the node is already explored or not.

Parameters:

`listOfNode` - The nodes inside the list is explored

`temp` - The current location

Returns:

True if it is already explored, False if not explored

checkValidNode

```
public boolean checkValidNode(List <Node> expandedNode,
                              List <Node> frontier,
                              int[] temp)
```

Check the location is valid to be a neighbor or not

Parameters:

`expandedNode` - A list of nodes which have already expanded

`frontier` - A list of nodes which are candidates to be expanded

`temp` - The current location

Returns:

True if not the node does not explore and within the maze, otherwise False

findNeighbor

```
public List <Node> findNeighbor(Node currentNode,
                                List <Node> expandedNode,
                                List <Node> frontier)
```

Find all neighbor near the current node

Parameters:

`currentNode` - The current node

`expandedNode` - A list of nodes which have already expanded

`frontier` - A list of nodes which are candidates to be expanded

Returns:

A list of valid neighbor with type Node

pathGeneratorByAStar

```
public List <int[]> pathGeneratorByAStar()
```

Generate the shortest path between Tom and Jerry

Returns:

A list of coordinate with type int[] which form the shortest path

tomNextMovement

```
public int[] tomNextMovement()
```

Tom's next action

Returns:

The coordinate of Tom's next movement.

Package FunctionC_TomCatchJerry

```
package FunctionC_TomCatchJerry
```

All Classes and Interfaces Classes Enum Classes

Class	Description
Character	Character class is used to record and update the location of an instance according to the key pressing or algorithm.
CheckEndGame	CheckEndGame class is to update the game state of character and check whether a game is ended.
GameMain	GameMain class is used to initialize game objects before game starts.
GameMazeGUI	GameMazeGUI class sets up the interface and the background music for the game.
InfoGUI	InfoGUI is the interface for game rule
KeyBoardListener	KeyBoardListener class is used to update the direction of the specific moving object according to the input received from key press by player.
MainGUI	MainGUI is the interface for the player to choose the features of the game.
MainGUI.Speed	Initialize of the speed value of different speed level

Package FunctionC_TomCatchJerry

Class Character

java.lang.Object
    FunctionC_TomCatchJerry.Character

```
public class Character
extends Object
```

Character class is used to record and update the location of an instance according to the key pressing or algorithm.

Field Summary

Fields

Modifier and Type	Field	Description
boolean	Game_state	
int	lastPos	
int[]	location	
int	newCol	
int	newRow	
int	speed	

Constructor Summary

Constructors

Constructor	Description
Character()	Constructor of Character Initialize all the attributes

Method Summary

All Methods Static Methods Instance Methods Concrete Methods

Modifier and Type	Method	Description
boolean	IsPath(int row, int col)	Check whether the new location is valid for the object to move
boolean	IsWithinBoundary(int coordinate)	Check whether the new coordinate of the location is valid in maze
void	move()	Player controls the direction of the moving object using keyboard 1.
void	MoveWithShortestPath()	Computer controls the movement of the moving object using algorithm from function B 1.
void	reset(int row, int col)	Reset the status of the character when the game is restarted 1.
static int	toIndex(int[] location)	Convert the 2D location to 1D index based on the length of the maze

Methods inherited from class java.lang.Object

clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait , wait , wait

Field Details

newCol

```
public int newCol
```

newRow

```
public int newRow
```

lastPos

```
public int lastPos
```

location

```
public int[] location
```

Game_state

```
public boolean Game_state
```

speed

```
public int speed
```

Constructor Details

Character

```
public Character()
```

Constructor of Character Initialize all the attributes

Method Details

move

```
public void move()
```

Player controls the direction of the moving object using keyboard 1. Check whether the new location is valid for moving 1.1 If valid, 1.1.1 Store the previous location 1.1.2 Update the new location 1.1.3 Update the current location to the algorithm for calculating the new shortest path between Tom and Jerry 1.2 If invalid, location remains unchanged

MoveWithShortestPath

```
public void MoveWithShortestPath()
```

Computer controls the movement of the moving object using algorithm from function B 1. Get the current NEXT shortest path 2. Check whether the next step is valid 2.1 If valid, 2.1.1 Update the last position and the current location 2.1.2 Update Tom's Location to the algorithm to find the newest shortest path 2.2 If invalid, location remains unchanged

IsWithinBoundary

```
public boolean IsWithinBoundary(int coordinate)
```

Check whether the new coordinate of the location is valid in maze

Parameters:

`coordinate` - The coordinate of the row/column of the location

Returns:

true if the coordinate is non-negative and within the maze size

IsPath

```
public boolean IsPath(int row,
                      int col)
```

Check whether the new location is valid for the object to move

Parameters:

`row` - The row value of the new location

`col` - The column value of the new location

Returns:

true if the location on the maze is not a BLOCK

toIndex

```
public static int toIndex(int[] location)
```

Convert the 2D location to 1D index based on the length of the maze

Parameters:

`location` - An int array storing the row and column value of a location

Returns:

the index value in 1D

reset

```
public void reset(int row,
                  int col)
```

Reset the status of the character when the game is restarted 1. Store the current location to last position 2. Reset the character's location back to spawn point 3. Reset the newRow and newCol to 0 indicating no movement exists 4. Set the game state to false The value of lastPos and location is used to update the game maze interface

Parameters:

`row` - The row value of the spawn point

`col` - The column value of the spawn point

Package FunctionC_TomCatchJerry

Class GameMazeGUI

java.lang.Object
  javafx.application.Application
    FunctionC_TomCatchJerry.GameMazeGUI

```
public class GameMazeGUI
extends javafx.application.Application
```

GameMazeGUI class sets up the interface and the background music for the game. The maze of initial state will be set up in SetGridPane method. When the game is started, each character (Tom and Jerry) is worked with its own thread. Multithreading ensures that Tom and Jerry can move simultaneously with different speeds. The updated movement of the character is shown on the screen by updateGridPane method. Interactions with the player are achieved by message display and the button.

Nested Class Summary

Nested classes/interfaces inherited from class javafx.application.Application

javafx.application.Application.Parameters

Field Summary

Fields

Modifier and Type	Field	Description
javafx.scene.paint.ImagePattern	block	
static final int	CELL_SIZE	
List <javafx.scene.shape.Rectangle>	cells	
int	entryIndex	
int	exitIndex	
javafx.scene.layout.GridPane	gridPane	
javafx.scene.paint.ImagePattern	JerryJerry	
javafx.scene.paint.ImagePattern	TomTom	

Fields inherited from class javafx.application.Application

STYLESHEET_CASPIAN, STYLESHEET_MODENA

Constructor Summary

Constructors

Constructor	Description
GameMazeGUI()	

Method Summary

All Methods Instance Methods Concrete Methods

Modifier and Type	Method	Description
void	SetGridPane()	Set up the scene of the maze and store the spawn point of Jerry and Tom to their location respectively

		Update the grid pane after initialising the location of Tom and Jerry
void	start(javafx.stage.Stage primaryStage)	Build up the GUI for the maze game.
void	updatedGridPane(Character c, javafx.scene.paint.ImagePattern imagePattern)	Update the interface of the maze game to show the object movement Fill the grid where the character located with the corresponding image Repaint the last position of the character back to the original color of that grid If the lastPos is an BLOCK, do nothing Else if the lastPos is entry point, fill the grid with #F1CD85 Else if the lastPos is exit point, fill the grid with #808990 Else, fill the grid with white color which refers to a path

Methods inherited from class javafx.application.Application

getHostServices, getParameters, getUserAgentStylesheet, init, launch, launch, notifyPreloader, setUserAgentStylesheet, stop

Methods inherited from class java.lang.Object

clone , equals , finalize , getClass , hashCode , notify , notifyAll , toString , wait , wait , wait

Field Details

CELL_SIZE

```
public static final int CELL_SIZE
```

See Also:

Constant Field Values

entryIndex

```
public int entryIndex
```

exitIndex

```
public int exitIndex
```

gridPane

```
public javafx.scene.layout.GridPane gridPane
```

cells

```
public List <javafx.scene.shape.Rectangle> cells
```

TomTom

```
public javafx.scene.paint.ImagePattern TomTom
```

JerryJerry

```
public javafx.scene.paint.ImagePattern JerryJerry
```

block

```
public javafx.scene.paint.ImagePattern block
```

Constructor Details

GameMazeGUI

```
public GameMazeGUI()
```

Method Details

SetGridPane

```
public void SetGridPane()
```

Set up the scene of the maze and store the spawn point of Jerry and Tom to their location respectively Update the grid pane after initialising the location of Tom and Jerry

updatedGridPane

```
public void updatedGridPane(Character c,
                            javafx.scene.paint.ImagePattern imagePattern)
```

Update the interface of the maze game to show the object movement Fill the grid where the character located with the corresponding image Repaint the last position of the character back to the original color of that grid If the lastPos is an BLOCK, do nothing Else if the lastPos is entry point, fill the grid with #F1CD85 Else if the lastPos is exit point, fill the grid with #808990 Else, fill the grid with white color which refers to a path

Parameters:

`c` - The character moving on the maze

`imagePattern` - The image which represents the corresponding character on the maze

start

```
public void start(javafx.stage.Stage primaryStage)
```

Build up the GUI for the maze game. Multithreading allows different events can work on different characters concurrently. The game interface has to update once the character's movement is finished. The thread should keep track on the game state. When the game is ended, no further actions on the corresponding characters are allowed. Home button always exist on the GUI. The game is forced to terminate if the game does not ended yet. The restart button and the end game message will only appear when the game ends. Reset the status of all characters and the game interface as initial state.

Specified by:

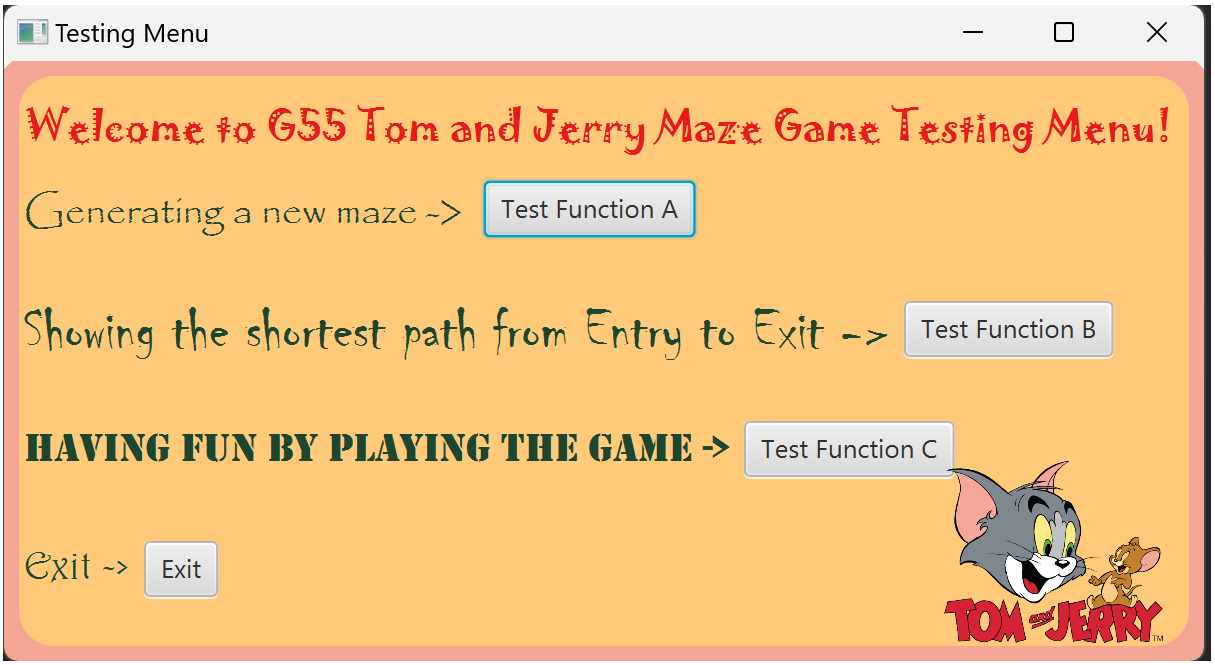
`start` in class `javafx.application.Application`

Parameters:

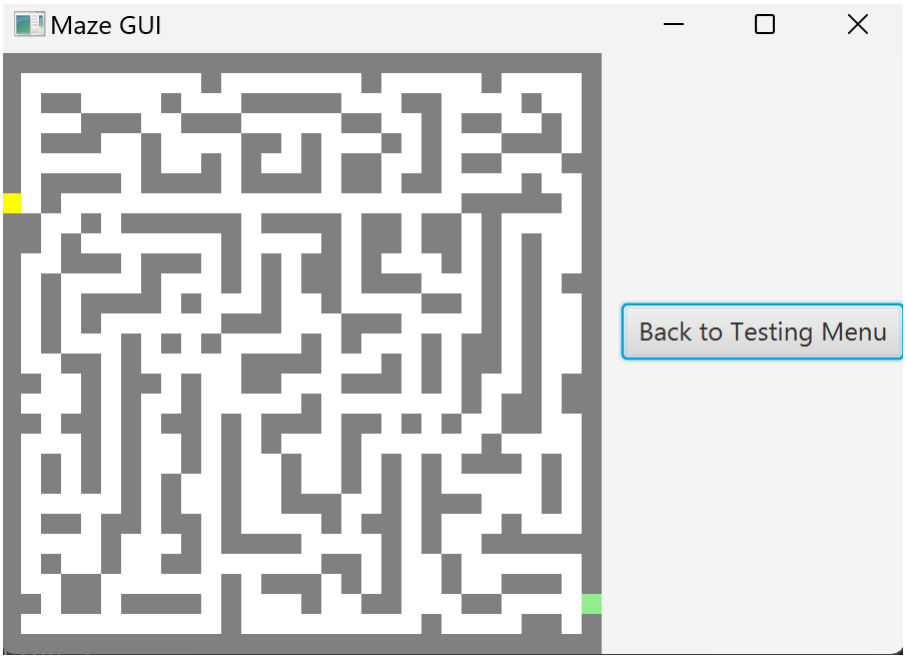
`primaryStage` - The stage shown on the screen

215 Screenshots of Application Software

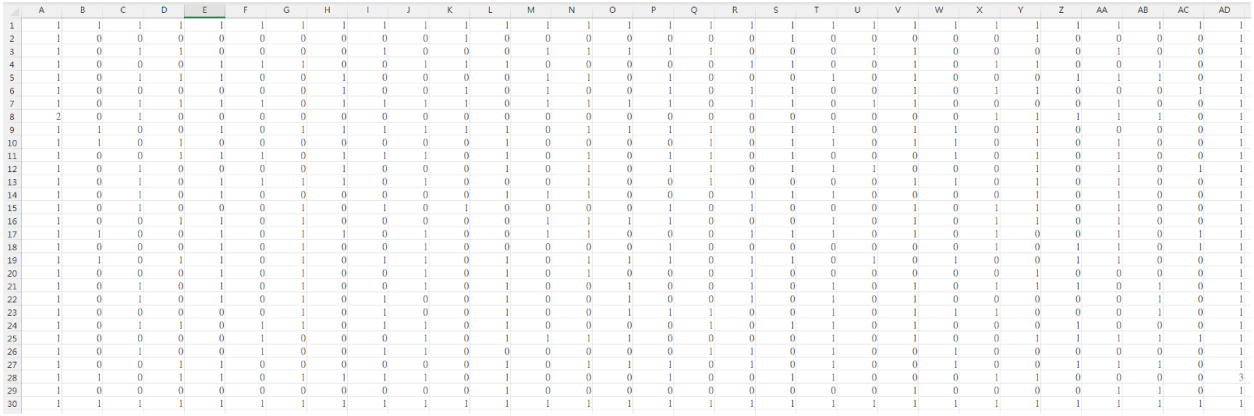
[Main Menu] Testing Menu GUI:



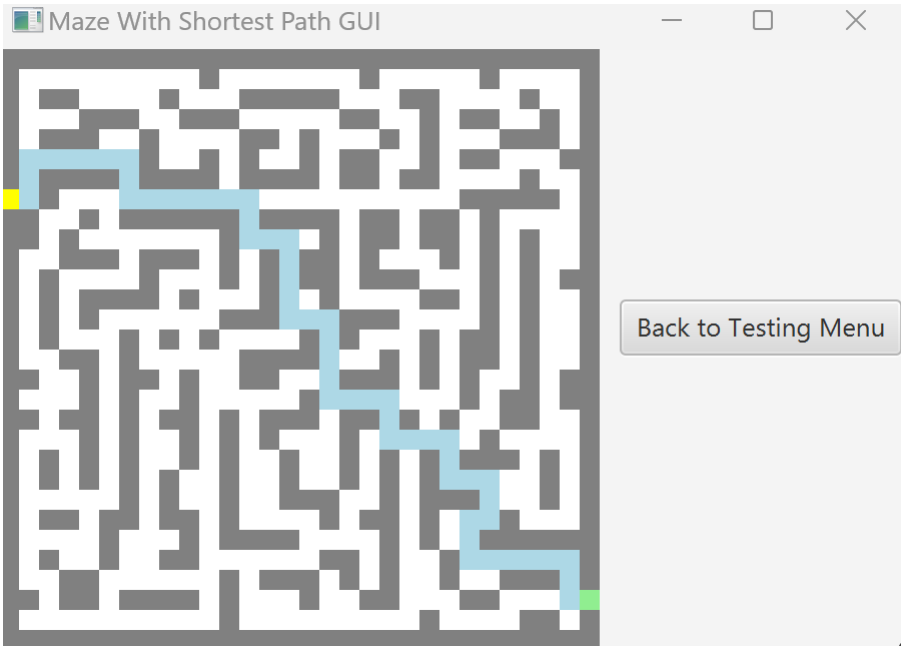
[Function A] Maze’s GUI:



[Function A] Maze’s CSV:



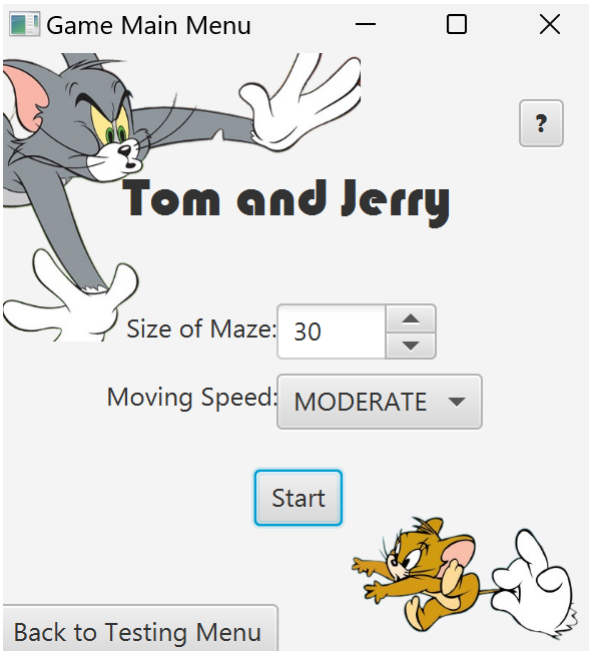
[Function B] Maze's GUI with Shortest Path:



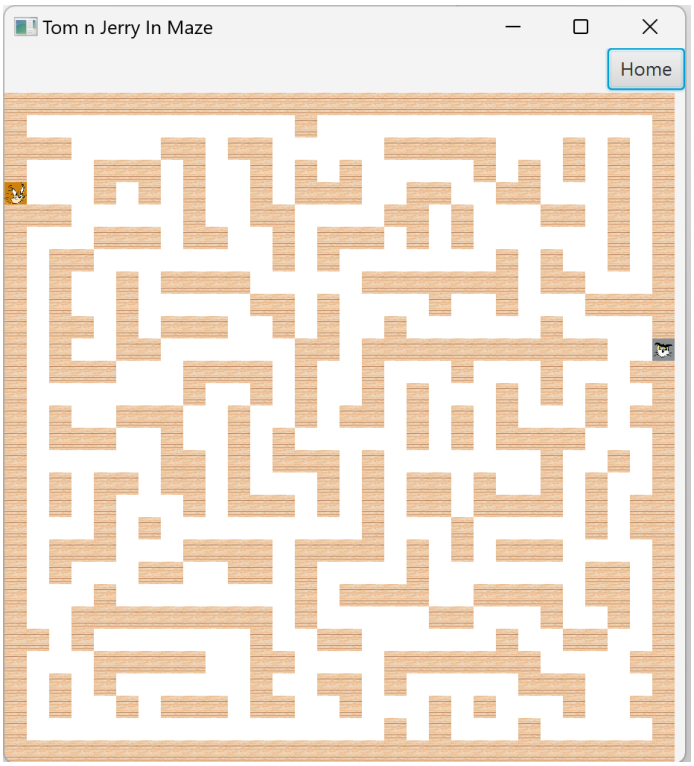
[Function B] Shortest Path Coordinate's CSV:

	A	B	C	D	E	F	G	H	I	J	K
1	Shortest Path (Read the coordinate line by line)										
2	Start->	27;29	27;28	26;28	25;28	25;27	25;26	25;25	25;24	25;23	24;23
3		23;23	23;24	22;24	21;24	21;23	21;22	20;22	19;22	19;21	19;20
4		19;19	18;19	17;19	17;18	17;17	17;16	16;16	15;16	14;16	13;16
5		13;15	13;14	12;14	11;14	10;14	9;14	9;13	9;12	8;12	7;12
6		7;11	7;10	7;9	7;8	7;7	7;6	6;6	5;6	5;5	5;4
7		5;3	5;2	5;1	6;1	7;1	7;0	-> End			

[Function C] Game Main Menu's GUI:



[Function C] Game's GUI:



Game Info


Game Main Menu
GAME RULE
Don't Be Caught by TOM & Go To The Exit Point
W
UP
A
S
D
LEFT
DOWN
RIGHT
OK!
Jerry
(You)
Tom
(Comp)

232 Unit Testing Report

		BigMainGUI	Cell		CSVOutput	MazeGUI	MazeLoader
Test Case	Target Function	start	Cell	equals	outputCSVFile	start	loadMazeFromCSV
bigMainGUIStartWithButton	BigMainGUI.start()	All Covered	na	na	na	na	na
CellTest	Cell.Cell()	na	All Covered	na	na	na	na
CellEqualsTest	Cell.equals	na	na	All Covered	na	na	na
CSVOutputTestA	CSVOutput.outputCSVFile()	na	na	na	All Covered	na	na
testMazeGUI	MazeGUI.start()	na	na	na	na	All Covered	na
loadMazefromCSVTest	MazeLoader.loadMazeFromCSV()	na	na	na	na	na	All Covered

		MazeGenerator					
Test Case	Target Function	MazeGenerator	EntryPointGenerator	initializeMaze	generateMaze	checkValidNeighbors	getValidNeighbors
MazeGenerator	MazeGenerator.MazeGenerator()	All Covered	na	na	na	na	na
EntryPointGeneratorTest	MazeGenerator.EntryPointGenerator()	na	All Covered	na	na	na	na
initializeMazeTest	MazeGenerator.initializeMaze()	na	na	All Covered	na	na	na
generateMazeTest	MazeGenerator.generateMaze()	na	na	na	All Covered	na	na
checkValidNeighborsTest	MazeGenerator.checkValidNeighbors()	na	na	na	na	All Covered	na
getValidNeighborsTest	MazeGenerator.getValidNeighbors()	na	na	na	na	na	149,153-158, 166-169,171-173, 181-187, 192,196-201

		MazeGenerator						
Test Case	Target Function	cellOnGrid	cellOnEdge	cellOnCorner	checkIfEntryPoint	checkIfExitPoint	CellOnOppositeEdge	getMaze
cellOnGridTest	MazeGenerator.cellOnGrid()	All Covered	na	na	na	na	na	na
cellOnEdgeTest	MazeGenerator.cellOnEdge()	na	All Covered	na	na	na	na	na
cellOnCornerTest	MazeGenerator.cellOnCorner()	na	na	All Covered	na	na	na	na
checkIfEntryPointTest	MazeGenerator.checkIfEntryPoint()	na	na	na	All Covered	na	na	na
checkIfExitPointTest	MazeGenerator.checkIfExitPoint()	na	na	na	na	All Covered	na	na
cellOnOppositeEdgeTest	MazeGenerator.CellOnOppositeEdge()	na	na	na	na	na	All Covered	na
getMaze	MazeGenerator.getMaze()	na	na	na	na	na	na	All Covered

		Node	CSVOutput	CSVOutputForGUI	MazeWithShortestPathGUI
Test Case	Target Function	Node	outputCSVFile	outputCSVFile	start
Node	Node.Node()	All Covered	na	na	na
CSVOutput	CSVOutput.outputCSVFile()	na	All Covered	na	na
CSVOutputForGUI	CSVOutputForGUI.outputCSVFile()	na	na	All Covered	na
MazeWithShortestPathGUIButton	MazeWithShortestPathGUI.start()	na	na	na	All Covered

		AStarAlgorithm						
Test Case	Target Function	AStarAlgorithm	changeLocation	checkExplored	checkValidNode	findNeighbor	pathGeneratorByAStar	tomNextMovement
AStarAlgorithm	AStarAlgorithm.AStarAlgorithm()	All Covered	na	na	na	na	na	na
changeLocation	AStarAlgorithm.changeLocation()	na	All Covered	na	na	na	na	na
checkExplored	AStarAlgorithm.checkExplored()	na	na	All Covered	na	na	na	na
checkValidNode	AStarAlgorithm.checkValidNode()	na	na	na	All Covered	na	na	na
findNeighbor	AStarAlgorithm.findNeighbor()	na	na	na	na	All Covered	na	na
pathGeneratorByAStar	AStarAlgorithm.pathGeneratorByAStar()	na	na	na	na	na	All Covered	na
tomNextMovement	AStarAlgorithm.tomNextMovement()	na	na	na	na	na	na	All Covered

		Character						
Test Case	Target Function	Character	move	MoveWithShortestPath	IsWithinBoundary	IsPath	toIndex	reset
Character	Character.Character()	All Covered	na	na	na	na	na	na
move	Character.move()	na	All Covered	na	na	na	na	na
moveWithShortestPath	Character.MoveWithShortestPath()	na	na	All Covered	na	na	na	na
IsWithinBoundary	Character.IsWithinBoundary()	na	na	na	All Covered	na	na	na
IsPath	Character.IsPath()	na	na	na	na	All Covered	na	na
toIndex	Character.toIndex()	na	na	na	na	na	All Covered	na
reset	Character.reset()	na	na	na	na	na	na	All Covered

		CheckEndGame	GameMain			GameMazeGUI			InfoGUI
Test Case	Target Function	isEndGame	newMaze	setGridPane	updateGridPane	start		handler	start
isEndGame	CheckEndGame.isEndGame()	All Covered	na	na	na	na	na	na	na
newMaze	GameMain.newMaze()	na	All Covered	na	na	na	na	na	na
setGridPane	GameMazeGUI.setGridPane()	na	na	All Covered	na	na	na	na	na
updateGridPane	GameMazeGUI.updateGridPane()	na	na	na	All Covered	na	na	na	na
GameMazeGUI_start	GameMazeGUI.start()	na	na	na	na	179-182, 184, 186,187, 193-195	na	na	na
testSceneOnKeyPress	GameMazeGUI.handler()	na	na	na	na	na	All Covered	na	na
InfoGUI_start	InfoGUI.start()	na	na	na	na	na	na	All Covered	

		KeyBoardListener	KeyBoardListener	MainGUI
Test Case	Target Function	KeyBoardListener	keyPressed	start
KeyBoardListener	KeyBoardListener.KeyBoardListener()	All Covered	na	na
keyPress	KeyBoardListener.keyPressed()	na	All Covered	na
MainGUI_start	MainGUI.start()	na	na	All Covered

232 Unit Testing Report

Element	Class, %	Method, %	Line, %
all	100% (22/22)	100% (69/69)	100% (735/735)
FunctionA_CreateMaze	100% (6/6)	100% (21/21)	100% (159/159)
constant	100% (1/1)	100% (2/2)	100% (5/5)
Cell	100% (1/1)	100% (2/2)	100% (8/8)
CSVOutput	100% (1/1)	100% (1/1)	100% (10/10)
MazeGenerator	100% (1/1)	100% (13/13)	100% (96/96)
MazeGUI	100% (1/1)	100% (2/2)	100% (27/27)
MazeLoader	100% (1/1)	100% (1/1)	100% (13/13)
FunctionB_ShortestPath	100% (5/5)	100% (12/12)	100% (134/134)
AStarAlgorithm	100% (1/1)	100% (7/7)	100% (61/61)
CSVOutput	100% (1/1)	100% (1/1)	100% (19/19)
CSVOutputForGUI	100% (1/1)	100% (1/1)	100% (20/20)
MazeWithShortestPathGUI	100% (1/1)	100% (2/2)	100% (28/28)
Node	100% (1/1)	100% (1/1)	100% (6/6)
FunctionC_TomCatchJerry	100% (10/10)	100% (32/32)	100% (355/355)
Character	100% (1/1)	100% (7/7)	100% (28/28)
CheckEndGame	100% (1/1)	100% (1/1)	100% (7/7)
GameMain	100% (1/1)	100% (2/2)	100% (12/12)
GameMazeGUI	100% (2/2)	100% (10/10)	100% (112/112)
InfoGUI	100% (1/1)	100% (2/2)	100% (109/109)
KeyBoardListener	100% (2/2)	100% (3/3)	100% (16/16)
MainGUI	100% (2/2)	100% (7/7)	100% (71/71)
Main	100% (1/1)	100% (4/4)	100% (87/87)
BigMainGUI	100% (1/1)	100% (4/4)	100% (87/87)

Test-per-function ratio

$$= \frac{47}{\max\left(160 - \max(47,80), 1\right)}$$

$$= \frac{47}{\max\left(160 - 80, 1\right)}$$

$$= \frac{47}{\max\left(80, 1\right)}$$

$$= \frac{47}{80}$$

1. BigMainGUI

```
public class BigMainGUI extends Application {
    timlaw0919
    @Override
    public void start (Stage stage) throws Exception{
        Pane pane = new Pane();

        Image tomAndJerryHead = new Image( s: "file:TomAndJerryHead.png");
        ImageView tomAndJerryHeadView = new ImageView(tomAndJerryHead);
        tomAndJerryHeadView.setFitHeight(91.2);
        tomAndJerryHeadView.setFitWidth(109.4);
        tomAndJerryHeadView.setLayoutX(470);
        tomAndJerryHeadView.setLayoutY(200);

        Rectangle box = new Rectangle( v: 600,  v1: 300);
        box.setFill(Color.web( s: "#FFC97A"));
        box.setStroke(Color.web( s: "#F4A594"));
        box.setStrokeWidth(15);
        box.setArcHeight(50);
        box.setArcWidth(50);

        Label welcomeLabel = new Label( s: "Welcome to G55 Tom and Jerry Maze Game Testing Menu!");
        welcomeLabel.setId("welcomeLabel");
        welcomeLabel.setFont(new Font( s: "Jokerman",  v: 20));
        welcomeLabel.setTextFill(Color.web( s: "#e61c1c"));
        welcomeLabel.setLayoutX(10);
        welcomeLabel.setLayoutY(15);

        Label generateMazeLabel = new Label( s: "Generating a new maze -> ");
        generateMazeLabel.setFont(new Font( s: "Papyrus",  v: 20));
        generateMazeLabel.setTextFill(Color.web( s: "#1e4631"));
        generateMazeLabel.setLayoutX(10);
        generateMazeLabel.setLayoutY(50);
```

```
        Label generateShortestPathMazeLabel = new Label( s: "Showing the shortest path from Entry to Exit -> ");
        generateShortestPathMazeLabel.setFont(new Font( s: "Chiller",  v: 30));
        generateShortestPathMazeLabel.setTextFill(Color.web( s: "#1e4631"));
        generateShortestPathMazeLabel.setLayoutX(10);
        generateShortestPathMazeLabel.setLayoutY(110);

        Label playTheGameLabel = new Label( s: "Having Fun by playing the game -> ");
        playTheGameLabel.setFont(new Font( s: "Stencil",  v: 20));
        playTheGameLabel.setTextFill(Color.web( s: "#1e4631"));
        playTheGameLabel.setLayoutX(10);
        playTheGameLabel.setLayoutY(180);

        Label exitLabel = new Label( s: "Exit -> ");
        exitLabel.setFont(new Font( s: "Harrington",  v: 20));
        exitLabel.setTextFill(Color.web( s: "#1e4631"));
        exitLabel.setLayoutX(10);
        exitLabel.setLayoutY(240);

        Button functionAButton = new Button( s: "Test Function A");
        functionAButton.setLayoutX(240);
        functionAButton.setLayoutY(60);

        Button functionBButton = new Button( s: "Test Function B");
        functionBButton.setLayoutX(450);
        functionBButton.setLayoutY(120);

        Button functionCButton = new Button( s: "Test Function C");
        functionCButton.setLayoutX(370);
        functionCButton.setLayoutY(180);
```

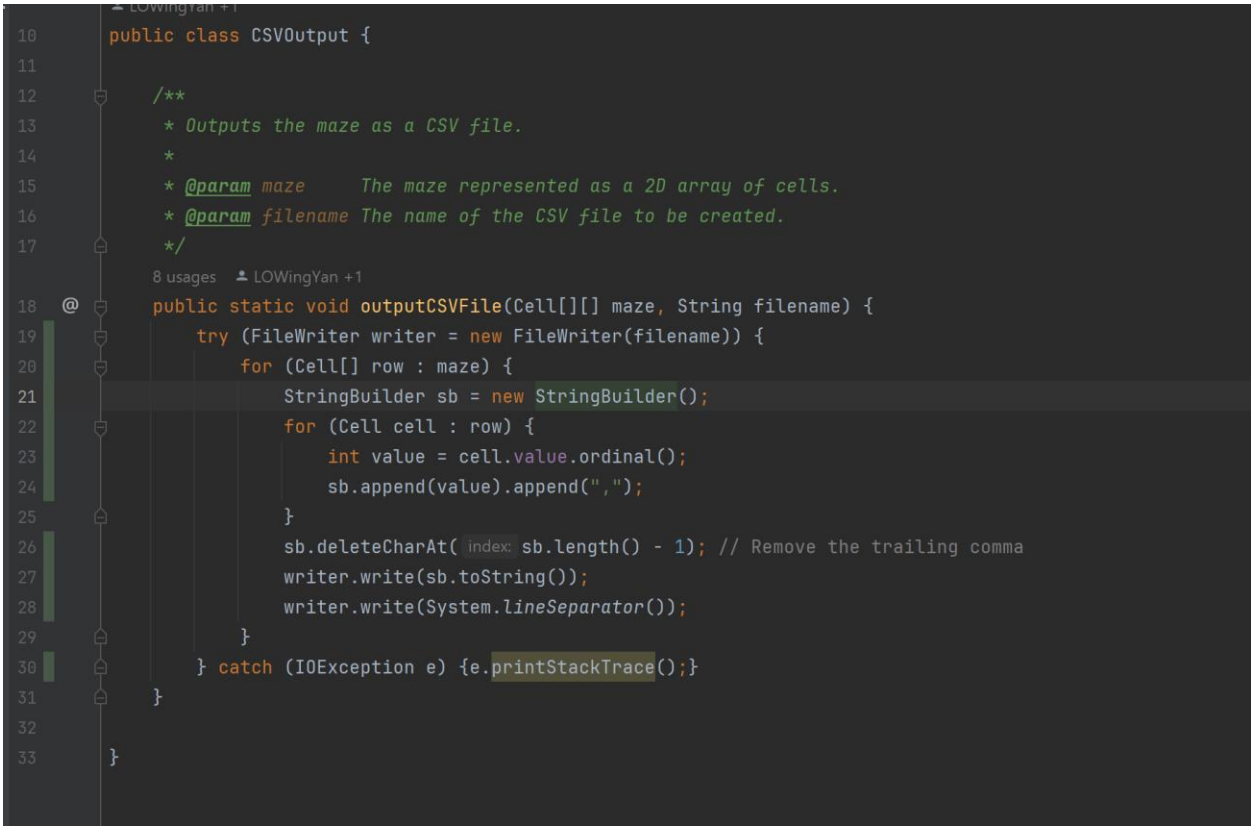
```
        Button exitButton = new Button( s: "Exit");
        exitButton.setLayoutX(70);
        exitButton.setLayoutY(240);

        pane.getChildren().addAll(box, welcomeLabel, generateMazeLabel, generateShortestPathMazeLabel,
        playTheGameLabel, exitLabel, functionAButton, functionBButton, functionCButton, exitButton,
        tomAndJerryHeadView);

        // Set the Click action
        functionAButton.setOnAction(actionEvent -> {
            FunctionA_CreateMaze.MazeGUI mazeGUI = new MazeGUI();
            try {
                int rows = 30; // Number of rows in the maze
                int cols = 30; // Number of columns in the maze
                // Generator the maze
                MazeGenerator mazeGenerator = new MazeGenerator(rows, cols);
                mazeGenerator.generateMaze();
                // Output the maze as csv file
                Cell[][] maze = mazeGenerator.getMaze();
                outputCSVFile(maze, filename: "maze_map.csv");
                mazeGUI.start(stage);} catch (Exception e) {throw new RuntimeException(e);}
        });
        functionBButton.setOnAction(actionEvent -> {
            MazeWithShortestPathGUI mazeWithShortestPathGUI = new MazeWithShortestPathGUI();
            try {
                int[] tomLocation = {0,0};
                int[] jerryLocation = {0,0};
                int[][] maze = MazeLoader.loadMazeFromCSV( filePath: "maze_map.csv");
                for (int i = 0; i < 30; i++){
                    for (int j = 0; j < 30; j++){
                        if (maze[i][j] == 2) {
                            jerryLocation[0] = i;
                            jerryLocation[1] = j;
                        }
                        if (maze[i][j] == 3) {
                            tomLocation[0] = i;
                            tomLocation[1] = j;
                        }
                    }
                }
                AStarAlgorithm obj1 = new AStarAlgorithm(tomLocation, jerryLocation, maze: "maze_map.csv");
                CSVOutputForGUI.outputCSVFile(obj1.pathGeneratorByAStar(), outputFilename: "maze_map_with_path.csv", inputFilename: "maze_map.csv");
                FunctionB_ShortestPath.CSVOutput.outputCSVFile(obj1.pathGeneratorByAStar(), filename: "path_coordinates.csv");
                mazeWithShortestPathGUI.start(stage);} catch (Exception e) {throw new RuntimeException(e);}
        });
        functionCButton.setOnAction(actionEvent -> {
            FunctionC_TomCatchJerry.MainGUI mainGUI = new MainGUI();
            try {
                mainGUI.start(stage);} catch (IOException e) {throw new RuntimeException(e);}
        });
        exitButton.setOnAction(actionEvent -> System.exit( status: 0));

        Scene startPage = new Scene(pane, v: 600, v1: 300);
        stage.setScene(startPage);
        stage.setTitle("Testing Menu");
        stage.show();
    }
}
```

2. Cell

```java
LOWingYan
public class Cell {
    public int row;             // The row index of the cell
    27 usages
    public int col;             // The column index of the cell
    public CellState value;     // The state of the cell: PATH, BLOCK, ENTRY, EXIT
    4 usages
    boolean visited;            // Indicates whether the cell has been visited or not

    /**
     * Constructs a new Cell object.
     *
     * @param row   The row index of the cell.
     * @param col   The column index of the cell.
     * @param value The state/value of the cell.
     */
    12 usages  LOWingYan
    public Cell(int row, int col, CellState value) {
        this.row = row;
        this.col = col;
        this.visited = false;
        this.value = value;
    }
```

```java
    }

    /**
     * Checks if the current cell is equal to the specified object.
     * Equal if the object is Cell. Also, it has the same row, col with the current cell
     *
     * @param obj The object to compare.
     * @return True if the cells are equal, false otherwise.
     */
    LOWingYan
    @Override
    public boolean equals(Object obj) {
        if(obj instanceof Cell)
            return (this.row == ((Cell)obj).row) && (this.col == ((Cell)obj).col);

        return false;
    }
}
```

3. Function A - CSVOutput



```java
public class CSVOutput {

    /**
     * Outputs the maze as a CSV file.
     *
     * @param maze     The maze represented as a 2D array of cells.
     * @param filename The name of the CSV file to be created.
     */
    8 usages  LOWingYan +1
    public static void outputCSVFile(Cell[][] maze, String filename) {
        try (FileWriter writer = new FileWriter(filename)) {
            for (Cell[] row : maze) {
                StringBuilder sb = new StringBuilder();
                for (Cell cell : row) {
                    int value = cell.value.ordinal();
                    sb.append(value).append(",");
                }
                sb.deleteCharAt( index: sb.length() - 1); // Remove the trailing comma
                writer.write(sb.toString());
                writer.write(System.lineSeparator());
            }
        } catch (IOException e) {e.printStackTrace();}
    }

}
```

4. MazeGUI

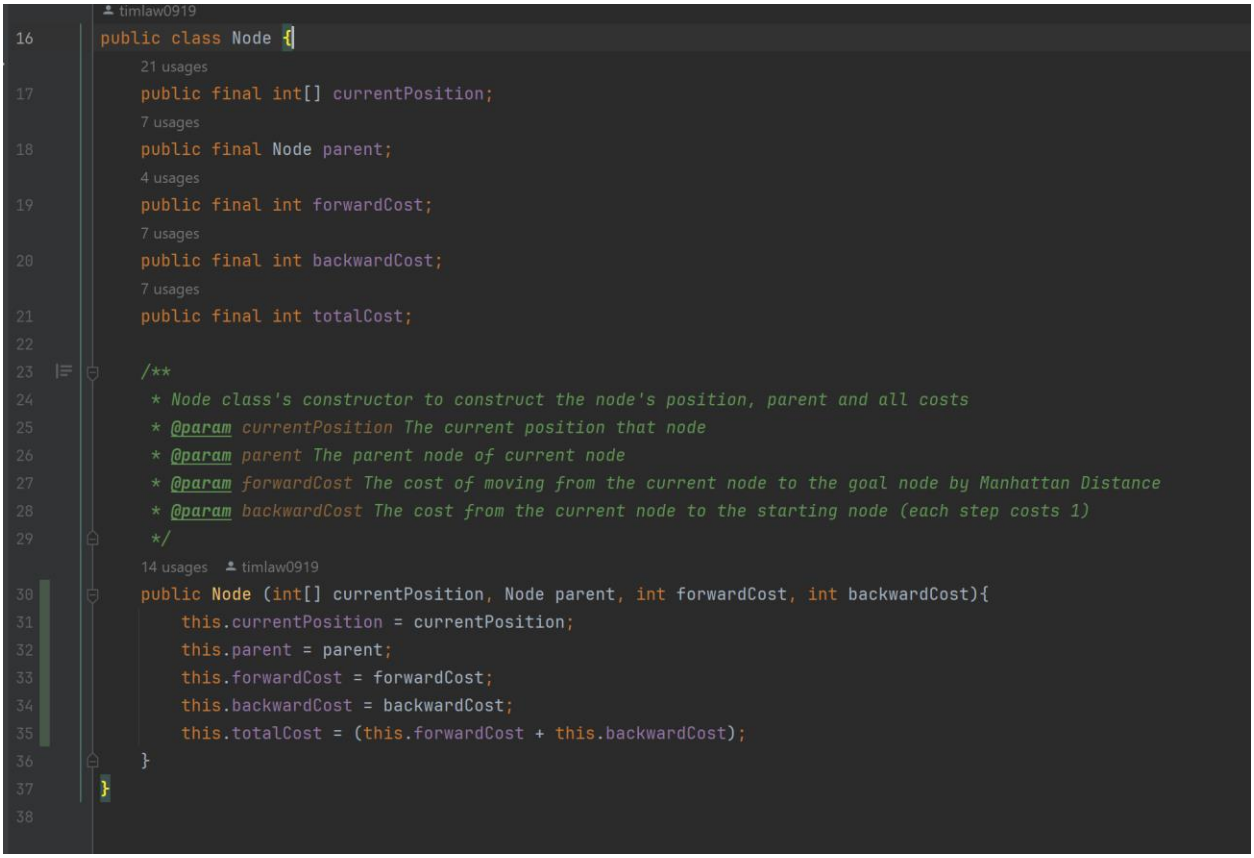
```java
public class MazeGUI extends Application {

    3 usages
    private static final int CELL_SIZE = 10;

    /**
     * Starts the maze GUI.
     *
     * @param primaryStage The primary stage to display the GUI.
     */
    LOWingYan +1
    @Override
    public void start(Stage primaryStage) {
        // Load the maze data from the CSV file
        int[][] mazeData = MazeLoader.loadMazeFromCSV( filePath: "maze_map.csv");

        // Create a GridPane to hold the maze cells
        GridPane gridPane = new GridPane();

        // Populate the GridPane with rectangles representing the maze cells
        for (int i = 0; i < mazeData.length; i++) {
            for (int j = 0; j < mazeData[i].length; j++) {
                Rectangle cell = new Rectangle(CELL_SIZE, CELL_SIZE);

                // Set the color of the cell based on the value in the maze data
                if (mazeData[i][j] == CellState.BLOCK.ordinal()) {
                    cell.setFill(Color.GRAY); // Block
                }
                else if(mazeData[i][j] == CellState.PATH.ordinal()){
                    cell.setFill(Color.WHITE); // Path
                }
```

```java
                    cell.setFill(Color.WHITE); // Path
                }
                else if(mazeData[i][j]==CellState.ENTRY.ordinal()){
                    cell.setFill(Color.YELLOW); // Entry
                }
                else if(mazeData[i][j]==CellState.EXIT.ordinal()){
                    cell.setFill(Color.LIGHTGREEN); //Exit
                }

                // Add the cell to the GridPane
                gridPane.add(cell, j, i);
            }
        }

        Pane pane = new Pane();
        Button backTestingMenuButton = new Button( s: "Back to Testing Menu");
        backTestingMenuButton.setLayoutX(CELL_SIZE * (mazeData.length ) + 10);
        backTestingMenuButton.setLayoutY(125);
        pane.getChildren().addAll(gridPane, backTestingMenuButton);

        backTestingMenuButton.setOnAction(actionEvent -> {
            BigMainGUI bigMainGUI = new BigMainGUI();
            try {
                bigMainGUI.start(primaryStage);} catch (Exception e) {throw new RuntimeException(e);}
        });

        // Create the scene and set it on the stage
        Scene scene = new Scene(pane);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Maze GUI");
        primaryStage.show();
    }
}
```

5. MazeLoader

```java
public class MazeLoader {
    /**
     * Loads a maze from a CSV file.
     *
     * @param filePath The path of the CSV file to load.
     * @return A 2D array representing the loaded maze.
     */
    16 usages  LOWingYan +1
    public static int[][] loadMazeFromCSV(String filePath) {
        int[][] maze = null;

        try (BufferedReader br = new BufferedReader(new FileReader(filePath))) {
            String line;
            int rowCount = 0;
            while ((line = br.readLine()) != null) {
                String[] cells = line.split( regex: ",");
                if (maze == null) {
                    maze = new int[cells.length][cells.length];
                }
                for (int colCount = 0; colCount < cells.length; colCount++) {
                    maze[rowCount][colCount] = Integer.parseInt(cells[colCount]);
                }
                rowCount++;
            }
        } catch (IOException e) {e.printStackTrace();}

        return maze;
    }
}
```

6. Node

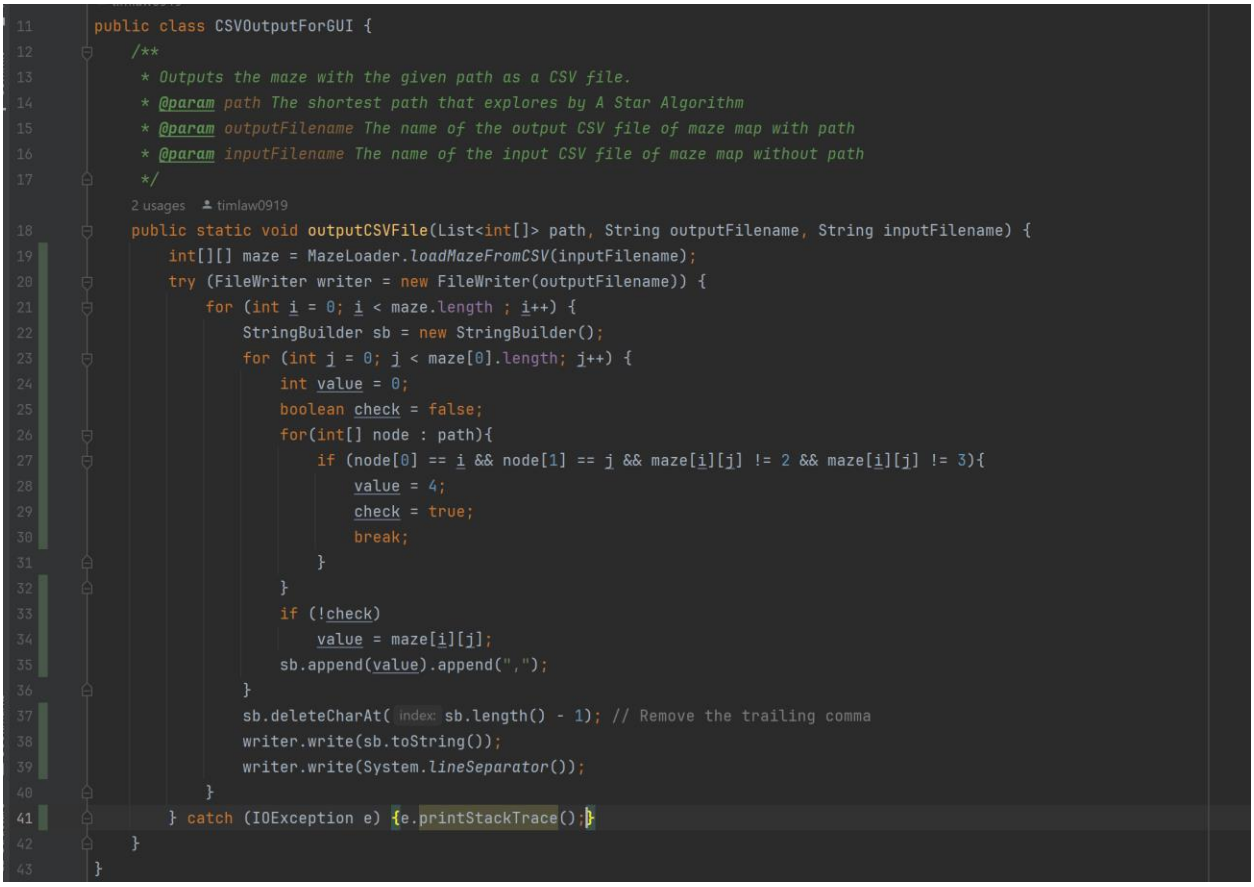


```
timlaw0919
public class Node {
    21 usages
    public final int[] currentPosition;
    7 usages
    public final Node parent;
    4 usages
    public final int forwardCost;
    7 usages
    public final int backwardCost;
    7 usages
    public final int totalCost;

    /**
     * Node class's constructor to construct the node's position, parent and all costs
     * @param currentPosition The current position that node
     * @param parent The parent node of current node
     * @param forwardCost The cost of moving from the current node to the goal node by Manhattan Distance
     * @param backwardCost The cost from the current node to the starting node (each step costs 1)
     */
    14 usages  timlaw0919
    public Node (int[] currentPosition, Node parent, int forwardCost, int backwardCost){
        this.currentPosition = currentPosition;
        this.parent = parent;
        this.forwardCost = forwardCost;
        this.backwardCost = backwardCost;
        this.totalCost = (this.forwardCost + this.backwardCost);
    }
}
```

7. Function B - CSVOutput

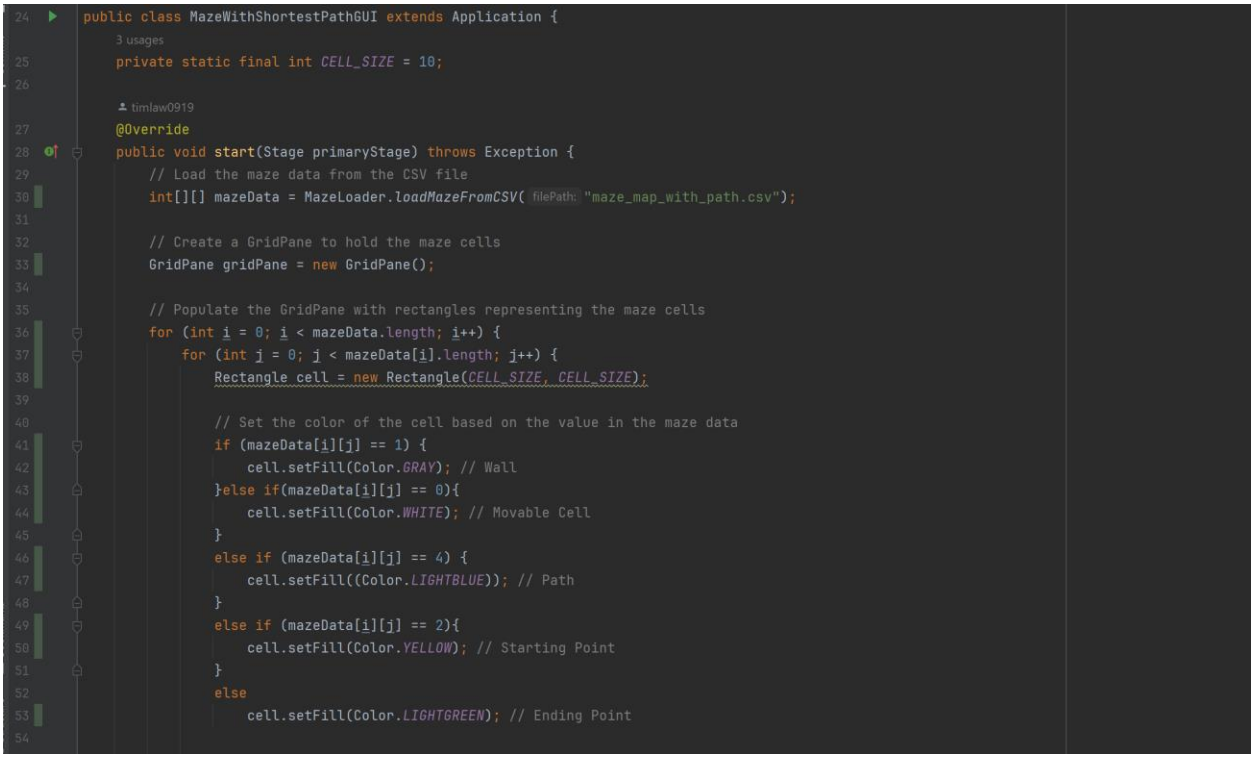
```java
public class CSVOutput {
    /**
     * Output a CSV file containing the shortest path coordinates.
     * @param path The shortest path that explores by A Star Algorithm
     * @param filename The file name of output
     */
    2 usages  timlaw0919
    public static void outputCSVFile(List<int[]> path, String filename) {

        try (BufferedWriter writer = new BufferedWriter(new FileWriter(filename))) {
            writer.write( str: "Shortest Path (Read the coordinate line by line)");
            writer.newLine();
            writer.write( str: "Start-> ");
            writer.write( str: ",");
            int j = 0;
            for (int k = path.size() - 1; k >= 0; k--) {
                int[] array = path.get(k);
                for (int i = 0; i < array.length; i++) {
                    writer.write(Integer.toString(array[i]));
                    if (i == 0)
                        writer.write( c: ';');
                }
                writer.write( c: ',');
                if ((j + 1) % 10 == 0){
                    writer.newLine();
                    writer.write( str: ",");
                }
                j++;
            }
            writer.write( str: " -> End");
        } catch (IOException e) {e.printStackTrace();}
    }
}
```

8. CSVOutuptForGUI

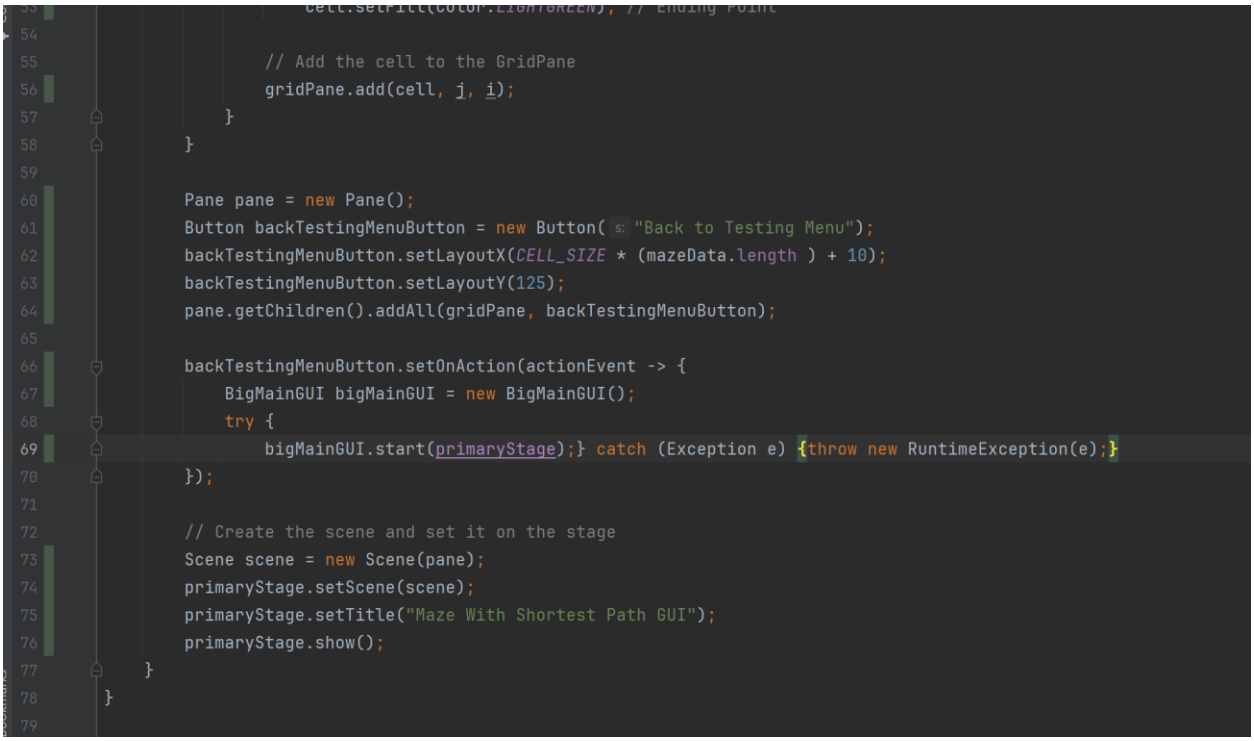


```java
public class CSVOutputForGUI {
    /**
     * Outputs the maze with the given path as a CSV file.
     * @param path The shortest path that explores by A Star Algorithm
     * @param outputFilename The name of the output CSV file of maze map with path
     * @param inputFilename The name of the input CSV file of maze map without path
     */
    2 usages  timlaw0919
    public static void outputCSVFile(List<int[]> path, String outputFilename, String inputFilename) {
        int[][] maze = MazeLoader.loadMazeFromCSV(inputFilename);
        try (FileWriter writer = new FileWriter(outputFilename)) {
            for (int i = 0; i < maze.length ; i++) {
                StringBuilder sb = new StringBuilder();
                for (int j = 0; j < maze[0].length; j++) {
                    int value = 0;
                    boolean check = false;
                    for(int[] node : path){
                        if (node[0] == i && node[1] == j && maze[i][j] != 2 && maze[i][j] != 3){
                            value = 4;
                            check = true;
                            break;
                        }
                    }
                    if (!check)
                        value = maze[i][j];
                    sb.append(value).append(",");
                }
                sb.deleteCharAt( index: sb.length() - 1); // Remove the trailing comma
                writer.write(sb.toString());
                writer.write(System.lineSeparator());
            }
        } catch (IOException e) {e.printStackTrace();}
    }
}
```

9. MazeWithShortestPathGUI

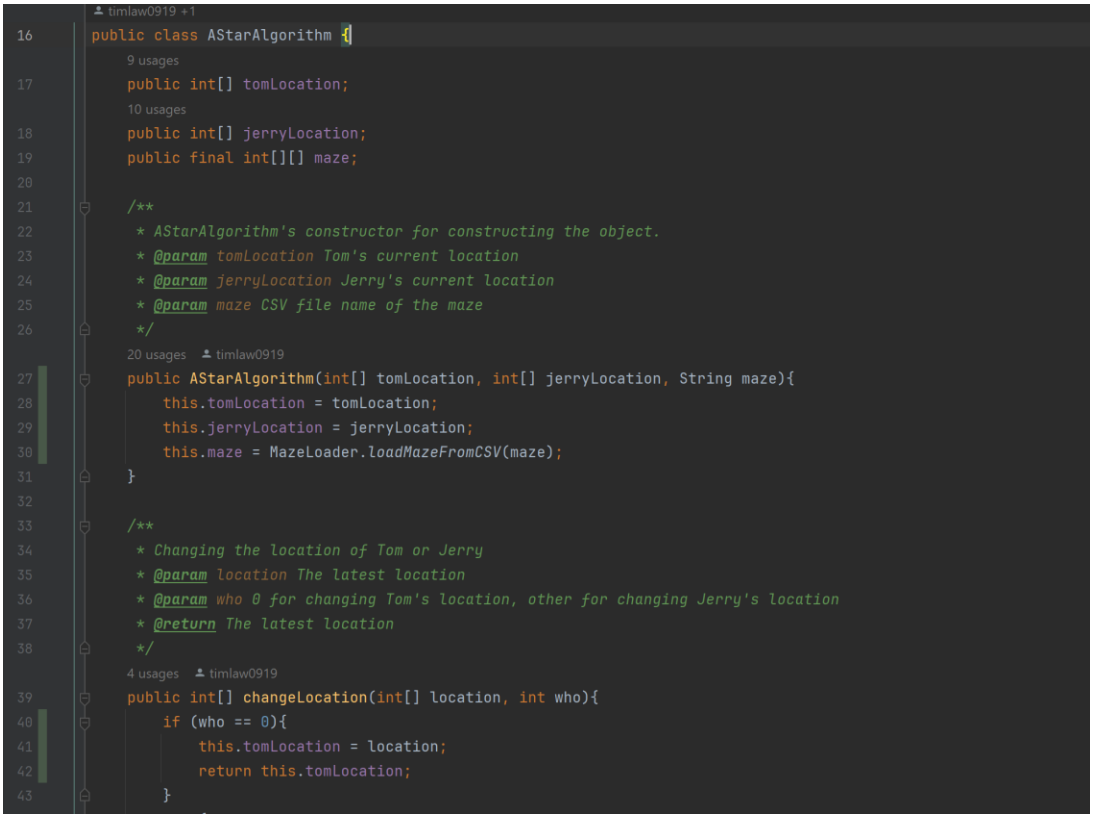


```
public class MazeWithShortestPathGUI extends Application {
    3 usages
    private static final int CELL_SIZE = 10;

    timlaw0919
    @Override
    public void start(Stage primaryStage) throws Exception {
        // Load the maze data from the CSV file
        int[][] mazeData = MazeLoader.loadMazeFromCSV( filePath: "maze_map_with_path.csv");

        // Create a GridPane to hold the maze cells
        GridPane gridPane = new GridPane();

        // Populate the GridPane with rectangles representing the maze cells
        for (int i = 0; i < mazeData.length; i++) {
            for (int j = 0; j < mazeData[i].length; j++) {
                Rectangle cell = new Rectangle(CELL_SIZE, CELL_SIZE);

                // Set the color of the cell based on the value in the maze data
                if (mazeData[i][j] == 1) {
                    cell.setFill(Color.GRAY); // Wall
                }else if(mazeData[i][j] == 0){
                    cell.setFill(Color.WHITE); // Movable Cell
                }
                else if (mazeData[i][j] == 4) {
                    cell.setFill((Color.LIGHTBLUE)); // Path
                }
                else if (mazeData[i][j] == 2){
                    cell.setFill(Color.YELLOW); // Starting Point
                }
                else
                    cell.setFill(Color.LIGHTGREEN); // Ending Point
```

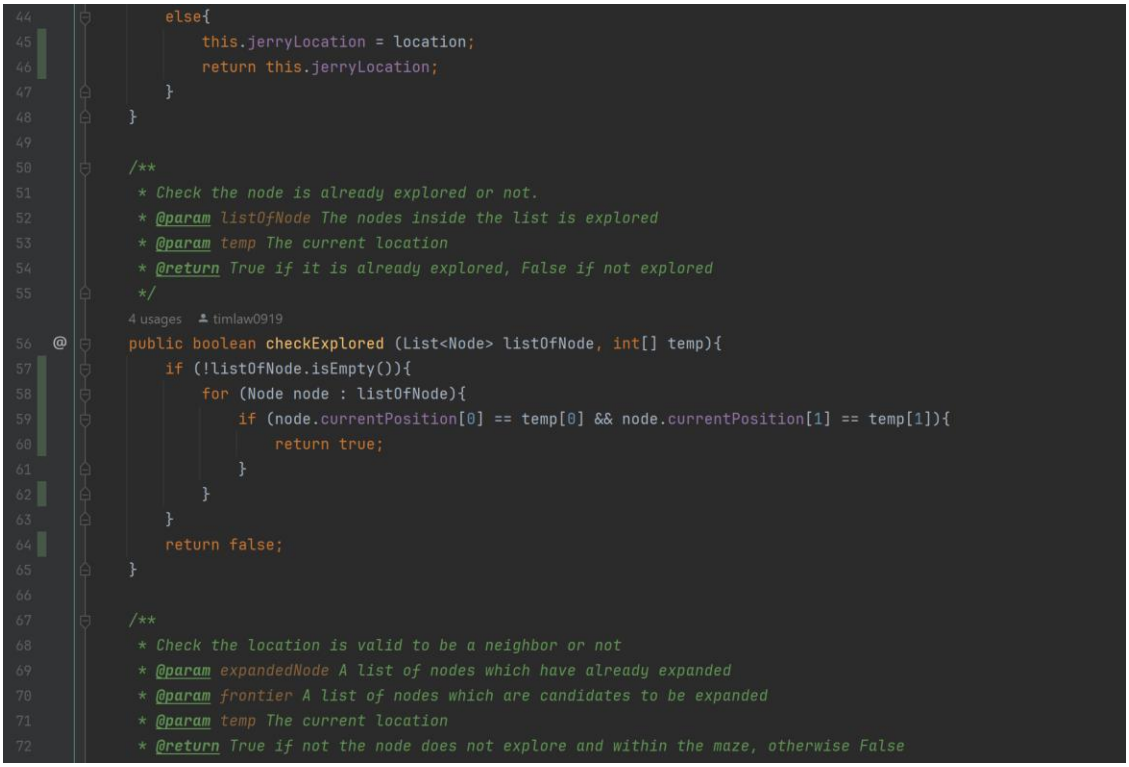


```
                // Add the cell to the GridPane
                gridPane.add(cell, j, i);
            }
        }

        Pane pane = new Pane();
        Button backTestingMenuButton = new Button( s: "Back to Testing Menu");
        backTestingMenuButton.setLayoutX(CELL_SIZE * (mazeData.length ) + 10);
        backTestingMenuButton.setLayoutY(125);
        pane.getChildren().addAll(gridPane, backTestingMenuButton);

        backTestingMenuButton.setOnAction(actionEvent -> {
            BigMainGUI bigMainGUI = new BigMainGUI();
            try {
                bigMainGUI.start(primaryStage);} catch (Exception e) {throw new RuntimeException(e);}
        });

        // Create the scene and set it on the stage
        Scene scene = new Scene(pane);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Maze With Shortest Path GUI");
        primaryStage.show();
    }
}
```

10. AStarAlgorithm



```
timlaw0919 +1
public class AStarAlgorithm {
    9 usages
    public int[] tomLocation;
    10 usages
    public int[] jerryLocation;
    public final int[][] maze;

    /**
     * AStarAlgorithm's constructor for constructing the object.
     * @param tomLocation Tom's current location
     * @param jerryLocation Jerry's current location
     * @param maze CSV file name of the maze
     */
    20 usages  timlaw0919
    public AStarAlgorithm(int[] tomLocation, int[] jerryLocation, String maze){
        this.tomLocation = tomLocation;
        this.jerryLocation = jerryLocation;
        this.maze = MazeLoader.loadMazeFromCSV(maze);
    }

    /**
     * Changing the location of Tom or Jerry
     * @param location The latest location
     * @param who 0 for changing Tom's location, other for changing Jerry's location
     * @return The latest location
     */
    4 usages  timlaw0919
    public int[] changeLocation(int[] location, int who){
        if (who == 0){
            this.tomLocation = location;
            return this.tomLocation;
        }
```



```
        else{
            this.jerryLocation = location;
            return this.jerryLocation;
        }
    }

    /**
     * Check the node is already explored or not.
     * @param listOfNode The nodes inside the list is explored
     * @param temp The current location
     * @return True if it is already explored, False if not explored
     */
    4 usages  timlaw0919
    public boolean checkExplored (List<Node> listOfNode, int[] temp){
        if (!listOfNode.isEmpty()){
            for (Node node : listOfNode){
                if (node.currentPosition[0] == temp[0] && node.currentPosition[1] == temp[1]){
                    return true;
                }
            }
        }
        return false;
    }

    /**
     * Check the location is valid to be a neighbor or not
     * @param expandedNode A list of nodes which have already expanded
     * @param frontier A list of nodes which are candidates to be expanded
     * @param temp The current location
     * @return True if not the node does not explore and within the maze, otherwise False
```

```java
     * @return True if not the node does not explore and within the maze, otherwise False
     */
    3 usages  timlaw0919
    public boolean checkValidNode(List<Node> expandedNode, List<Node> frontier, int[] temp){
        if (!checkExplored(expandedNode, temp) && !checkExplored(frontier, temp) && temp[0] >= 0 && temp[0] < this.maze.length
                && temp[1] >= 0 && temp[1] < this.maze[0].length && this.maze[temp[0]][temp[1]] != 1){
            return true;
        }
        return false;
    }

    /**
     * Find all neighbor near the current node
     * @param currentNode The current node
     * @param expandedNode A list of nodes which have already expanded
     * @param frontier A list of nodes which are candidates to be expanded
     * @return A list of valid neighbor with type Node
     */
    3 usages  timlaw0919
    public List<Node> findNeighbor(Node currentNode, List<Node> expandedNode, List<Node> frontier){
        List<Node> neighbor = new ArrayList<>();
        int[][] fourDirection = {{0,1}, {1,0}, {0,-1}, {-1,0}};
        for (int[] direction : fourDirection){
            int[] temp = Arrays.copyOf(currentNode.currentPosition, currentNode.currentPosition.length);
            temp[0] += direction[0];
            temp[1] += direction[1];
            if (checkValidNode(expandedNode, frontier, temp)){
                neighbor.add(new Node(temp, currentNode, (Math.abs(temp[0] - this.jerryLocation[0]) + Math.abs(temp[1] - this.jerryLocation[1])), backwardCost: current
            }
        }
        return neighbor;
    }

    /**
```

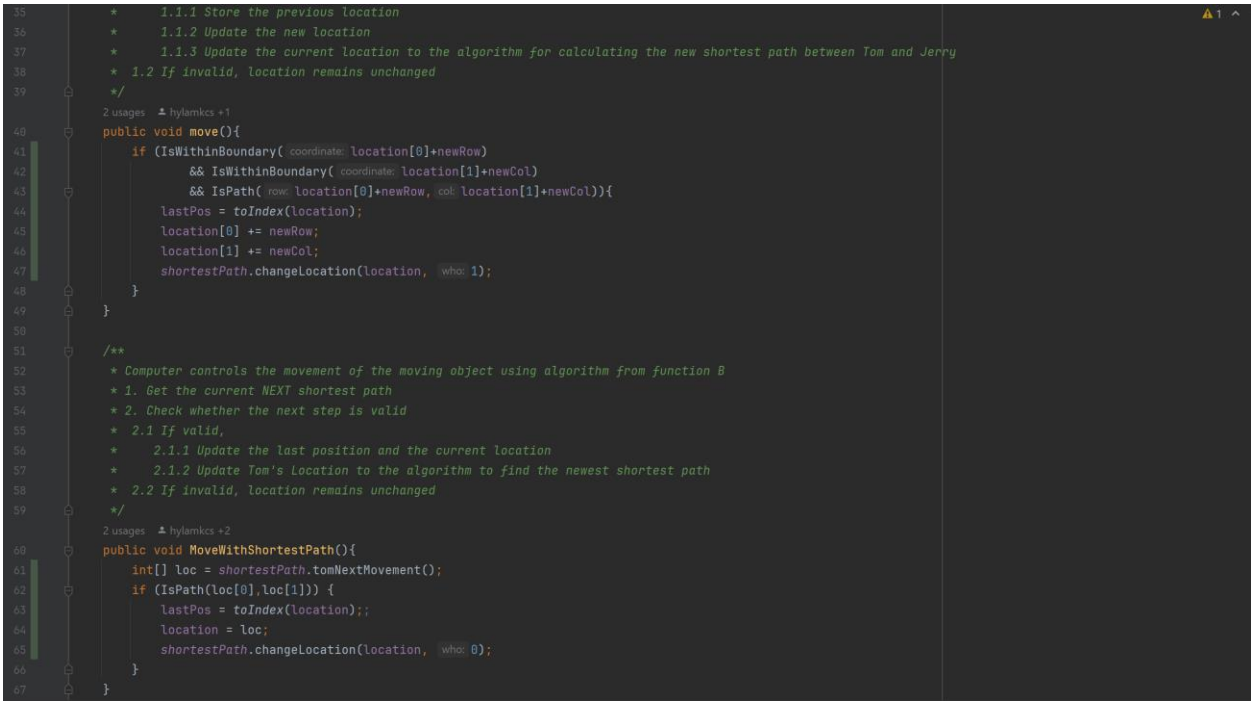
```java
    /**
     * Generate the shortest path between Tom and Jerry
     * @return A list of coordinate with type int[] which form the shortest path
     */
    6 usages  timlaw0919
    public List<int[]> pathGeneratorByAStar(){
        List<int[]> path = new ArrayList<>(); //The final shortest path
        List<Node> expandedNode = new ArrayList<>(); // The nodes that are expanded (Only if the node is first time expands, then it may be the shortest path)
        List<Node> frontier = new ArrayList<>(); // The potential candidates for expansion

        Node tomNode = new Node(tomLocation, parent: null, (Math.abs(tomLocation[0] - this.jerryLocation[0]) + Math.abs(tomLocation[1] - this.jerryLocation[1])), ba
        frontier.add(tomNode);

        while (!(frontier.isEmpty())){
            // Find out the lowest total cost node + Remove it from frontier + Add to expanded node
            int minimumTotalCost = 10000;
            int index = -1;
            for (Node temp : frontier){
                if (temp.totalCost < minimumTotalCost) {
                    minimumTotalCost = temp.totalCost;
                    index = frontier.indexOf(temp);
                }
            }
            Node current = frontier.get(index);
            expandedNode.add(frontier.get(index));
            frontier.remove(index);

            // Reach the goal state, add the path coordinate
            if (current.currentPosition[0] == jerryLocation[0] && current.currentPosition[1] == jerryLocation[1]){
                Node temp = current;
                while (temp.parent != null) {
                    path.add(temp.currentPosition);
                    temp = temp.parent;
                }
                path.add(temp.currentPosition);
```

```
                path.add(temp.currentPosition);
                break;
            }

            // Find the valid neighbor of current node + add into frontier
            frontier.addAll(findNeighbor(current, expandedNode, frontier));
        }
        return path;
    }

    /**
     * Tom's next action
     * @return The coordinate of Tom's next movement.
     */
    4 usages  timlaw0919 +1
    public int[] tomNextMovement(){
        List<int[]> path = pathGeneratorByAStar();

        // At least having one path between Tom and Jerry
        if (path.size() >= 2){
            return path.get(path.size() - 2);
        }

        // No path between Tom and Jerry
        else {
            List<Node> temp = new ArrayList<>();
            List<Node> neighbor = findNeighbor(new Node(this.tomLocation,  parent: null,  forwardCost: 0,  backwardCost: 0), temp, temp);
            // Some movable cell near Tom
            if (!neighbor.isEmpty()) {
                return neighbor.get(0).currentPosition;
            }
            // Tom is surrounded by barrier
            else {
                return this.tomLocation;
            }
```

```
                return this.tomLocation;
            }
        }
    }
}
```

11. Character

```java
public class Character {

    15 usages
    public int newCol;
    15 usages
    public int newRow;
    19 usages
    public int lastPos;                // Previous Location (index)
    public int[] location;             // Current location (Coordinates)
    16 usages
    public boolean Game_state;         // Current Game state
    7 usages
    public int speed;                  // Speed of the moving character

    /**
     * Constructor of Character
     * Initialize all the attributes
     */
    hylamkcs
    public Character() {
        newRow = 0;
        newCol = 0;
        location = new int[]{0, 0};
        Game_state = false;
        speed = 0;
        lastPos = toIndex(location);
    }

    /**
     * Player controls the direction of the moving object using keyboard
     * 1. Check whether the new location is valid for moving
     *  1.1 If valid,
```



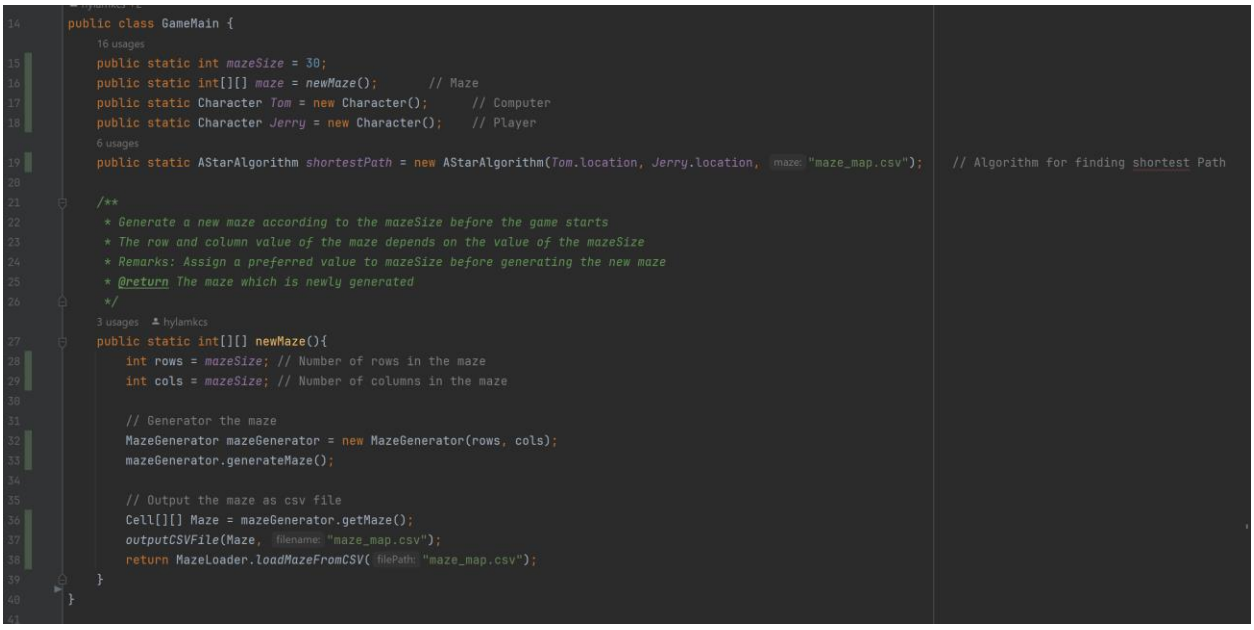
```java
     *      1.1.1 Store the previous location
     *      1.1.2 Update the new location
     *      1.1.3 Update the current location to the algorithm for calculating the new shortest path between Tom and Jerry
     *  1.2 If invalid, location remains unchanged
     */
    2 usages  hylamkcs +1
    public void move(){
        if (IsWithinBoundary( coordinate: location[0]+newRow)
                && IsWithinBoundary( coordinate: location[1]+newCol)
                && IsPath( row: location[0]+newRow, col: location[1]+newCol)){
            lastPos = toIndex(location);
            location[0] += newRow;
            location[1] += newCol;
            shortestPath.changeLocation(location,  who: 1);
        }
    }

    /**
     * Computer controls the movement of the moving object using algorithm from function B
     * 1. Get the current NEXT shortest path
     * 2. Check whether the next step is valid
     *  2.1 If valid,
     *     2.1.1 Update the last position and the current location
     *     2.1.2 Update Tom's Location to the algorithm to find the newest shortest path
     *  2.2 If invalid, location remains unchanged
     */
    2 usages  hylamkcs +2
    public void MoveWithShortestPath(){
        int[] loc = shortestPath.tomNextMovement();
        if (IsPath(loc[0],loc[1])) {
            lastPos = toIndex(location);;
            location = loc;
            shortestPath.changeLocation(location,  who: 0);
        }
    }
```

```java
    /**
     * Check whether the new coordinate of the location is valid in maze
     * @param coordinate The coordinate of the row/column of the location
     * @return true if the coordinate is non-negative and within the maze size
     */
    4 usages  hylamkcs
    public boolean IsWithinBoundary(int coordinate) { return coordinate >= 0 && coordinate < maze.length; }

    /**
     * Check whether the new location is valid for the object to move
     * @param row The row value of the new location
     * @param col The column value of the new location
     * @return true if the location on the maze is not a BLOCK
     */
    4 usages  hylamkcs
    public boolean IsPath(int row, int col) { return maze[row][col] != 1; }

    /**
     * Convert the 2D location to 1D index based on the length of the maze
     * @param location An int array storing the row and column value of a location
     * @return the index value in 1D
     */
    12 usages  hylamkcs
    public static int toIndex(int[] location) { return maze.length*location[0]+location[1]; }

    /**
     * Reset the status of the character when the game is restarted
     * 1. Store the current location to last position
     * 2. Reset the character's location back to spawn point
     * 3. Reset the newRow and newCol to 0 indicating no movement exists
     * 4. Set the game state to false
     * The value of lastPos and location is used to update the game maze interface
     * @param row The row value of the spawn point
     * @param col The column value of the spawn point
     */
    3 usages  hylamkcs
    public void reset(int row, int col){
        lastPos = toIndex(location);
        location[0] = row;
        location[1] = col;
        newRow = 0;
        newCol = 0;
        Game_state = false;
    }

}
```

12. CheckEndGame

```java
package FunctionC_TomCatchJerry;

import ...

/**
 * CheckEndGame class is to update the game state of character and check whether a game is ended.
 */
hylamkcs
public class CheckEndGame {
    2 usages
    public final int ExitPoint = toIndex(Tom.location);

    /**
     * Check whether the game is finished
     * If Jerry locates at exit point, Jerry wins
     * If Jerry has the same location with Tom, Tom wins
     * @return true if either Tom or Jerry wins the game
     */
    5 usages  hylamkcs
    public boolean isEndGame(){
        if (toIndex(Jerry.location) == ExitPoint) {
            Jerry.Game_state = true;
        }
        else if (toIndex(Jerry.location) == toIndex(Tom.location)) {
            Tom.Game_state = true;
        }
        return (Jerry.Game_state && !Tom.Game_state) ||
                (!Jerry.Game_state && Tom.Game_state);
    }
}
```

13. GameMain



```java
public class GameMain {
    16 usages
    public static int mazeSize = 30;
    public static int[][] maze = newMaze();       // Maze
    public static Character Tom = new Character();      // Computer
    public static Character Jerry = new Character();    // Player
    6 usages
    public static AStarAlgorithm shortestPath = new AStarAlgorithm(Tom.location, Jerry.location, maze: "maze_map.csv");    // Algorithm for finding shortest Path

    /**
     * Generate a new maze according to the mazeSize before the game starts
     * The row and column value of the maze depends on the value of the mazeSize
     * Remarks: Assign a preferred value to mazeSize before generating the new maze
     * @return The maze which is newly generated
     */
    3 usages  hylamkcs
    public static int[][] newMaze(){
        int rows = mazeSize; // Number of rows in the maze
        int cols = mazeSize; // Number of columns in the maze

        // Generator the maze
        MazeGenerator mazeGenerator = new MazeGenerator(rows, cols);
        mazeGenerator.generateMaze();

        // Output the maze as csv file
        Cell[][] Maze = mazeGenerator.getMaze();
        outputCSVFile(Maze, filename: "maze_map.csv");
        return MazeLoader.loadMazeFromCSV( filePath: "maze_map.csv");
    }
}
```

14. InfoGUI

```java
LOWingYan +2
public class InfoGUI extends Application {
    /**
     * Build the GUI for the Info page of the game.
     * Game rule is shown on this page.
     * @param stage Set the scene of the Info page of the maze game
     */
    LOWingYan +1
    public void start(Stage stage) throws Exception {
        Pane root = new Pane();

        Rectangle box = new Rectangle( v: 300,  v1: 300);
        box.setFill(Color.web( s: "#e2d5bb"));
        box.setStroke(Color.web( s: "#c08129bd"));
        box.setStrokeWidth(10);
        box.setArcHeight(5);
        box.setArcWidth(5);

        Label gameRuleLabel = new Label( s: "GAME RULE");
        gameRuleLabel.setFont(new Font( s: "Berlin Sans FB Demi Bold",  v: 22));
        gameRuleLabel.setTextFill(Color.web( s: "#1e4631"));
        gameRuleLabel.setLayoutX(87);
        gameRuleLabel.setLayoutY(26);

        Button okButton = new Button( s: "OK!");
        okButton.setLayoutX(38);
        okButton.setLayoutY(249);

        ImageView tomImageView = new ImageView(new Image( s: "file:tom.png"));
        tomImageView.setLayoutX(214);
        tomImageView.setLayoutY(249);
        tomImageView.setFitWidth(27);
        tomImageView.setFitHeight(27);
```

```java
        ImageView jerryImageView = new ImageView(new Image( s: "file:jerry.png"));
        jerryImageView.setLayoutX(134);
        jerryImageView.setLayoutY(249);
        jerryImageView.setFitWidth(27);
        jerryImageView.setFitHeight(27);

        Label donBeCaughtLabel = new Label( s: "Don't Be Caught by TOM");
        donBeCaughtLabel.setFont(new Font( s: "Berlin Sans FB",  v: 12));
        donBeCaughtLabel.setTextFill(Color.web( s: "#d52332"));
        donBeCaughtLabel.setLayoutX(29);
        donBeCaughtLabel.setLayoutY(59);

        Label jerryLabel = new Label( s: "Jerry");
        jerryLabel.setFont(Font.font( s: "Berlin Sans FB Demi Bold",  v: 12));
        jerryLabel.setTextFill(Color.web( s: "#c09129"));
        jerryLabel.setLayoutX(165);
        jerryLabel.setLayoutY(254);

        Label tomLabel = new Label( s: "Tom");
        tomLabel.setFont(Font.font( s: "Berlin Sans FB Demi Bold",  v: 12));
        tomLabel.setTextFill(Color.web( s: "#808990"));
        tomLabel.setLayoutX(245);
        tomLabel.setLayoutY(255);

        Label goToExitLabel = new Label( s: "Go To The Exit Point");
        goToExitLabel.setFont(Font.font( s: "Berlin Sans FB",  v: 12));
        goToExitLabel.setTextFill(Color.web( s: "#d52332"));
        goToExitLabel.setLayoutX(173);
        goToExitLabel.setLayoutY(59);
```

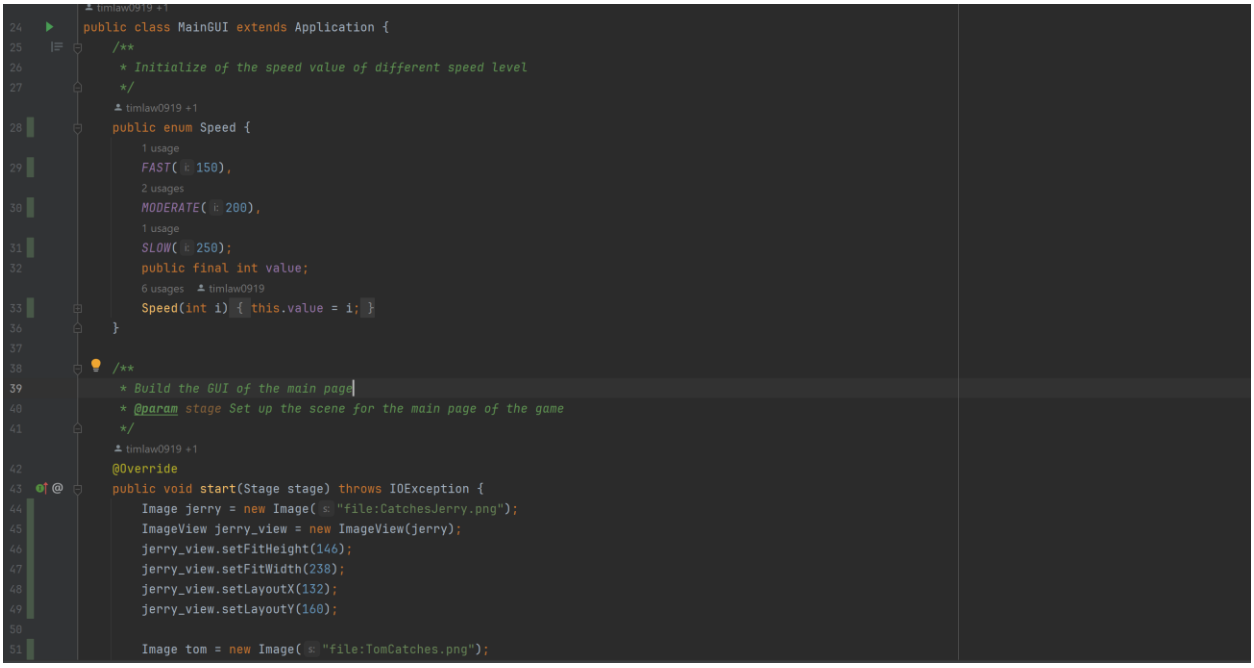
```
81
82          Label andLabel = new Label( s: "&");
83          andLabel.setFont(Font.font( s: "Berlin Sans FB",  v: 12));
84          andLabel.setTextFill(Color.web( s: "#d9838b"));
85          andLabel.setLayoutX(160);
86          andLabel.setLayoutY(59);
87
88          Font font = new Font( s: "Berlin Sans FB Demi Bold",  v: 12.0);
89
90          Button wButton = new Button( s: "W");
91          wButton.setFont(font);
92          wButton.setLayoutX(138);
93          wButton.setLayoutY(103);
94          wButton.setMinSize( v: 27,  v1: 27);
95
96          Button dButton = new Button( s: "D");
97          dButton.setFont(font);
98          dButton.setLayoutX(181);
99          dButton.setLayoutY(151);
100         dButton.setMinSize( v: 27,  v1: 27);
101
102         Button sButton = new Button( s: "S");
103         sButton.setFont(font);
104         sButton.setLayoutX(138);
105         sButton.setLayoutY(151);
106         sButton.setMinSize( v: 27,  v1: 27);
107
108         Button aButton = new Button( s: "A");
109         aButton.setFont(font);
110         aButton.setLayoutX(95);
111         aButton.setLayoutY(151);
112         aButton.setMinSize( v: 27,  v1: 27);
113
114         Label upLabel = new Label( s: "UP");
115         upLabel.setFont(font);
116         upLabel.setTextFill(Color.web( s: "#040404"));
117         upLabel.setLayoutX(144);
118         upLabel.setLayoutY(133);
119
120         Label leftLabel = new Label( s: "LEFT");
121         leftLabel.setFont(font);
122         leftLabel.setTextFill(Color.web( s: "#040404"));
123         leftLabel.setLayoutX(93);
124         leftLabel.setLayoutY(182);
125
126         Label downLabel = new Label( s: "DOWN");
127         downLabel.setFont(font);
128         downLabel.setTextFill(Color.web( s: "#040404"));
129         downLabel.setLayoutX(132);
130         downLabel.setLayoutY(182);
131
132         Label rightLabel = new Label( s: "RIGHT");
133         rightLabel.setFont(font);
134         rightLabel.setTextFill(Color.web( s: "#040404"));
135         rightLabel.setLayoutX(181);
136         rightLabel.setLayoutY(182);
137
138         Label you = new Label( s: "(You)");
139         you.setFont(font);
140         you.setTextFill(Color.web( s: "#c09129"));
141         you.setLayoutX(162);
142         you.setLayoutY(267);
143
144         Label comp = new Label( s: "(Comp)");
145         comp.setFont(font);
146         comp.setTextFill(Color.web( s: "#808990"));
147         comp.setLayoutX(244);
```

```
        comp.setTextFill(Color.web( s: "#808990"));
        comp.setLayoutX(244);
        comp.setLayoutY(266);

        root.getChildren().addAll(box,gameRuleLabel,okButton,tomImageView,jerryImageView,donBeCaughtLabel,jerryLabel,
                tomLabel,goToExitLabel,andLabel,wButton,sButton,aButton,dButton,upLabel,leftLabel, rightLabel,downLabel,you,comp);

        okButton.setOnAction(actionEvent -> {
            MainGUI mainGUI = new MainGUI();
            try {
                mainGUI.start(stage);} catch (IOException e) {throw new RuntimeException(e);}
        });

        Scene startPage = new Scene(root, v: 300, v1: 300);
        stage.setScene(startPage);
        stage.show();
    }

//    public static void main(String[] args) {
//        launch(args);
//    }
}
```

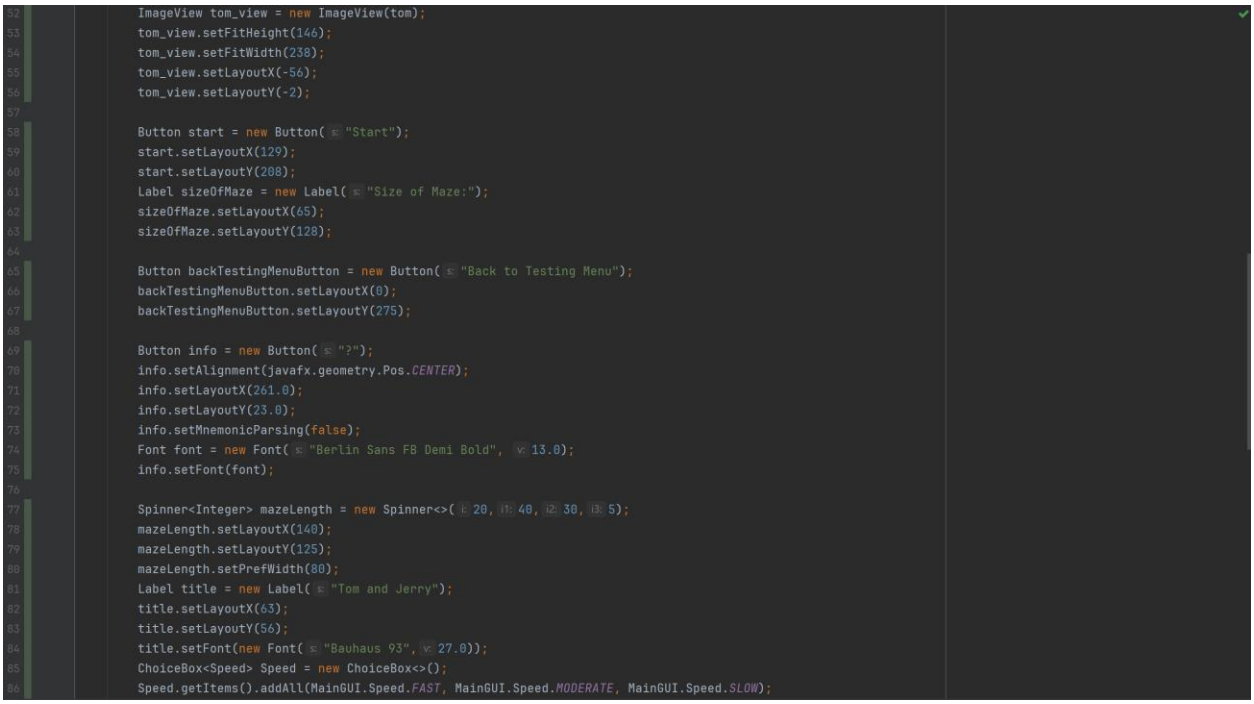
15. KeyBoardListener

```
hylamkcs
public class KeyBoardListener extends KeyAdapter{
    10 usages
    public Character player;

    /**
     * @param player The character controlled by player
     */
    6 usages  hylamkcs
    public KeyBoardListener (Character player) { this.player = player; }

    /**
     * Get the inputs from the keyboard to update the direction of the moving object
     * @param e The input received from the keyboard
     */
    6 usages  hylamkcs
    public void keyPressed(KeyEvent e){
        switch(e.getCode()){
            case W:     // Upward
                player.newCol = 0;
                player.newRow = -1;
                break;
            case A:     // Left
                player.newCol = -1;
                player.newRow = 0;
                break;

            case S:     // Downward
                player.newCol = 0;
                player.newRow = 1;
                break;

            case D:     // Right
                player.newCol = 1;
                player.newRow = 0;
                break;

            default:
                break;
        }
    }
}
```

16. MainGUI



```
timlaw0919 +1
public class MainGUI extends Application {
    /**
     * Initialize of the speed value of different speed level
     */
    timlaw0919 +1
    public enum Speed {
        1 usage
        FAST( i: 150),
        2 usages
        MODERATE( i: 200),
        1 usage
        SLOW( i: 250);
        public final int value;
        6 usages  timlaw0919
        Speed(int i) { this.value = i; }
    }

    /**
     * Build the GUI of the main page
     * @param stage Set up the scene for the main page of the game
     */
    timlaw0919 +1
    @Override
    public void start(Stage stage) throws IOException {
        Image jerry = new Image( s: "file:CatchesJerry.png");
        ImageView jerry_view = new ImageView(jerry);
        jerry_view.setFitHeight(146);
        jerry_view.setFitWidth(238);
        jerry_view.setLayoutX(132);
        jerry_view.setLayoutY(160);

        Image tom = new Image( s: "file:TomCatches.png");
```

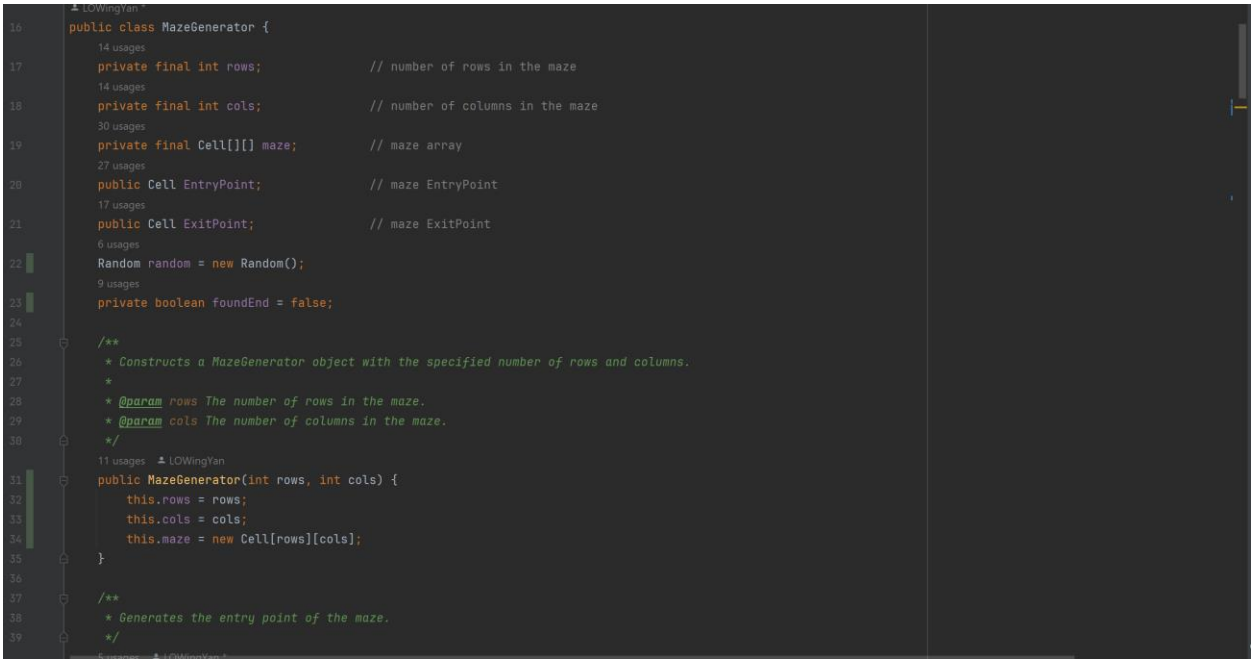


```
        ImageView tom_view = new ImageView(tom);
        tom_view.setFitHeight(146);
        tom_view.setFitWidth(238);
        tom_view.setLayoutX(-56);
        tom_view.setLayoutY(-2);

        Button start = new Button( s: "Start");
        start.setLayoutX(129);
        start.setLayoutY(208);
        Label sizeOfMaze = new Label( s: "Size of Maze:");
        sizeOfMaze.setLayoutX(65);
        sizeOfMaze.setLayoutY(128);

        Button backTestingMenuButton = new Button( s: "Back to Testing Menu");
        backTestingMenuButton.setLayoutX(0);
        backTestingMenuButton.setLayoutY(275);

        Button info = new Button( s: "?");
        info.setAlignment(javafx.geometry.Pos.CENTER);
        info.setLayoutX(261.0);
        info.setLayoutY(23.0);
        info.setMnemonicParsing(false);
        Font font = new Font( s: "Berlin Sans FB Demi Bold",  v: 13.0);
        info.setFont(font);

        Spinner<Integer> mazeLength = new Spinner<>( i: 20, i1: 40, i2: 30, i3: 5);
        mazeLength.setLayoutX(140);
        mazeLength.setLayoutY(125);
        mazeLength.setPrefWidth(80);
        Label title = new Label( s: "Tom and Jerry");
        title.setLayoutX(63);
        title.setLayoutY(56);
        title.setFont(new Font( s: "Bauhaus 93",  v: 27.0));
        ChoiceBox<Speed> Speed = new ChoiceBox<>();
        Speed.getItems().addAll(MainGUI.Speed.FAST, MainGUI.Speed.MODERATE, MainGUI.Speed.SLOW);
```

```
        Speed.setValue(MainGUI.Speed.MODERATE);
        Speed.setLayoutX(140);
        Speed.setLayoutY(160);

        Label movingSpeed = new Label( s: "Moving Speed:");
        movingSpeed.setLayoutX(55);
        movingSpeed.setLayoutY(162);
        Pane pane = new Pane();
        pane.getChildren().addAll(jerry_view,tom_view,start,mazeLength,title,sizeOfMaze,Speed,movingSpeed,info, backTestingMenuButton);

        info.setOnAction(actionEvent -> {
            InfoGUI infoGUI = new InfoGUI();
            try {
                infoGUI.start(stage);} catch (Exception e) {throw new RuntimeException(e);}
        });

        start.setOnAction(actionEvent -> {
            mazeSize = mazeLength.getValue();
            maze = newMaze();
            Tom.speed = (Speed.getValue().value-20);
            Jerry.speed = (Speed.getValue().value);
            shortestPath = new AStarAlgorithm(Tom.location, Jerry.location,  maze: "maze_map.csv");
            GameMazeGUI gameMazeGUI = new GameMazeGUI();
            gameMazeGUI.start(stage);
        });

        backTestingMenuButton.setOnAction(actionEvent -> {
            BigMainGUI bigMainGUI = new BigMainGUI();
            try {
                bigMainGUI.start(stage);} catch (Exception e) {throw new RuntimeException(e);}
        });

        Scene startPage = new Scene(pane, v: 300, v1: 300);
        stage.setScene(startPage);
        stage.setTitle("Game Main Menu");
```

```
            Scene startPage = new Scene(pane, v: 300, v1: 300);
            stage.setScene(startPage);
            stage.setTitle("Game Main Menu");
            stage.show();
        }
    }

```

17. MazeGenerator (without EntryPointGenerator & getValidNeighbor)



```java
public class MazeGenerator {
    private final int rows;               // number of rows in the maze
    private final int cols;               // number of columns in the maze
    private final Cell[][] maze;          // maze array
    public Cell EntryPoint;               // maze EntryPoint
    public Cell ExitPoint;                // maze ExitPoint
    Random random = new Random();
    private boolean foundEnd = false;

    /**
     * Constructs a MazeGenerator object with the specified number of rows and columns.
     *
     * @param rows The number of rows in the maze.
     * @param cols The number of columns in the maze.
     */
    public MazeGenerator(int rows, int cols) {
        this.rows = rows;
        this.cols = cols;
        this.maze = new Cell[rows][cols];
    }

    /**
     * Generates the entry point of the maze.
     */
```

```java
    public void initializeMaze() {
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                maze[i][j] = new Cell(i, j, CellState.BLOCK);
            }
        }
        // Get the Random EntryPoint
        EntryPointGenerator( testRandomNumber: null);
    }

    /**
     * Generates the maze by DFS algorithm.
     */
    public void generateMaze() {
        initializeMaze();
        EntryPoint.visited = true;

        Stack<Cell> stack = new Stack<>();
        stack.push(EntryPoint);

        while (!stack.isEmpty()) {
            Cell currentCell = stack.peek();
            List<Cell> neighbors = getValidNeighbors(currentCell);

            if (!neighbors.isEmpty()) {
                Cell randomNeighbor = neighbors.get(random.nextInt(neighbors.size()));
                randomNeighbor.visited = true;
                if(!checkIfExitPoint(randomNeighbor))
                    randomNeighbor.value = CellState.PATH;
                stack.push(randomNeighbor);
            } else {
```

```java
                    randomNeighbor.value = CellState.PATH;
                stack.push(randomNeighbor);
            } else {
                stack.pop();
            }
        }

    }
```

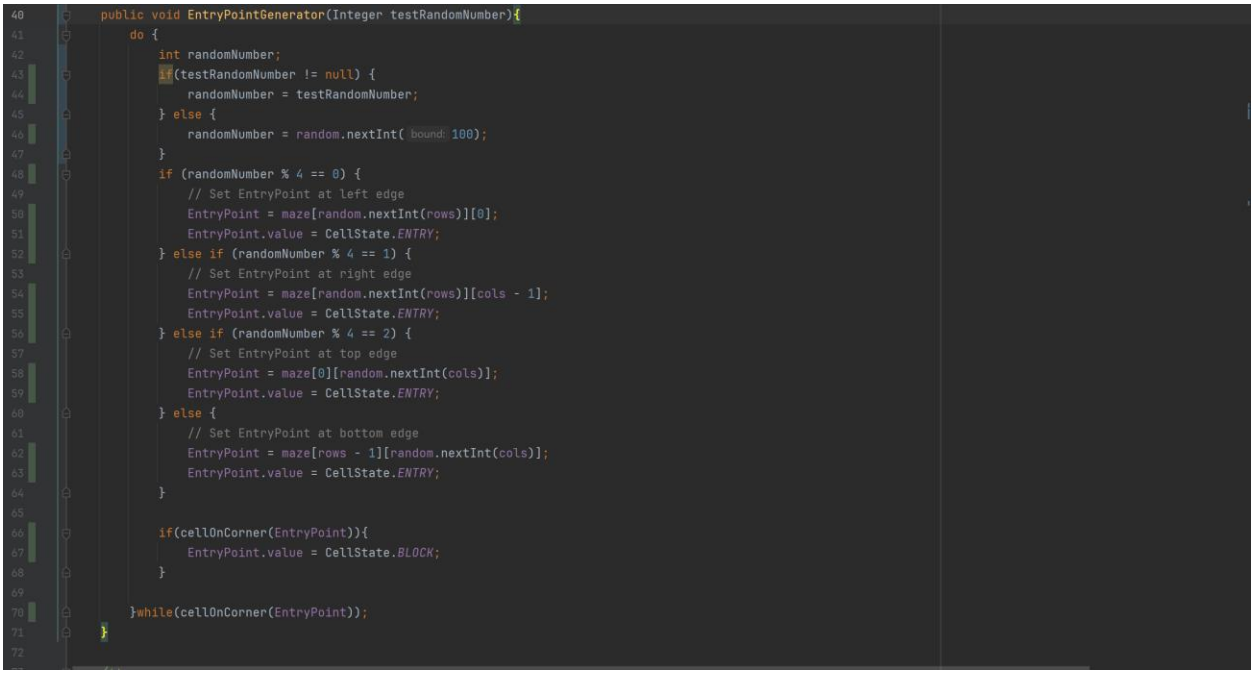
```java
public boolean checkValidNeighbors(Cell cell){
    int numNeighboringZeros = 0;
    for (int col = cell.col-1; col < cell.col+2; col++) {
        for (int row = cell.row-1; row < cell.row+2; row++) {
            if (cellOnGrid(row, col) && !cell.equals(maze[row][col]) && maze[row][col].value==CellState.PATH) {
                numNeighboringZeros++;
            }
        }
    }

    return (numNeighboringZeros < 4) && !maze[cell.row][cell.col].visited && !cellOnEdge(cell) && !checkIfEntryPoint(cell);
}
```

```java
public Boolean cellOnGrid(int row, int col) { return row >= 0 && col >= 0 && row < rows && col < cols; }

/**
 * Checks if a cell is on the edge of the maze.
 *
 * @param cell The cell to check.
 * @return True if the cell is on the edge, false otherwise.
 */
5 usages  LOWingYan
public Boolean cellOnEdge(Cell cell){
    return (cell.row == 0 || cell.col == 0 || cell.row == rows-1 || cell.col == cols-1);
}

/**
 * Checks if a cell is on the corner of the maze.
 *
 * @param cell The cell to check.
 * @return True if the cell is on the corner, false otherwise.
 */
5 usages  LOWingYan
public Boolean cellOnCorner(Cell cell){
    return ((cell.row==0 && cell.col==0) || (cell.row==0 && cell.col==cols-1) ||
            (cell.row==rows-1 && cell.col==0) || (cell.row==rows-1 && cell.col==cols-1));
}

/**
 * Checks if a cell is the entry point of the maze.
 *
 * @param cell The cell to check.
 * @return True if the cell is the entry point, false otherwise.
 */
```

```java
public Boolean checkIfEntryPoint(Cell cell) { return (EntryPoint.equals(cell)); }

/**
 * Checks if a cell is the exit point of the maze.
 *
 * @param cell The cell to check.
 * @return True if the cell is the exit point, false otherwise.
 */
27 usages  LOWingYan
public Boolean checkIfExitPoint(Cell cell){
    if(foundEnd)
        return (ExitPoint.equals(cell));
    else
        return (cell.row==rows-1 || cell.row==0 || cell.col==cols-1 || cell.col==0) && CellOnOppositeEdge(EntryPoint,cell);

}

/**
 * Checks if two cells are on the opposite edges of the maze.
 *
 * @param cell_1 The first cell.
 * @param cell_2 The second cell.
 * @return True if the cells are on opposite edges, false otherwise.
 */
10 usages  LOWingYan
public Boolean CellOnOppositeEdge(Cell cell_1, Cell cell_2){
    return ((cell_1.row==0 && cell_2.row==rows-1) || (cell_1.col==0 && cell_2.col==cols-1) ||
            (cell_1.row==rows-1 && cell_2.row==0) || (cell_1.col==cols-1 && cell_2.col==0));

}

/**
 * Retrieves the maze array.
 *
```

```java
     * @return The maze array.
     */
    LOWingYan
    public Cell[][] getMaze() { return maze; }

}
```

MazeGeneratoe.EntryPointGenerator()



```
40    public void EntryPointGenerator(Integer testRandomNumber){
41        do {
42            int randomNumber;
43            if(testRandomNumber != null) {
44                randomNumber = testRandomNumber;
45            } else {
46                randomNumber = random.nextInt( bound: 100);
47            }
48            if (randomNumber % 4 == 0) {
49                // Set EntryPoint at left edge
50                EntryPoint = maze[random.nextInt(rows)][0];
51                EntryPoint.value = CellState.ENTRY;
52            } else if (randomNumber % 4 == 1) {
53                // Set EntryPoint at right edge
54                EntryPoint = maze[random.nextInt(rows)][cols - 1];
55                EntryPoint.value = CellState.ENTRY;
56            } else if (randomNumber % 4 == 2) {
57                // Set EntryPoint at top edge
58                EntryPoint = maze[0][random.nextInt(cols)];
59                EntryPoint.value = CellState.ENTRY;
60            } else {
61                // Set EntryPoint at bottom edge
62                EntryPoint = maze[rows - 1][random.nextInt(cols)];
63                EntryPoint.value = CellState.ENTRY;
64            }
65
66            if(cellOnCorner(EntryPoint)){
67                EntryPoint.value = CellState.BLOCK;
68            }
69
70        }while(cellOnCorner(EntryPoint));
71    }
72
```

Sometimes Line 67 will not be covered since it is totally dependent on random.

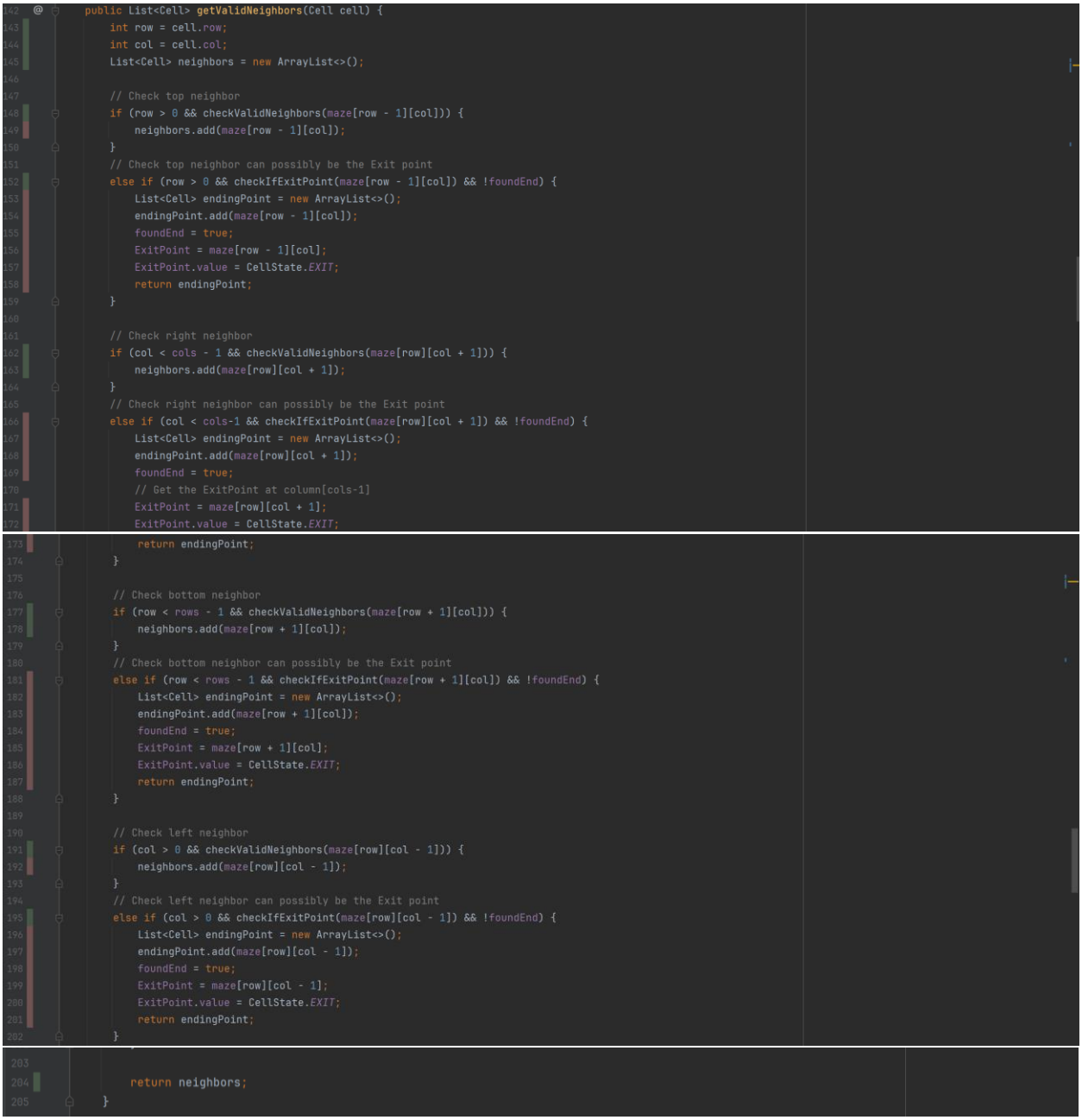
GameMazeGUI.start()

```
    @Override
    public void start(Stage primaryStage) {
        String musicFilePath = "TomAndJerryBackgroundMusic.mp3";
        Media media = new Media(new File(musicFilePath).toURI().toString());

        // Create a MediaPlayer with the Media object
        MediaPlayer mediaPlayer = new MediaPlayer(media);
        mediaPlayer.play();

        SetGridPane();
        updatedGridPane(Jerry, JerryJerry);
        updatedGridPane(Tom, TomTom);
        // Create the scene and set it on the stage
        StackPane stackPane = new StackPane();
        Label TomWin = new Label( s: "You Are Caught by Tom!");
        Label JerryWin = new Label( s: "Successfully Escape!");
        TomWin.setStyle("-fx-font-size: 24; -fx-font-weight: bold;");
        JerryWin.setStyle("-fx-font-size: 24; -fx-font-weight: bold;");


        Button reT = new Button( s: "Restart");
        Button reJ = new Button( s: "Restart");
        VBox T_Win = new VBox( v: 20,TomWin,reT);
        VBox J_Win = new VBox( v: 20,JerryWin,reJ);
        T_Win.setAlignment(Pos.CENTER);
        J_Win.setAlignment(Pos.CENTER);

        T_Win.setVisible(false);
        J_Win.setVisible(false);
        stackPane.getChildren().addAll(gridPane,T_Win,J_Win);

        Pane pane = new Pane();
        Button home = new Button( s: "Home");
        home.setLayoutX(CELL_SIZE*(mazeSize-3));
        home.setLayoutY(0);
```

```
        stackPane.setLayoutY(30);
        pane.getChildren().addAll(stackPane,home);

        Scene scene = new Scene(pane);
        KeyBoardListener JerryMove = new KeyBoardListener(Jerry);
        CheckEndGame endGame = new CheckEndGame();
        // Create a thread for Jerry and Tom respectively to make them run simultaneously
        Runnable startGame = () -> {
            Thread Player = new Thread(() -> {
                while (!endGame.isEndGame()){
                    Jerry.move();
                    updatedGridPane(Jerry, JerryJerry);
                    try {
                        sleep(Jerry.speed);} catch (InterruptedException e) {e.printStackTrace();}
                }
                if (Jerry.Game_state)  {
                    J_Win.setVisible(true);
                }
            });

            Thread Computer = new Thread(() -> {
                try {
                    sleep( millis: 1500);} catch (InterruptedException e) {throw new RuntimeException(e);}
                while (!endGame.isEndGame()){
                    Tom.MoveWithShortestPath();
                    updatedGridPane(Tom, TomTom);
                    try {
                        sleep(Tom.speed);} catch (InterruptedException e) {e.printStackTrace();}
                }
                if (Tom.Game_state) {
                    T_Win.setVisible(true);
                }
            });
            Player.start();
//          Computer starts moving when player starts to play
```

```
            scene.setOnKeyPressed(keyEvent -> {
                JerryMove.keyPressed(keyEvent);
                if (Computer.getState() == Thread.State.NEW) {
                    Computer.start();
                }
            });
        };
        startGame.run();
//      Function of restart button
        EventHandler handler = event -> {
            T_Win.setVisible(false);
            J_Win.setVisible(false);

            Jerry.reset( row: entryIndex/maze.length, col: entryIndex%maze.length);
            Tom.reset( row: exitIndex/maze.length, col: exitIndex%maze.length);
            updatedGridPane(Jerry, JerryJerry);
            updatedGridPane(Tom, TomTom);
            startGame.run();
        };
        reT.setOnAction(handler);
        reJ.setOnAction(handler);
//      Function of Home button
        LOWingYan +2
        home.setOnAction(new EventHandler<ActionEvent>() {
            LOWingYan +2
            @Override
            public void handle(ActionEvent event) {
                mediaPlayer.stop();
                Tom.Game_state = true;
                try {
                    sleep( millis: 500);} catch (InterruptedException e) {throw new RuntimeException(e);}

                Tom = new Character();
                Jerry = new Character();

                primaryStage.close();
                MainGUI mainGUI = new MainGUI();
                Stage mainStage = new Stage();
                try {
                    mainGUI.start(mainStage);} catch (IOException e) {throw new RuntimeException(e);}
            }
        });


        primaryStage.setScene(scene);
        primaryStage.setTitle("Tom n Jerry In Maze");
        primaryStage.show();


    }

}
```

MazeGenerator.getValidNeighbor()



```
public List<Cell> getValidNeighbors(Cell cell) {
    int row = cell.row;
    int col = cell.col;
    List<Cell> neighbors = new ArrayList<>();

    // Check top neighbor
    if (row > 0 && checkValidNeighbors(maze[row - 1][col])) {
        neighbors.add(maze[row - 1][col]);
    }
    // Check top neighbor can possibly be the Exit point
    else if (row > 0 && checkIfExitPoint(maze[row - 1][col]) && !foundEnd) {
        List<Cell> endingPoint = new ArrayList<>();
        endingPoint.add(maze[row - 1][col]);
        foundEnd = true;
        ExitPoint = maze[row - 1][col];
        ExitPoint.value = CellState.EXIT;
        return endingPoint;
    }

    // Check right neighbor
    if (col < cols - 1 && checkValidNeighbors(maze[row][col + 1])) {
        neighbors.add(maze[row][col + 1]);
    }
    // Check right neighbor can possibly be the Exit point
    else if (col < cols-1 && checkIfExitPoint(maze[row][col + 1]) && !foundEnd) {
        List<Cell> endingPoint = new ArrayList<>();
        endingPoint.add(maze[row][col + 1]);
        foundEnd = true;
        // Get the ExitPoint at column[cols-1]
        ExitPoint = maze[row][col + 1];
        ExitPoint.value = CellState.EXIT;
        return endingPoint;
    }

    // Check bottom neighbor
    if (row < rows - 1 && checkValidNeighbors(maze[row + 1][col])) {
        neighbors.add(maze[row + 1][col]);
    }
    // Check bottom neighbor can possibly be the Exit point
    else if (row < rows - 1 && checkIfExitPoint(maze[row + 1][col]) && !foundEnd) {
        List<Cell> endingPoint = new ArrayList<>();
        endingPoint.add(maze[row + 1][col]);
        foundEnd = true;
        ExitPoint = maze[row + 1][col];
        ExitPoint.value = CellState.EXIT;
        return endingPoint;
    }

    // Check left neighbor
    if (col > 0 && checkValidNeighbors(maze[row][col - 1])) {
        neighbors.add(maze[row][col - 1]);
    }
    // Check left neighbor can possibly be the Exit point
    else if (col > 0 && checkIfExitPoint(maze[row][col - 1]) && !foundEnd) {
        List<Cell> endingPoint = new ArrayList<>();
        endingPoint.add(maze[row][col - 1]);
        foundEnd = true;
        ExitPoint = maze[row][col - 1];
        ExitPoint.value = CellState.EXIT;
        return endingPoint;
    }


    return neighbors;
}
```

Since the maze is randomly generated, so for this function’s result is non-deterministic.