COMP3111 GRP34

Documentation

Link to demo video: https://www.youtube.com/watch?v=DN3FrAQz7y0

Repository: https://github.com/yxiaoaz/Comp3111F23G34

CHAN, Chun Hin	20853893	chchanec@connect.ust.hk
WONG Yin Lam Anthony	20855918	ylawongab@connect.ust.hk
XIAO Yicong	20632366	yxiaoaz@connect.ust.hk

Meeting Minutes

Minutes of the 1st Project Scrum Meeting
Tom and Jerry in Maze Game

Date	28 September, 2023
Time	15:00
Duration	40 minutes
Venue	Face-to-face @ HKUST Library (LG1 Study Room 353)
Attending	ALL (Anthony WONG Yin Lam, Edward XIAO Yicong, Vincent CHAN Chun Hin)
Absent	None
Recorder	Vincent CHAN Chun Hin
Agenda	1. Sync the understanding about project specification 2. Devise preliminary plan for completing project

I. Report on progress during the past week

Name	Tasks worked on in the past week	Total hours
Anthony WONG Yin Lam	N/A	N/A
Edward XIAO Yicong	N/A	N/A
Vincent CHAN Chun Hin	N/A	N/A

II. Discussion details

1. Project specification
    - All members went over the project manual and discussed teammates' confusions, and consensus reached
    - All members reached consensus on project requirements, including expected output of program, requirement of files to be submitted, and grading specifications

2. Division of function implementation task
    - All members discussed the detailed requirements of the 3 functions
    - All members discussed the overall implementation method, difficulty and

estimated implementation time of each function
- Edward XIAO divided the task according to the personal interest and technical knowledge of each teammate
- Vincent CHAN would do function A, Anthony WONG would do function B and Edward XIAO would do function C

3. Preliminary project timeline
    - All members discussed the implementation order and dependency between different functions
    - All members reached consensus on weekly/biweekly meeting for progress update until end of project

III. Goals for the coming week

Name	**Tasks that will be worked on in the coming week**
Anthony WONG Yin Lam	Implement the own version of class diagram
Edward XIAO Yicong	Implement the own version of class diagram
Vincent CHAN Chun Hin	Implement the own version of class diagram

IV. Meeting adjournment and next meeting

The meeting was adjourned at 15:40.

The next meeting will be held on 5 October, 2023.

Minutes of the 2nd Project Scrum Meeting
Tom and Jerry in Maze Game

Date	5 October, 2023
Time	15:15
Duration	90 minutes
Venue	Face-to-face @ HKUST CSD Teaching Lab 4 (Room 4210)
Attending	ALL (Anthony WONG Yin Lam, Edward XIAO Yicong, Vincent CHAN Chun Hin)
Absent	None
Recorder	Vincent CHAN Chun Hin
Agenda	1. Exchange each member opinions on the class diagram 2. Finalize the class diagram 3. Brief discussion on the use case specification

I. Report on progress during the past week

Name	Tasks worked on in the past week	Total hours
Anthony WONG Yin Lam	Implement the own version of class diagram	2
Edward XIAO Yicong	Implement the own version of class diagram	2
Vincent CHAN Chun Hin	Implement the own version of class diagram	2

II. Discussion details

1. Class diagram
    - All members showed their class diagram to other members
    - All members discussed which part of their class diagram was correct and which parts were wrong, and could be rectified by which member's diagram's part
    - All members reached consensus on the finalized version of the class diagram, where it adopted majority part of the design from Edward XIAO and some

minor part from Vincent CHAN

2. Use case specification
    - All members started to discuss how to write the use case specification
    - Vincent CHAN showed his basic draft of the use case specification
    - All members reached consensus that all members would start working on the use case specification of the function they were responsible for and allowed further discussion on WhatsApp

III. Goals for the coming week

Name	**Tasks that will be worked on in the coming week**
Anthony WONG Yin Lam	Write the use case specification for function B
Edward XIAO Yicong	Write the use case specification for function C
Vincent CHAN Chun Hin	Write the use case specification for function A

IV. Meeting adjournment and next meeting

The meeting was adjourned at 16:45.

The next meeting will be held on 19 October, 2023.

Minutes of the 3rd Project Scrum Meeting
Tom and Jerry in Maze Game

Date	19 October, 2023
Time	15:00
Duration	100 minutes
Venue	Face-to-face @ HKUST LG1 AV Editing Suite A (room LC-102)
Attending	ALL (Anthony WONG Yin Lam, Edward XIAO Yicong, Vincent CHAN Chun Hin)
Absent	None
Recorder	Vincent CHAN Chun Hin
Agenda	1. Exchange every member's progress on the implementation of the code 2. Do Lab 5 together

I. Report on progress during the past week

Name	Tasks worked on in the past week	Total hours
Anthony WONG Yin Lam	Write the use case specification for function B Work on the implementation of function B	11
Edward XIAO Yicong	Write the use case specification for function C Work on the implementation of function C	13
Vincent CHAN Chun Hin	Write the use case specification for function A Work on the implementation of function A	10

II. Discussion details

1. Progress on the implementation of the code
    - All members showed their progress on the implementation of the code
    - Vincent CHAN has done the algorithm part on how to generate the maze in

character format
- Edward XIAO has done how to show the GUI of the maze
- Anthony WONG has partly done the shortest path
- All members agreed to explore on how to export Java list array to a csv file

2. COMP3111/COMP3111H Lab 5
   - All members do Lab 5 together

III. Goals for the coming week

Name	**Tasks that will be worked on in the coming week**
Anthony WONG Yin Lam	Continue working on the implementation of function B Explore how to export Java list array to a csv file
Edward XIAO Yicong	Continue working on the implementation of function C Explore how to export Java list array to a csv file
Vincent CHAN Chun Hin	Continue working on the implementation of function A Explore how to export Java list array to a csv file

IV. Meeting adjournment and next meeting

The meeting was adjourned at 16:40.

The next meeting will be held on 2 November, 2023.

Minutes of the 4th Project Scrum Meeting
Tom and Jerry in Maze Game

Date	2 November, 2023
Time	15:00
Duration	17 minutes
Venue	Face-to-face @ HKUST LG1 AV Editing Suite C (room LC-104)
Attending	ALL (Anthony WONG Yin Lam, Edward XIAO Yicong, Vincent CHAN Chun Hin)
Absent	None
Recorder	Vincent CHAN Chun Hin
Agenda	1. Exchange every member's progress on the implementation of the code

I. Report on progress during the past week

Name	Tasks worked on in the past week	Total hours
Anthony WONG Yin Lam	Working on the implementation of function B Explore how to export Java list array to a csv file	14
Edward XIAO Yicong	Working on the implementation of function C Explore how to export Java list array to a csv file	15
Vincent CHAN Chun Hin	Working on the implementation of function A Explore how to export Java list array to a csv file	11

II. Discussion details

1. Progress on the implementation of the code

- All members showed and explained their progress on the implementation of the code
- Vincent CHAN has already finished all the coding work for function A
- Anthony WONG has already finished all the coding work for function B
- Edward XIAO has finished most of the coding work for function C, but his code would have a bug since when Tom caught Jerry, the game could not be terminated
- Edward XIAO promised that he would try to debug function C and fixed his function C by the next meeting
- Edward XIAO suggested all members should start working on the unit testing of their responsible function and finished them before the next meeting, all members agreed with that
- Edward XIAO planned that the next meeting would be related to the video shooting of the game

III. Goals for the coming week

Name	**Tasks that will be worked on in the coming week**
Anthony WONG Yin Lam	Work on the unit testing for function B
Edward XIAO Yicong	Fix the bug for function C Work on the unit testing for function C
Vincent CHAN Chun Hin	Work on the unit testing for function A

IV. Meeting adjournment and next meeting

The meeting was adjourned at 15:17.

The next meeting will be held on 9 November, 2023

Minutes of the 5th Project Scrum Meeting
Tom and Jerry in Maze Game

Date	9 November, 2023
Time	15:00
Duration	50 minutes
Venue	Face-to-face @ HKUST CSD Teaching Lab 4 (Room 4210)
Attending	ALL (Anthony WONG Yin Lam, Edward XIAO Yicong, Vincent CHAN Chun Hin)
Absent	None
Recorder	Vincent CHAN Chun Hin
Agenda	1. Exchange every member's progress on the unit testing

I. Report on progress during the past week

Name	Tasks worked on in the past week	Total hours
Anthony WONG Yin Lam	Work on the unit testing for function B	3
Edward XIAO Yicong	Fix the bug for function C Work on the unit testing for function C	9
Vincent CHAN Chun Hin	Work on the unit testing for function A	3

II. Discussion details

1. Progress on the unit testing
    - Edward XIAO has done part of the unit testing, and merge his function C with function A and function B into a game that can run successfully
    - Edward XIAO suggested all members should do unit testing on the merged code, but not on the individual function they worked on previously as there are some big changes on the class names and the class structures
    - Vincent CHAN and Anthony WONG thus needed to redo the unit testing, and all members agreed with that
    - Edward XIAO hoped that all members should complete the unit testing, Javadoc documentation, burndown chart, and Gantt chart by next week, and all

members agreed with that

- Edward XIAO considered next week's focus would be on the discussion of the video shooting of the game

III. Goals for the coming week

Name	**Tasks that will be worked on in the coming week**
Anthony WONG Yin Lam	Work on the unit testing of function B Work on the Javadoc documentation of function B Work on the burndown chart Work on the Gantt chart
Edward XIAO Yicong	Work on the unit testing of function C Work on the Javadoc documentation of function C Work on the burndown chart Work on the Gantt chart
Vincent CHAN Chun Hin	Work on the unit testing of function A Work on the Javadoc documentation of function A Work on the burndown chart Work on the Gantt chart

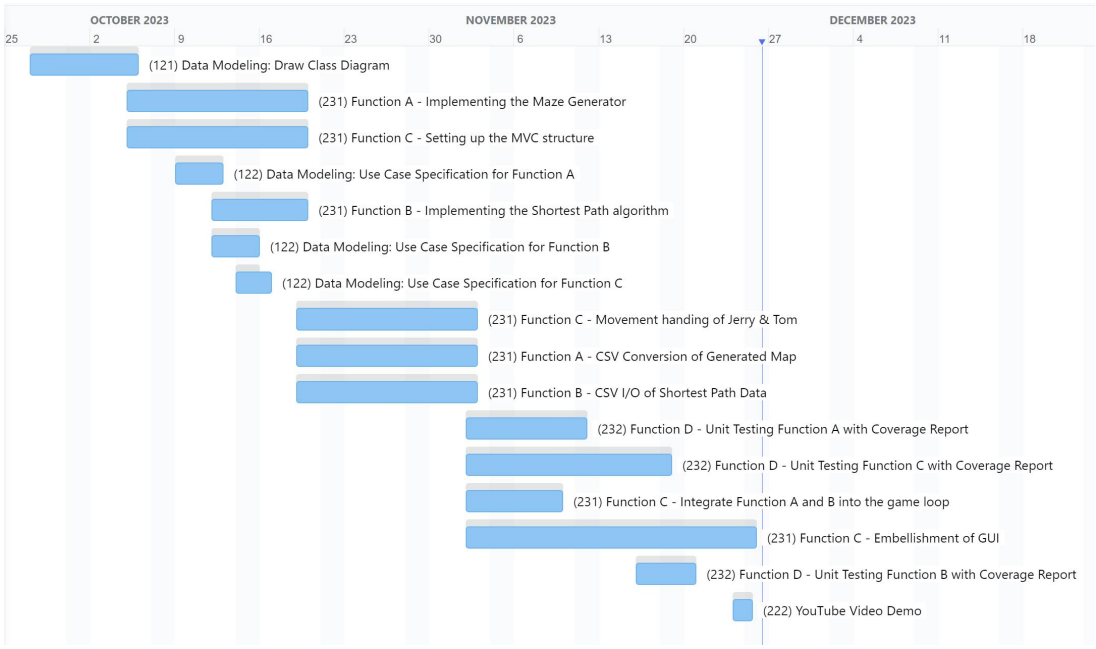
IV. Meeting adjournment and next meeting

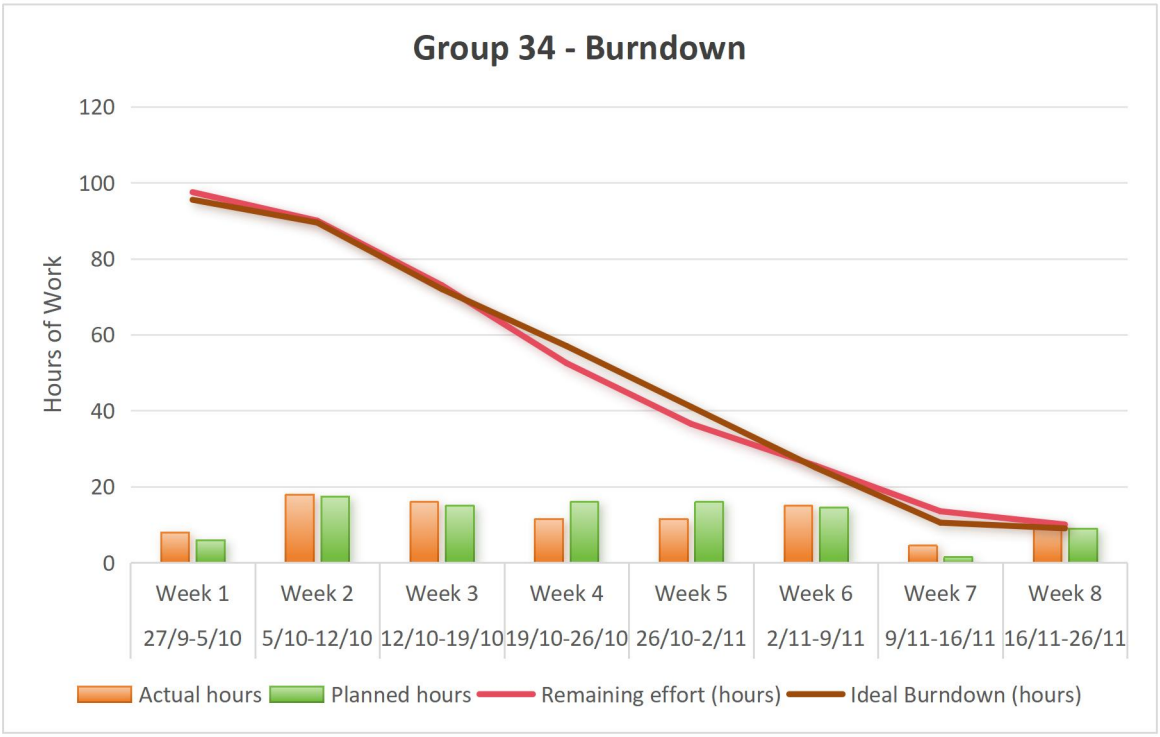
The meeting was adjourned at 15:50.

Edward XIAO stated that this would be the last official meeting as the remaining tasks are less difficult to finish. The remaining things related to the documentation and video shooting would be addressed through contact in WhatsApp. No official meeting would be held unless any emergency situations occured.

Gantt Chart & Burndown Chart

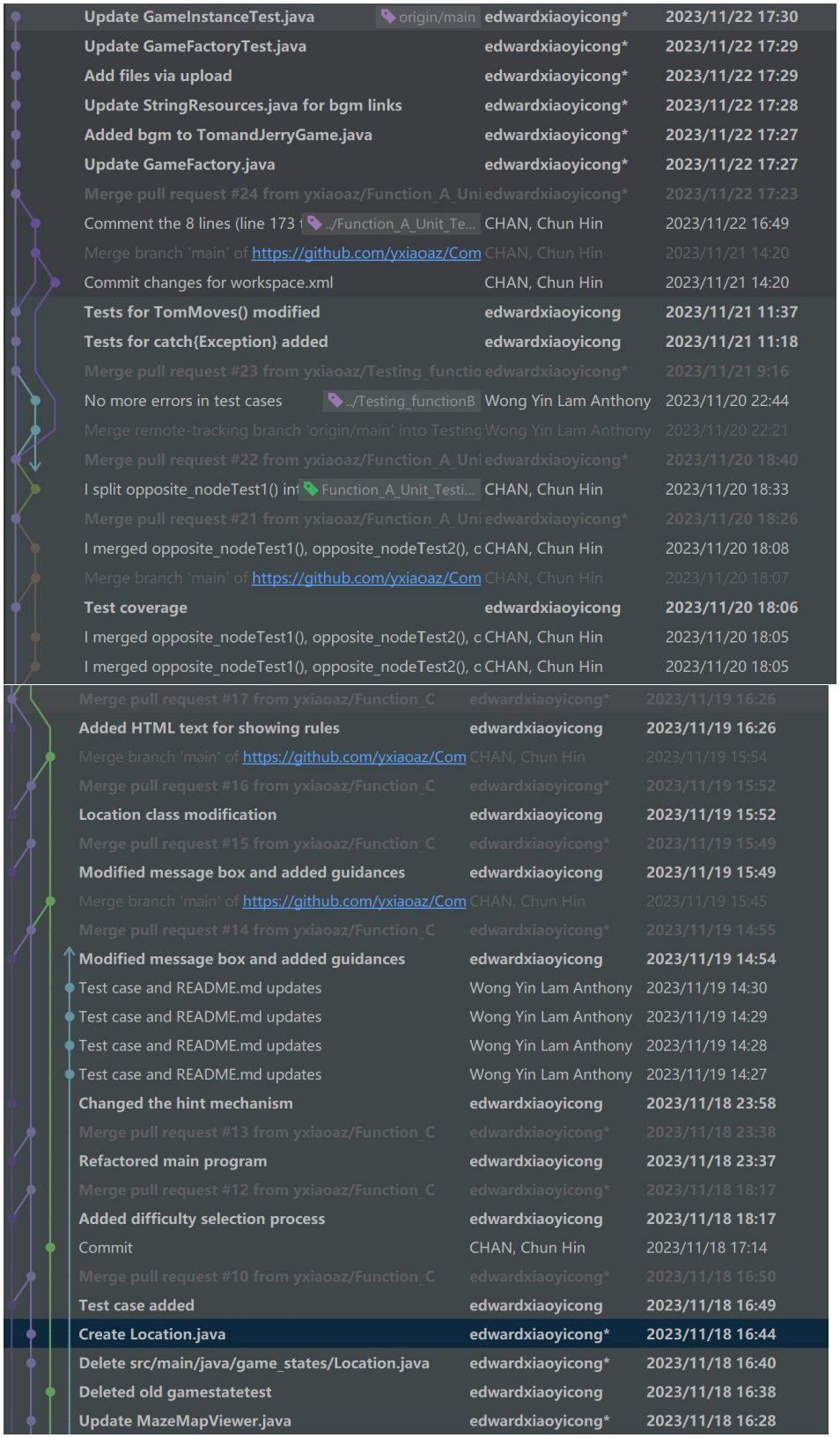
Week	Start Date	End Date	Task	Actual Days	Actual hours	Planned hours
Week 1	2023/9/27	2023/10/5	(121) Data Modeling: Draw Class Diagram	8	8.00	6.00
Week 2	2023/10/5	2023/10/19	(231) Function A - Implementing the Maze Generator	14	10.00	10.00
	2023/10/5	2023/10/19	(231) Function C - Setting up the MVC structure	14	20.00	20.00
	2023/10/9	2023/10/12	(122) Data Modeling: Use Case Specification for Function A	3	3.00	2.50
Week 3	2023/10/12	2023/10/15	(122) Data Modeling: Use Case Specification for Function B	3	3.00	2.50
	2023/10/12	2023/10/19	(231) Function B - Implementing the Shortest Path algorithm	7	10.00	10.00
	2023/10/14	2023/10/15	(122) Data Modeling: Use Case Specification for Function C	1	3.00	2.50
Week 4	2023/10/19	2023/11/2	(231) Function B - CSV I/O of Shortest Path Data	14	10.00	15.00
	2023/10/19	2023/11/2	(231) Function A - CSV Conversion of Generated Map	14	10.00	15.00
	2023/10/19	2023/11/2	(231) Function C - Movement handing of Jerry & Tom (control panel setup)	14	3.00	2.00
Week 6	2023/11/2	2023/11/9	(231) Function C - Integrate Function A and B into the game loop	7	1.00	2.00
	2023/11/2	2023/11/25	(231) Function C - Embellishment of GUI (BGM, animation, landing page)	23	15.00	15.00
	2023/11/2	2023/11/18	(232) Function D - Unit Testing Function C with Coverage Report	16	15.00	10.00
	2023/11/2	2023/11/11	(232) Function D - Unit Testing Function A with Coverage Report	9	3.00	5.00
Week 8	2023/11/16	2023/11/20	(232) Function D - Unit Testing Function B with Coverage Report	4	3.00	5.00
	2023/11/24	2023/11/24	(222) YouTube Video Demo	1	6.00	5.00



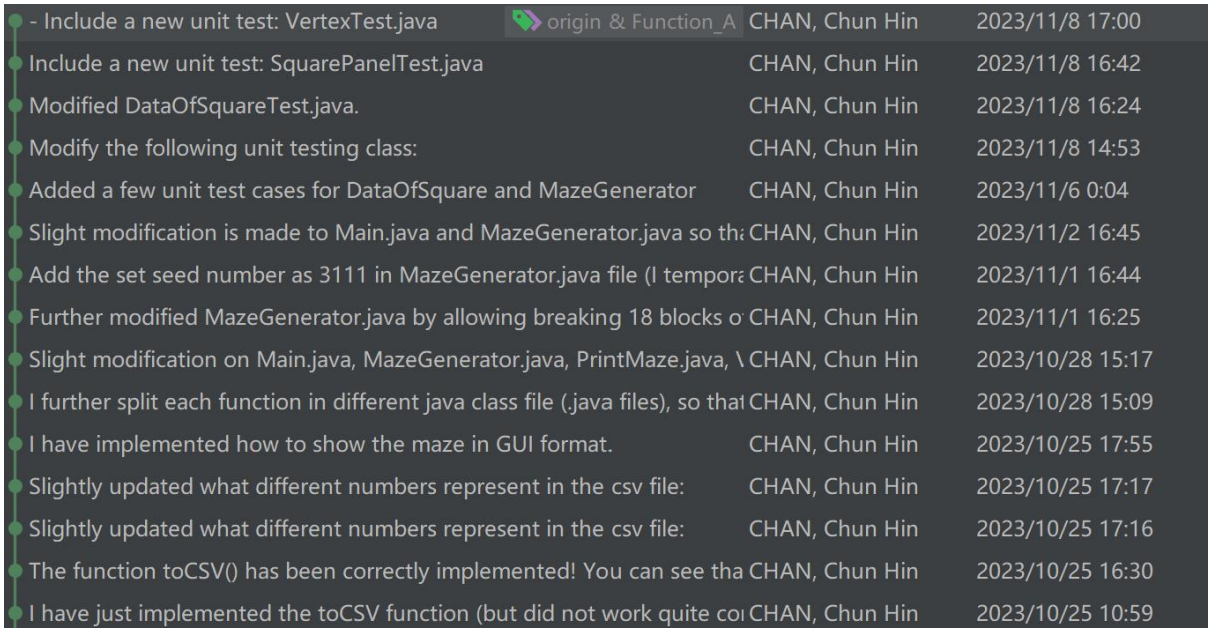

Group 34 - Burndown
Hours of Work
120
100
80
60
40
20
0
Week 1
27/9-5/10
Week 2
5/10-12/10
Week 3
12/10-19/10
Week 4
19/10-26/10
Week 5
26/10-2/11
Week 6
2/11-9/11
Week 7
9/11-16/11
Week 8
16/11-26/11
Actual hours
Planned hours
Remaining effort (hours)
Ideal Burndown (hours)

Git Commit Log

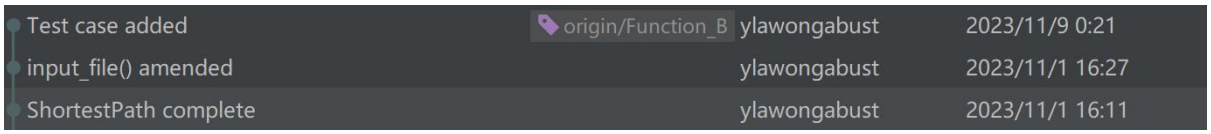
Main branch


Update GameInstanceTest.java origin/main edwardxiaoyicong* 2023/11/22 17:30
Update GameFactoryTest.java edwardxiaoyicong* 2023/11/22 17:29
Add files via upload edwardxiaoyicong* 2023/11/22 17:29
Update StringResources.java for bgm links edwardxiaoyicong* 2023/11/22 17:28
Added bgm to TomandJerryGame.java edwardxiaoyicong* 2023/11/22 17:27
Update GameFactory.java edwardxiaoyicong* 2023/11/22 17:27
Merge pull request #24 from yxiaoaz/Function_A_Uni edwardxiaoyicong* 2023/11/22 17:23
Comment the 8 lines (line 173 t ../Function_A_Unit_Te... CHAN, Chun Hin 2023/11/22 16:49
Merge branch 'main' of https://github.com/yxiaoaz/Com CHAN, Chun Hin 2023/11/21 14:20
Commit changes for workspace.xml CHAN, Chun Hin 2023/11/21 14:20
Tests for TomMoves() modified edwardxiaoyicong 2023/11/21 11:37
Tests for catch{Exception} added edwardxiaoyicong 2023/11/21 11:18
Merge pull request #23 from yxiaoaz/Testing_functio edwardxiaoyicong* 2023/11/21 9:16
No more errors in test cases ../Testing_functionB Wong Yin Lam Anthony 2023/11/20 22:44
Merge remote-tracking branch 'origin/main' into Testing Wong Yin Lam Anthony 2023/11/20 22:21
Merge pull request #22 from yxiaoaz/Function_A_Uni edwardxiaoyicong* 2023/11/20 18:40
I split opposite_nodeTest1() int Function_A_Unit_Testi... CHAN, Chun Hin 2023/11/20 18:33
Merge pull request #21 from yxiaoaz/Function_A_Uni edwardxiaoyicong* 2023/11/20 18:26
I merged opposite_nodeTest1(), opposite_nodeTest2(), c CHAN, Chun Hin 2023/11/20 18:08
Merge branch 'main' of https://github.com/yxiaoaz/Com CHAN, Chun Hin 2023/11/20 18:07
Test coverage edwardxiaoyicong 2023/11/20 18:06
I merged opposite_nodeTest1(), opposite_nodeTest2(), c CHAN, Chun Hin 2023/11/20 18:05
I merged opposite_nodeTest1(), opposite_nodeTest2(), c CHAN, Chun Hin 2023/11/20 18:05
Merge pull request #17 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/19 16:26
Added HTML text for showing rules edwardxiaoyicong 2023/11/19 16:26
Merge branch 'main' of https://github.com/yxiaoaz/Com CHAN, Chun Hin 2023/11/19 15:54
Merge pull request #16 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/19 15:52
Location class modification edwardxiaoyicong 2023/11/19 15:52
Merge pull request #15 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/19 15:49
Modified message box and added guidances edwardxiaoyicong 2023/11/19 15:49
Merge branch 'main' of https://github.com/yxiaoaz/Com CHAN, Chun Hin 2023/11/19 15:45
Merge pull request #14 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/19 14:55
Modified message box and added guidances edwardxiaoyicong 2023/11/19 14:54
Test case and README.md updates Wong Yin Lam Anthony 2023/11/19 14:30
Test case and README.md updates Wong Yin Lam Anthony 2023/11/19 14:29
Test case and README.md updates Wong Yin Lam Anthony 2023/11/19 14:28
Test case and README.md updates Wong Yin Lam Anthony 2023/11/19 14:27
Changed the hint mechanism edwardxiaoyicong 2023/11/18 23:58
Merge pull request #13 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/18 23:38
Refactored main program edwardxiaoyicong 2023/11/18 23:37
Merge pull request #12 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/18 18:17
Added difficulty selection process edwardxiaoyicong 2023/11/18 18:17
Commit CHAN, Chun Hin 2023/11/18 17:14
Merge pull request #10 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/18 16:50
Test case added edwardxiaoyicong 2023/11/18 16:49
Create Location.java edwardxiaoyicong* 2023/11/18 16:44
Delete src/main/java/game_states/Location.java edwardxiaoyicong* 2023/11/18 16:40
Deleted old gamestatetest edwardxiaoyicong 2023/11/18 16:38
Update MazeMapViewer.java edwardxiaoyicong* 2023/11/18 16:28

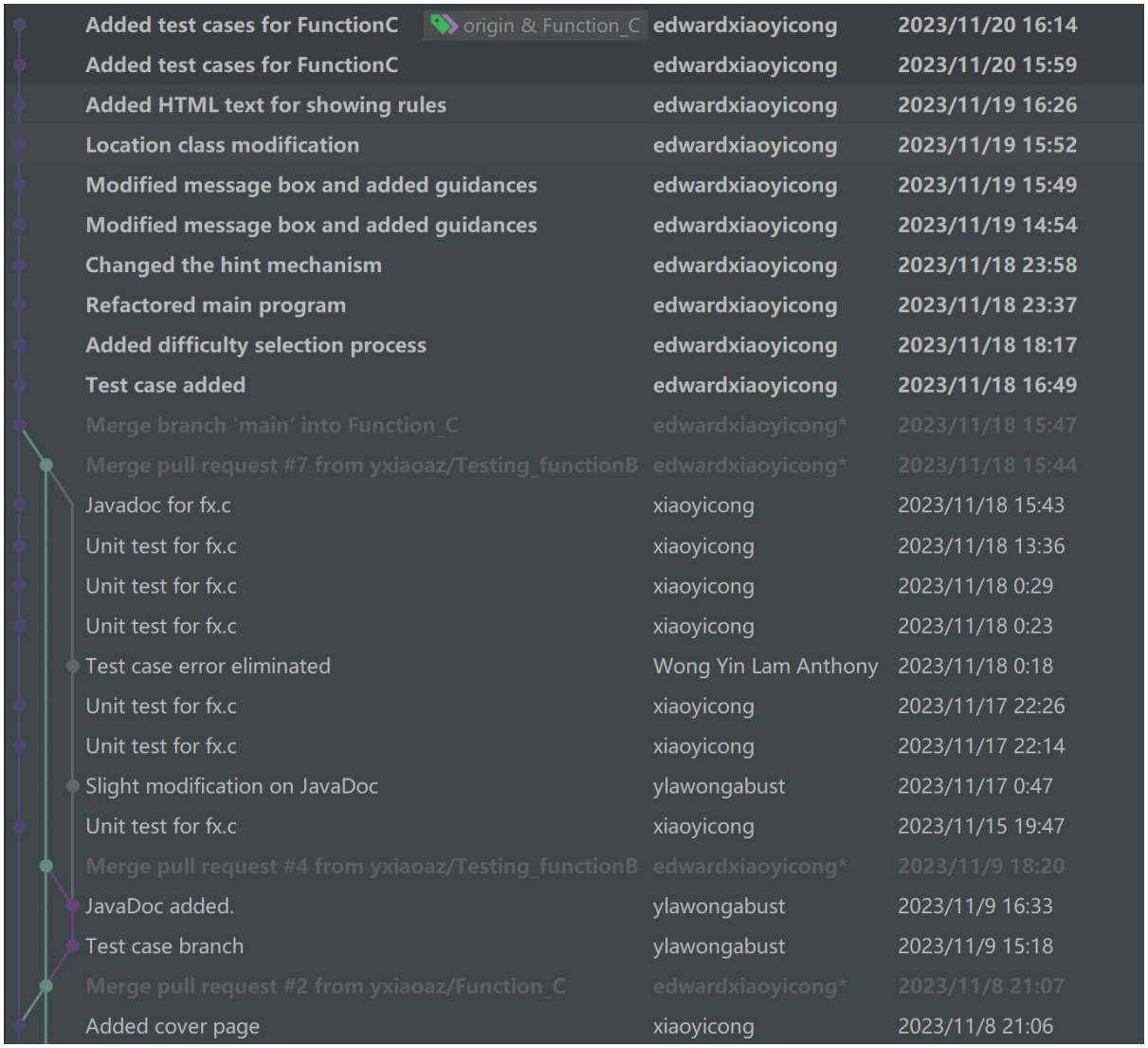
Function A



Function B



Function C


Added test cases for FunctionC origin & Function_C edwardxiaoyicong 2023/11/20 16:14
Added test cases for FunctionC edwardxiaoyicong 2023/11/20 15:59
Added HTML text for showing rules edwardxiaoyicong 2023/11/19 16:26
Location class modification edwardxiaoyicong 2023/11/19 15:52
Modified message box and added guidances edwardxiaoyicong 2023/11/19 15:49
Modified message box and added guidances edwardxiaoyicong 2023/11/19 14:54
Changed the hint mechanism edwardxiaoyicong 2023/11/18 23:58
Refactored main program edwardxiaoyicong 2023/11/18 23:37
Added difficulty selection process edwardxiaoyicong 2023/11/18 18:17
Test case added edwardxiaoyicong 2023/11/18 16:49
Merge branch 'main' into Function_C edwardxiaoyicong* 2023/11/18 15:47
Merge pull request #7 from yxiaoaz/Testing_functionB edwardxiaoyicong* 2023/11/18 15:44
Javadoc for fx.c xiaoyicong 2023/11/18 15:43
Unit test for fx.c xiaoyicong 2023/11/18 13:36
Unit test for fx.c xiaoyicong 2023/11/18 0:29
Unit test for fx.c xiaoyicong 2023/11/18 0:23
Test case error eliminated Wong Yin Lam Anthony 2023/11/18 0:18
Unit test for fx.c xiaoyicong 2023/11/17 22:26
Unit test for fx.c xiaoyicong 2023/11/17 22:14
Slight modification on JavaDoc ylawongabust 2023/11/17 0:47
Unit test for fx.c xiaoyicong 2023/11/15 19:47
Merge pull request #4 from yxiaoaz/Testing_functionB edwardxiaoyicong* 2023/11/9 18:20
JavaDoc added. ylawongabust 2023/11/9 16:33
Test case branch ylawongabust 2023/11/9 15:18
Merge pull request #2 from yxiaoaz/Function_C edwardxiaoyicong* 2023/11/8 21:07
Added cover page xiaoyicong 2023/11/8 21:06

Java Doc

game_algorithm (Function A & B)

程序包 game_algorithm

package game_algorithm

类

类	说明
GameMapGenerator	The class GameMapGenerator involves 4 data members, in which 1 of them is public and the remaining 3 of them are private.
ShortestPathGenerator	This class is used to calculate the shortest path in a maze.

程序包 game_algorithm

类 GameMapGenerator

java.lang.Object
    game_algorithm.GameMapGenerator

public class GameMapGenerator
extends Object

The class GameMapGenerator involves 4 data members, in which 1 of them is public and the remaining 3 of them are private.

- rand: store a java.util.Random class instance
- ROW: store the number of rows in the maze
- COL: stores the number of columns in the maze
- maze_csv_file: stores the filename of the csv file

This class involves 1 constructor and 2 member functions.

- The constructor GameMapGenerator() construct a new Random instance and put it in the class's variable rand. It also put the string stored in the input parameter "maze_csv_file" to the class's variable maze_csv_file.
- The function PrimMazeGenerator() will create a random maze of size ROW*COL at anytime, and will return the maze in a 2D char array.
- The function to_csv() will write the data of the maze to a csv file.

To create a random maze map, programmers should create a GameMapGenerator class instance, and then call the PrimMazeGenerator() method to generate a maze of 2D char array.

To write the maze data to a csv file, call the to_csv() method by passing the maze of 2D char array as the input parameter. The maze data can then ben found in the csv file pecified.

字段概要

字段

修饰符和类型	字段	说明
Random	rand	rand: stores a java.util.Random class instance

构造器概要

构造器

构造器	说明
GameMapGenerator(String maze_csv_file)	GameMapGenerator() is a constructor for GameMapGenerator.

方法概要

所有方法 | 实例方法 | 具体方法

修饰符和类型	方法	说明
char[][]	PrimMazeGenerator()	Randomly generates a maze map at any time, and return the data of the maze map as 2D char array.
void	to_csv(char[][] maze_data)	Writes the maze map data generated by PrimMazeGenerator() to a csv file with the filename stored in this.maze_csv_file

从类继承的方法 java.lang.Object

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

字段详细资料

rand

```
public Random rand
```

rand: stores a java.util.Random class instance

构造器详细资料

GameMapGenerator

```
public GameMapGenerator(String maze_csv_file)
```

GameMapGenerator() is a constructor for GameMapGenerator. It initializes the data members "rand" by creating a new instance of java.util.Random class It initializes the data members "this.maze_csv_file" by assigning maze_csv_file to it.

参数:

maze_csv_file - : The filename of the csv file that stores the maze data.

方法详细资料

PrimMazeGenerator

```
public char[][] PrimMazeGenerator()
```

Randomly generates a maze map at any time, and return the data of the maze map as 2D char array.

This maze generator uses the idea of Prim's Minimum Spanning Tree (MST) algorithm.

返回:

maze: It stores the data of the randomly generated maze map.

to_csv

```
public void to_csv(char[][] maze_data)
            throws IOException
```

Writes the maze map data generated by PrimMazeGenerator() to a csv file with the filename stored in this.maze_csv_file

参数:

maze_data - : the maze map data that are stored in a 2D char array

抛出:

IOException - if the reading/writing operation is not successful

程序包 game_algorithm

类 ShortestPathGenerator

java.lang.Object
    game_algorithm.ShortestPathGenerator

```
public class ShortestPathGenerator
extends Object
```

This class is used to calculate the shortest path in a maze.

To calculate the shortest path in a maze, an object is needed to be created. Then, calling calculate_shortest_path() will return an ArrayList object. This ArrayList object contains a list of Location objects for each vertex along the shortest path in the maze.

To generate a .csv file that contains the coordinates along the vertices in the shortest path, calling output_file() is required after calculate_shortest_path(). The .csv file will be created in the directory specified in the constructor.

构造器概要

构造器

构造器	说明
ShortestPathGenerator(String map_file_path, String output_file_path)	This is a constructor for ShortestPathGenerator class.

方法概要

所有方法 **实例方法** **具体方法**

修饰符和类型	方法
说明	
ArrayList<Location>	calculate_shortest_path(Location entry, Location exit)
This function calculates the shortest path according the given maze data.	
void	load_csv(String map_file_path)
This function is called by the constructor.	
ArrayList<Location>	output_array(Location entry, Location exit, int[][][] previous, ArrayList<Location> Shortest_path_vertices)
void	output_file(ArrayList<Location> Shortest_path_vertices)
This function outputs the coordinates along the shortest path to a .csv file.	

从类继承的方法 java.lang.Object

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

构造器详细资料

ShortestPathGenerator

```
public ShortestPathGenerator(String map_file_path,
                             String output_file_path)
```

This is a constructor for ShortestPathGenerator class.

参数:

`map_file_path` - Directory of the map data file in .csv format. For example, "src/main/java/MazeData.csv".

`output_file_path` - Directory of the file in .csv format for outputing the coordinates along the shortest path vertices. For example, "src/main/java/shortest_path_at_beginning.csv".

方法详细资料

calculate_shortest_path

```
public ArrayList<Location> calculate_shortest_path(Location entry,
                                                   Location exit)
```

This function calculates the shortest path according the given maze data. Only the vertices travelled in the shortest path will be saved.

参数:

`entry` - Location object that indicates the entry point in the maze.

`exit` - Location object that indicates the exit point in the maze.

返回:

ArrayList consists of a list of Location objects for each vertex along the shortest path in the maze.

output_file

```
public void output_file(ArrayList<Location> Shortest_path_vertices)
```

This function outputs the coordinates along the shortest path to a .csv file.

In the .csv file, the first line in the file will be "s1". Lines afterwards will be in the format of "row, col", where row and col are the row index and column index of a vertex respectively.

To output the file, an ArrayList containing a list of Location objects is passed to this function.

参数:

`Shortest_path_vertices` - An ArrayList object that contains a list of Location objects for each vertex along the shortest path.

load_csv

`public void load_csv(String map_file_path)`

This function is called by the constructor. It is to import the maze data into the object for further calculation.

Each char in .csv file represent a vertex in the map. This function converts every char into a int[30][30] array, where given the map has 30 rows and 30 columns.

参数:

`map_file_path` - Directory of the map data file in .csv format.

game_states（Function C）

程序包 game_states

package game_states

All Classes and Interfaces | 类 | Enum Classes | Record Classes

类	说明
GameState	The fixed representation of three possible game ending conditions
GameStateController	The monitor of the game state.
Location	The class Location involves 2 data members, which are row and col.
Move	An action of moving a player.
Move.Down	The action of moving down.
Move.Left	The action of moving left.
Move.Right	The action of moving right.
Move.Up	The action of moving up.

程序包 game_states

类 GameStateController

java.lang.Object
    game_states.GameStateController

public class GameStateController
extends Object

The monitor of the game state. As the game goes, the GameStateController takes a snapshot of the game after each move by recording the updated locations of Tom and Jerry. **The original game map is not modified.**

字段概要

字段

修饰符和类型	字段
说明	
Location	entryLocation
The location of entry point, IMMUTABLE	
Location	exitLocation
The location of exit point, IMMUTABLE	
Location	JerryLocation
The latest location of Tom	
Map<Location, game_entities.Vertex>	location2vertex
The hashmap storing the vertex entity at each location	
static final int	SIDE_LENGTH
The side length of the game map	
Location	TomLocation
The latest location of Tom	

构造器概要

构造器

构造器	说明
GameStateController(ArrayList<ArrayList<Integer>> MazeMap, Location start, Location end)	Constructor of this class

方法概要

所有方法 | 实例方法 | 具体方法

修饰符和类型	方法	说明
boolean	canMove(int charId, Move move)	Determines whether a proposed move request is valid
GameState	gameStateOutcome()	Determine the current state of the game
Location	getCharacterLocation(int charId)	Fetch the latest location of a character
boolean	moveCharacter(int charId, Move move)	Honor the move request by a character after checking its validity.
ArrayList<Location>	reachablePositions(int characterID, int allowed_num_move)	Given a quota of allowed_num_move, return all reachable positions by the given character

字段详细资料

location2vertex

public Map<Location, game_entities.Vertex> location2vertex

The hashmap storing the vertex entity at each location

TomLocation

public Location TomLocation

The latest location of Tom

JerryLocation

public Location JerryLocation

The latest location of Tom

entryLocation

public Location entryLocation

The location of entry point, IMMUTABLE

exitLocation

public Location exitLocation

The location of exit point, IMMUTABLE

SIDE_LENGTH

```
public static final  int  SIDE_LENGTH
```

The side length of the game map

另请参阅:

常量字段值

构造器详细资料

GameStateController

```
public  GameStateController(ArrayList<ArrayList<Integer>>  MazeMap,
                            Location  start,
                            Location  end)
```

Constructor of this class

参数:

`MazeMap` - the underlying game map that initiates the first snapshot of game state

`start` - the entry location of the map (starting point of Jerry)

`end` - the exit location of the map (starting point of Tom)

方法详细资料

canMove

```
public  boolean  canMove(int  charId,
                         Move  move)
```

Determines whether a proposed move request is valid

A move is NOT valid if either:

- The resulted move puts the character at a location out of the map
- The resulted move puts the character at a location occupied by a barrier

参数:

`charId` - initiator of the move request, 0 for Tom, 1 for Jerry

`move` - the requested move

moveCharacter

```
public  boolean  moveCharacter(int  charId,
                               Move  move)
```

Honor the move request by a character after checking its validity.

Update the records of the locations of Tom and Jerry

参数:

`charId` - the initiator of the move request

`move` - the move request

getCharacterLocation

```
public Location getCharacterLocation(int charId)
```

Fetch the latest location of a character

参数:

`charId` - the target character to get location from

返回:

the location of the specified character

gameStateOutcome

```
public GameState gameStateOutcome()
```

Determine the current state of the game

The game is ended if either one of these conditions happen:

- Tom reaches Jerry's Location
- Jerry reaches exit point Location

Otherwise, the game goes on

返回:

the current game state

reachablePositions

```
public ArrayList<Location> reachablePositions(int characterID,
                                              int allowed_num_move)
```

Given a quota of allowed_num_move, return all reachable positions by the given character

参数:

`characterID` - the character concerned

`allowed_num_move` - the number of moves that can be made

程序包 game_states

类 Move

java.lang.Object
    game_states.Move

直接已知子类:

Move.Down, Move.Left, Move.Right, Move.Up

```
public abstract sealed class Move
extends Object
permits Move.Down, Move.Left, Move.Right, Move.Up
```

An action of moving a player.

嵌套类概要

嵌套类

修饰符和类型	类 说明
static final class	Move.Down The action of moving down.
static final class	Move.Left The action of moving left.
static final class	Move.Right The action of moving right.
static final class	Move.Up The action of moving up.

方法概要

所有方法 | **实例方法** | **抽象方法**

修饰符和类型	方法 说明
abstract @NotNull Location	nextPosition(@NotNull Location currentLocation) Generates the next position after the move based on the current position.

从类继承的方法 java.lang.Object

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

方法详细资料

nextPosition

```
@NotNull
public abstract @NotNull Location nextPosition(@NotNull
                                               @NotNull Location currentLocation)
```

Generates the next position after the move based on the current position.

参数:

currentLocation - The current position.

返回:

The next position.

game_scene (Function C)

程序包 game_scene

package game_scene

类

类	说明
LandingPageController	The controller of the landing page the user sees first when the program starts
LandingPageView	The viewer of the landing page which user sees first after starting the program
MazeMapController	The controller of the maze map
MazeMapViewer	The user view of the maze map
VertexController	The controller of a vertex on the map
VertexViewer	The user view of a vertex on the map

程序包 game_scene

类 LandingPageController

java.lang.Object
    game_scene.LandingPageController

public class LandingPageController
extends Object

The controller of the landing page the user sees first when the program starts

字段概要

字段

修饰符和类型	字段
说明	
static LinkedBlockingQueue<String>	buttonHitRecords
The FIFO blocking buffer for storing the users' button pressing movements	

构造器概要

构造器

构造器	说明
LandingPageController(JButton startButton, JFrame LandingPage)	Constructor of this class

方法概要

所有方法 | 实例方法 | 具体方法

修饰符和类型	方法	说明
LinkedBlockingQueue<String>	getButtonHitRecords()	

从类继承的方法 java.lang.Object

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

字段详细资料

buttonHitRecords

```
public static LinkedBlockingQueue<String> buttonHitRecords
```

The FIFO blocking buffer for storing the users' button pressing movements

构造器详细资料

LandingPageController

```
public LandingPageController(JButton startButton,
                             JFrame LandingPage)
```

Constructor of this class

参数:

`startButton` - this class will assign action listener for ths startButton

`LandingPage` - the landing page this class controls

方法详细资料

getButtonHitRecords

```
public LinkedBlockingQueue<String> getButtonHitRecords()
```

返回:

the FIFO blocking buffer for storing the users' button pressing movements

程序包 game_scene

类 MazeMapController

java.lang.Object
    game_scene.MazeMapController

public class MazeMapController
extends Object

The controller of the maze map

字段概要

字段

修饰符和类型	字段 说明
static final int	SIDE_LENGTH The side length of the square map

构造器概要

构造器

构造器	说明
MazeMapController(ArrayList<ArrayList<Integer>> MazeMap, Location entry, Location exit)	Constructor of this class

方法概要

所有方法 | 实例方法 | 具体方法

修饰符和类型	方法 说明
HashMap<Location, VertexController>	getLocationVertexControllerMap() Accessor function
void	highlightPath(ArrayList<Location> pathComponents, String color_code) Highlight the vertices that form a path, called when showing the shortest path to the user
void	insertImage(Location location, String imagepath) Insert an image on a vertex, e.g. the image of Tom and Jerry will be inserted into the vertices they reside on
void	removeHighlightPath() Restore the appearance of the path that was previously highlighted
void	renderMap(Location newtom, Location newjerry, Location oldtom, Location oldjerry, boolean unittesting) Update the appearance of the map based on the latest location of Tom and Jerry

字段详细资料

SIDE_LENGTH

```
public static final int SIDE_LENGTH
```

The side length of the square map

另请参阅:

常量字段值

构造器详细资料

MazeMapController

```
public MazeMapController(ArrayList<ArrayList<Integer>> MazeMap,
                         Location entry,
                         Location exit)
```

Constructor of this class

参数:

`MazeMap` - The maze map data based on which this controller will monitor the game state

`entry` - The entry point of the maze map

`exit` - The exit point of the maze map

方法详细资料

getLocationVertexControllerMap

```
public HashMap<Location,VertexController> getLocationVertexControllerMap()
```

Accessor function

返回:

the map storing the controller instance of the vertex at eacb specified location

insertImage

```
public void insertImage(Location location,
                        String imagepath)
                 throws IOException
```

Insert an image on a vertex, e.g. the image of Tom and Jerry will be inserted into the vertices they reside on

参数:

`location` - The location of the vertex to insert image

`imagepath` - The path of the image to be inserted

抛出:

`IOException` - if the imagepath is invalid

renderMap

```
public void renderMap(Location newtom,
                      Location newjerry,
                      Location oldtom,
                      Location oldjerry,
                      boolean unittesting)
               throws IOException
```

Update the appearance of the map based on the latest location of Tom and Jerry

参数:

`newjerry` - the new location that jerry is at, insert Jerry's image into the vertex at this location

`newtom` - the new location that tom is at, insert Tom's image into the vertex at this location

`oldjerry` - the last location jerry was at, remove Jerry's image from the vertex at this location

`oldtom` - the last location tom was at, remove Tom's image from the vertex at this location

抛出:

`IOException` - if there is error in image insertion operation

highlightPath

```
public void highlightPath(ArrayList<Location> pathComponents,
                          String color_code)
```

Highlight the vertices that form a path, called when showing the shortest path to the user

参数:

`pathComponents` - the vertices on the path to highlight

removeHighlightPath

```
public void removeHighlightPath()
```

Restore the appearance of the path that was previously highlighted

TomJerryGame.java (Function C)

类 TomJerryGame

java.lang.Object
    TomJerryGame

public class **TomJerryGame**
extends Object

Each instance of this class is an independent game with its own maze map, game states, and GUI components

字段概要

字段

修饰符和类型	字段	说明
String	difficulty	The difficulty of the game, to be selected by player
Location	entry	Entry position (start point of Jerry)
Location	exit	Exit position (start point of Tom)
int	jerrySpeed	The speed of Jerry, to be determined by difficulty selected
ArrayList<ArrayList<Integer>>	mazeMap	The 2D maze map
MazeMapController	mazeMapController	The controller of the maze map GUI view
ShortestPathGenerator	shortestPathGenerator	The generator of the shortest path
final int	SIDE_LENGTH	The side length of the square game map
GameStateController	stateController	The monitor of the game state
int	tomSpeed	The speed of Tom, to be determined by difficulty selected
game_scene.WindowsView	windowsView	The main frame view containing all GUI elements of the game

构造器概要

构造器

构造器	说明
TomJerryGame(String map_file_path, String sp_output_path)	Constructor of this class

方法详细资料

readData

```
public ArrayList<ArrayList<Integer>> readData(String file,
                                              String separator)
                                       throws IOException
```

Read maze map data from a specified csv file

参数:

`file` - the path of the csv file containing the maze map data

`separator` - the separator used by the csv file

抛出:

`IOException` - if the filepath points to invalid csv file

setDifficulty

```
public boolean setDifficulty(boolean unittesting,
                             String unittesting_fixed_difficulty)
```

Choose mode of difficulty

- Easy: PLAYER_SPEED = 3, COMPUTER_SPEED = 3
- Medium: PLAYER_SPEED = 4, COMPUTER_SPEED = 6
- Hard: PLAYER_SPEED = 6, COMPUTER_SPEED = 9

参数:

`unittesting` - ONLY for unit testing, true when this method is evoked when unit testing

`unittesting_fixed_difficulty` - ONLY for unit testing, the fixed difficulty set when unit testing

返回:

whether user successfully selected a valid difficulty level

TomMovesOneStep

```
public  void  TomMovesOneStep(int  remainingMoves,
                              boolean  unittesting,
                              String  unit_test_direction)
                       throws IOException
```

Calculate the shortest path from Tom to Jerry Move Tom by one step following this path

参数:

`remainingMoves` - the remaining quota of moves by Tom

`unittesting` - ONLY for unit testing, to fix the movements of tom

`unit_test_direction` - ONLY for unit testing, the hard-coded direction to move

抛出:

`IOException` - when exception occurs in updating the GUI of the map

TomMoves

```
public  void  TomMoves()
              throws IOException
```

Called when it is Tom's turn to move

抛出:

`IOException`

JerryMoves

```
public  void  JerryMoves(boolean  highlighted,
                         boolean  unittesting,
                         LinkedBlockingQueue<Move>  unit_test_movements)
                  throws InterruptedException,
                         IOException,
                         UnsupportedAudioFileException,
                         LineUnavailableException
```

A single round for the player Receives button pressing actions from player and change game state accordingly

参数:

`highlighted` - whether any of the vertices are currently highlighted

`unittesting` - ONLY for unit testing

`unit_test_movements` - ONLY for unit testing, the hard coded movements when unit testing

抛出:

`InterruptedException`

`IOException`

`UnsupportedAudioFileException`

`LineUnavailableException`

showHintsOnMap

```
public  void  showHintsOnMap(boolean  do_SP,
                             boolean  do_TRL)
```

Show the shortest path from Jerry to the exit AND the reachable positions by Tom Shown at end of every even-number rounds

参数:

`do_SP` - highlight the shortest path

`do_TRL` - highlight the reachable locations by Tom

run

```
public  void  run(LandingPageView  landingPageView,
                  boolean  unittesting,
                  boolean  unittesting_from_main,
                  LinkedBlockingQueue<Move>  moves_for_testing)
         throws IOException,
                InterruptedException,
                UnsupportedAudioFileException,
                LineUnavailableException
```

The lifecycle of a single game instance

参数:

`landingPageView` - the landing page that popped up before this game instance, needs to be closed as game starts

`unittesting` - ONLY for unit testing, true when this method is run under testing mode

`unittesting_from_main` - ONLY for unit testing, true when this method is called by GameFactory when unit testing

`moves_for_testing` - ONLY for unit testing, the automatic moves for players when this method is called when unit testing

抛出:

`IOException`

`InterruptedException`

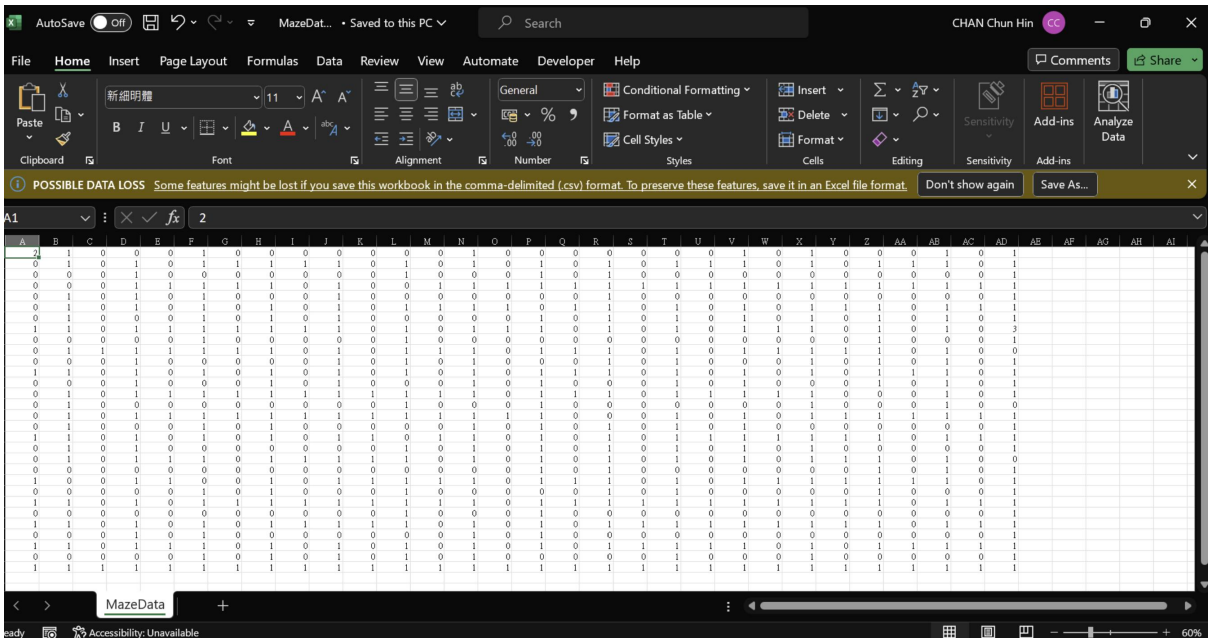
`UnsupportedAudioFileException`

`LineUnavailableException`

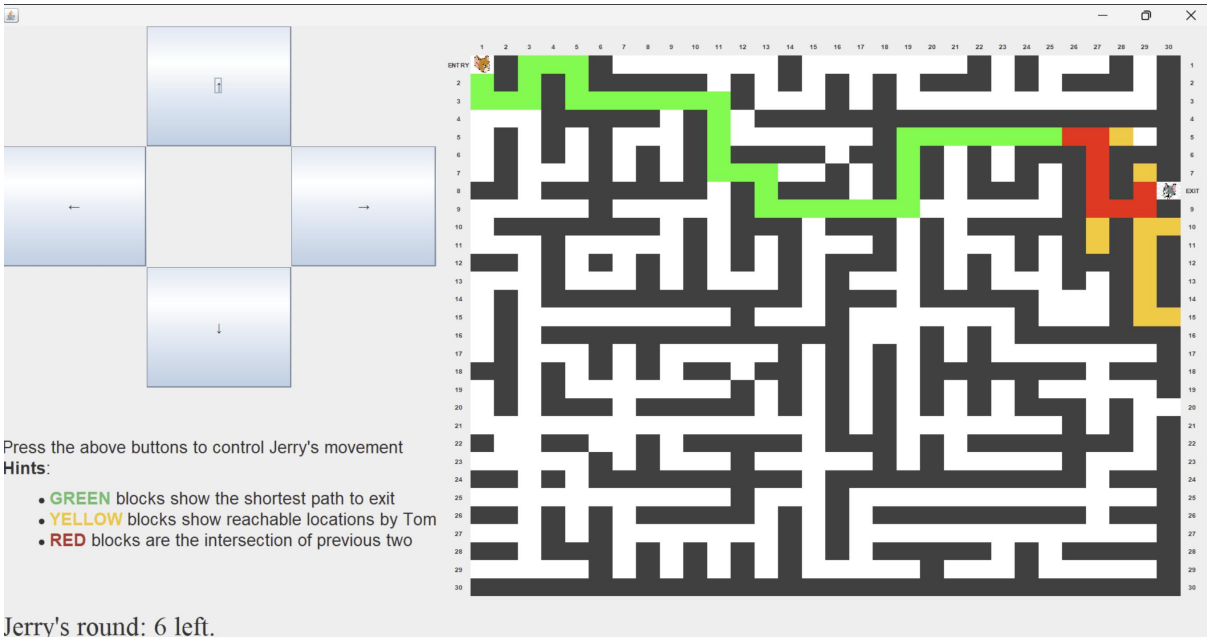
Screenshots

Function A - Maze maps in CSV and GUI format

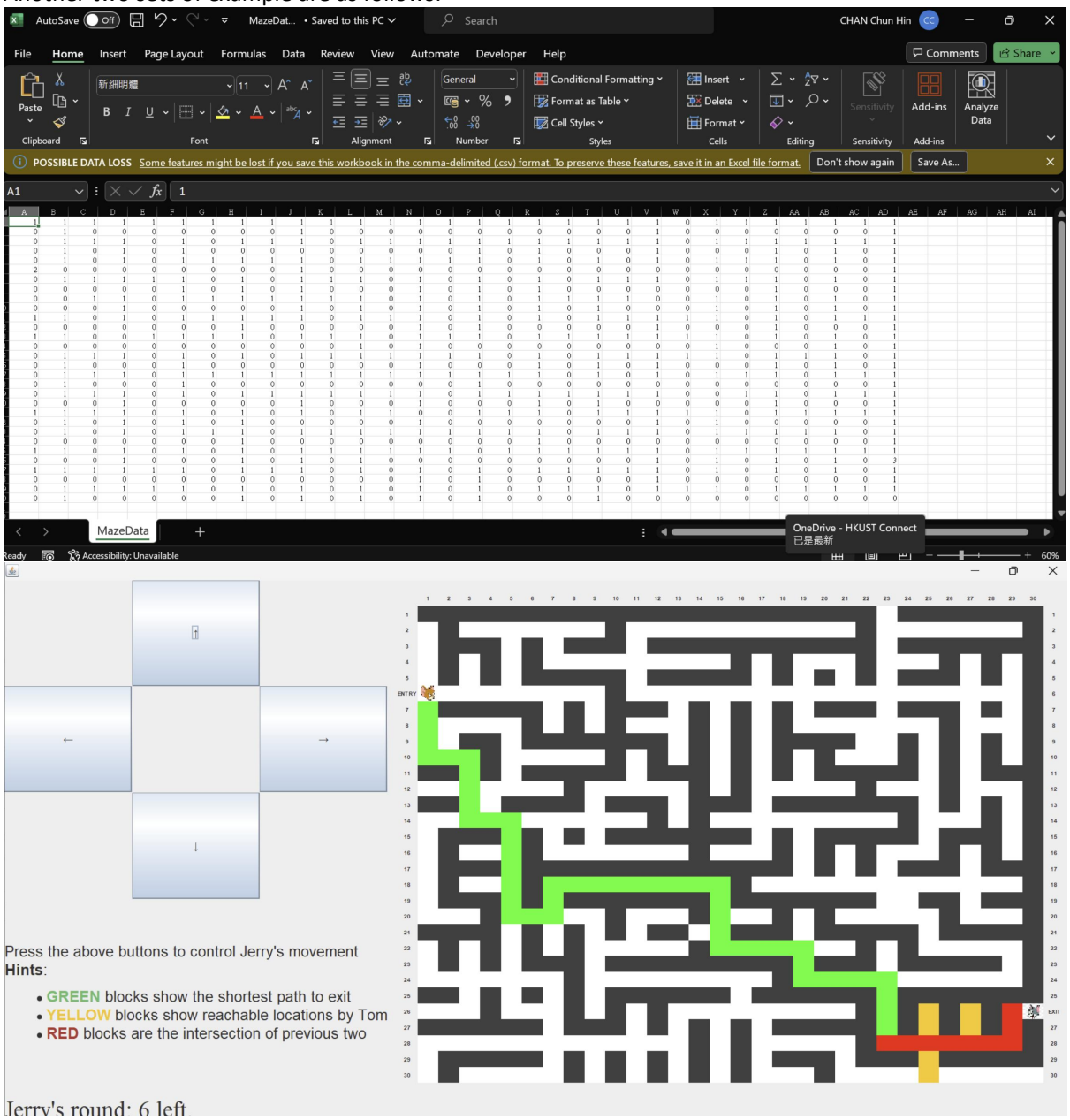
The generated maze map in csv file is stored under src/main/java/MazeData.csv
(0: clear vertex, 1: barrier, 2: entry, 3: exit)

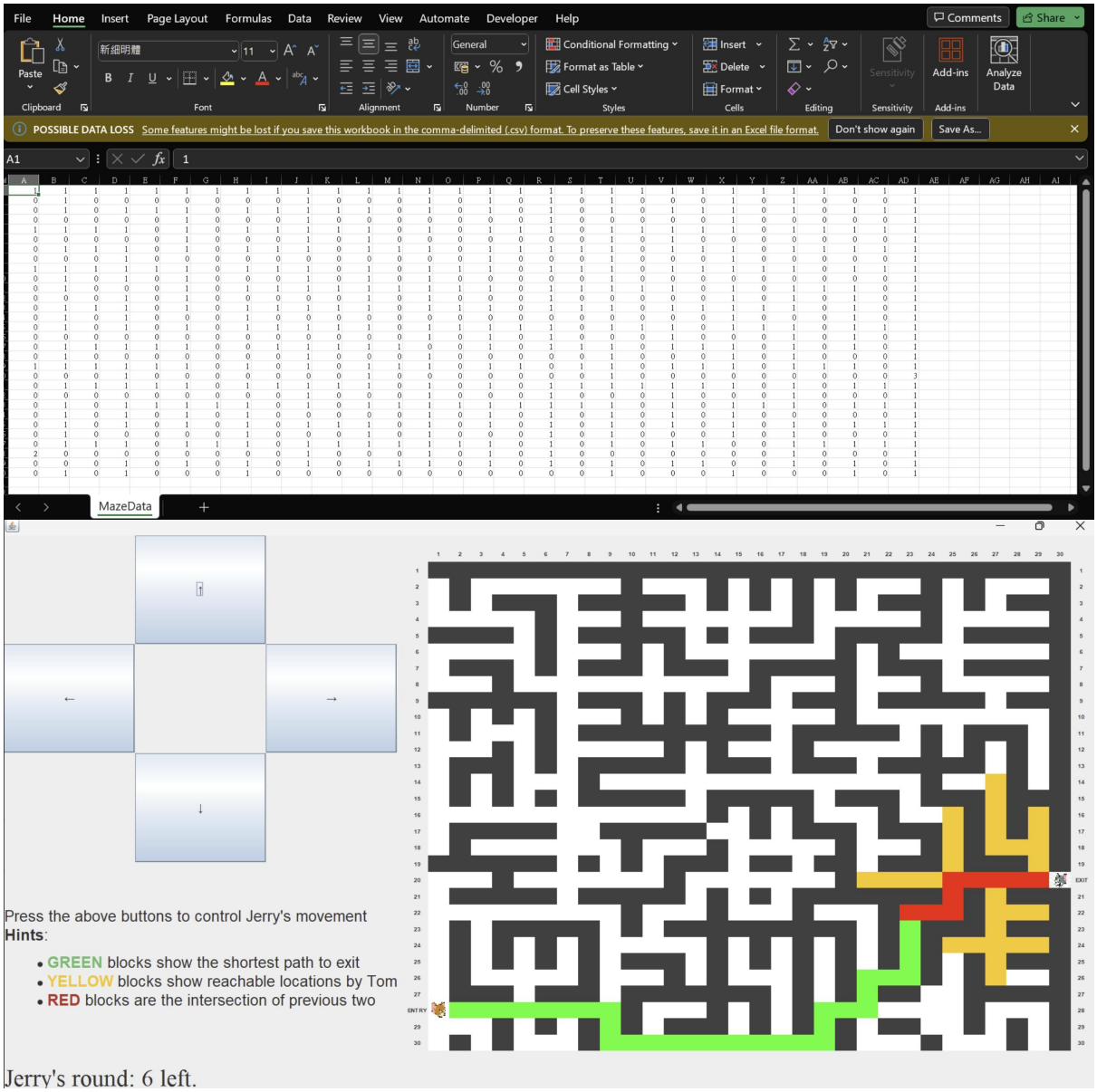


The generated maze map in GUI format, which represents the same map as the above csv file.



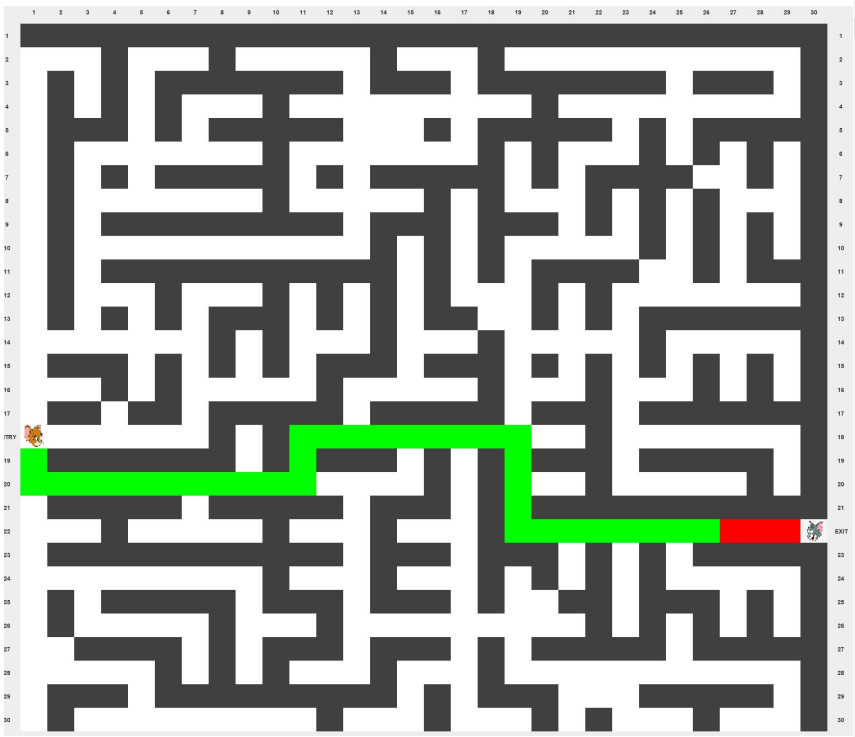
Another two sets of example are as follows:




File Home Insert Page Layout Formulas Data Review View Automate Developer Help
Comments Share
Paste
Clipboard Font Alignment Number Styles Cells Editing Sensitivity Add-ins Analyze Data
POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format. Don't show again Save As...
A1
MazeData
Press the above buttons to control Jerry's movement
Hints:
GREEN blocks show the shortest path to exit
YELLOW blocks show reachable locations by Tom
RED blocks are the intersection of previous two
EXIT
ENTRY
Jerry's round: 6 left.

Function B: shortest path from entry to exit

The csv file can be found at src/main/java/shortest_path_at_beginning.csv
Column A stores the row and B stores the column of the path (starting from entry at row 2 of the file and exit at the last row of the file)
Note that the coordinates in the csv file are **0-indexed** as opposed to the coordinates shown on the GUI



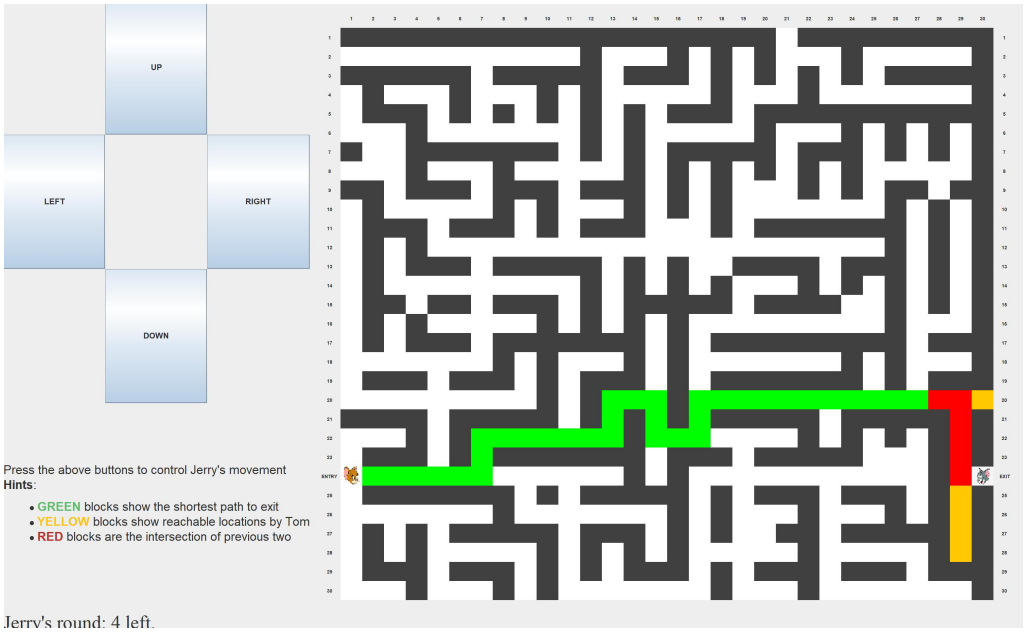
	A	B
1	s1	
2	17	0
3	18	0
4	19	0
5	19	1
6	19	2
7	19	3
8	19	4
9	19	5
10	19	6
11	19	7
12	19	8
13	19	9
14	19	10
15	18	10
16	17	10
17	17	11
18	17	12
19	17	13
20	17	14
21	17	15
22	17	16
23	17	17
24	17	18
25	18	18
26	19	18
27	20	18
28	21	18
29	21	19
30	21	20
31	21	21
32	21	22
33	21	23
34	21	24
35	21	25
36	21	26
37	21	27
38	21	28
39	21	29

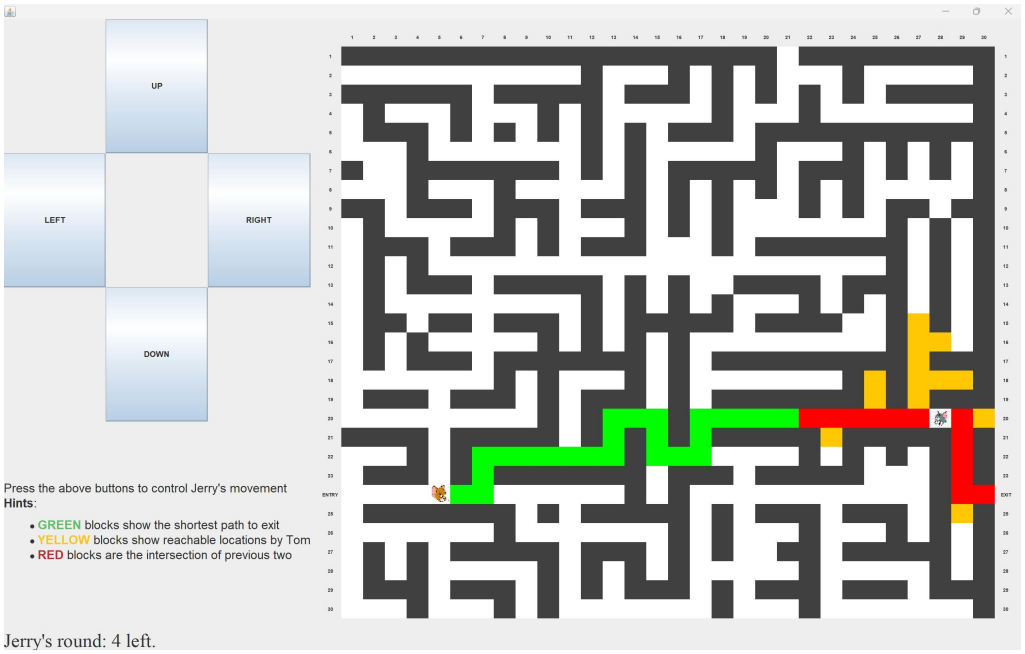
Function C: landing page, game playing interface, winning&losing

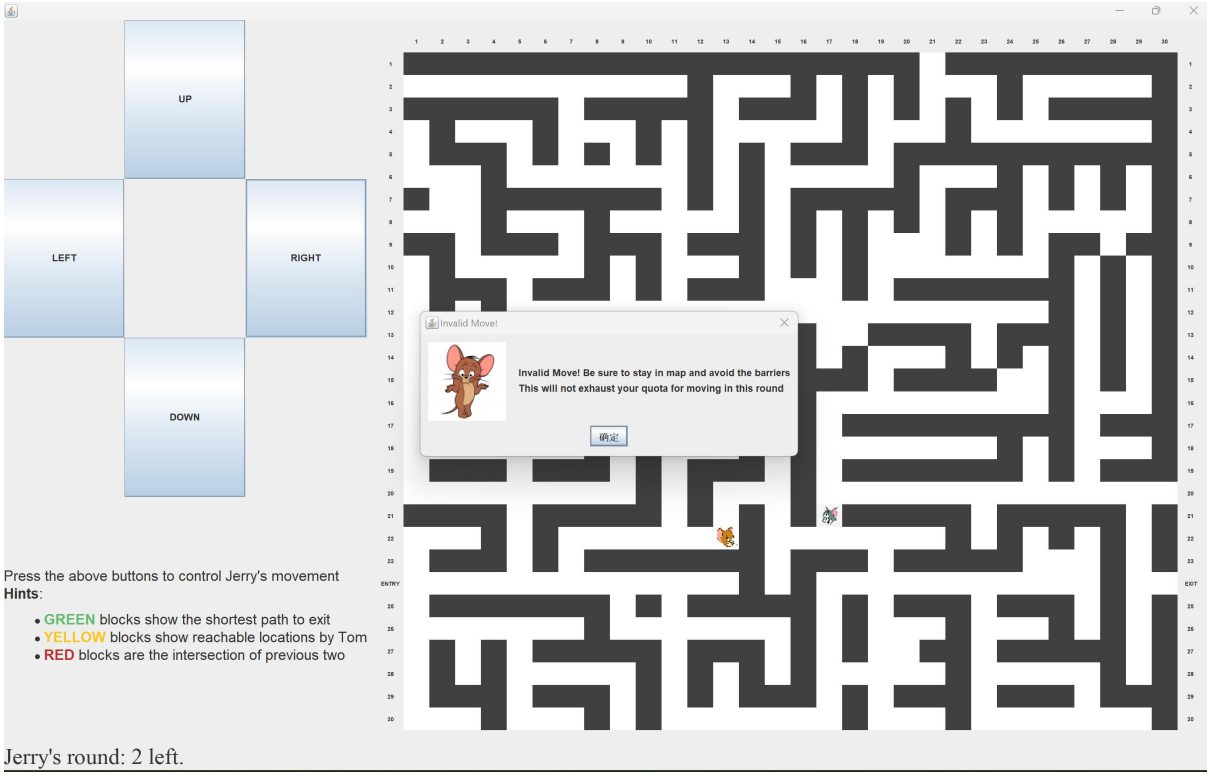
1. Landing Page

2. Select difficulty

3. Game play interface, all notes and hints are clearly marked with html

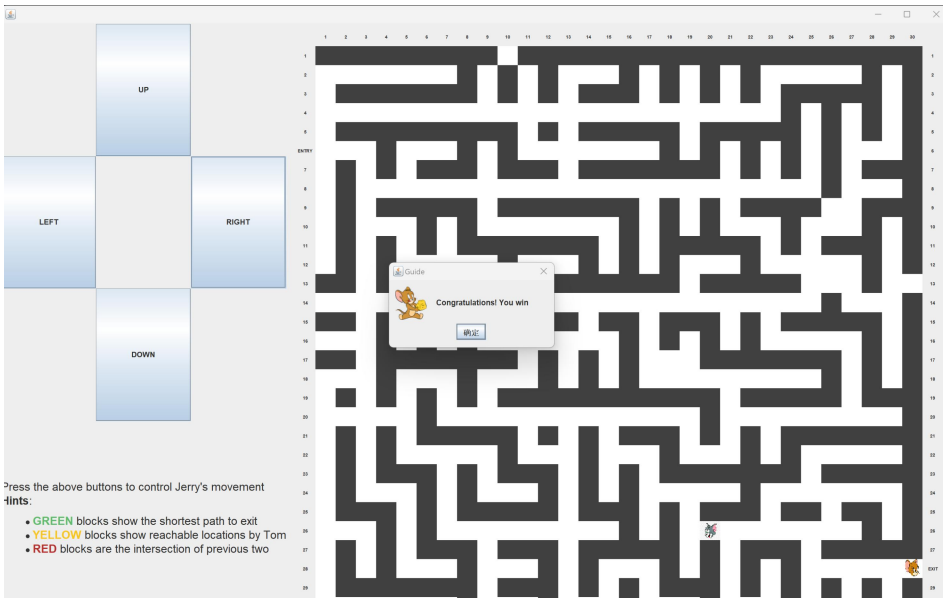


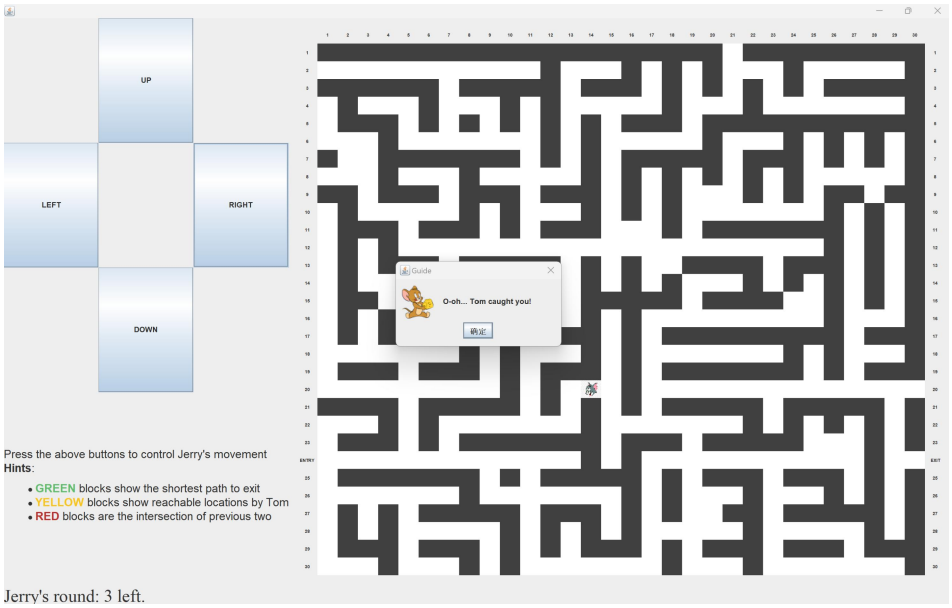




(if hit onto the wall or outside the map)

4. Winning & Losing

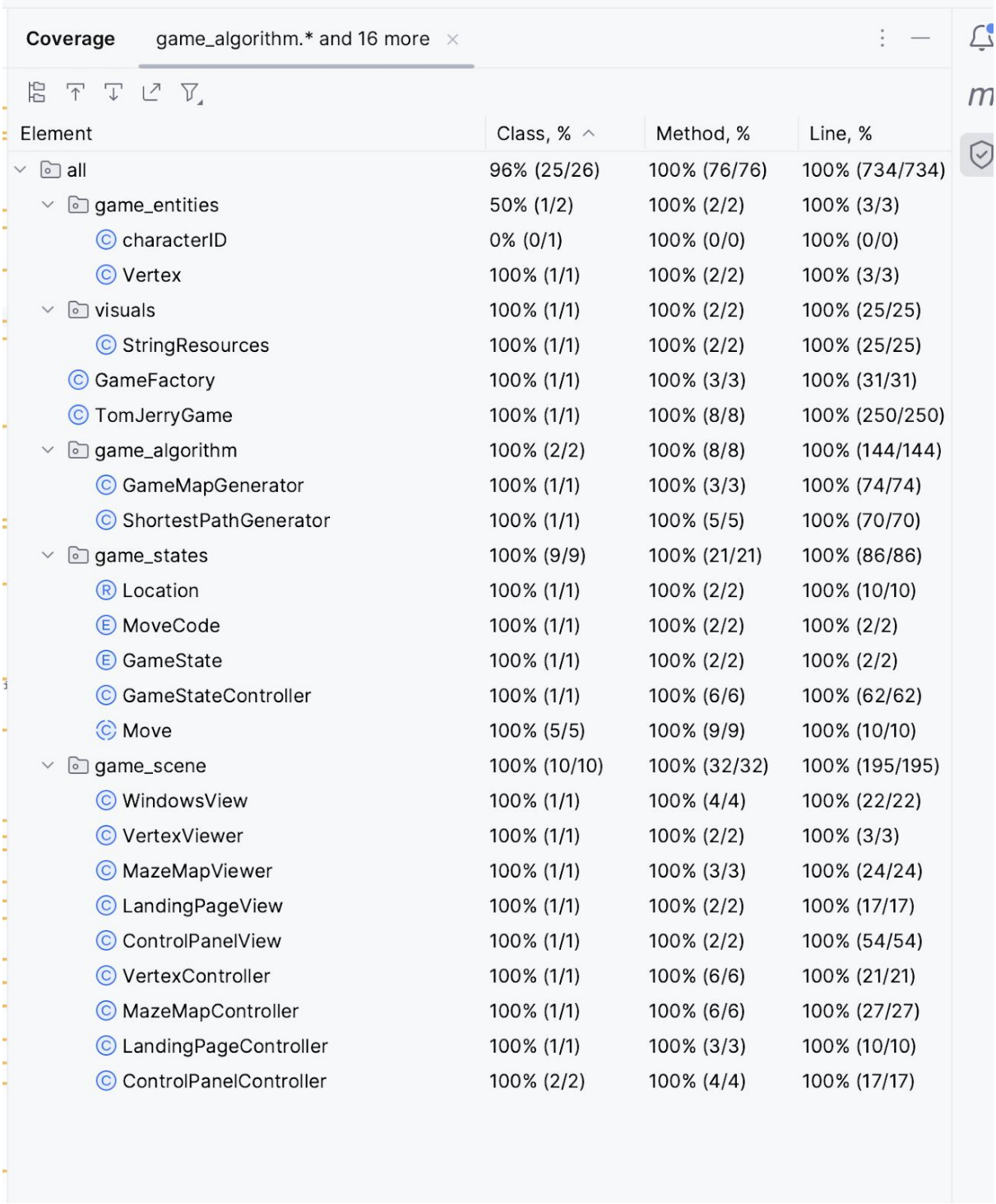




Unit Testing

JUnit version: 4.13.2

1. Line Coverage report generated by JUnit:



Coverage | game_algorithm.* and 16 more

Element	Class, %	Method, %	Line, %
all	96% (25/26)	100% (76/76)	100% (734/734)
game_entities	50% (1/2)	100% (2/2)	100% (3/3)
characterID	0% (0/1)	100% (0/0)	100% (0/0)
Vertex	100% (1/1)	100% (2/2)	100% (3/3)
visuals	100% (1/1)	100% (2/2)	100% (25/25)
StringResources	100% (1/1)	100% (2/2)	100% (25/25)
GameFactory	100% (1/1)	100% (3/3)	100% (31/31)
TomJerryGame	100% (1/1)	100% (8/8)	100% (250/250)
game_algorithm	100% (2/2)	100% (8/8)	100% (144/144)
GameMapGenerator	100% (1/1)	100% (3/3)	100% (74/74)
ShortestPathGenerator	100% (1/1)	100% (5/5)	100% (70/70)
game_states	100% (9/9)	100% (21/21)	100% (86/86)
Location	100% (1/1)	100% (2/2)	100% (10/10)
MoveCode	100% (1/1)	100% (2/2)	100% (2/2)
GameState	100% (1/1)	100% (2/2)	100% (2/2)
GameStateController	100% (1/1)	100% (6/6)	100% (62/62)
Move	100% (5/5)	100% (9/9)	100% (10/10)
game_scene	100% (10/10)	100% (32/32)	100% (195/195)
WindowsView	100% (1/1)	100% (4/4)	100% (22/22)
VertexViewer	100% (1/1)	100% (2/2)	100% (3/3)
MazeMapViewer	100% (1/1)	100% (3/3)	100% (24/24)
LandingPageView	100% (1/1)	100% (2/2)	100% (17/17)
ControlPanelView	100% (1/1)	100% (2/2)	100% (54/54)
VertexController	100% (1/1)	100% (6/6)	100% (21/21)
MazeMapController	100% (1/1)	100% (6/6)	100% (27/27)
LandingPageController	100% (1/1)	100% (3/3)	100% (10/10)
ControlPanelController	100% (2/2)	100% (4/4)	100% (17/17)

Line coverage report for testing every class in the project: 734/734, 100%

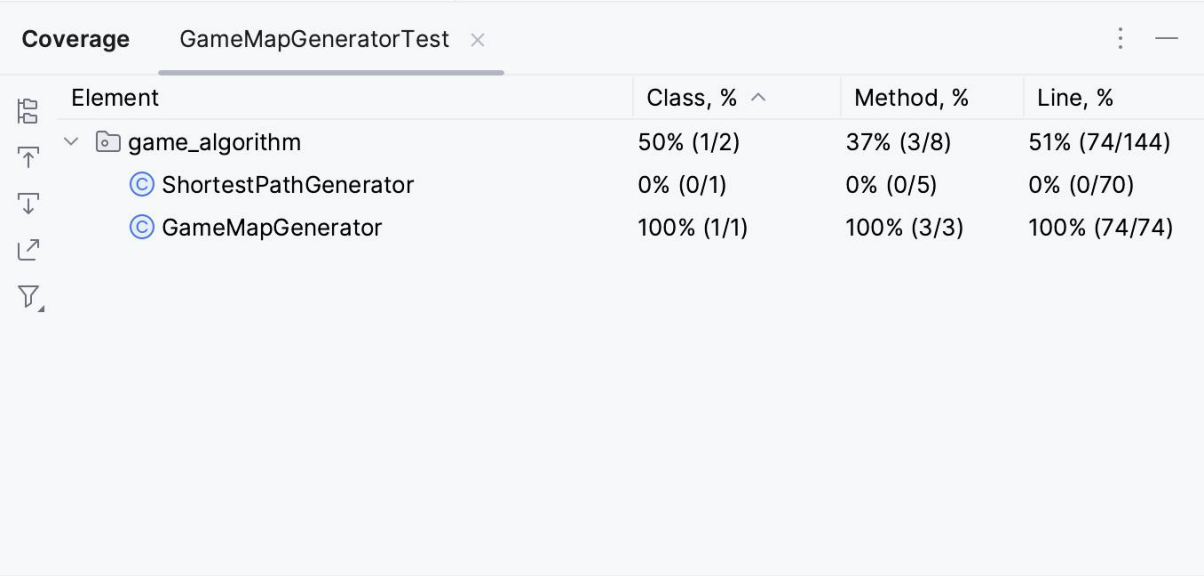
2. Put down which lines are not covered by each test case (e.g., provide the line numbers):

All lines are covered by the test cases. Here is a detailed report for the test cases.

a. GameMapGeneratorTest.java

Total number of test cases in GameMapGeneratorTest.java: 3
Testing target class: GameMapGenerator.java

Coverage report for GameMapGeneratorTest.java



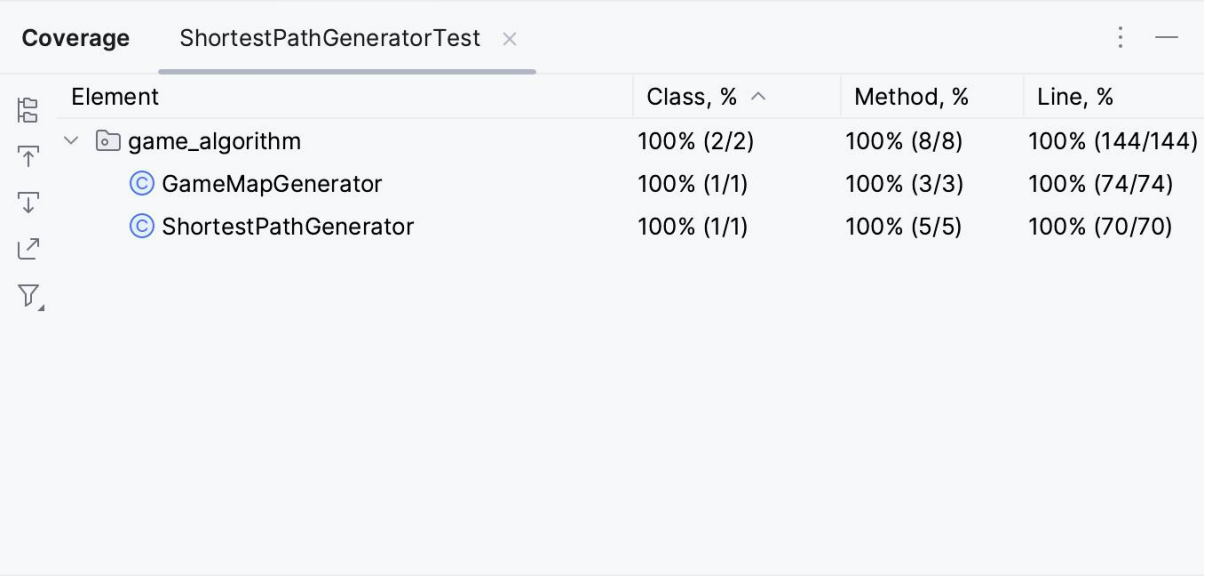
Report for functions in GameMapGenerator.java

		All implemented functions in GameMapGenerator.java		
Test case	Target function	GameMapGenerator	PrimMazeGenerator	to_csv
GameMapGeneratorTest1	GameMapGenerator.GameMapGenerator()	All covered	N/A	N/A
PrimMazeGeneratorTest1	GameMapGenerator.PrimMazeGenerator()	N/A	All covered	N/A
to_csvTest1	GameMapGenerator.to_csv()	N/A	N/A	All covered

b. ShortestPathGeneratorTest.java

Total number of test cases in ShortestPathGeneratorTest.java: 5
Testing target class: ShortestPathGenerator.java

Coverage report for ShortestPathGeneratorTest.java



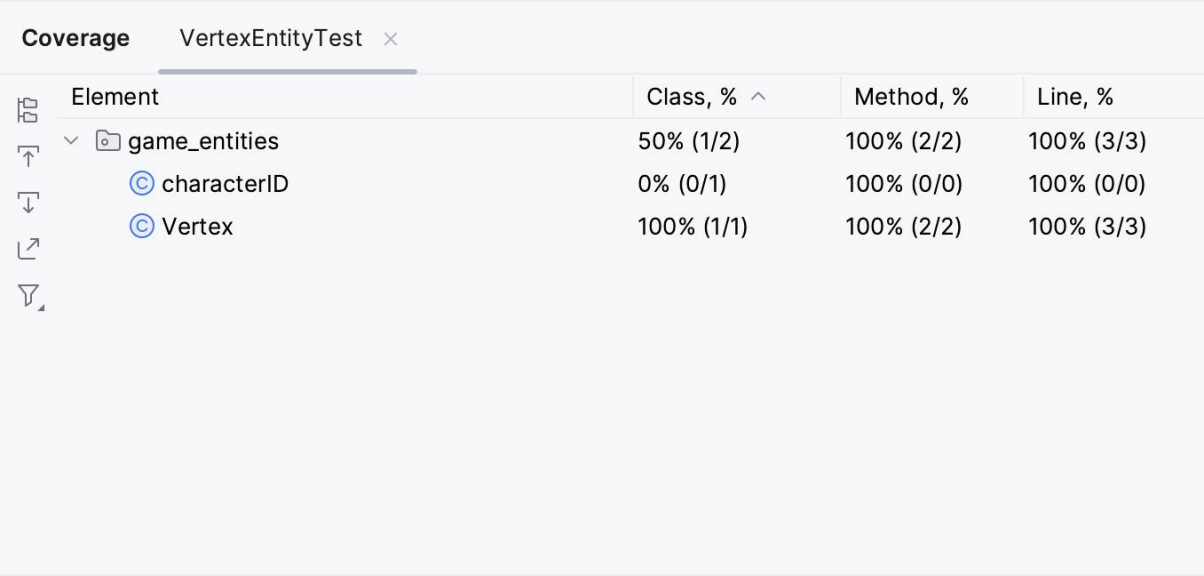
Report for functions in ShortestPathGenerator.java

		All implemented functions in ShortestPathGenerator.java				
Test case	Target function	ShortestPathGenerator	calculate_shortest_path	output_file	output_array	load_csv
test_constructor	ShortestPathGenerator.ShortestPathGenerator()	All covered	N/A	N/A	N/A	N/A
test_calculate_shortest_path	ShortestPathGenerator.calculate_shortest_path()	N/A	All covered	N/A	N/A	N/A
test_output_file	ShortestPathGenerator.output_file()	N/A	N/A	All covered	N/A	N/A
test_output_array	ShortestPathGenerator.output_array()	N/A	N/A	N/A	All covered	N/A
test_load_csv	ShortestPathGenerator.load_csv()	N/A	N/A	N/A	N/A	All covered

c. VertexEntityTest.java

Total number of test cases in VertexEntityTest.java: 2
Testing target class: Vertex.java

Coverage report for VertexEntityTest.java



Report for functions in Vertex.java

		All implemented functions in Vertex.java	
Test case	Target function	Vertex	isClear
testConstructor	Vertex.Vertex()	All covered	N/A
isClear	Vertex.isClear()	N/A	All covered

d. ControlPanelTest.java

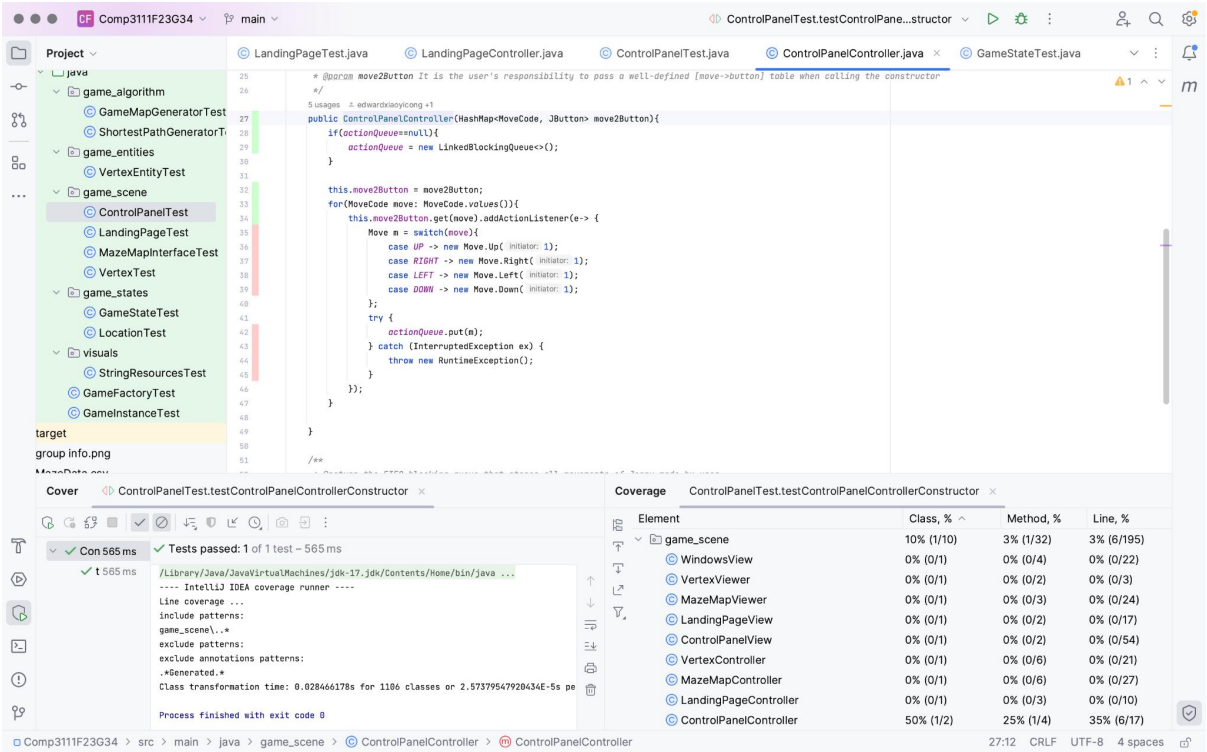
Total number of test cases in ControlPanelTest.java: 6
Testing target class: ControlPanelView.java, ControlPanelController.java

Coverage report for ControlPanelTest.java

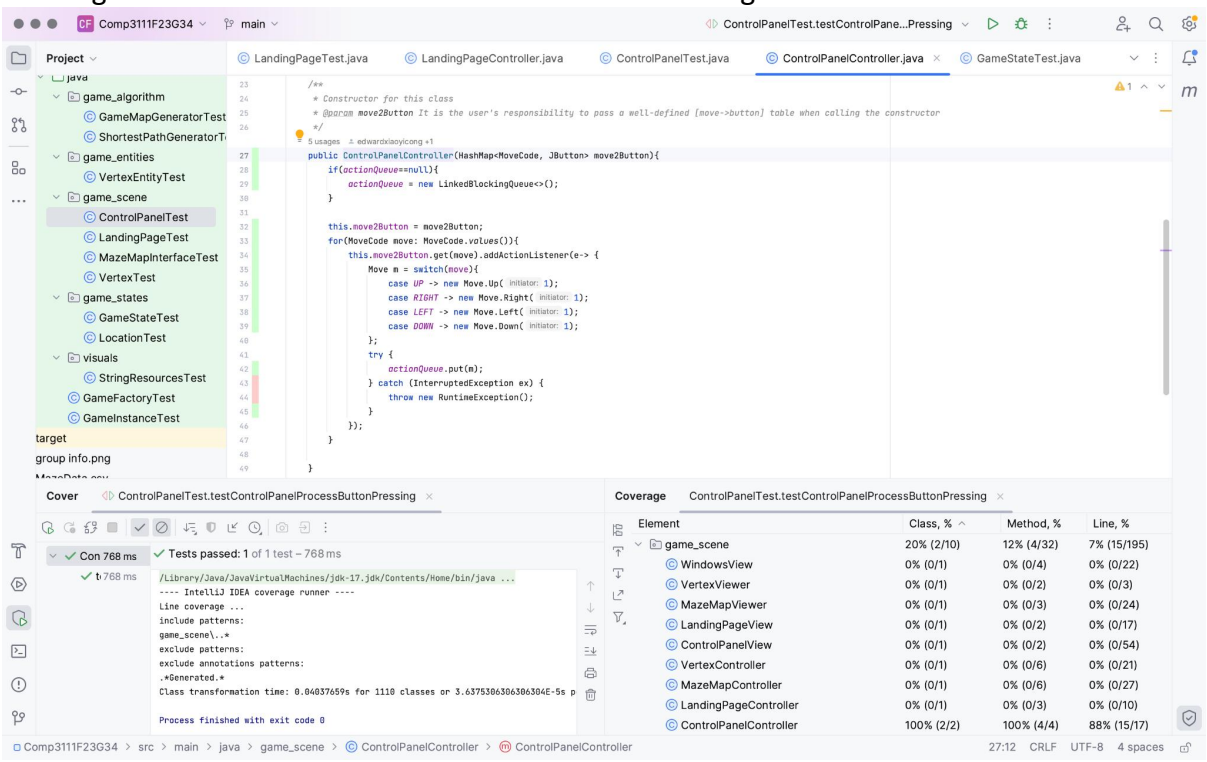
Coverage | ControlPanelTest

Element	Class, %	Method, %	Line, %
game_scene	30% (3/10)	18% (6/32)	36% (71/195)
WindowsView	0% (0/1)	0% (0/4)	0% (0/22)
VertexViewer	0% (0/1)	0% (0/2)	0% (0/3)
MazeMapViewer	0% (0/1)	0% (0/3)	0% (0/24)
LandingPageView	0% (0/1)	0% (0/2)	0% (0/17)
VertexController	0% (0/1)	0% (0/6)	0% (0/21)
MazeMapController	0% (0/1)	0% (0/6)	0% (0/27)
LandingPageController	0% (0/1)	0% (0/3)	0% (0/10)
ControlPanelView	100% (1/1)	100% (2/2)	100% (54/54)
ControlPanelController	100% (2/2)	100% (4/4)	100% (17/17)

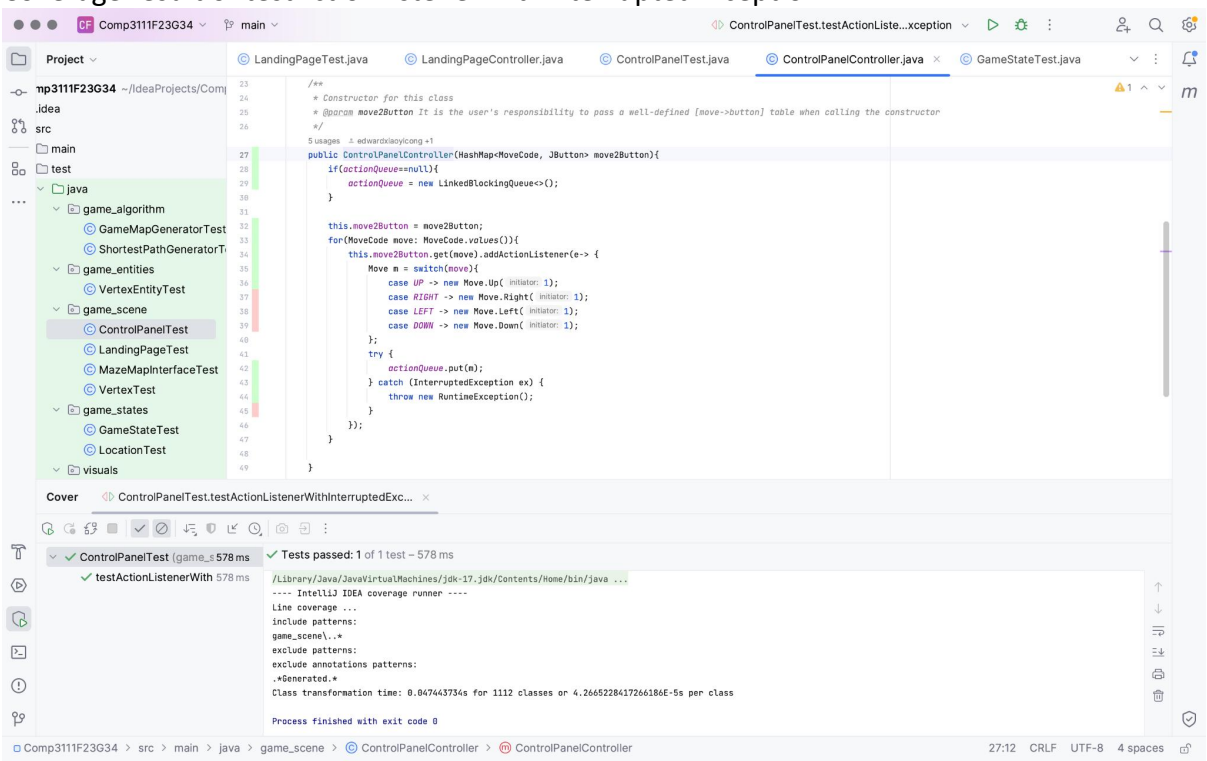
Coverage result of testControlPanelControllerConstructor



Coverage result of testControlPanelProcessButtonPressing



Coverage result of testActionListenerWithInterruptedException



Report for functions in ControlPanelView.java, ControlPanelController.java

		All implemented functions in ControlPanelView.java		All implemented functions in ControlPanelController.java	
Test case	Target function	ControlPanelView	getControlPanelController	ControlPanelController	getActionQueue
testControlPanelViewConstructor	ControlPanelView.ControlPanelView()	All covered	N/A	N/A	N/A
testgetControlPanelController	ControlPanelView.getControlPanelController()	N/A	All covered	N/A	N/A
testControlPanelControllerConstructor	ControlPanelView.ControlPanelController()	N/A	N/A	35 - 45	N/A
testgetControlPanelControllerActionQueue	ControlPanelView.getActionQueue()	N/A	N/A	N/A	All covered
testControlPanelProcessButtonPressing	ControlPanelView.ControlPanelController()	N/A	N/A	43 - 44	N/A
testActionListenerWithInterruptedException	ControlPanelView.ControlPanelController()	N/A	N/A	37 - 39, 45	N/A

e. LandingPageTest.java

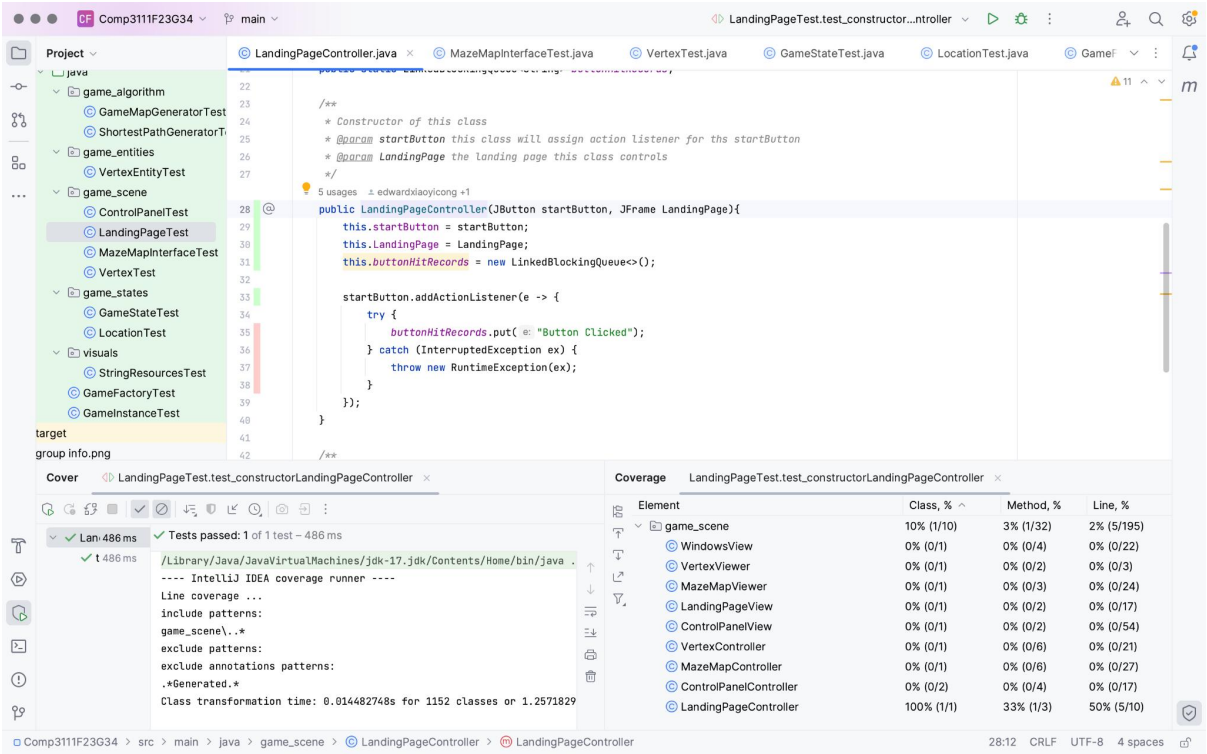
Total number of test cases in LandingPageTest.java: 6
Testing target class: LandingPageView.java, LandingPageController.java

Coverage report for LandingPageTest.java

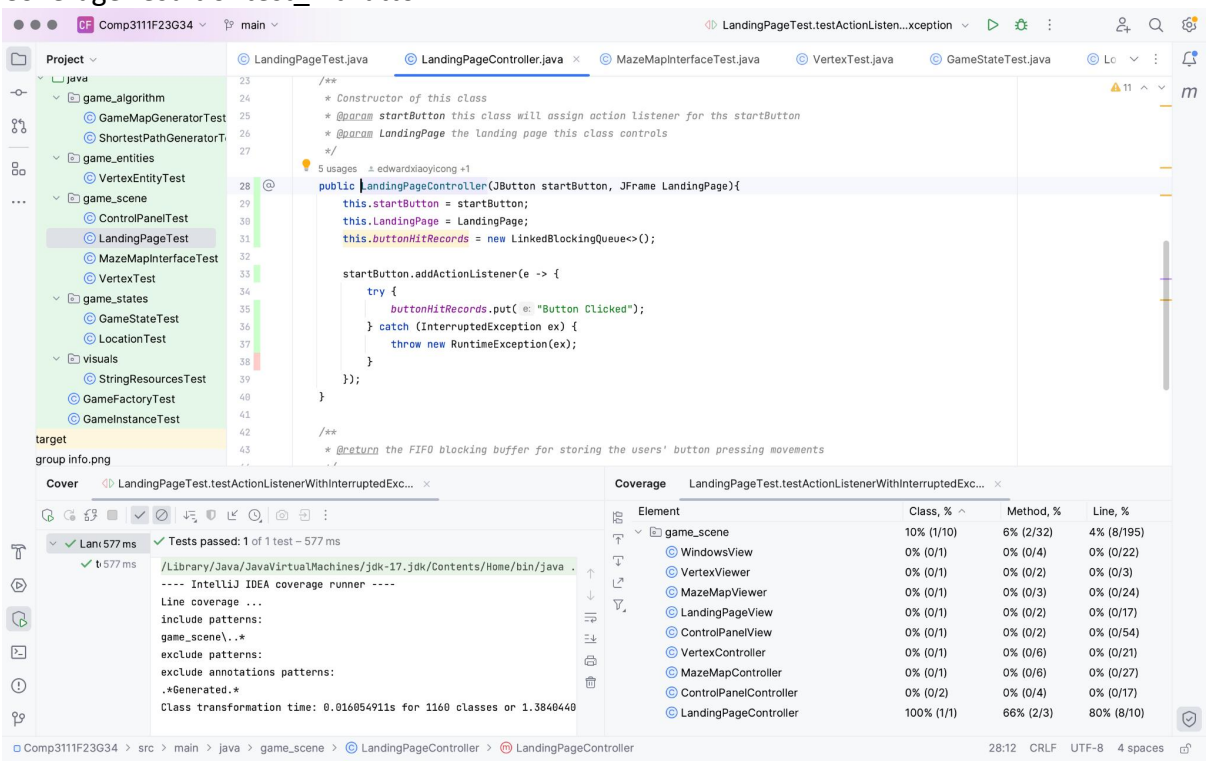
Coverage | LandingPageTest

Element	Class, %	Method, %	Line, %
game_scene	20% (2/10)	15% (5/32)	13% (27/195)
WindowsView	0% (0/1)	0% (0/4)	0% (0/22)
VertexViewer	0% (0/1)	0% (0/2)	0% (0/3)
MazeMapViewer	0% (0/1)	0% (0/3)	0% (0/24)
ControlPanelView	0% (0/1)	0% (0/2)	0% (0/54)
VertexController	0% (0/1)	0% (0/6)	0% (0/21)
MazeMapController	0% (0/1)	0% (0/6)	0% (0/27)
ControlPanelController	0% (0/2)	0% (0/4)	0% (0/17)
LandingPageView	100% (1/1)	100% (2/2)	100% (17/17)
LandingPageController	100% (1/1)	100% (3/3)	100% (10/10)

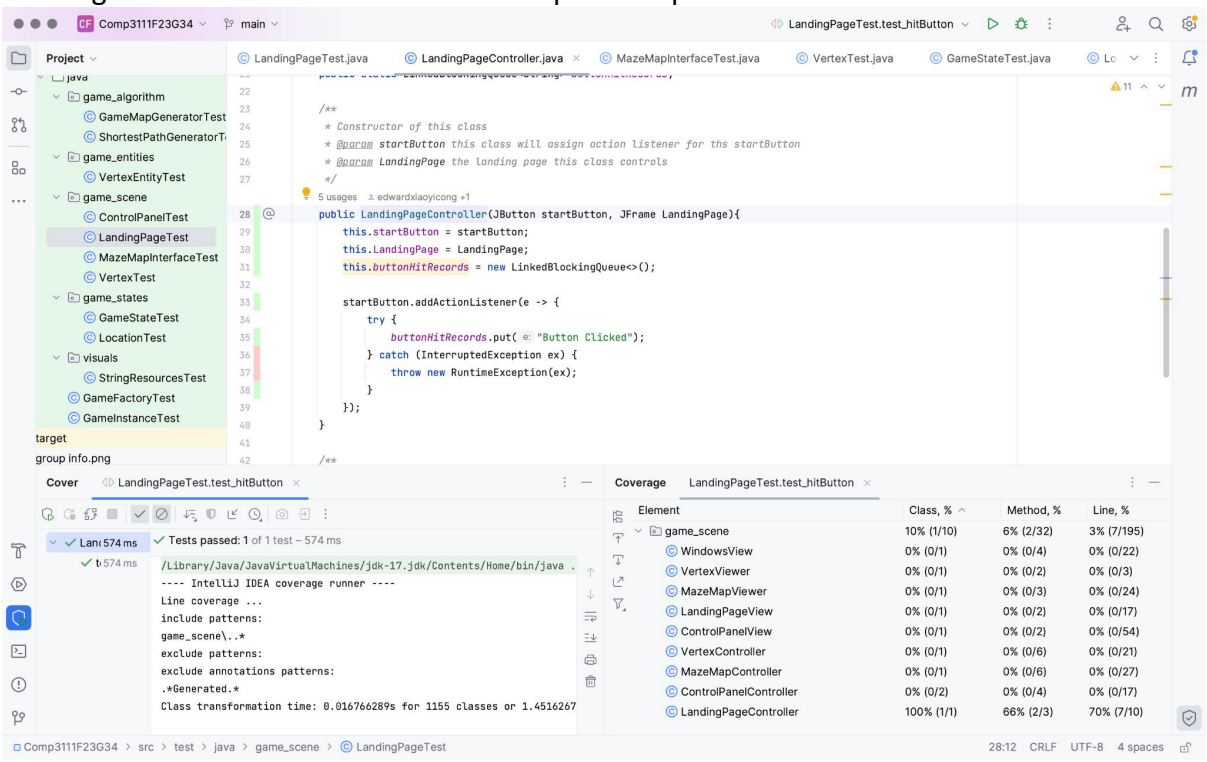
Coverage result of test_constructorLandingPageController



Coverage result of test_hitButton



Coverage of testActionListenerWithInterruptedException



Report for functions in LandingPageView.java, LandingPageController.java

		All implemented functions in LandingPageView.java		All implemented functions in LandingPageController.java	
Test case	Target function	LandingPageView	getController	LandingPageController	getButtonHitRecords
test_constructorLandingPageView	LandingPageView.LandingPageView()	All covered	N/A	N/A	N/A
test_getController	LandingPageView.getController()	N/A	All covered	N/A	N/A
test_constructorLandingPageController	LandingPageController.LandingPageController()	N/A	N/A	35 - 38	N/A
test_hitButton	LandingPageController.LandingPageController()	N/A	N/A	36 - 37	N/A
test_getButtonHitRecords	LandingPageController.getButtonHitRecords()	N/A	N/A	N/A	All covered
testActionListenerWithInterruptedException	LandingPageController.LandingPageController()	N/A	N/A	38	N/A

f. MazeMapInterfaceTest.java

Total number of test cases in LandingPageTest.java: 13
Testing target class: MazeMapViewer.java, MazeMapController.java, WindowsView.java

Coverage report for MazeMapInterfaceTest.java

Coverage | MazeMapInterfaceTest

Element	Class, %	Method, %	Line, %
game_scene	70% (7/10)	71% (23/32)	80% (156/195)
LandingPageView	0% (0/1)	0% (0/2)	0% (0/17)
LandingPageController	0% (0/1)	0% (0/3)	0% (0/10)
ControlPanelController	50% (1/2)	25% (1/4)	35% (6/17)
WindowsView	100% (1/1)	100% (4/4)	100% (22/22)
VertexViewer	100% (1/1)	100% (2/2)	100% (3/3)
MazeMapViewer	100% (1/1)	100% (3/3)	100% (24/24)
ControlPanelView	100% (1/1)	50% (1/2)	98% (53/54)
VertexController	100% (1/1)	100% (6/6)	100% (21/21)
MazeMapController	100% (1/1)	100% (6/6)	100% (27/27)

Report for functions in MazeMapViewer.java and MazeMapController.java

		All implemented functions in MazeMapViewer.java			Part of the implemented functions in MazeMapController.java			
Test case	Target function	MazeMapViewer	getController	createAxisLabel	MazeMapController	getLocationVertexControllerMap	highlightPath	removeHighlightPath
testMazeMapViewer_constructor	MazeMapViewer.MazeMapViewer()	All covered	N/A	N/A	N/A	N/A	N/A	N/A
testMazeMapViewer_getController	MazeMapViewer.getController()	N/A	All covered	N/A	N/A	N/A	N/A	N/A
testMazeMapViewer_createAxisLabel	MazeMapViewer.creatAxisLabel()	N/A	N/A	All covered	N/A	N/A	N/A	N/A
testMazeMapController_constructor	MazeMapController.MazeMapController()	N/A	N/A	N/A	All covered	N/A	N/A	N/A
testMazeMapController_getLocationVertexControllerMap	MazeMapController.getLocationVertexControllerMap()	N/A	N/A	N/A	N/A	All covered	N/A	N/A
testMazeMapController_highlightPath	MazeMapController.highlightPath()	N/A	N/A	N/A	N/A	N/A	All covered	N/A
testMazeMapController_removeHighlightPath	MazeMapController.removeHighlightPath()	N/A	N/A	N/A	N/A	N/A	N/A	All covered

Report for functions in WindowsView.java and MazeMapController.java

		All implemented functions in WindowsView.java				Part of the implemented functions in MazeMapController.java	
Test case	Target function	WindowsView	setTextBillboard	getControlPanelView	getMapViewer	insertImage	renderMap
testWindowsView_constructor	WindowsView.WindowsView()	All covered	N/A	N/A	N/A	N/A	N/A
testWindowsView_setTextBillboard	WindowsView.setTextBillboard()	N/A	All covered	N/A	N/A	N/A	N/A
testWindowsView_getControlPanelView	WindowsView.getControlPanelView()	N/A	N/A	All covered	N/A	N/A	N/A
testWindowsView_getMapView	WindowsView.getMapViewer()	N/A	N/A	N/A	All covered	N/A	N/A
testMazeMapController_insertImage	MazeMapController.insertImage()	N/A	N/A	N/A	N/A	All covered	N/A
testMazeMapController_removeImage	MazeMapController.renderMap()	N/A	N/A	N/A	N/A	N/A	All covered

g. VertexTest.java

Total number of test cases in VertexTest.java: 7
Testing target class: VertexController.java, VertexViewer.java

Coverage report for VertexTest.java

Coverage | VertexTest

Element	Class, %	Method, %	Line, %
game_scene	70% (7/10)	53% (17/32)	69% (136/195)
LandingPageView	0% (0/1)	0% (0/2)	0% (0/17)
LandingPageController	0% (0/1)	0% (0/3)	0% (0/10)
ControlPanelController	50% (1/2)	25% (1/4)	35% (6/17)
WindowsView	100% (1/1)	50% (2/4)	81% (18/22)
VertexViewer	100% (1/1)	100% (2/2)	100% (3/3)
MazeMapViewer	100% (1/1)	100% (3/3)	100% (24/24)
ControlPanelView	100% (1/1)	50% (1/2)	98% (53/54)
VertexController	100% (1/1)	100% (6/6)	100% (21/21)
MazeMapController	100% (1/1)	33% (2/6)	40% (11/27)

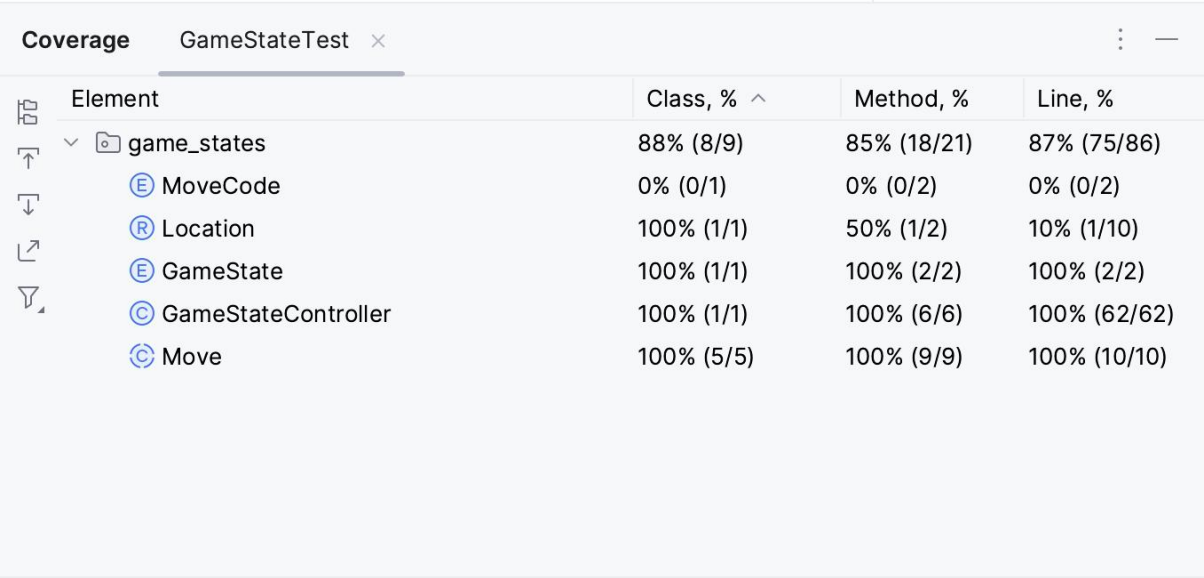
Report for functions in VertexController.java, VertexViewer.java

		All implemented functions in VertexController.java					All implemented functions in VertexViewer.java	
Test case	Target function	VertexController	changeVertexColor	getVertex	insertImage	removeImage	ChangeColor	VertexViewer
test_ConstructorVertexController	VertexController.VertexController()	All covered	N/A	N/A	N/A	N/A	N/A	N/A
test_changeVertexColor	VertexController.changeVertexColor()	N/A	All covered	N/A	N/A	N/A	N/A	N/A
test_getVertex	VertexController.getVertex()	N/A	N/A	All covered	N/A	N/A	N/A	N/A
test_insertImage	VertexController.insertImage()	N/A	N/A	N/A	All covered	N/A	N/A	N/A
test_removeImage	VertexController.removeImage()	N/A	N/A	N/A	N/A	All covered	N/A	N/A
test_vertexViewerchangeColor	VertexViewer.ChangeColor()	N/A	N/A	N/A	N/A	N/A	All covered	N/A
test_constructorVertexView	VertexViewer.VertexViewer()	N/A	N/A	N/A	N/A	N/A	N/A	All covered

h. GameStateTest.java

Total number of test cases in GameStateTest.java: 11
Testing target class: Move.java, GameStateController.java

Coverage report for GameStateTest.java



Report for functions in Move.java

		All implemented functions in Move.java				
Test case	Target function	Left.Left	Right.Right	Up.Up	Down.Down	nextPosition
test_constructorMoveLeft	Move.Left.Left()	All covered	N/A	N/A	N/A	N/A
test_constructorMoveRight	Move.Right.Right()	N/A	All covered	N/A	N/A	N/A
test_constructorMoveUp	Move.Up.Up()	N/A	N/A	All covered	N/A	N/A
test_constructorMoveDown	Move.Down.Down()	N/A	N/A	N/A	All covered	N/A
test_nextPosition	Move.nextPosition()	N/A	N/A	N/A	N/A	All covered

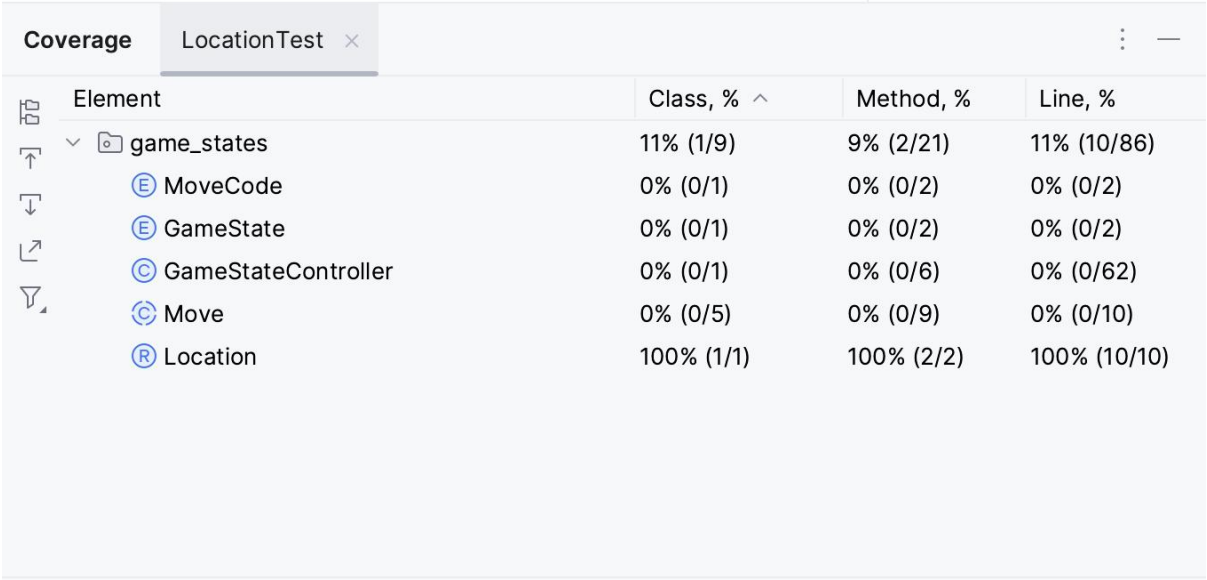
Report for functions in GameStateController.java

		All implemented functions in GameStateController.java					
Test case	Target function	GameStateController	canMove	moveCharacter	getCharacterLocation	gameStateOutcome	reachablePositions
test_constructorGameStateController	GameStateController.GameStateController()	All covered	N/A	N/A	N/A	N/A	N/A
test_canMove	GameStateController.canMove()	N/A	All covered	N/A	N/A	N/A	N/A
test_moveCharacter	GameStateController.moveCharacter()	N/A	N/A	All covered	N/A	N/A	N/A
test_getCharacterLocation	GameStateController.getCharacterLocation()	N/A	N/A	N/A	All covered	N/A	N/A
test_gameStateOutcome	GameStateController.gameStateOutcome()	N/A	N/A	N/A	N/A	All covered	N/A
test_reachableLocationByTom	GameStateController.reachablePositions()	N/A	N/A	N/A	N/A	N/A	All covered

i. LocationTest.java

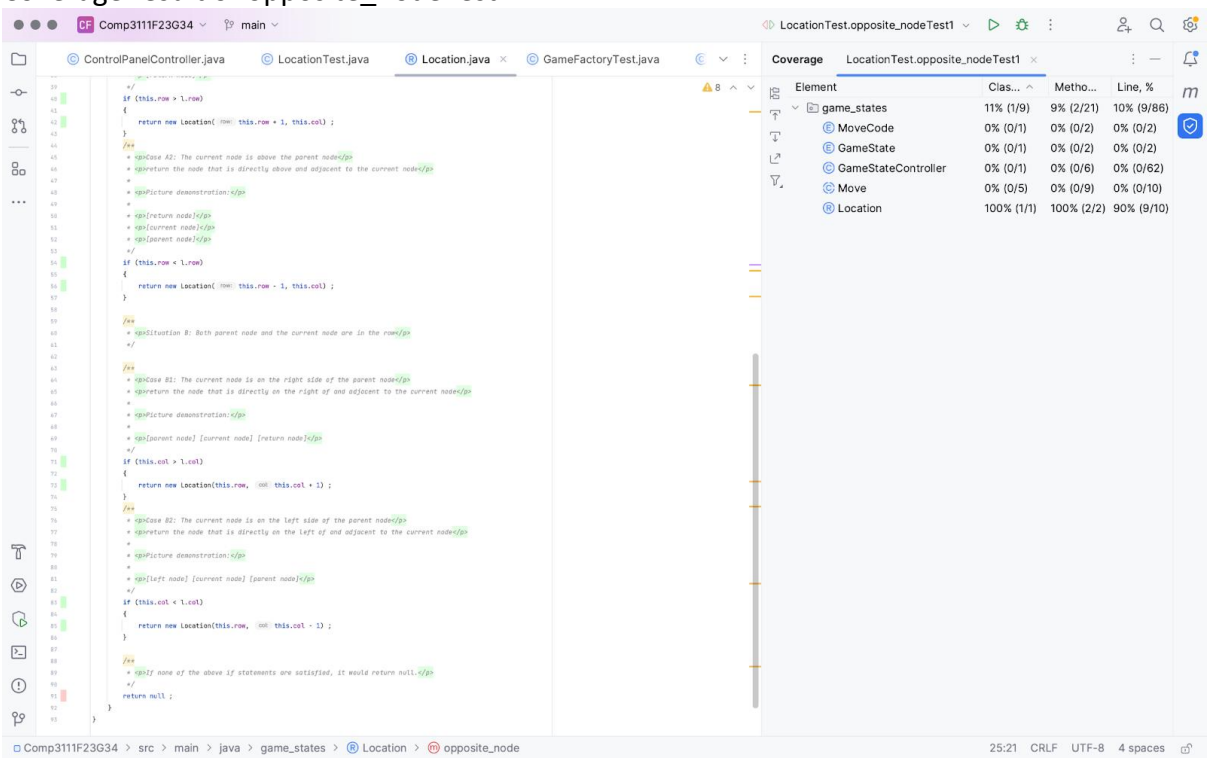
Total number of test cases in LocationTest.java: 3
Testing target class: Location.java

Coverage report for LocationTest.java

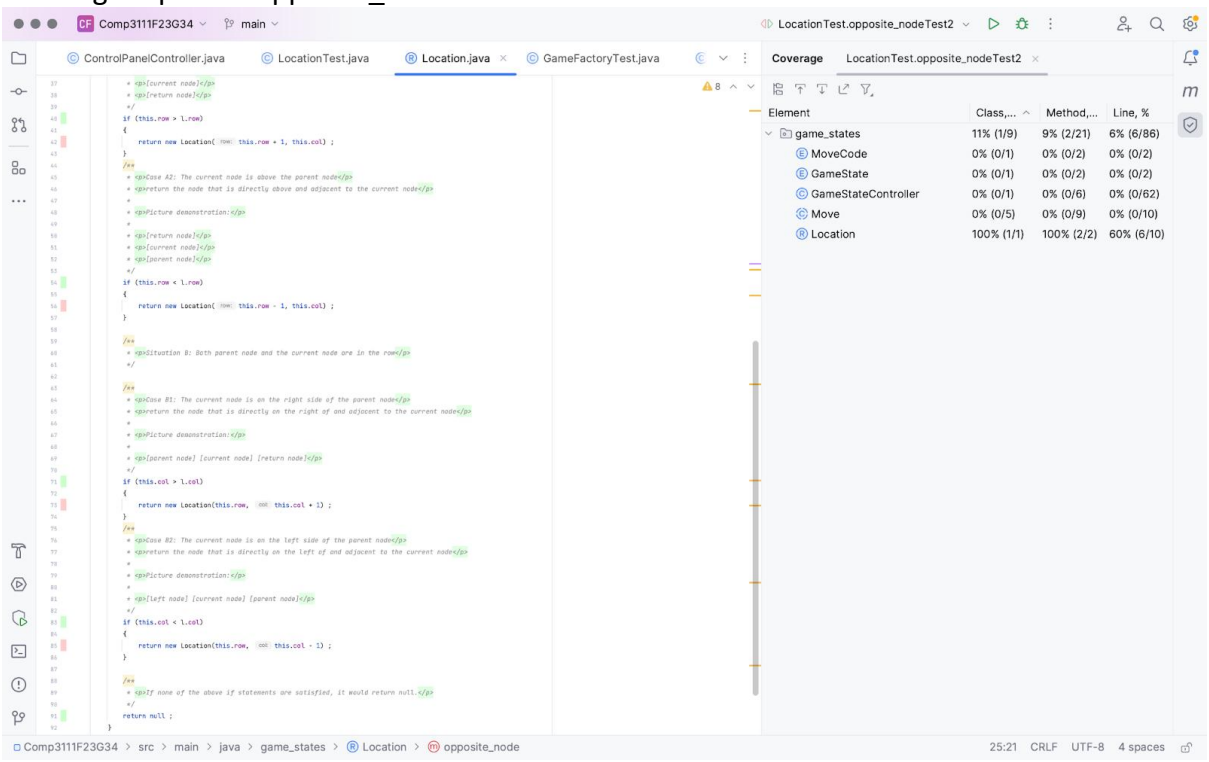


Element	Class, %	Method, %	Line, %
game_states	11% (1/9)	9% (2/21)	11% (10/86)
MoveCode	0% (0/1)	0% (0/2)	0% (0/2)
GameState	0% (0/1)	0% (0/2)	0% (0/2)
GameStateController	0% (0/1)	0% (0/6)	0% (0/62)
Move	0% (0/5)	0% (0/9)	0% (0/10)
Location	100% (1/1)	100% (2/2)	100% (10/10)

Coverage result of opposite_nodeTest1



Coverage report of opposite_nodeTest2



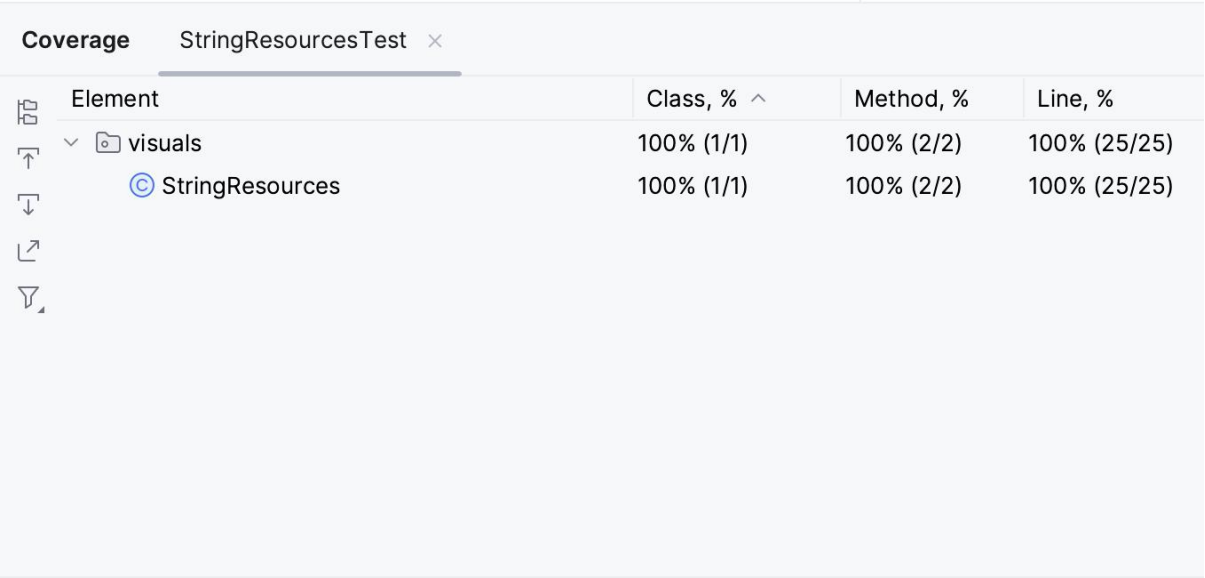
Report for functions in Location.java

		All implemented functions in Location.java	
Test case	Target function	Location	opposite_node
testEqual	Location.Location()	All covered	N/A
opposite_nodeTest1	Location.opposite_node()	N/A	91
opposite_nodeTest2	Location.opposite_node()	N/A	42, 56, 73, 85

j. StringResourcesTest.java

Total number of test cases in StringResourcesTest.java: 1
Testing target class: StringResources.java

Coverage report for StringResourcesTest.java



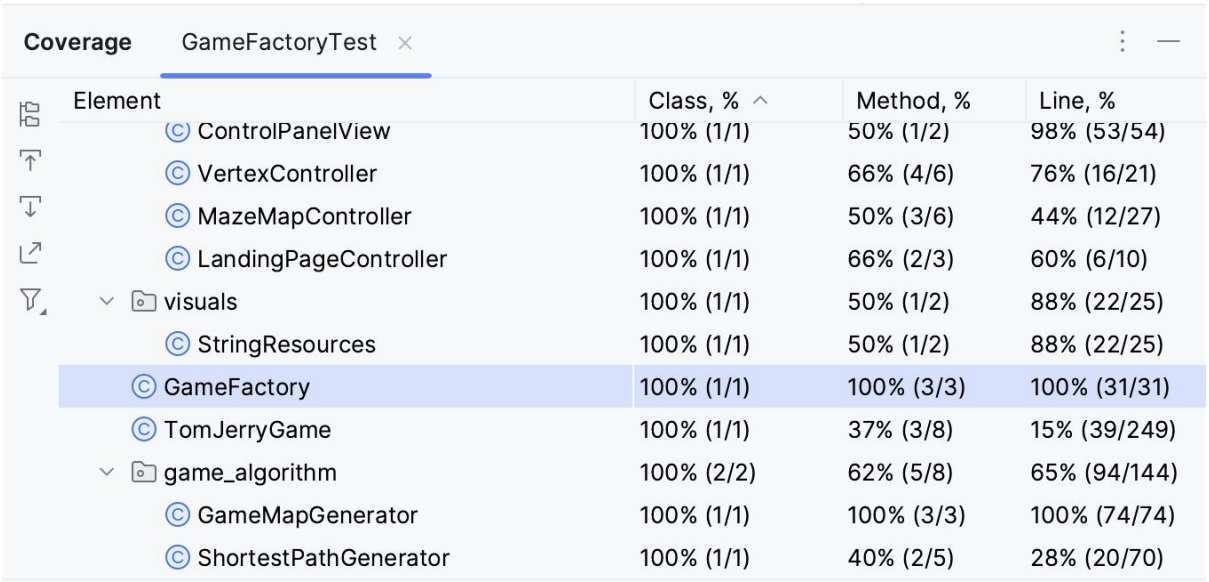
Report for functions in StringResources.java

		All implemented functions in StringResources.java
Test case	Target function	showRemainingMoves
testShowRemainingMoves	StringResources.showRemainingMoves	All covered

k. GameFactoryTest.java

Total number of test cases in GameFactoryTest.java: 2
Testing target class: GameFactory.java

Coverage report for GameFactoryTest.java



Element	Class, %	Method, %	Line, %
ControlPanelView	100% (1/1)	50% (1/2)	98% (53/54)
VertexController	100% (1/1)	66% (4/6)	76% (16/21)
MazeMapController	100% (1/1)	50% (3/6)	44% (12/27)
LandingPageController	100% (1/1)	66% (2/3)	60% (6/10)
visuals	100% (1/1)	50% (1/2)	88% (22/25)
StringResources	100% (1/1)	50% (1/2)	88% (22/25)
GameFactory	100% (1/1)	100% (3/3)	100% (31/31)
TomJerryGame	100% (1/1)	37% (3/8)	15% (39/249)
game_algorithm	100% (2/2)	62% (5/8)	65% (94/144)
GameMapGenerator	100% (1/1)	100% (3/3)	100% (74/74)
ShortestPathGenerator	100% (1/1)	40% (2/5)	28% (20/70)

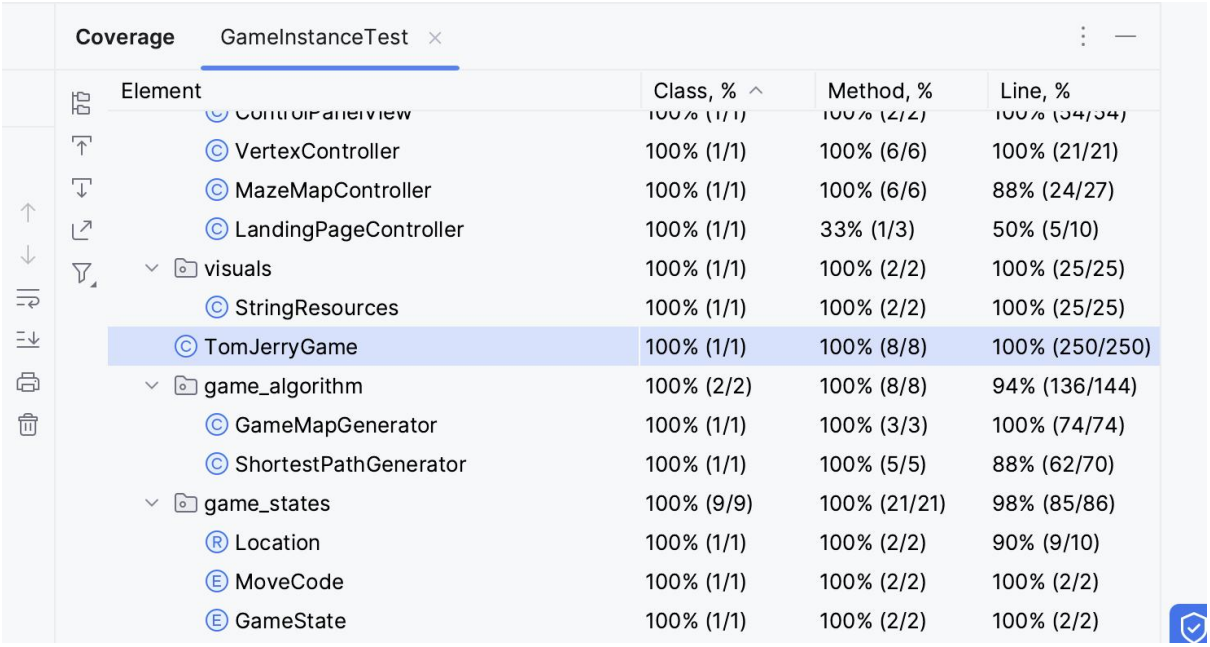
Report for functions in GameFactory.java

		All implemented functions in GameFactory.java	
Test case	Target function	createGame	main
test_creategame	GameFactory.createGame()	All covered	N/A
test_main	GameFactory.main()	N/A	All covered

I. GameInstanceTest.java

Total number of test cases in GameInstanceTest.java: 8
Testing target class: TomJerryGame.java

Coverage report for GameInstanceTest.java



Element	Class, %	Method, %	Line, %
VertexController	100% (1/1)	100% (6/6)	100% (21/21)
MazeMapController	100% (1/1)	100% (6/6)	88% (24/27)
LandingPageController	100% (1/1)	33% (1/3)	50% (5/10)
visuals	100% (1/1)	100% (2/2)	100% (25/25)
StringResources	100% (1/1)	100% (2/2)	100% (25/25)
TomJerryGame	100% (1/1)	100% (8/8)	100% (250/250)
game_algorithm	100% (2/2)	100% (8/8)	94% (136/144)
GameMapGenerator	100% (1/1)	100% (3/3)	100% (74/74)
ShortestPathGenerator	100% (1/1)	100% (5/5)	88% (62/70)
game_states	100% (9/9)	100% (21/21)	98% (85/86)
Location	100% (1/1)	100% (2/2)	90% (9/10)
MoveCode	100% (1/1)	100% (2/2)	100% (2/2)
GameState	100% (1/1)	100% (2/2)	100% (2/2)

Report for functions in TomJerryGame.java

		All implemented functions in TomJerryGame.java							
Test case	Target function	TomJerryGame	readData	setDifficulty	TomMovesOneStep	showHintsOnMap	TomMoves	JerryMoves	run
test_GameInstanceConstructor	TomJerryGame.TomJerryGame()	All covered	N/A	N/A	N/A	N/A	N/A	N/A	N/A
test_readData_invalidmap	TomJerryGame.readData()	N/A	All covered	N/A	N/A	N/A	N/A	N/A	N/A
test_selectDifficulty	TomJerryGame.setDifficulty()	N/A	N/A	All covered	N/A	N/A	N/A	N/A	N/A
test_TomMovesOneStep	TomJerryGame.TomMovesOneStep()	N/A	N/A	N/A	All covered	N/A	N/A	N/A	N/A
test_showHints	TomJerryGame.showHintsOnMap()	N/A	N/A	N/A	N/A	All covered	N/A	N/A	N/A
test_TomMoves	TomJerryGame.TomMoves()	N/A	N/A	N/A	N/A	N/A	All covered	N/A	N/A
test_JerryMoves	TomJerryGame.JerryMoves()	N/A	N/A	N/A	N/A	N/A	N/A	All covered	N/A
test_run	TomJerryGame.run()	N/A	N/A	N/A	N/A	N/A	N/A	N/A	All covered

3) Describe which lines are never covered by any test case, the total number of these uncovered lines, as well as the Test-per-function ratio (shows how it is computed)

All lines are covered by all test cases.

Here, we lists out the number of functions in each class:

game_algorithm folder

Number of functions in GameMapGenerator.java: 3
Number of functions in ShortestPathGenerator.java: 5

game_entities folder

Number of functions in characterID.java: 0
Number of functions in Vertex.java: 2

game_scene folder

Number of functions in ControlPanelController.java: 2
Number of functions in ControlPanelView.java: 2
Number of functions in LandingPageController.java: 2
Number of functions in LandingPageView.java: 2
Number of functions in MazeMapController.java: 6
Number of functions in MazeMapViewer.java: 3
Number of functions in VertexController: 5
Number of functions in VertexViewer.java: 2
Number of functions in WindowsView.java: 4

game_states folder

Number of functions in GameState.java: 0
Number of functions in GameStateController.java: 6
Number of functions in Location.java: 2
Number of functions in Move.java: 5
Number of functions in MoveCode.java: 0

visuals folder

Number of functions in StringResources.java: 1

Number of functions in GameFactory.java: 2
Number of functions in TomJerryGame.java: 8

Total number of functions = 62

Here, we list out the number of test cases:

game_algorithm folder

Number of test cases in GameMapGeneratorTest.java: 3
Number of test cases in ShortestPathGeneratorTest.java: 5

game_entities folder

Number of test cases in VertexEntityTest.java: 2

game_scene folder

Number of test cases in ControlPanelTest.java: 6
Number of test cases in LandingPageTest.java: 6
Number of test cases in MazeMapInterfaceTest.java: 13
Number of test cases in VertexTest: 7

game_states folder

Number of test cases in GameStateTest.java: 11
Number of test cases in LocationTest.java: 3

visuals folder

Number of test cases in StringResourcesTest.java: 1

Number of test cases in GameFactoryTest.java: 2
Number of test cases in GameInstanceTest.java: 8

Total number of test cases = 67

Test-per-function ratio:

$$= \frac{number\ of\ test\ cases}{max(160 - max(number\ of\ functions\ in\ the\ code, 80)\ , 1)}$$

$$= \frac{67}{max\ (160 - max(62,80)\ ,1)}$$

$$= \frac{67}{max\ (160 - 80,1)}$$

$$= \frac{67}{max\ (80,1)}$$

$$= \frac{67}{80}$$

$$= 0.8375$$