Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

The testing objective test these objects.

- The objective here is to test the entering implementation of the last iteration.
- There will be test two use_case by running dynamic manual two test cases for each. There will be writing an automated unit test that is testing two methods used in the rk222jy_asg2 class in the class rk222jy_asg2Test va

Time Plan

task	Estimated Time	Actual Time
Manual TC	2 hours & 30 min	3 hours
Unit Tests	4 hours	6 hours
Running manual tests	30 min	1 hour
Code inspection	2 hour	2 hours& 30 min
Test Report	40 min	1 hour& 20 min

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Use-Cases

This is an app game called “The HangMan” with two use-cases.

UC1 check player input while player success.

Precondition: the player chose from the menu to starts the game.
Postcondition: the player entered a letter.

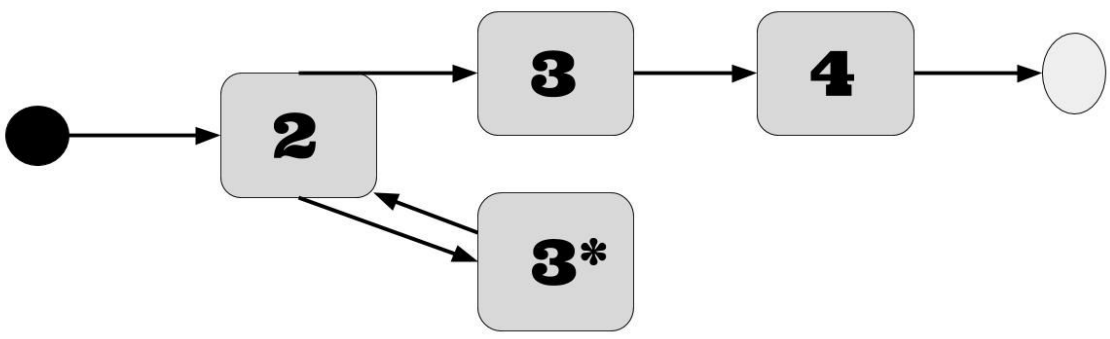
Main scenario

1. Starts the game when the player enters the first guessing (a letter)
2. The system shows a number of stars that describe the number of guessing word letters.
3. The player enters a letter.
4. The system shows the entered letter by storing it, then continues to the next new guessing.

Alternate scenarios

- 3* The player enters a number (EX”4”, or any singe but letter).
- The system shows an error message.
- Enter new valid Input

Activity Diagram

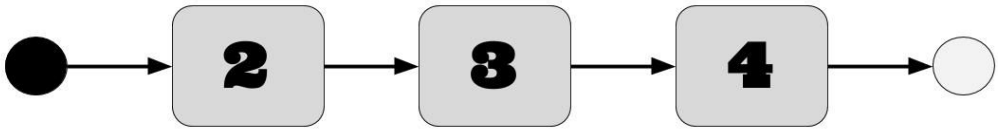


Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Manual Test-Cases

TC1.1 player enters valid input.

Use case: UC1: check the player input while playing.
Scenario: check the player input success.



The main scenario of UC1 is tested when the player makes a valid input.
Precondition: run the game.

Test steps

- Start the game
- The system shows: “GUESS THE WORD BEHIND THE STARS".
- The player enters a letter (EX: x) to guess the word and press enter.

Expected

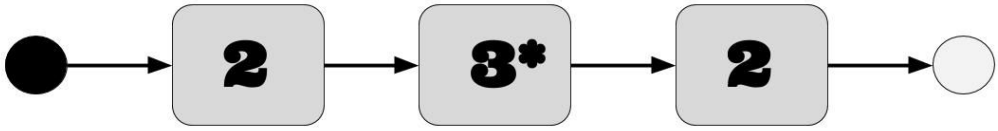
The system should show the letter result of the entered letter “x” if it included in the guess word or not.

The system continues running and asking the player continues entering letters to find the guess word.

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

TC1.2 player enters wrong input.

Use case: UC1: check the player input while playing.
Scenario: wrong input force to input again.
When the player enters a number (wrong input"7") the alternate scenario force the player to enter a letter instead.



Precondition: The game is running.

Test steps

- Start the game
- The system shows: "GUESS THE WORD BEHIND THE STARS".
- The player enters "7" and press enter.

Expected

The system should show " wrong input"(error).

The system shows " you can not input a number the input must be a letter, Please Input again" and wait for the player to enter a new input.

UC2 check player input.

Precondition: chose from the menu to start the game.
Postcondition: the player decides to enter a name or play as a gest.

Main scenario

1. Start when the plyer runt he code

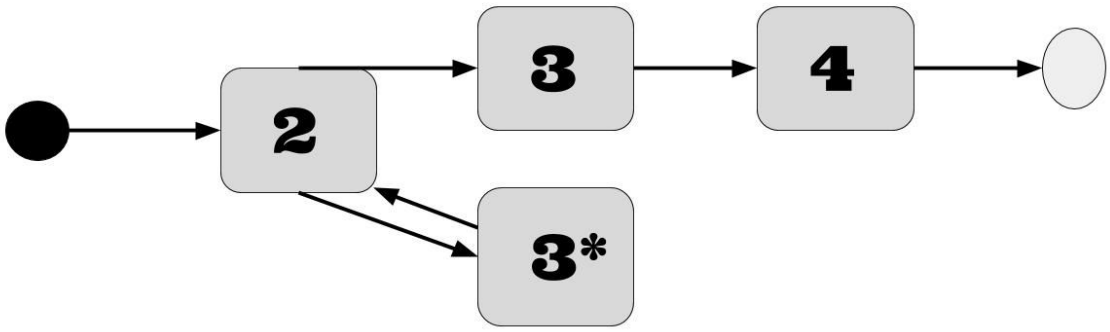
Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

2. System ask to choose from the menu
3. The player chose from the menu to play.
4. The system starts the game.

Alternate scenario

- 3* The player enters invalid input (not exist in the menu EX: “3”).
- The system shows an error message.
- The system goto "UC1” asks chose from the menu (EX: 1 or 2).

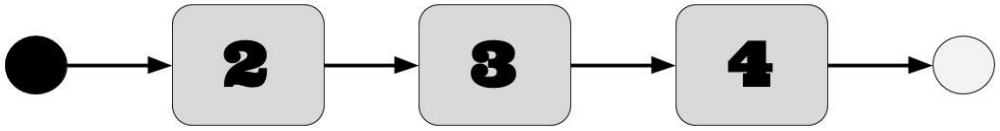
Activity Diagram



Manual Test-Cases

TC1.2 The player makes a valid input from the menu

Use case: UC2 check player input.
Scenario: check player input successful.



The main scenario of uc2 is checked when the player makes a valid input from the menu.

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Precondition: the system shows a welcome message and starts the game.

Test steps

- Start the game
- The system shows the menu.
- 1_Enter your name
- 2_Play as a guest.
- Enter a valid chose from the menu (EX: "2") to start the game.

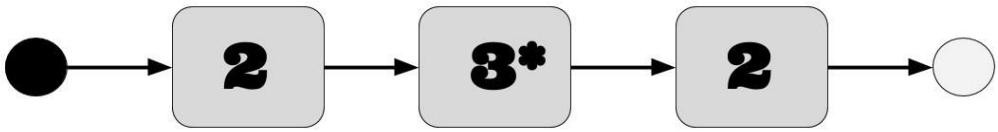
Expected

- The system starts the game.

TC2.2 player makes an invalid input from menu to start the game.

---

1. Use-case: UC2 check player input to start the game.
2. Scenario: invalid input force to input again.
3. The alternate scenario when the player enters an invalid chose from the menu and is forced to enter a valid a chose again.



Precondition: the game starts.

Test steps

- Start the game
- System shows the menu.
- 1_Enter your name
- 2_Play as a guest.
- Enter a number "3" and press enter.

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Expected

- The system should show “Invalid input”
- System shows "wrong input you must choose from the menu, please enter again" and waits for the new input.

Unit Test source code 1

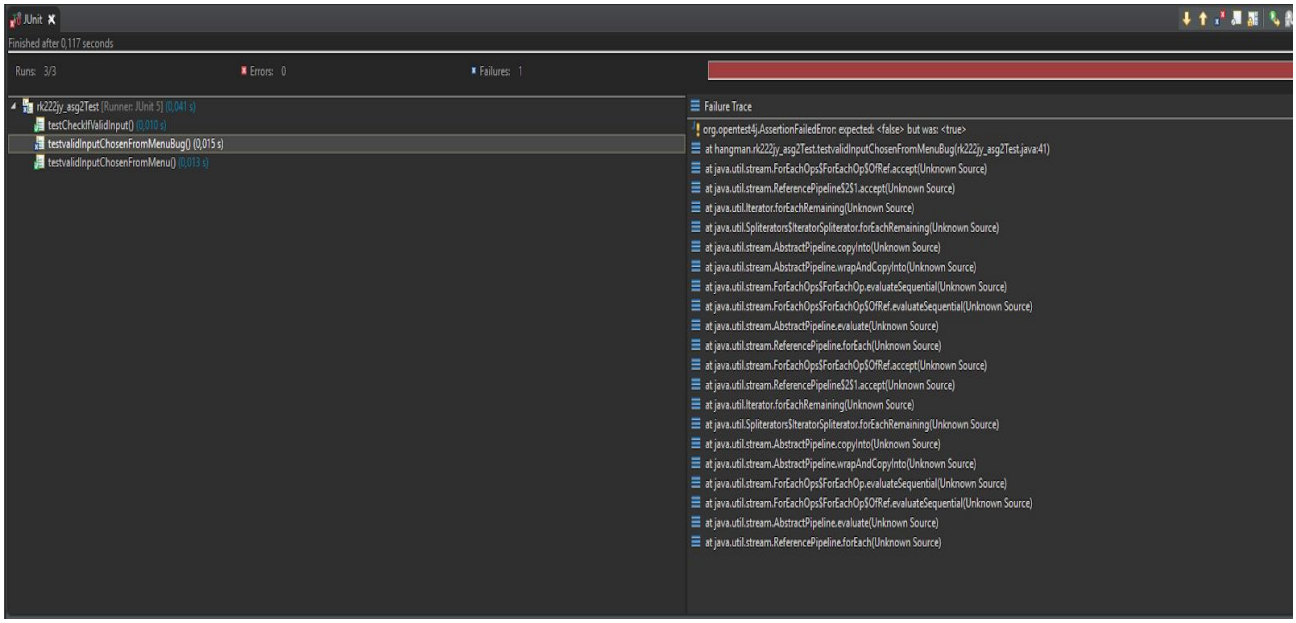
```
protected static boolean checkIfValidInput(String input){
    Character check = input.charAt(0);
    if(Character.isLetter(check) && input.length() == 1)
        return true;
    else
        return false;

}}
```

Unit Test 1

```
private String validInput = "f";
private String inValidInput = "that's long invalid input ";
private String inValidIntegerInput = "6";

private String validInputToChooseFromMenu = "1";
private String validInputToChooseFromMenu2 = "2";

rk222jy_asg2 hangman = new rk222jy_asg2();

@Test
void testCheckIfValidInput() {
    assertEquals(true, hangman.checkIfValidInput(validInput));
    assertEquals(false, hangman.checkIfValidInput(inValidIntegerInput));
    assertEquals(false, hangman.checkIfValidInput(inValidInput));

}
```

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

The result of Test 1



Unit Test source code 2

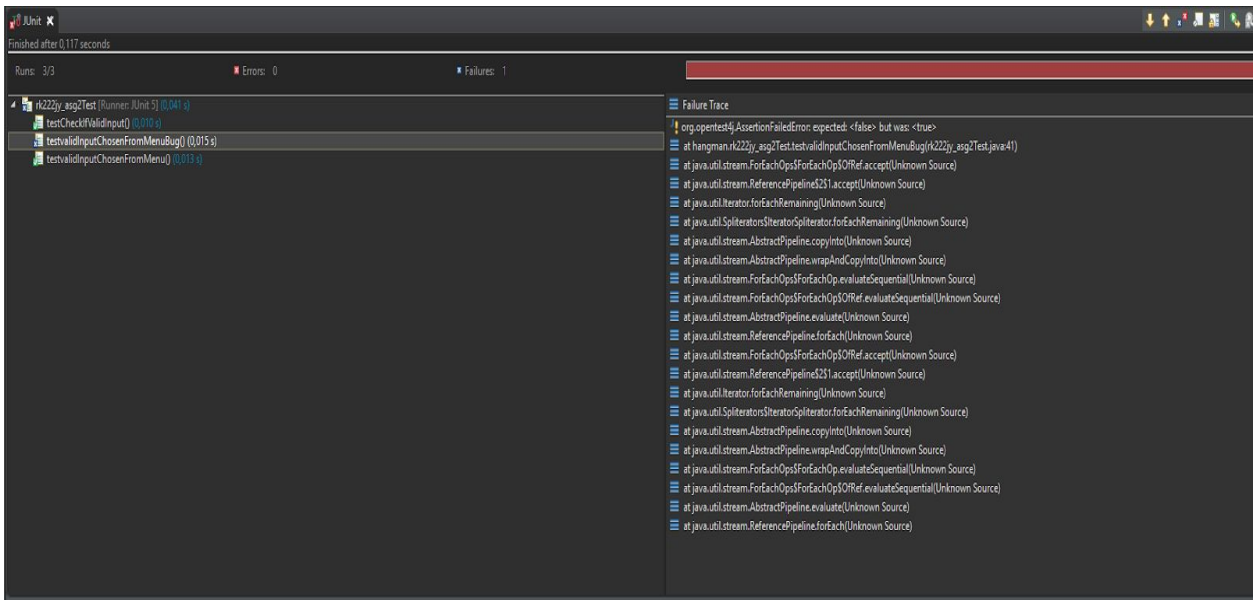
```
protected static boolean validInputChosenFromMenu(String input) {
    if(input.equals("1")) {
        isEnterName  = true;
        return true;
    }
    else if(input.equals("2")) {
        return true;
    }
    else if(input.equals("3"))
    {
        isExit = true;
        return true;
    }
    return false;
}
```

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Unit test 2

```
private String validInput = "f";
private String inValidInput = "that's long invalid input ";
private String inValidIntegerInput = "6";

private String validInputToChooseFromMenu = "1";
private String validInputToChooseFromMenu2 = "2";

rk222jy_asg2 hangman = new rk222jy_asg2();

@Test
void testvalidInputChosenFromMenu() {
    assertEquals(true, hangman.validInputChosenFromMenu(validInputToChooseFromMenu));
    assertEquals(true, hangman.validInputChosenFromMenu(validInputToChooseFromMenu2));
    assertEquals(false, hangman.validInputChosenFromMenu(inValidInput));
    assertEquals(false, hangman.validInputChosenFromMenu(inValidIntegerInput));
}
```

The result of Test 2



Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Unit Test Source code 3

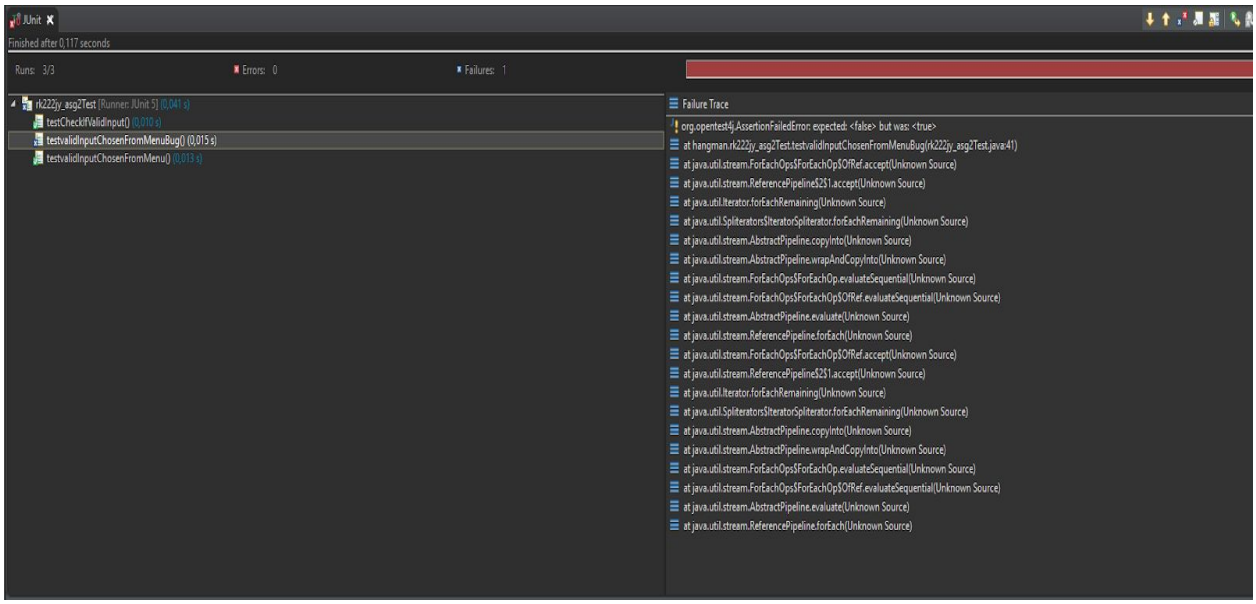
```
protected static boolean validInputChosenFromMenu(String input) {
    if(input.equals("1")) {
        isEnterName  = true;
        return true;
    }
    else if(input.equals("2")) {
        return true;
    }
    else if(input.equals("3"))
    {
        isExit = true;
        return true;
    }
    return false;
}
```

Unit Test 3

```
private String validInput = "f";
private String inValidInput = "that's long invalid input ";
private String inValidIntegerInput = "6";

private String validInputToChooseFromMenu = "1";
private String validInputToChooseFromMenu2 = "2";

rk222jy_asg2 hangman = new rk222jy_asg2();

// when we press number 3 the game is finished and the return is true for valid input
// but it should return false because it's a bug.
@Test
void testvalidInputChosenFromMenuBug() {
    assertEquals(true, hangman.validInputChosenFromMenu(validInputToChooseFromMenu));
    assertEquals(true, hangman.validInputChosenFromMenu(validInputToChooseFromMenu2));
    assertEquals(false, hangman.validInputChosenFromMenu("3"));
}
```

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

The result of Unit Test 3



Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Test Report

Test traceability matrix and success

Test	UC1	UC2
TC1.1	1/OK	0
TC1.2	1/OK	0

Test	UC1	UC2
TC2.1	0	2/OK
TC2.2	0	2/OK
COVERAGE & SUCCESS	2/OK	2/OK

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master

Automated unit test coverage and success

Test	Console View	GameEntity Controller	Main	rk222jy_asg2
testvalidInputChosenFromMenu	0	0	0	100%/Ok
testvalidInputChosenFromMenuBug	0	0	0	33%/OK
testCheckIfValidInput	0	0	0	100%/Ok
Coverage and Successo	0/NA	0/NA	0/NA	100%/OK

Reflection

Code testing: for the hangman game give the programmer more confidence in his program. This Testing task is an independent code performed after finishing the greeted program code. helps to test the code, find the bugs learn how to manage the time to have a clear program In addition to the written report which develops the skills learning applied and used.
in this task, the programmer works more deep within the results of coding (the input results after running the code).
make a reflective discussion on the program code that reflects on how the individual skills coding have developed which affected the implementation and outcome.

Name : Reham Khaezran.
Lnu-email: rk222jy@student.lnu.se .
Githublink: https://github.com/Rihamoo/Rk222jy_1dv600/tree/master