

**CS4004 Software Testing and  
Inspection Autumn 2020 Project  
Specification**

By  
Richard Alexander – 19270151  
Micheál Hogan – 15172856  
Fiston Kulimushi – 19258879  
Benjamin Setterfield - 19247125

## Table of Contents

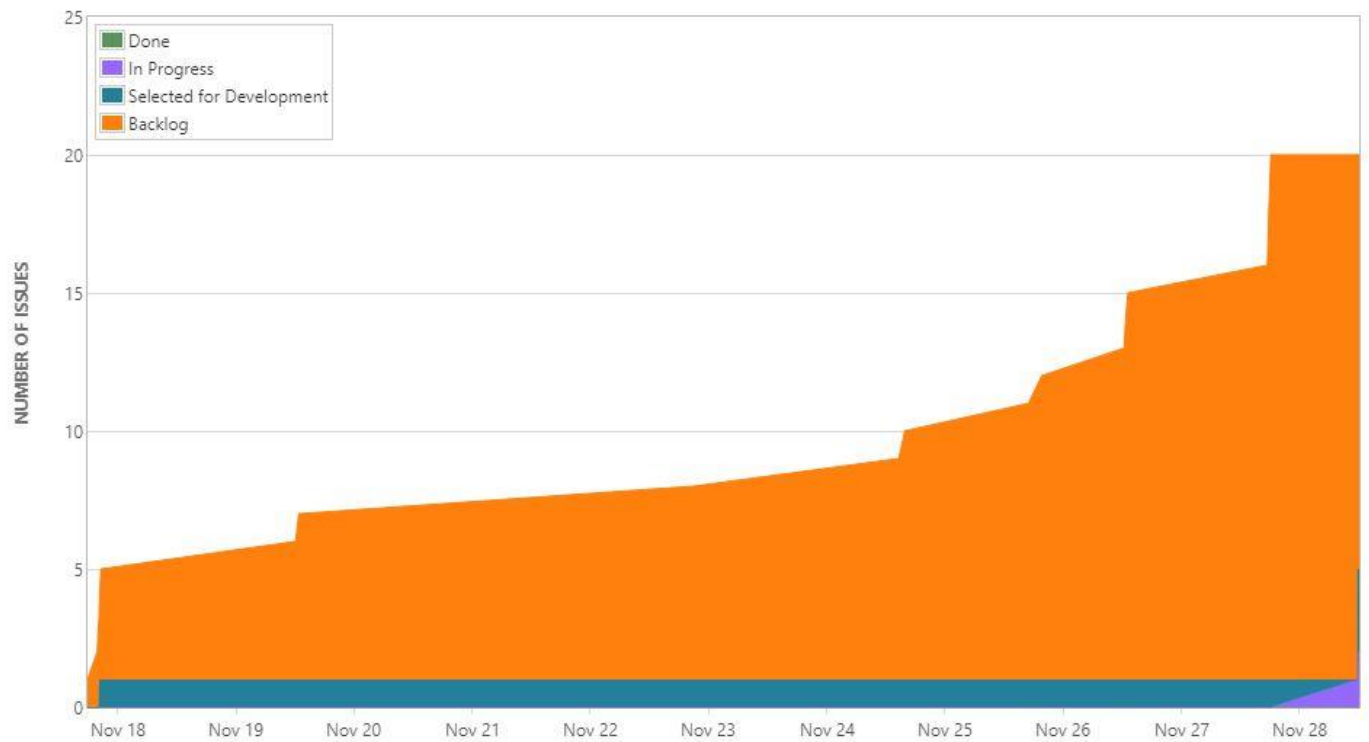
Task #1 Bug Reporting .....	2
A. The log: .....	2
B. Reflections: .....	3
C. Bug reports.....	4
Task #2: Test Design and Development using JUnit .....	26
1. Test-Driven-Design Approach .....	26
2. Coverage Criteria Based Testing .....	48
A. The log: .....	48
B. Reflections: .....	49
C. Coverage Criteria code.....	50

## Task #1 Bug Reporting

### A. The log:

Report was sourced from our Jira project, we continuously made bugs as we went and managed to finish our bug reporting within time. We began on the 18<sup>th</sup> of November and finished at the 28<sup>th</sup> of November.

Cumulative flow diagram:



## B. Reflections:

### Reflection Benjamin Setterfield:

The Bug reports were particularly difficult to write as I had no code or frame of reference to begin with but as I settled into thinking as an employee of such a place and thinking of possible bugs was my best option.

### Reflection Fiston Kulimushi:

The project was about finding bug in a library website and I had to think and come up with the kind of bugs that may be found on it. I used the knowledge that I gained during the lectures slides to write the bug report. What I learned is the theory part of how to write an effective report showing each step for finding a defect in a system. Once finished I felt confident writing bug report.

### Reflection Micheál Hogan:

What did we find difficult?

What I found difficult in task 1 of the project was creating realistic hypothetical bugs that could arise in a library system based on the requirements given in the case study.

What I also found was that I had to always put myself in the shoes of the user and accurately and concisely explain the steps taken to reproduce a bug in the system.

How did we go about doing some things?

We decided as a group to have an equal number of bugs to report on from varying levels of priority in the system. Ranging from the lowest priority to the highest priority.

### Reflection Richard Alexander:

When beginning to write my bug reports it was strange to be bug reporting code that did not exist. However, when I got the first one done the rest came to me easily. I just thought about what might go wrong in the UL library with things such as the card swipe system. Overall, I think this task wasn't too bad as long as you space out your bug reporting, so you don't have to do it all in one go. This task also helped gain experience to learn how Jira works.

## C. Bug reports

CS4004-1

Give feedback 1

### Searching computer science journals do not display correct search result

Attach Create subtask Link issue

#### Description

Testing the library's website search functionality, After we log in and select book genre category and sub-category and click on Search, we are not getting the expected output.

Steps to reproduce the bug:

1. Go to the website
2. Log in as a faculty/staff member
3. Select book genre category = computer science
4. Select Subcategory = journals
5. Click on the Search button

Expected Result is: It should display 788 results

Actual: We are getting 2000 results

#### Environment

Browser: Google Chrome.

Operating system: Windows 10 professional 64 bit.

Hardware specification: PC, CPU: i56400, GPU: msi 1060, Memory: 16gb ram.

#### Activity

Show: Comments History Work log



Add a comment...

Pro tip: press **M** to comment

Backlog

Assignee **FK** fiston kulimushi

Reporter **MH** micheal hogan

Labels None

Priority **High**

Show 6 more fields

Original Estimate, Time tracking, Epic Link, Components, Fix vers...

Created November 17, 2020, 5:52 PM

Configure

Updated 15 hours ago



**1. Summary Required:**

The Science subsection Physics is incorrectly alphabetized

**Description:**

The subsection "Physics" under the science tab has incorrect order and does not follow the standard alphabetization.

To reproduce:

- 1: Log in
- 2: Open the Science tab and navigate to Physics
- 3: Scroll to the end of the entries that begin with numbers

Expected: Following the entries that begin with numbers, the standard alphabetization should occur

Actual: The entries that follow the numerical are random, based on the order on entry instead of the alphabetization that is implemented elsewhere

**Reporter:**

Benjamin Setterfield

**Priority**

Low

**Environment:**

Browser: Google Chrome.

Operating system: Windows XP.

Hardware specification: PC, CPU: i3-6600 GPU: Intel onboard graphics, Memory: 4gb ram.

**Assignee:**

Fiston Kulimushi

**2. Summary Required:**

There's Incompatible data between departments into the library's system.

**Description:**

There is a set of useful data present in two departments that are not accounted for in the new database. They need to be reviewed and added to the new database where it is applicable.

To reproduce:

- 1: Log in as a Staff member of the art and history departments
- 2: Open the old department system alongside the new
- 3: Compare the two systems.

Expected: A comprehensive transfer of all data to the new system

Actual: The data set is omitted as the new system cannot support it.

**Reporter:**

Benjamin Setterfield

**Priority**

Medium

**Environment:**

Browser: FireFox

Operating system: Windows XP.

Hardware specification: PC, CPU: i5-10400 GPU: Intel onboard graphics, Memory: 4gb ram.

**Assignee:**

Richard Alexander



### **3. Summary Required:**

There's typo's in staff data entry menu in the system.

#### **Description:**

There are some typos on the data entry menu for staff. Minor ones being the number following the data field names.

To recreate:

1. Log into the new system as a staff member
2. Click "New Entry"
3. Enter the first page of data and navigate to the second
4. The fifth data field name on the second entry page is "Year of Release7"
5. Similar typos are present in the 7th and 10th data field

Expected: That The data field names are accurate and do not have typo's present.

Actual: Minor typo's that are not visible to the public and do not currently affect the new system

#### **Reporter:**

Benjamin Setterfield

#### **Priority**

Lowest

#### **Environment:**

Browser: Google Chrome.

Operating system: Windows XP.

Hardware specification: PC, CPU: i5-9400F GPU: rtx 950, Memory: 8gb ram.

**Assignee:** Micheál Hogan

**4. Summary Required:**

The mobile version of the website is Non-responsive.

**Description:**

The mobile version of the new system is non-responsive with frequent errors and delays. The functionality of the mobile system is minimal and poorly adapted.

To recreate:

1. Enter the website name into the search engine
2. Enter the website and attempt to sign in as a student/user.

Expected: An adapted version of the website that is compatible with mobile devices and allows ease of use.

Actual: Non-responsive and error prone version of the website that is poorly adjusted for mobile devices

**Reporter:**

Benjamin Setterfield

**Priority**

Highest

**Environment:**

Browser: Microsoft edge

Operating system: Windows 10.

Hardware specification: PC, CPU: i7 47090k GPU: rtx 950, Memory: 12gb ram.

**Assignee:** Richard Alexander

5.

**Summary Required:**

There is a minor issue with copy pasted text on the website.

**Description:**

Some copy pasted text can appear much larger in the data entry system for staff if copy pasted but will enter the system with the correct size.

To recreate:

1. Log in as a staff member
2. Click "New Entry"
3. Prepare a document with text more than font size 18
4. Copy paste the large text into the New Entry menu

Expected: That the data entered will be of uniform size

Actual: The data appears large to the staff member entering it with copy paste but is converted when the data entry is completed.

**Reporter:**

Benjamin Setterfield

**Priority**

Low

**Environment:**

Browser: Internet Explorer

Operating system: Windows 7.

Hardware specification: PC, CPU: AMD 2600x GPU: rtx 950, Memory: 8gb ram.

**Assignee:** Micheál Hogan

6.

**Summary Required:**

Searching for computer science journals does not display correct search results.

**Description:**

Testing the library's website search functionality, after we log in and select book genre category and sub-category and click on Search, we are not getting the expected output.

Steps to reproduce the bug:

1. Go to the website
2. Log in as a faculty/staff member
3. Select book genre category = computer science
4. Select Subcategory = journals
5. Click the Search button

Expected Result is: It should display 788 results

Actual: We are getting 2000 results

**Reporter:**

Micheal Hogan

**Priority**

High

**Environment:**

Browser: Google Chrome.

Operating system: Windows 10 professional 64 bit.

Hardware specification: PC, CPU: i56400, GPU: msi 1060, Memory: 16gb ram.

**Assignee:**

Fiston Kulimushi

**7.**

**Summary Required:**

The search result for the location of medical journals in the library is incorrect on the website.

**Description:**

Testing the library's website search functionality, after we log in and select book genre category and sub-category and click on Search, we are not getting the expected output.

Steps to reproduce the bug:

1. Go to the website
2. Log in as a regular library member
3. Select book genre category = medical
4. Select Subcategory = journals
5. Click the Search button

Expected Result is: It should display the location of the bookshelf as c1456

Actual: The result being displayed shows the location as bookshelf fg4523

**Reporter:**

Micheal Hogan

**Priority**

Medium

**Environment:**

Browser: Chromium

Operating system: Linux distro: Fedora

Hardware specifications: PC, CPU: i56400, GPU: msi 1070, Memory: 8gb ram.

**Assignee:**

Richard Alexander

**8.**

**Summary Required:**

The Logo is not displaying on the home page.

**Description:**

Testing the graphical components of the home page. The main central logo of the library's website is no longer being displayed.

Steps to reproduce the bug:

1. Go to the website
2. View from the homepage

Expected Result is: All elements of the home page should be displayed

Actual: An empty outline of the logo is being displayed on the home page.

**Reporter:**

Micheal Hogan

**Priority**

Medium

**Environment:**

Browser: Firefox.

Operating system: Windows 10 Home 64 bit.

Hardware specification: Laptop, CPU: i36300, Memory: 4gb ram.

**Assignee:**

Micheal Hogan

9.

**Summary Required:**

Transition animation for the library's section button is not functioning.

**Description:**

Testing the website's homepage functionality, After hovering over the about section button on the main home page. There is no transition animation when the mouse cursor is on the button.

Steps to reproduce the bug:

1. Go to the website
2. Place the mouse cursor onto the top of the section button.

Expected Result is: A transition animation should occur where the outline of the button should change to a darker shade of orange.

Actual: No response, the button remains unchanged.

**Reporter:**

Micheal Hogan

**Priority**

Lowest

**Environment:**

Browser: Safari

Operating system: macOS 10.14: Mojave- 2018.

Hardware specification: Mac book pro-2016, CPU: i7, GPU: Radeon pro-450, Memory: 16GB DDR3

**Assignee:**

Benjamin Setterfield

**10.**

**Summary Required:**

Login button for user has no functionality after the button has been clicked.

**Description:**

Testing the website's homepage functionality, after clicking the login button on the main home page. There was no response from the website.

Steps to reproduce the bug:

1. Go to the website
2. Click on the login button

Expected Result is: A window should pop up on the main interface requesting the user's username and password

Actual: No response from clicking the button. The user cannot log in.

**Reporter:**

Micheal Hogan

**Priority**

Highest

**Environment:**

Browser: Safari

Operating system: macOS 10.14: Mojave- 2018.

Hardware specification: Mac book pro-2016, CPU: i7, GPU: Radeon pro-450, Memory: 16GB DDR3

**Assignee:**

Benjamin Setterfield



**11.**

**Summary Required:**

Searching for a specific journal gives a duplicate of journals from a different department.

**Description:**

Tuesday evening, I was testing the library website, getting journals for a specific department but I got duplicate journals of different departments.

Steps to reproduce the bug

- 1)Login in the library website as a student in google chrome
- 2) navigate through the tabs.
- 3)search for book genre category = business.
- 4)select subcategory = journals.
- 5)click on the search button.

Expected result: business related journals should be displayed not duplicate.

Actual result: I am getting the duplicate of business journals and other department duplicate journals.

**Reporter:**

Fiston Kulimushi

**Priority**

High

Environment

Browser: Google chrome

Operating system: Windows10 Home 64 bits.

Hardware specification: PC, CPU: i56400, GPU: msi 1070, Memory: 10gb ram.

**Assignee:**

Micheal Hogan

**12.**

**Summary Required:**

Poor adjustment of text on the website as result words are being removed.

**Description:**

I was testing the pages on the website, but I could not finish reading due to words being cut at the end of the paragraph.

Steps to reproduce the b

1) Login on the website as a staff member.

2) navigate through the tab.

3) click on the tab.

4) read the words being displayed there.

Expected result: I should be able to read all the words being displayed.

Actual result: the last five words are cut or removed at the end of the paragraph.

**Reporter:**

Fiston Kulimushi

**Priority**

Medium

Environment

Browser: Microsoft edge.

Operating system: MacOS 10.13: High Sierra- 2017.

Hardware specification: PC, CPU: i56400, GPU: msi 1080, Memory: 8gb ram.

**Assignee:**

Richard Alexander

**13.**

**Summary Required:**

Subscription error, the subscription was still on after the expiration date.

**Description:**

Testing my library subscription to journals and eBooks, I checked my subscription after one month of subscription, and it was not terminated.

Steps to reproduce the bug

- 1) login into the library website as a new user member.
- 2) Go to subscribe.
- 3) Subscribe for one month.
- 4) log out.
- 5) login again after one month and check my subscription again.

Expected result: I should get a confirmation email stating that I borrowed the eBook.

Actual result: I did not get any confirmation email stating the due date of the book I borrowed.

**Reporter:**

Fiston Kulimushi

**Priority**

Medium

Environment

Browser: Internet explorer

Operating system: Linux distro: Ubuntu

Hardware specifications: PC, CPU: i56400, GPU: msi 1070, Memory: 8gb ram.

**Assignee:**

Fiston Kulimushi

**14.**

**Summary Required:**

No email confirmation after borrowing an eBook from the library website.

**Description:**

Friday I was testing the library website, I borrowed an eBook, but I did not get the confirmation email

Steps to reproduce the bug

- 1) Login into the library website as a student in google chrome
- 2) navigate through the tabs.
- 3) search for the book.
- 3) go to borrow and click borrow.
- 5) log out from the website.

Expected result: I should get a confirmation email stating that I borrowed the eBook.

Actual result: I did not get any confirmation email stating the due date of the book I borrowed.

**Reporter:**

Fiston Kulimushi

**Priority**

Highest

Environment

Browser: Microsoft edge.

Operating system: MacOS 10.14: Mojave- 2018.

Hardware specification: PC, CPU: i56400, GPU: msi 1060, Memory: 8gb ram.

**Assignee:**

Benjamin Setterfield

**15.**

**Summary Required:**

Website is crashing when buttons are clicked.

**Description:**

Description: I was using the library website and I clicked on the about page and suddenly the whole website crashed.

steps to reproduce the bug

- 1) login to the website as a student.
- 2) navigate through the navigation bar.
- 3) I clicked the about page and the website crashed.

Expected result: I should be able to see the written material.

Actual result: The website crashed.

**Reporter:**

Fiston Kulimushi

**Priority**

Lowest

Environment

Browser: Internet explorer

Operating system: Linux distro: Ubuntu

Hardware specifications: PC, CPU: i56400, GPU: msi 1070, Memory: 8gb ram.

**Assignee:**

Benjamin Setterfield

**16.**

**Summary Required:**

Browser incompatibility issue.

**Description:**

Tested the library website google chrome but when I went on Microsoft edge it did not work steps to reproduce the bug

1)inserting the website URL into the Microsoft Edge browser.

2) Error is shown

Expected result: I should be able to see the front page.

Actual result: No page is shown, I see just an error.

**Reporter:**

Richard Alexander

**Priority**

Medium

**Environment**

Browser: Microsoft edge.

Operating system: macOS 10.13: High Sierra- 2017.

Hardware specification: PC, CPU:i7 9700K, GPU: RTX 2060 SUPER, Memory: 8gb ram.

**Assignee:**

Micheal Hogan

**17.**

**Summary Required:**

Time to connect to website is longer than usual

**Description:**

Tested the loading time of the home page but it was so slow to load.

1)login to the website as a visitor

2) navigating through the tabs

3)clicked on the home page

4)And it took a few minutes to load, which is not normal.

Expected result: I expected the page to load in a matter of seconds.

Actual result: it took more than two minutes to load

**Reporter:**

Richard Alexander

**Priority**

Medium

**Environment**

Browser: FireFox

Operating system: Linux.

Hardware specification: PC, CPU: AMD 3600, GPU: rx 5700xt, Memory: 16gb ram.

**Assignee:**

Richard Alexander

**18.**

**Summary Required:**

**Email password resetting error.**

**Description:**

I forgot my email password and then I reset the password, but it was not changed when I went back to try login into the website.

steps to reproduce the bug

1) login into the website as a student

2)I realized that I forgot the password

3) I reset it

4)I tried to login back but it did not work

Expected result: I should be able to log in when I change the password.

Actual result: I was not able to log in.

**Reporter:**

Richard Alexander

**Priority**

Highest

**Environment**

Browser:Safari.

Operating system: macOS 10.13: High Sierra- 2017.

Hardware specification: PC, CPU: AMD 2700x, GPU: gtx 3070, Memory: 32gb ram.

**Assignee**

Fiston Kulimushi



**19.**

**Summary Required:**

**The Font size of paragraph are too big on a mobile.**

**Description:**

tested the website on my phone but the font size was not responsive to the mobile screen size.

steps to reproduce the bugs

1)login to the website as a staff member

2)I went on the about page

3) the paragraph was too bigger than normal.

Expected result: the words should be responsive to the mobile screen size.

Actual result: The words came up too big.

**Reporter:**

Richard Alexander

**Priority**

Lowest

**Environment**

Browser: Microsoft edge.

Operating system: Windows10 Home 64 bits.

Hardware specification: PC, CPU: AMD 3700x, GPU: 3090 OC, Memory: 64gb ram.

**Assignee:**

Micheal hogan

**20.**

**Summary Required:**

**Swiping library cards do not register on the system.**

**Description:**

Testing the card swipes functionality, after swiping our library card we get a responsive beep, however the library card details are not being registered to the system.

Steps to reproduce the bug:

1. Get an appropriate library card.
2. Swipe it past the scanner.
3. Check library registration on the library administrative computer.

**Expected result:** Library ID and details on card swiped to be uploaded and log time and date of swipe.

Actual result: No details uploaded to the system from the swiped card.

**Reporter:**

Richard Alexander

**Priority**

High

**Environment**

**Browser:** Internet explorer

Operating system: windows 7

Hardware specification: PC, CPU= i5-6500, GPU= Intel on board graphics, RAM= 8GB

**Assignee:**

Micheal hogan

## Task #2: Test Design and Development using JUnit

### 1. Test-Driven-Design Approach

#### Test #1:

Rationale: Tie

Our first expected output of the program is if the game is a tie. Knowing the bounds of the results we test if the outputs are the same and return a tie result if they are. (In this test it covers 3 of the 9 possible inputs [1,1] [2,2] and [3,3].)

	1	2	3
1	1,1	1,2	1,3
2	2,1	2,2	2,3
3	3,1	3,2	3,3

Our first expected output of the program is if the game is a tie. Knowing the bounds of the results we test if the outputs are the same and return a tie result if they are. (In this test it covers 3 of the 9 possible inputs [1,1] [2,2] and [3,3].)

Code developed for test:

```
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

import static org.junit.jupiter.api.Assertions.assertEquals;

class GameLogicTest {

    private GameLogic game;

    @BeforeEach
    void setUp() {
        game = new GameLogic();
    }

    //scissors(1), rock(2), paper(3)
    @Test
    public void Tie() {
        int computerPlayerTom = 2;
        int computerPlayerJerry = 2;

        String result = game.run(computerPlayerTom, computerPlayerJerry);
        assertEquals("Its a Tie", result);
    }
}
```

Code developed for programme:

```
public class GameLogic {
    public String run(int computerPlayerJerry, int computerPlayerTom) {

        String result = "Invalid";

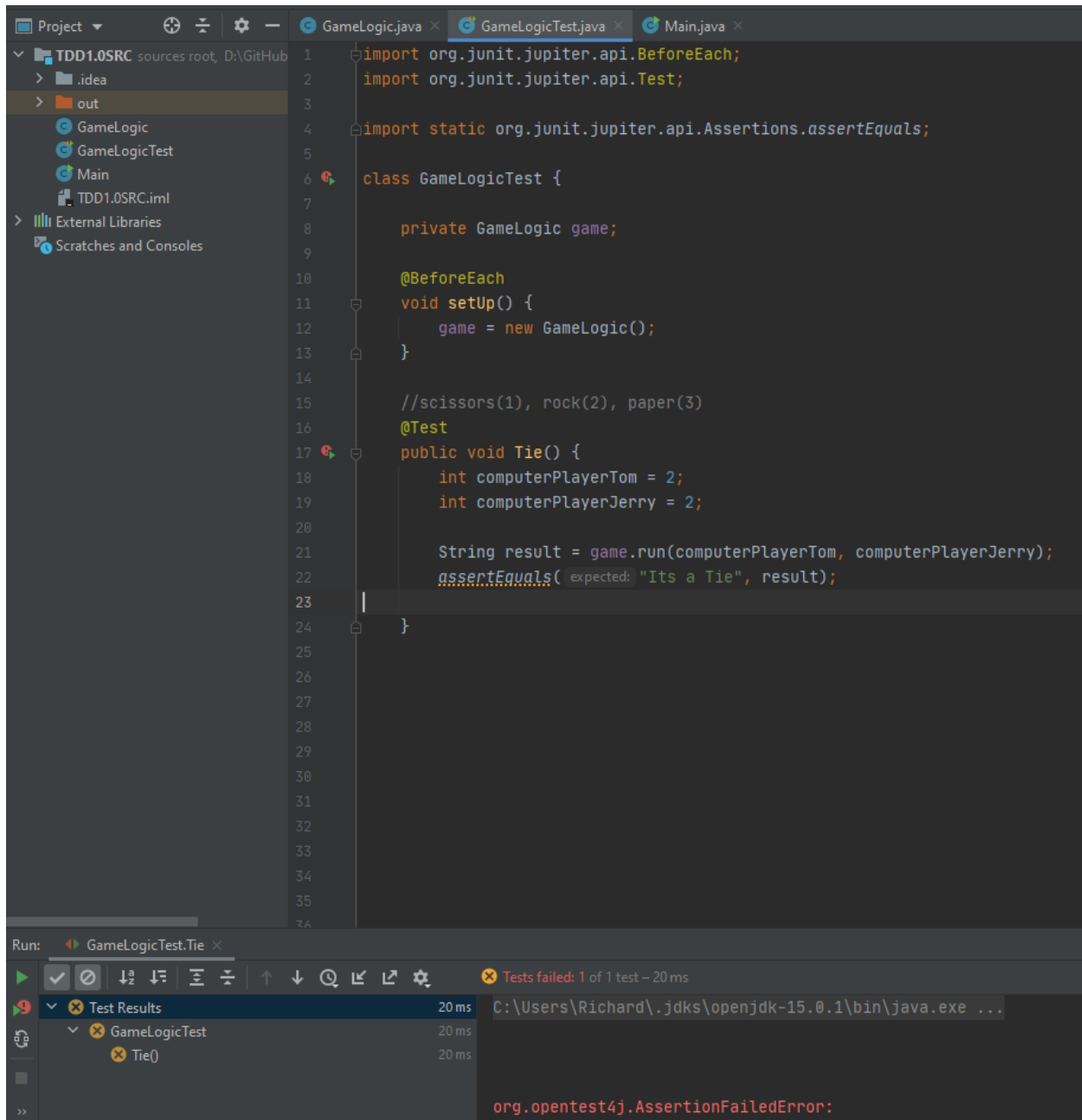
        if (computerPlayerJerry < 4 && computerPlayerJerry >= 1) {
            if (computerPlayerTom < 4 && computerPlayerTom >= 1) {

                if (computerPlayerJerry == computerPlayerTom) {

                    result = "Its a Tie";
                }
            }
        }
    }
}
```

Screenshots of failure and success.

Failure:



```
1 import org.junit.jupiter.api.BeforeEach;
2 import org.junit.jupiter.api.Test;
3
4 import static org.junit.jupiter.api.Assertions.assertEquals;
5
6 class GameLogicTest {
7
8     private GameLogic game;
9
10    @BeforeEach
11    void setUp() {
12        game = new GameLogic();
13    }
14
15    //scissors(1), rock(2), paper(3)
16    @Test
17    public void Tie() {
18        int computerPlayerTom = 2;
19        int computerPlayerJerry = 2;
20
21        String result = game.run(computerPlayerTom, computerPlayerJerry);
22        assertEquals("expected: 'Its a Tie', result");
23    }
24 }
```

Run: GameLogicTest.Tie

Tests failed: 1 of 1 test - 20 ms

Test Results	Time	Path
GameLogicTest	20 ms	C:\Users\Richard\.jdk\openjdk-15.0.1\bin\java.exe ...
GameLogicTest.Tie()	20 ms	

org.opentest4j.AssertionFailedError:

Success:

The screenshot shows an IDE with three tabs: GameLogic.java, GameLogicTest.java, and Main.java. The GameLogicTest.java tab is active, displaying the following code:

```
1 import org.junit.jupiter.api.BeforeEach;
2 import org.junit.jupiter.api.Test;
3
4 import static org.junit.jupiter.api.Assertions.assertEquals;
5
6 class GameLogicTest {
7
8     private GameLogic game;
9
10    @BeforeEach
11    void setUp() {
12        game = new GameLogic();
13    }
14
15    //scissors(1), rock(2), paper(3)
16    @Test
17    public void Tie() {
18        int computerPlayerTom = 2;
19        int computerPlayerJerry = 2;
20
21        String result = game.run(computerPlayerTom, computerPlayerJerry);
22        assertEquals( expected: "Its a Tie", result);
23    }
24 }
25
26
27
28
29
```

The Run tab at the bottom shows the test results for GameLogicTest.Tie(). The test passed in 24 ms. The console output shows the command: C:\Users\Richard\.jdk\openjdk-15.0.1\bin\java.exe ... and the message: Process finished with exit code 0.

Test Results	Time
GameLogicTest	24 ms
Tie()	24 ms

Tests passed: 1

Process finished with exit code 0

Test #2:

To cover the rest of the possible inputs we need more specific tests. Linked with a later Test (Test 4) where the inputs are swapped, we cover one more of the outputs here (That output being [1,2])

Code developed for test:

```
@Test
    public void ScissorsVsRock() {
        int computerPlayerJerry = 1;
        int computerPlayerTom = 2;

        String result = game.run(computerPlayerJerry, computerPlayerTom );
        assertEquals("Tom picked Rock and Jerry picked scissors: Tom
Wins!!!", result);
    }
```

Code developed for programme:

```
else if (computerPlayerJerry == 1) {

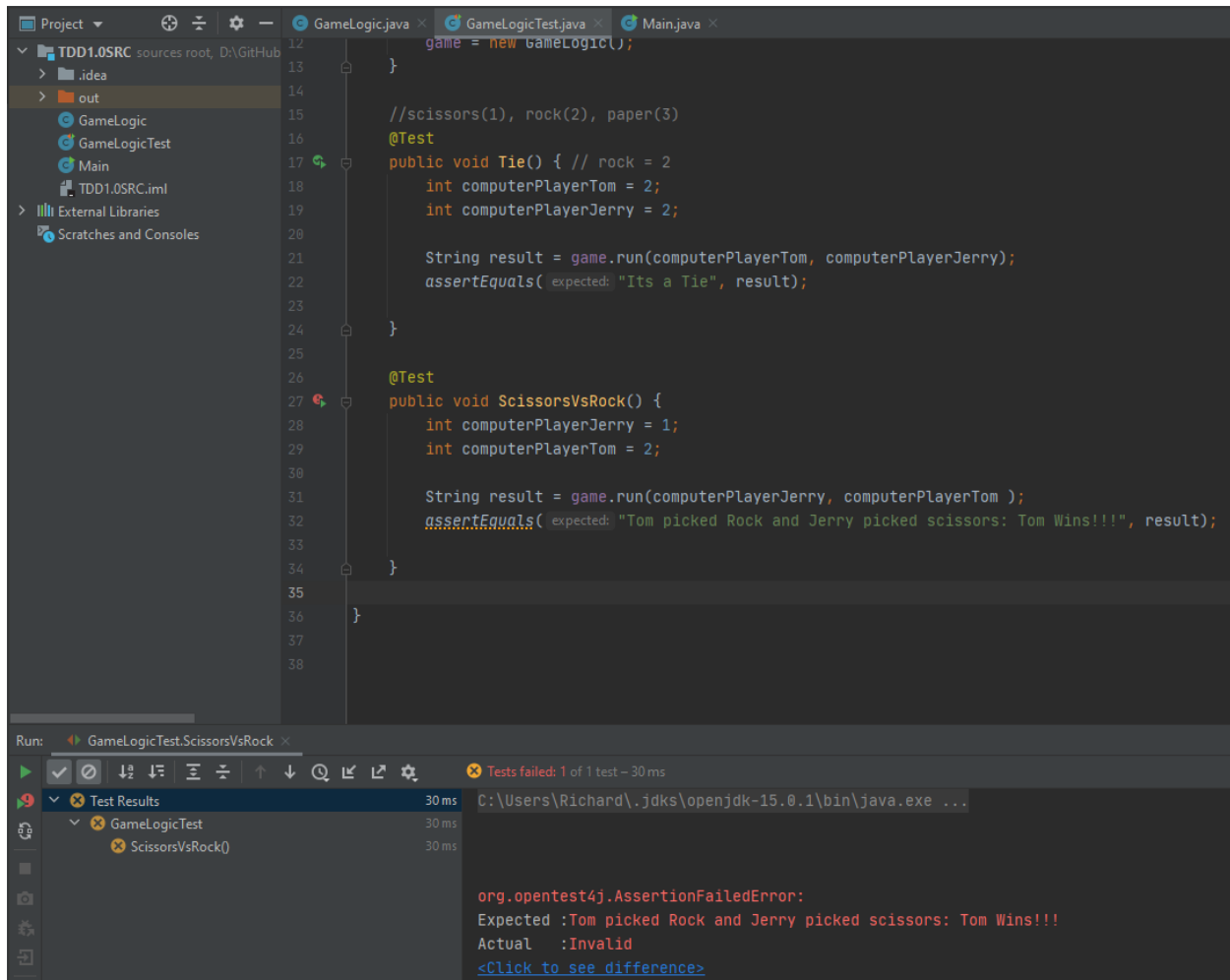
    if (computerPlayerTom == 2) {

        result = "Tom picked Rock and Jerry picked scissors: Tom
Wins!!!";

    }
```

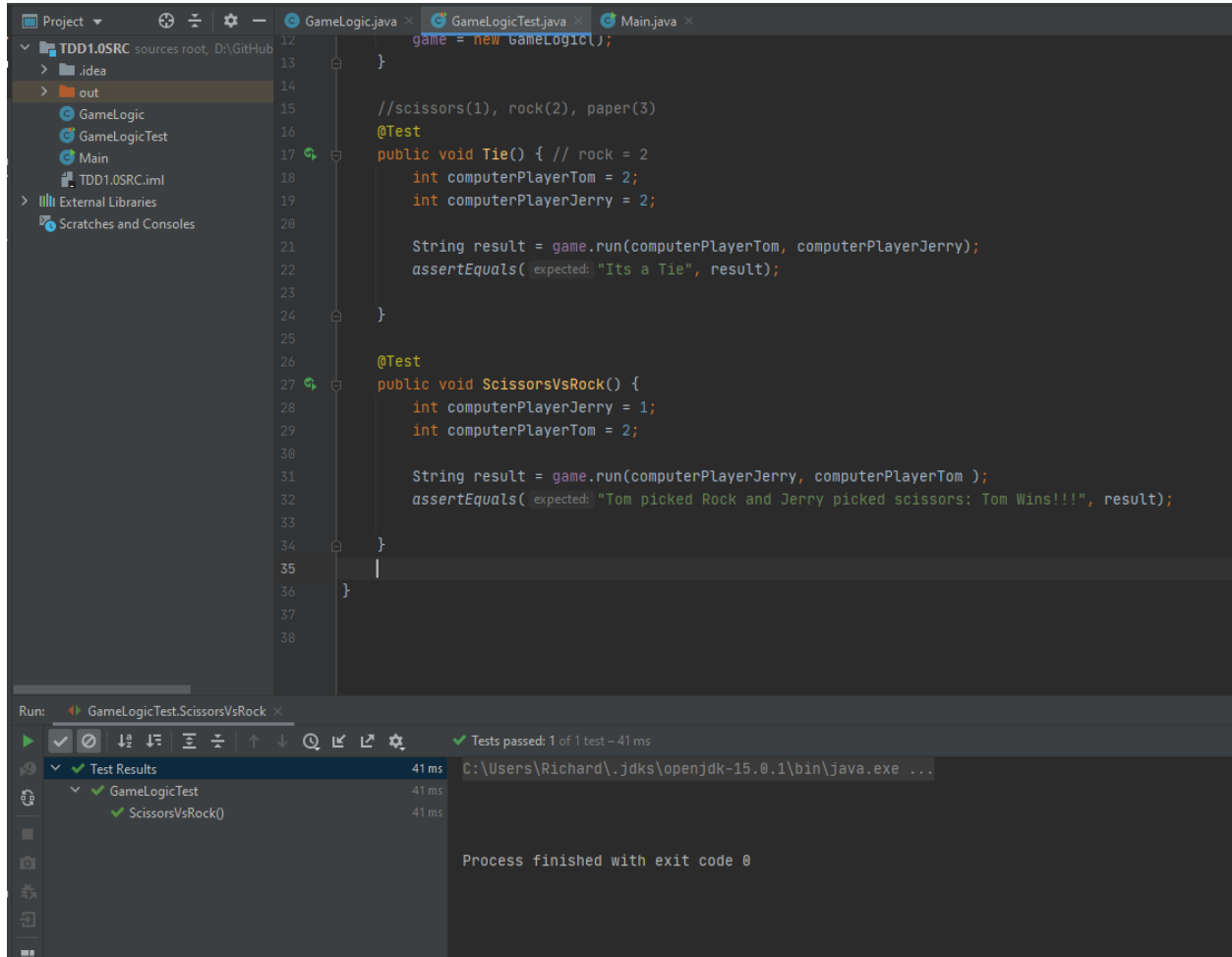
Screenshots of failure and success.

Failure:





Success:



The screenshot shows an IDE with three tabs: `GameLogic.java`, `GameLogicTest.java`, and `Main.java`. The `GameLogicTest.java` tab is active, displaying the following code:

```
12 game = new GameLogic();
13 }
14
15 //scissors(1), rock(2), paper(3)
16 @Test
17 public void Tie() { // rock = 2
18     int computerPlayerTom = 2;
19     int computerPlayerJerry = 2;
20
21     String result = game.run(computerPlayerTom, computerPlayerJerry);
22     assertEquals("expected: \"Its a Tie\", result);
23 }
24
25
26 @Test
27 public void ScissorsVsRock() {
28     int computerPlayerJerry = 1;
29     int computerPlayerTom = 2;
30
31     String result = game.run(computerPlayerJerry, computerPlayerTom );
32     assertEquals("expected: \"Tom picked Rock and Jerry picked scissors: Tom Wins!!!\", result);
33 }
34 }
35
36 }
37
38 }
```

The bottom panel shows the Run configuration for `GameLogicTest.ScissorsVsRock`. The test results indicate that the test passed successfully.

Test Results	Time	Command
Test Results	41 ms	C:\Users\Richard\.jdk\openjdk-15.0.1\bin\java.exe ...
GameLogicTest	41 ms	
ScissorsVsRock()	41 ms	

Process finished with exit code 0

### Test #3:

Like the second test, the third is a more specific test that covers another possible output and the inputs will be reversed for another test. The output of this test being [1,3] and is linked with Test 6

### Code developed for test:

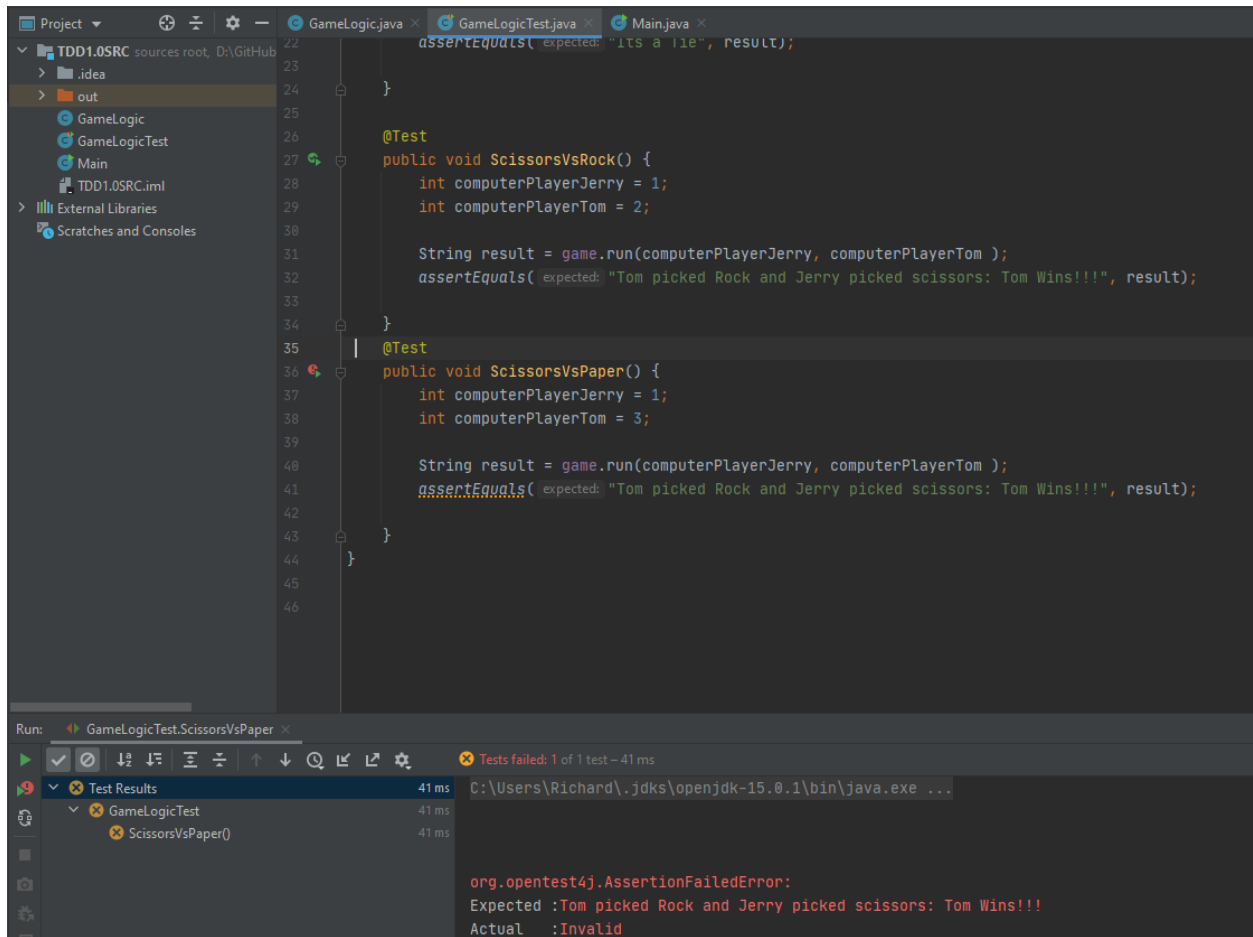
```
@Test
public void ScissorsVsPaper() {
    int computerPlayerJerry = 1;
    int computerPlayerTom = 3;
    String result = game.run(computerPlayerJerry, computerPlayerTom );
    assertEquals("Tom picked paper and Jerry picked scissors: Jerry Wins!!!",
        result);
}
```

### Code developed for programme:

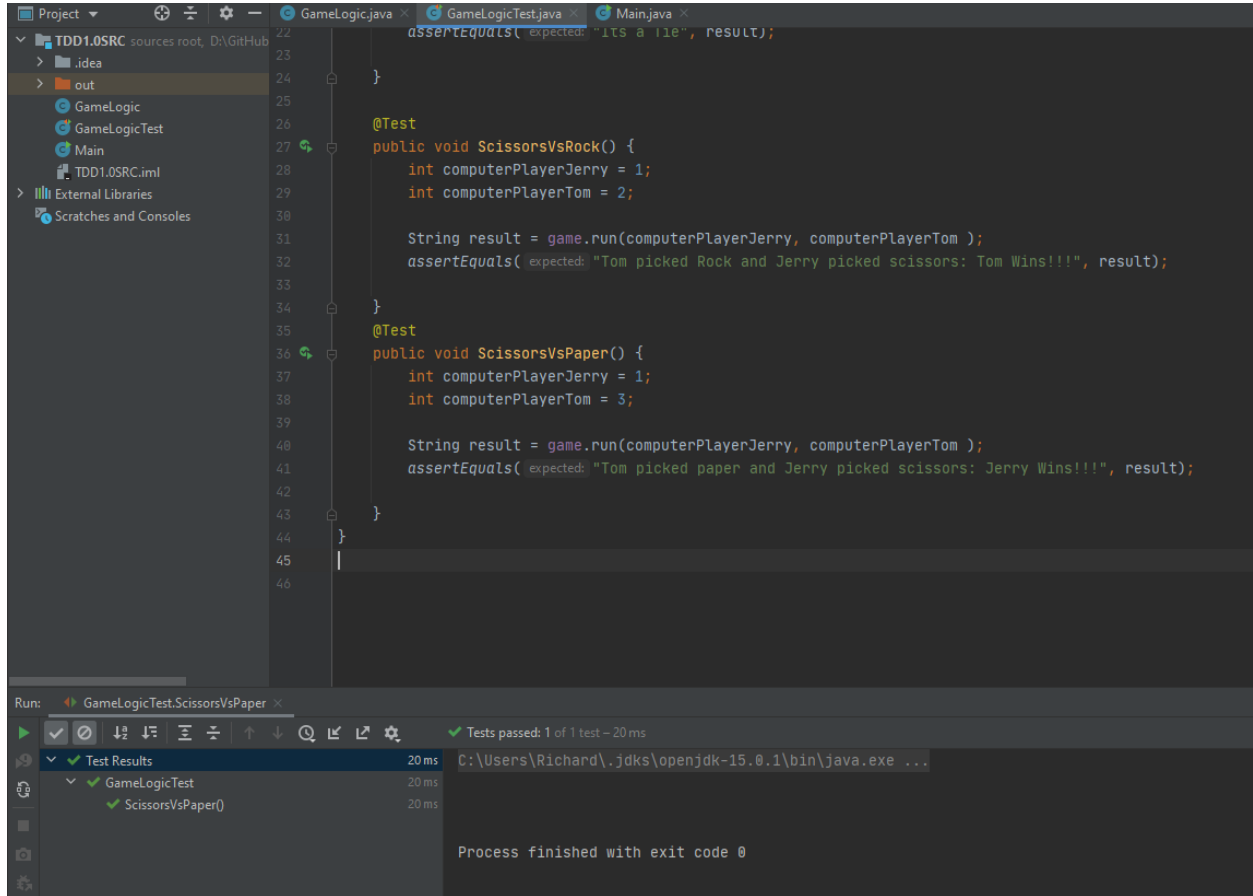
```
else {
    result = "Tom picked paper and Jerry picked scissors: Jerry Wins!!!";
}
```

Screenshots of failure and success.

Failure:



Success:



The screenshot shows an IDE with three tabs: `GameLogic.java`, `GameLogicTest.java`, and `Main.java`. The `GameLogicTest.java` tab is active, displaying the following code:

```
22 assertEquals( expected: "its a lie", result);
23
24 }
25
26 @Test
27 public void ScissorsVsRock() {
28     int computerPlayerJerry = 1;
29     int computerPlayerTom = 2;
30
31     String result = game.run(computerPlayerJerry, computerPlayerTom );
32     assertEquals( expected: "Tom picked Rock and Jerry picked scissors: Tom Wins!!!", result);
33 }
34
35 @Test
36 public void ScissorsVsPaper() {
37     int computerPlayerJerry = 1;
38     int computerPlayerTom = 3;
39
40     String result = game.run(computerPlayerJerry, computerPlayerTom );
41     assertEquals( expected: "Tom picked paper and Jerry picked scissors: Jerry Wins!!!", result);
42 }
43
44 }
45
46
```

The bottom of the IDE shows the Run window for `GameLogicTest.ScissorsVsPaper`. It indicates that the tests passed:

Run: `GameLogicTest.ScissorsVsPaper` Tests passed: 1 of 1 test - 20 ms

Test Results	Duration
GameLogicTest	20 ms
ScissorsVsPaper()	20 ms

Process finished with exit code 0

Test #4:

Linked with Test 2 we check the output [2,1]

Code developed for test:

```
@Test
    public void RockVsScissors() {
        int computerPlayerJerry = 2;
        int computerPlayerTom = 1;

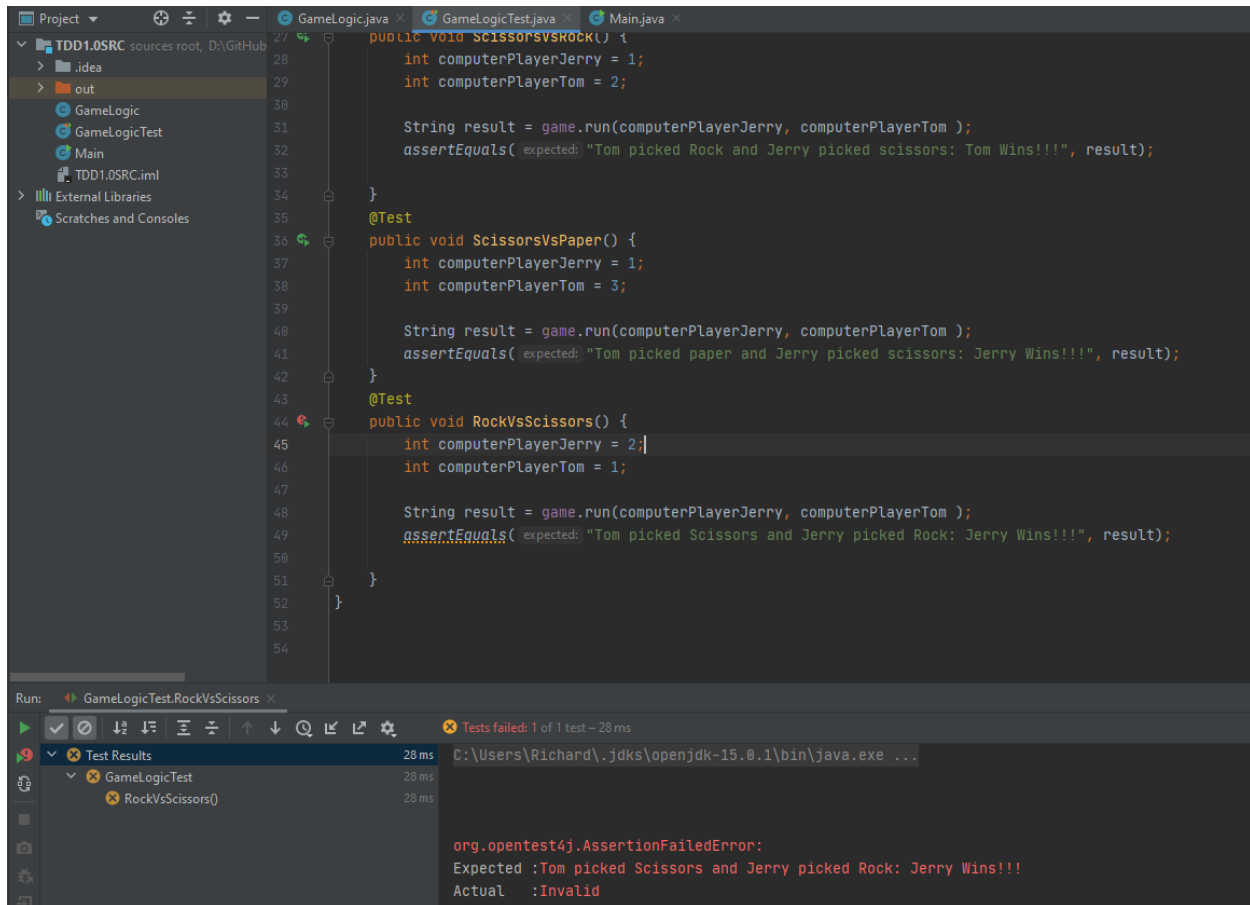
        String result = game.run(computerPlayerJerry, computerPlayerTom );
        assertEquals("Tom picked Scissors and Jerry picked Rock: Jerry Wins!!!", result);
    }
```

Code developed for programme:

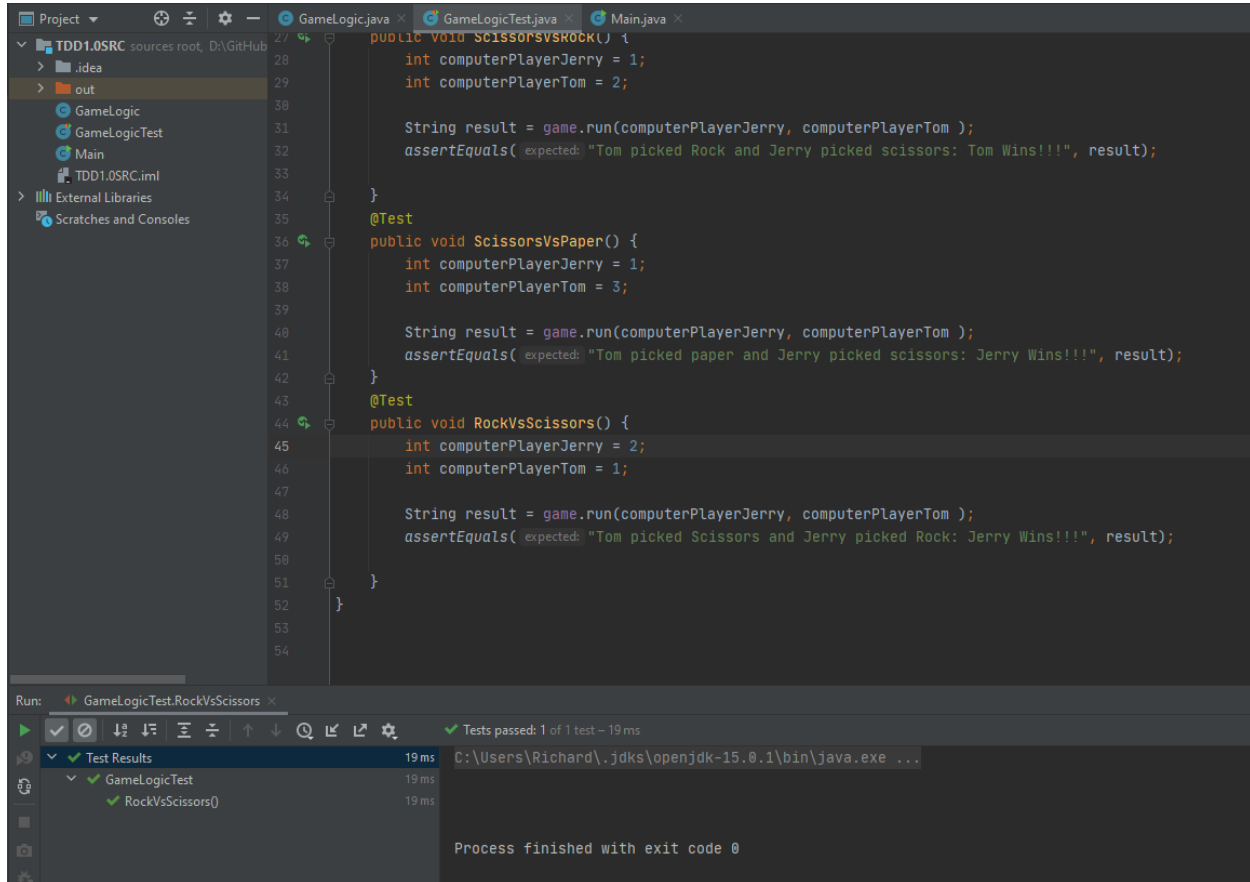
```
} else if (computerPlayerJerry == 2) {
    if (computerPlayerTom == 1) {
        result = "Tom picked Scissors and Jerry picked Rock: Jerry Wins!!!";
    }
```

Screenshots of failure and success.

Failure:



Success:



The screenshot shows an IDE with three tabs: `GameLogic.java`, `GameLogicTest.java`, and `Main.java`. The `GameLogicTest.java` tab is active, displaying the following code:

```
27 public void ScissorsVsRock() {
28     int computerPlayerJerry = 1;
29     int computerPlayerTom = 2;
30
31     String result = game.run(computerPlayerJerry, computerPlayerTom);
32     assertEquals("Tom picked Rock and Jerry picked scissors: Tom Wins!!!", result);
33 }
34
35 @Test
36 public void ScissorsVsPaper() {
37     int computerPlayerJerry = 1;
38     int computerPlayerTom = 3;
39
40     String result = game.run(computerPlayerJerry, computerPlayerTom);
41     assertEquals("Tom picked paper and Jerry picked scissors: Jerry Wins!!!", result);
42 }
43
44 @Test
45 public void RockVsScissors() {
46     int computerPlayerJerry = 2;
47     int computerPlayerTom = 1;
48
49     String result = game.run(computerPlayerJerry, computerPlayerTom);
50     assertEquals("Tom picked Scissors and Jerry picked Rock: Jerry Wins!!!", result);
51 }
52
53
54 }
```

Below the code editor, the `Run` tab is visible, showing the test results for `GameLogicTest.RockVsScissors`. The results indicate that the test passed successfully.

Test Results	Time
GameLogicTest	19 ms
RockVsScissors()	19 ms

Tests passed: 1 of 1 test - 19 ms  
C:\Users\Richard\.jdk\openjdk-15.0.1\bin\java.exe ...  
Process finished with exit code 0

Test #5:

This test covers the output of [2,3] and is linked to Test 7

Code developed for test:

```
@Test
public void RockVsPaper() {
    int computerPlayerJerry = 2;
    int computerPlayerTom = 3;
    String result = game.run(computerPlayerJerry, computerPlayerTom );
    assertEquals("Tom picked paper and Jerry picked rock: Tom Wins!!!", result);
}
```

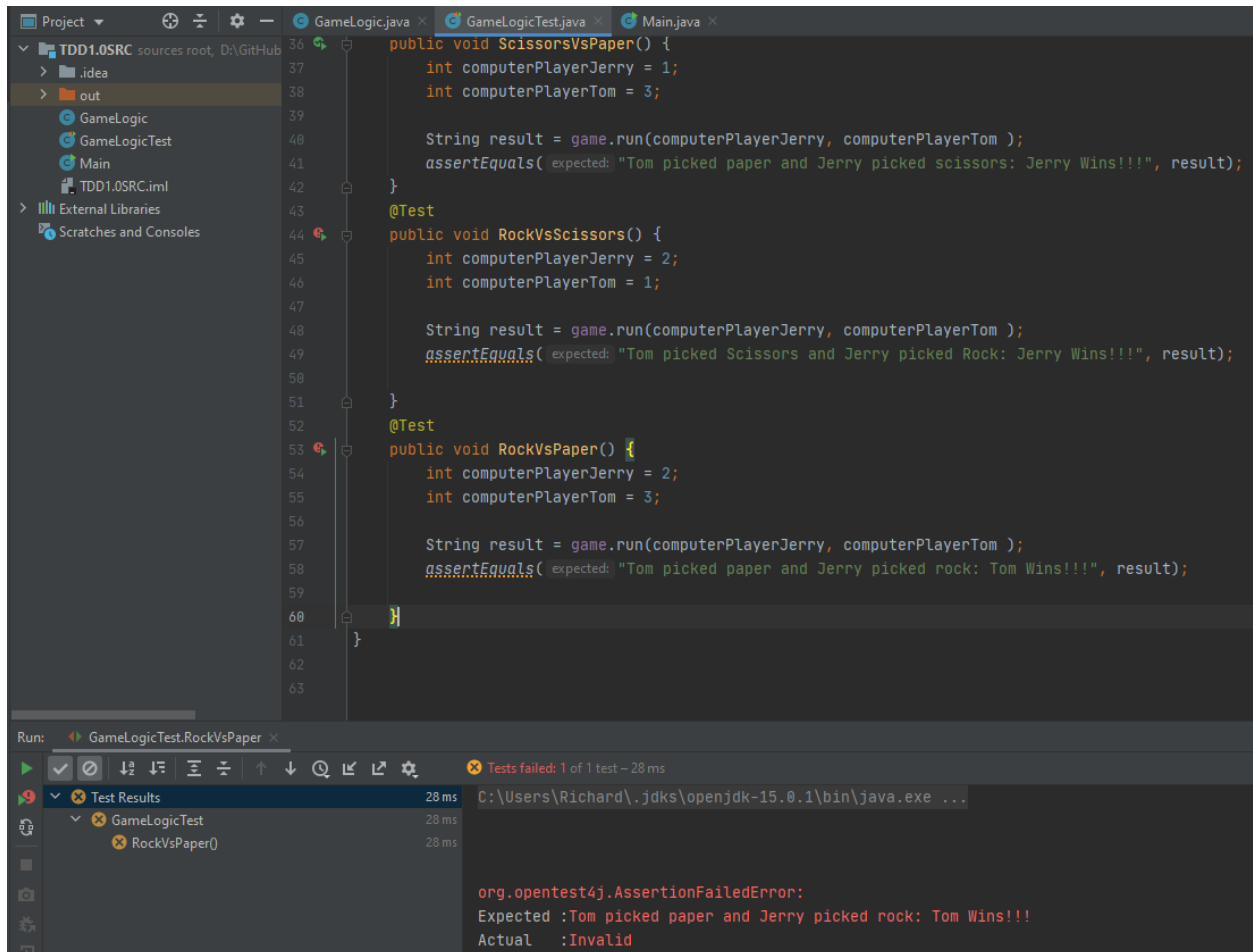
Code developed for programme:

```
else {
    result = "Tom picked paper and Jerry picked rock: Tom Wins!!!";
}
```

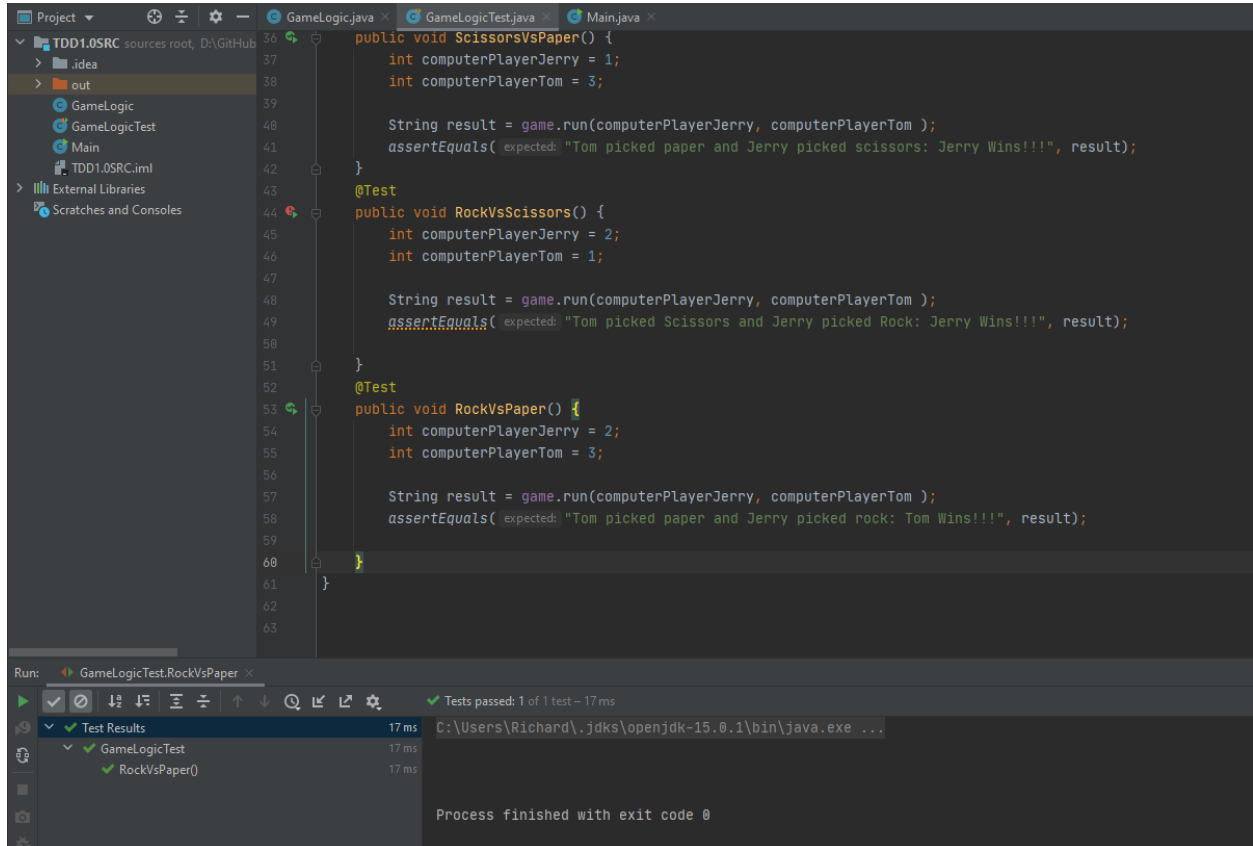


Screenshots of failure and success.

Failure:



Success:



The screenshot shows an IDE with three tabs: GameLogic.java, GameLogicTest.java, and Main.java. The GameLogicTest.java tab is active, displaying the following code:

```
36 public void ScissorsVsPaper() {
37     int computerPlayerJerry = 1;
38     int computerPlayerTom = 3;
39
40     String result = game.run(computerPlayerJerry, computerPlayerTom );
41     assertEquals( expected: "Tom picked paper and Jerry picked scissors: Jerry Wins!!!", result);
42 }
43
44 @Test
45 public void RockVsScissors() {
46     int computerPlayerJerry = 2;
47     int computerPlayerTom = 1;
48
49     String result = game.run(computerPlayerJerry, computerPlayerTom );
50     assertEquals( expected: "Tom picked Scissors and Jerry picked Rock: Jerry Wins!!!", result);
51 }
52
53 @Test
54 public void RockVsPaper() {
55     int computerPlayerJerry = 2;
56     int computerPlayerTom = 3;
57
58     String result = game.run(computerPlayerJerry, computerPlayerTom );
59     assertEquals( expected: "Tom picked paper and Jerry picked rock: Tom Wins!!!", result);
60 }
61
62
63 }
```

The bottom panel shows the Run configuration for GameLogicTest.RockVsPaper. The test results table indicates that the test passed in 17 ms.

Test Results	Time	Command
GameLogicTest	17 ms	C:\Users\Richard\.jdk\openjdk-15.0.1\bin\java.exe ...
RockVsPaper()	17 ms	

Process finished with exit code 0

Test #6:

This test covers the reverse inputs of Test 3. Covering the [3,1] output.

Code developed for test:

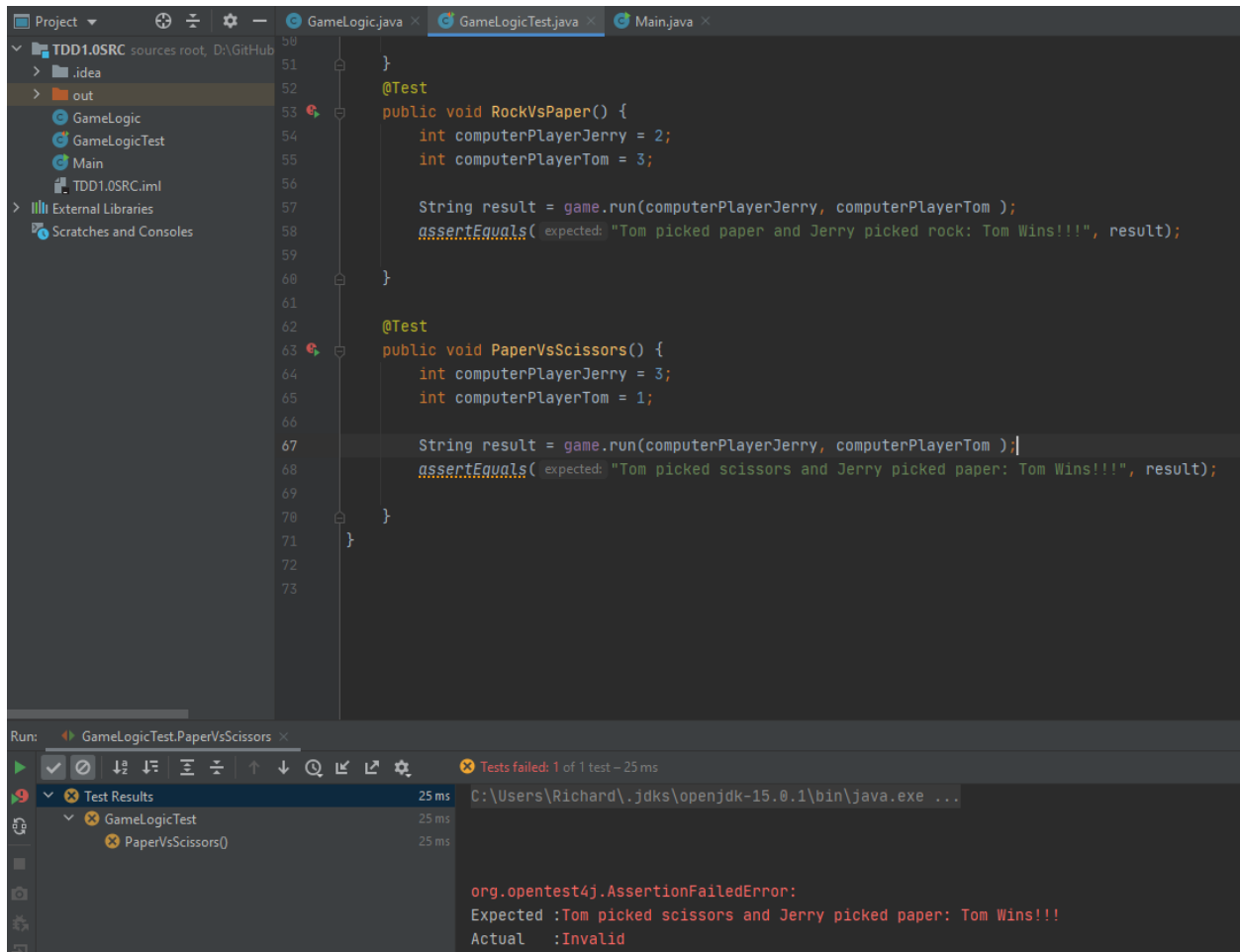
```
@Test
public void PaperVsScissors() {
    int computerPlayerJerry = 3;
    int computerPlayerTom = 1;
    String result = game.run(computerPlayerJerry, computerPlayerTom );
    assertEquals("Tom picked scissors and Jerry picked paper: Tom Wins!!!",
        result);
}
```

Code developed for programme

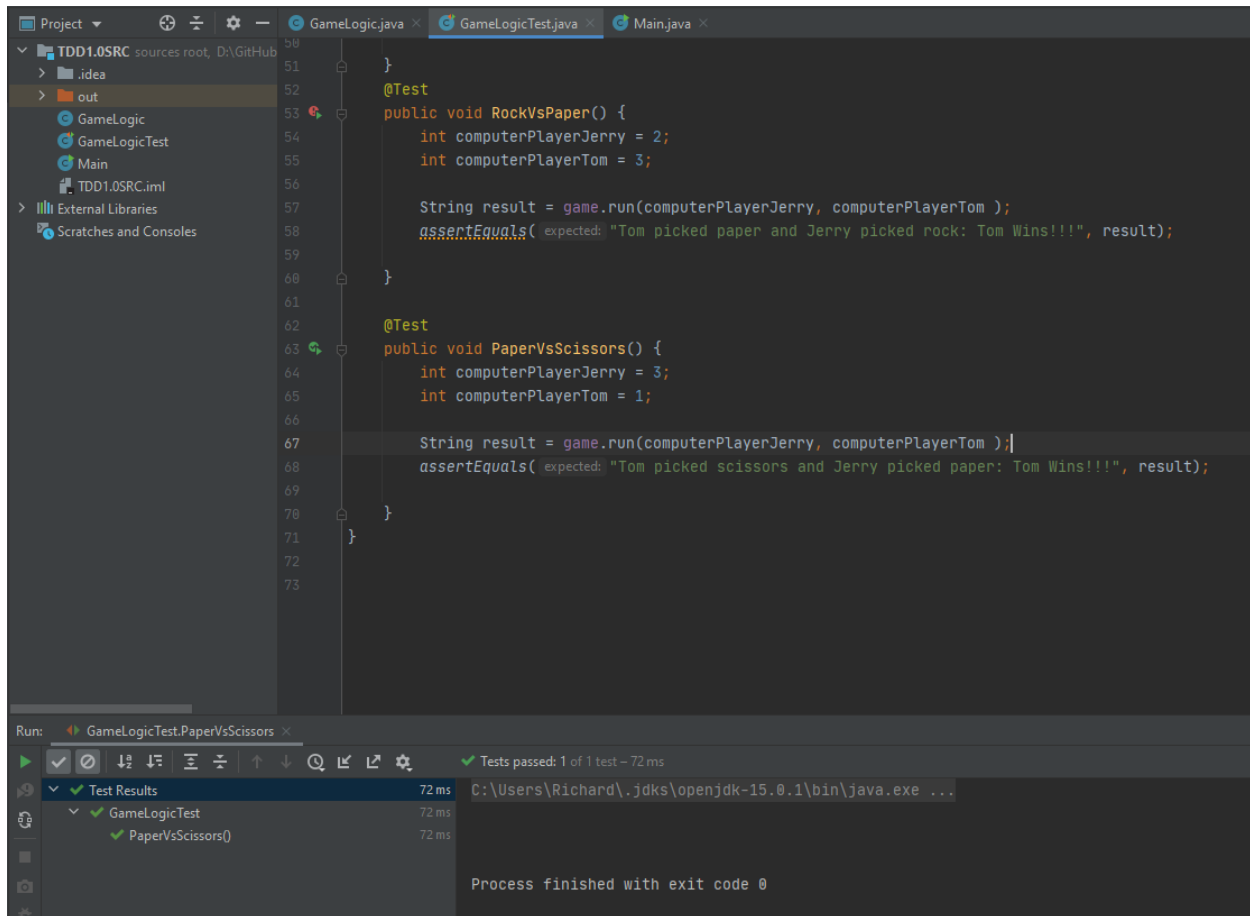
```
} else {
    if (computerPlayerTom == 1) {
        result = "Tom picked scissors and Jerry picked paper: Tom Wins!!!";
    }
```

Screenshots of failure and success.

Failure:



Success:



The screenshot displays an IDE interface with two main panels. The top panel shows the source code for `GameLogicTest.java`, which contains two test methods: `RockVsPaper()` and `PaperVsScissors()`. Both methods use `assertEquals()` to verify the output of `game.run()`. The bottom panel shows the 'Run' output, indicating that the tests passed successfully.

```
50 }
51
52 @Test
53 public void RockVsPaper() {
54     int computerPlayerJerry = 2;
55     int computerPlayerTom = 3;
56
57     String result = game.run(computerPlayerJerry, computerPlayerTom );
58     assertEquals( expected: "Tom picked paper and Jerry picked rock: Tom Wins!!!", result);
59 }
60
61
62 @Test
63 public void PaperVsScissors() {
64     int computerPlayerJerry = 3;
65     int computerPlayerTom = 1;
66
67     String result = game.run(computerPlayerJerry, computerPlayerTom );
68     assertEquals( expected: "Tom picked scissors and Jerry picked paper: Tom Wins!!!", result);
69 }
70 }
71
72
73 }
```

Run: `GameLogicTest.PaperVsScissors` Tests passed: 1 of 1 test - 72 ms

Test Results	Time	Command
GameLogicTest	72 ms	C:\Users\Richard\.jdk\openjdk-15.0.1\bin\java.exe ...
GameLogicTest.PaperVsScissors()	72 ms	

Process finished with exit code 0

## Test #7

Code developed for test:

```
@Test
    public void PaperVsRock() {
        int computerPlayerJerry = 3;
        int computerPlayerTom = 2;

        String result = game.run(computerPlayerJerry, computerPlayerTom );
        assertEquals("Tom picked Rock and Jerry picked Paper: Jerry Wins!!!",
result);
    }
}
```

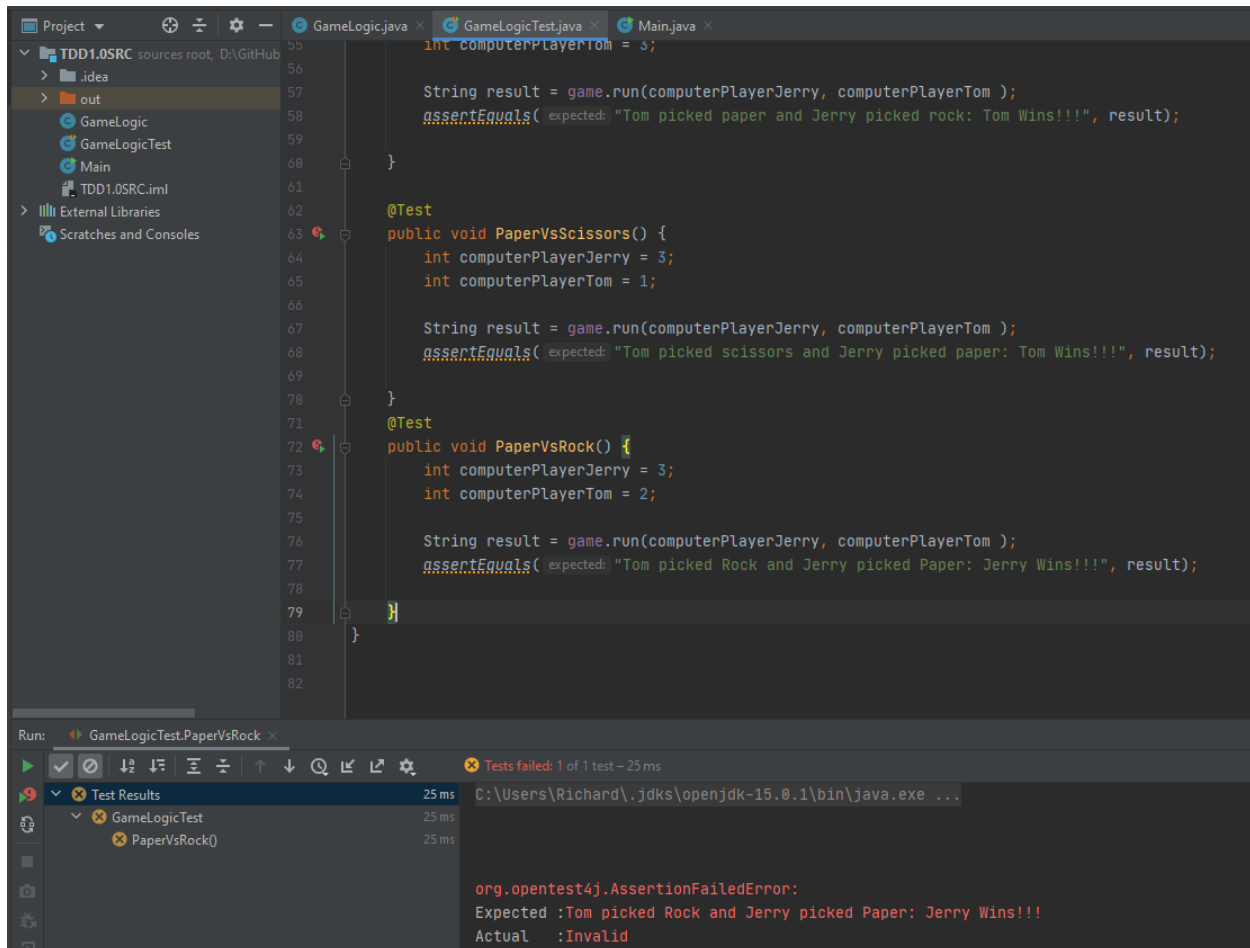
Code developed for programme:

```
else {
    result = "Tom picked Rock and Jerry picked Paper: Jerry
Wins!!!";
    }
    return result;
}

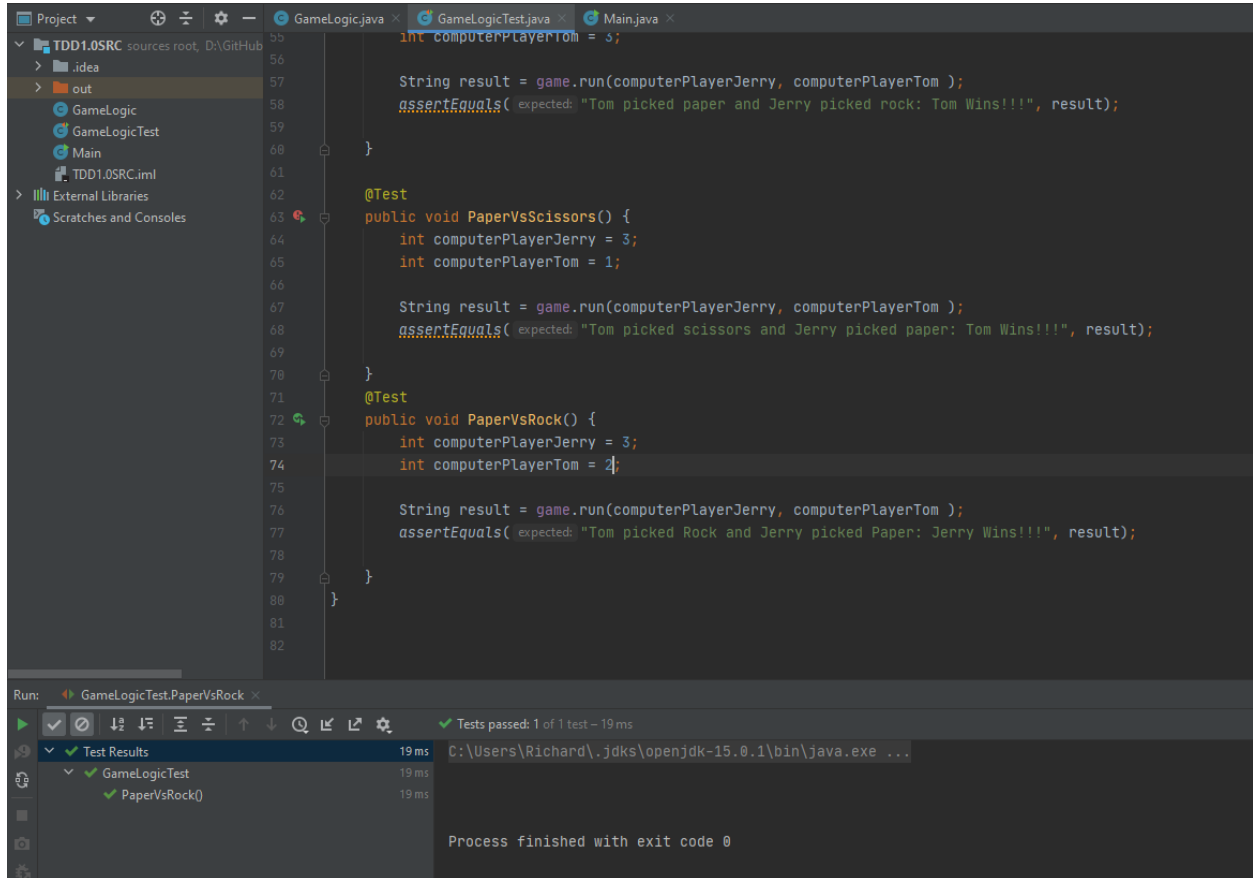
}
}
return result;
}
}
```

Screenshots of failure and success.

Failure:



Success:



The screenshot shows an IDE with three tabs: `GameLogic.java`, `GameLogicTest.java`, and `Main.java`. The `GameLogicTest.java` tab is active, displaying the following code:

```
55     int computerPlayerTom = 3;
56
57     String result = game.run(computerPlayerJerry, computerPlayerTom );
58     assertEquals( expected: "Tom picked paper and Jerry picked rock: Tom Wins!!!", result);
59
60 }
61
62 @Test
63 public void PaperVsScissors() {
64     int computerPlayerJerry = 3;
65     int computerPlayerTom = 1;
66
67     String result = game.run(computerPlayerJerry, computerPlayerTom );
68     assertEquals( expected: "Tom picked scissors and Jerry picked paper: Tom Wins!!!", result);
69
70 }
71
72 @Test
73 public void PaperVsRock() {
74     int computerPlayerJerry = 3;
75     int computerPlayerTom = 2;
76
77     String result = game.run(computerPlayerJerry, computerPlayerTom );
78     assertEquals( expected: "Tom picked Rock and Jerry picked Paper: Jerry Wins!!!", result);
79
80 }
81
82 }
```

Below the code editor, the `Run` tab is visible, showing the test results for `GameLogicTest.PaperVsRock`. The results indicate that the test passed successfully.

Test Results	Time	Command
✓ Test Results	19 ms	C:\Users\Richard\.jdk\openjdk-15.0.1\bin\java.exe ...
✓ GameLogicTest	19 ms	
✓ PaperVsRock()	19 ms	

Process finished with exit code 0



## 2.Coverage Criteria Based Testing

### A. The log:

#### **Part1 Test-Driven Approach:**

20/11/2020 Started Brain storming on what type of program we would like to create. We came up with the concept of a rock paper scissors game.

23/11/2020 Working on creating the first initial test for TDD.

24/11/2020 Finished all tests and developed code from using the TDD approach.

#### **Part2 Coverage Criteria Based Testing:**

25/11/2020 Started developing coverage tests for the code we developed from the previous part

26/11/2020 Altering the developed code so we would cover every branch and output a high-level percentile of coverage in our code. Leaving no branch untouched.

27/11/2020 Developed a test suite for Junit classes we created to test our code in one click.

## B. Reflections:

Reflection Micheál Hogan:

What did I find difficult?

What I found difficult in task 2 of the project was getting the grasp of how to go about using the TDD approach to develop a program. But eventually it clicked with me and I found the task quite rewarding. It was very satisfying to see that the various versions of the program were becoming closer and closer to what we wanted based off our tests which fulfil a project specification.

How did we go about doing some things?

We held online remote group sessions where we discussed and collaborated on the task either working on a live document, screen sharing and also, we created a GitHub repository, so all members had the most UpToDate version of the work.

Reflection Richard Alexander:

When starting this part of the project it was easy for us to decide on doing the rock paper scissors game. With our usual methodology we quickly made a simple rock paper scissors game, however we soon realised we needed take a step back and start again with a test-driven approach. Once we did this, things started to fall into place and take shape. We then refined our code and came up with coverage tests that covered up to 97.2%.

Reflection Benjamin Setterfield:

TDD and the coverage criteria was a very different experience as it is a new thought process to my experience of coding.

Thankfully the programme we selected was easy enough to plan logically.

Reflection Fiston Kulimushi:

This part of the project we had to come up with simple program that we are going to test, we finally used paper rock scissors game, I had to test one function of the program to see if it fails or pass the test. I learned how to use the Junit from testing this program and I feel good that if I get ask by an employer, I would be able to explain it perfectly.

### C. Coverage Criteria code

#### Test strategies:

When writing our test cases, we use parameterized tests to prevent large amounts of code duplication. We stored all the test cases arguments inside of a csvSource which allowed us to store multiple sets of test case sources to be executed. The number of test cases we used we kept to a minimum but still achieving a high level of coverage. We studied the control flow of our program. Looking closely at the control flow structures such as the if statements assuring that each path was taken. Our chosen selected test cases were dependent on how the program flowed. We developed a test suite so we could test both of our classes in the program with one click. This test strategy we used for both coverage test cases in both classes/methods we did below.

## Code Developed for the test:

```
import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvSource;

class GameLogicTest {

    // @Test
    // void test() {
    //     fail("Not yet implemented");
    // }

    @ParameterizedTest(name = "{index} GameOptions ({0}), ({1})) is {3}" )
    @CsvSource({ "1,1,0",
        "1,2,1",
        "1,3,2",
        "2,1,3",
        "2,2,0",
        "2,3,4",
        "3,1,5",
        "3,2,6",
        "3,3,0",
        "4,2,-1",
        "0,2,-1",
        "5,2,-1",
        "5,1,-1"
    })

    void testRun(int userPlayer, int computerPlayer, int expected) {

        GameLogic checker = new GameLogic();
        int result = checker.run(userPlayer, computerPlayer );

        assertEquals(expected, result);
    }
}
```

Code developed for the programme:

```
public class GameLogic {

    public int run(int computerPlayJerry, int computerPlayTom) {

        if (computerPlayerJerry < 4 && computerPlayerJerry >= 1)
        if (computerPlayerTom < 4 && computerPlayerTom >= 1){

            if (computerPlayerJerry == computerPlayerTom) {

                // System.out.println("Its a Tie");
                return 0;

            } else if (computerPlaeryJerry == 1) {

                if (computerPlayerTom == 2) {

                    //System.out.println("Tom picked Rock and Jerry picked
scissors: Tom Wins!!!");
                    return 1;
                } else {

                    // System.out.println("Tom picked paper and Jerry picked
scissors: Jerry Wins!!!");
                    return 2;
                }

            } else if (computerPlayerJerry == 2) {

                if (computerPlayerTom == 1) {

                    // System.out.println("Tom picked Scissors and Jerry
picked Rock: Jerry Wins!!!");
                    return 3;
                } else {

                    // System.out.println("Tom picked paper and Jerry picked
rock: Tom Wins!!!");
                    return 4;
                }

            } else {

                if (computerPlayerTom == 1) {

                    // System.out.println("Tom picked scissors and Jerry
picked paper: Tom Wins!!!");
                    return 5;

                }

            }

        }

    }

}
```

```
} else {  
    //      System.out.println("Tom picked Rock and Jerry picked  
Paper: Jerry Wins!!!");  
    return 6;  
    }  
    }  
    }  
    return -1;  
    }  
}
```

## Coverage test screenshot:

The screenshot displays an IDE with two main panels. The left panel shows the test results for 'OurTestSuite [Runner: JUnit 4]'. The tests are categorized into 'Unrooted Tests' and 'GameOptions' tests. The right panel shows the source code for 'GameLogicTest.java' and 'GameLogic.java'.

**Test Results (Left Panel):**

- OurTestSuite [Runner: JUnit 4]
  - Unrooted Tests [Runner: JUnit 4] (0.081 s)
    - 1 GameOptions ((1)\, (1)) is {3}(testRun(int\, int\, int)) (0.081 s)
    - 2 GameOptions ((1)\, (2)) is {3}(testRun(int\, int\, int)) (0.002 s)
    - 3 GameOptions ((1)\, (3)) is {3}(testRun(int\, int\, int)) (0.002 s)
    - 4 GameOptions ((2)\, (1)) is {3}(testRun(int\, int\, int)) (0.003 s)
    - 5 GameOptions ((2)\, (2)) is {3}(testRun(int\, int\, int)) (0.003 s)
    - 6 GameOptions ((2)\, (3)) is {3}(testRun(int\, int\, int)) (0.005 s)
    - 7 GameOptions ((3)\, (1)) is {3}(testRun(int\, int\, int)) (0.003 s)
    - 8 GameOptions ((3)\, (2)) is {3}(testRun(int\, int\, int)) (0.002 s)
    - 9 GameOptions ((3)\, (3)) is {3}(testRun(int\, int\, int)) (0.002 s)
    - 10 GameOptions ((4)\, (2)) is {3}(testRun(int\, int\, int)) (0.002 s)
    - 11 GameOptions ((0)\, (2)) is {3}(testRun(int\, int\, int)) (0.002 s)
    - 12 GameOptions ((5)\, (2)) is {3}(testRun(int\, int\, int)) (0.003 s)
    - 13 GameOptions ((5)\, (1)) is {3}(testRun(int\, int\, int)) (0.004 s)
    - 1 GameResults (-1) is Invalid(testResult(int\, String)) (0.008 s)
    - 2 GameResults (0) is It's a Tie(testResult(int\, String)) (0.001 s)
    - 3 GameResults (1) is Tom picked Rock and Jerry picked scissors: Tom Wins!!!(testResult(int\, String)) (0.003 s)
    - 4 GameResults (2) is Tom picked paper and Jerry picked scissors: Jerry Wins!!!(testResult(int\, String)) (0.002 s)
    - 5 GameResults (3) is Tom picked Scissors and Jerry picked Rock: Jerry Wins!!!(testResult(int\, String)) (0.001 s)
    - 6 GameResults (4) is Tom picked paper and Jerry picked rock: Tom Wins!!!(testResult(int\, String)) (0.003 s)
    - 7 GameResults (5) is Tom picked scissors and Jerry picked paper: Tom Wins!!!(testResult(int\, String)) (0.002 s)
    - 8 GameResults (6) is Tom picked Rock and Jerry picked Paper: Jerry Wins!!!(testResult(int\, String)) (0.003 s)
    - 9 GameResults (89) is INVALID(testResult(int\, String)) (0.007 s)

**Source Code (Right Panel):**

```

10 import static org.junit.jupiter.api.Assertions.*;
11
12 class GameLogicTest {
13
14     // @Test
15     // void test() {
16     //     fail("Not yet implemented");
17     // }
18
19     @ParameterizedTest(name = "{index} GameOptions ({0}), ({1}) is {3}")
20     @CsvSource({
21         "1,2,1",
22         "1,3,2",
23         "2,1,3",
24         "2,2,0",
25         "2,3,4",
26         "3,1,5",
27         "3,2,6",
28         "3,3,0",
29         "4,2,-1",
30         "0,2,-1",
31         "5,2,-1",
32         "5,1,-1"
33     })
34     void testRun(int userPlay, int computerPlay, int expected) {
35         GameLogic checker = new GameLogic();
36         int result = checker.run(userPlay, computerPlay);
37         assertEquals(expected, result);
38     }
39 }

```

**Coverage Table (Bottom Panel):**

Element	Coverage	Covered Instructions	Missed Instructions
RockPaperScissors	97.2 %	106	

Test #2:

Code Developed for the test:

```
import static org.junit.jupiter.api.Assertions.*;
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.CsvSource;

class WinnerTest {

    @ParameterizedTest(name = "{index} GameResults ({0}) is {1}" )
    @CsvSource({ "-1, Invalid",
        "0, It's a Tie",
        "1, Tom picked Rock and Jerry picked scissors: Tom Wins!!!",
        "2, Tom picked paper and Jerry picked scissors: Jerry Wins!!!",
        "3, Tom picked Scissors and Jerry picked Rock: Jerry Wins!!!",
        "4, Tom picked paper and Jerry picked rock: Tom Wins!!!",
        "5, Tom picked scissors and Jerry picked paper: Tom Wins!!!",
        "6, Tom picked Rock and Jerry picked Paper: Jerry Wins!!!",
        "89, INVALID"
    })

    void testResult(int result, String expected) {

        Winner checker = new Winner();
        String output = checker.gameResult(result);

        assertEquals(expected, output);
    }
}
```



Code developed for the programme:

```
public class Winner {
    public String gameResult(int result) {

        String temp = "INVALID";

        switch (result) {

            case -1:
                temp = "Invalid";
                break;
            case 0:
                temp = "It's a Tie";
                break;
            case 1:
                temp = "Tom picked Rock and Jerry picked scissors: Tom
Wins!!!";
                break;
            case 2:
                temp = "Tom picked paper and Jerry picked scissors: Jerry
Wins!!!";
                break;
            case 3:
                temp = "Tom picked Scissors and Jerry picked Rock: Jerry
Wins!!!";
                break;
            case 4:
                temp = "Tom picked paper and Jerry picked rock: Tom
Wins!!!";
                break;
            case 5:
                temp = "Tom picked scissors and Jerry picked paper: Tom
Wins!!!";
                break;
            case 6:
                temp = "Tom picked Rock and Jerry picked Paper: Jerry
Wins!!!";
                break;
        }
        return temp;
    }
}
```

Coverage test screenshot:

The screenshot shows an IDE with several tabs: Winner.java, OurTestSuite.java, GameLogicTest.java, WinnerTest.java (active), and GameLogic.java. The active tab displays the following Java code:

```
1 import static org.junit.jupiter.api.Assertions.*;
5
6 class WinnerTest {
7
8
9     @ParameterizedTest(name = "{index} GameResults ({0}) is {1}" )
10    @CsvSource({"-1, Invalid",
11               "0, It's a Tie",
12               "1, Tom picked Rock and Jerry picked scissors: Tom Wins!!!",
13               "2, Tom picked paper and Jerry picked scissors: Jerry Wins!!!",
14               "3, Tom picked Scissors and Jerry picked Rock: Jerry Wins!!!",
15               "4, Tom picked paper and Jerry picked rock: Tom Wins!!!",
16               "5, Tom picked scissors and Jerry picked paper: Tom Wins!!!",
17               "6, Tom picked Rock and Jerry picked Paper: Jerry Wins!!!",
18               "89, INVALID"
19            })
20
21    void testResult(int result, String expected) {
22
23        Winner checker = new Winner();
24        String output = checker.gameResult(result);
25
26        assertEquals(expected, output);
27    }
28
29
30
31
32
33 }
34
```

Below the code editor, the 'Coverage' tab is active, showing a table for 'OurTestSuite (28 Nov 2020 11:54:40)'. The table has four columns: Element, Coverage, Covered Instructions, and Missed Instructions.

Element	Coverage	Covered Instructions	Missed Instructions
> RockPaperScissors	97.2 %	106	3