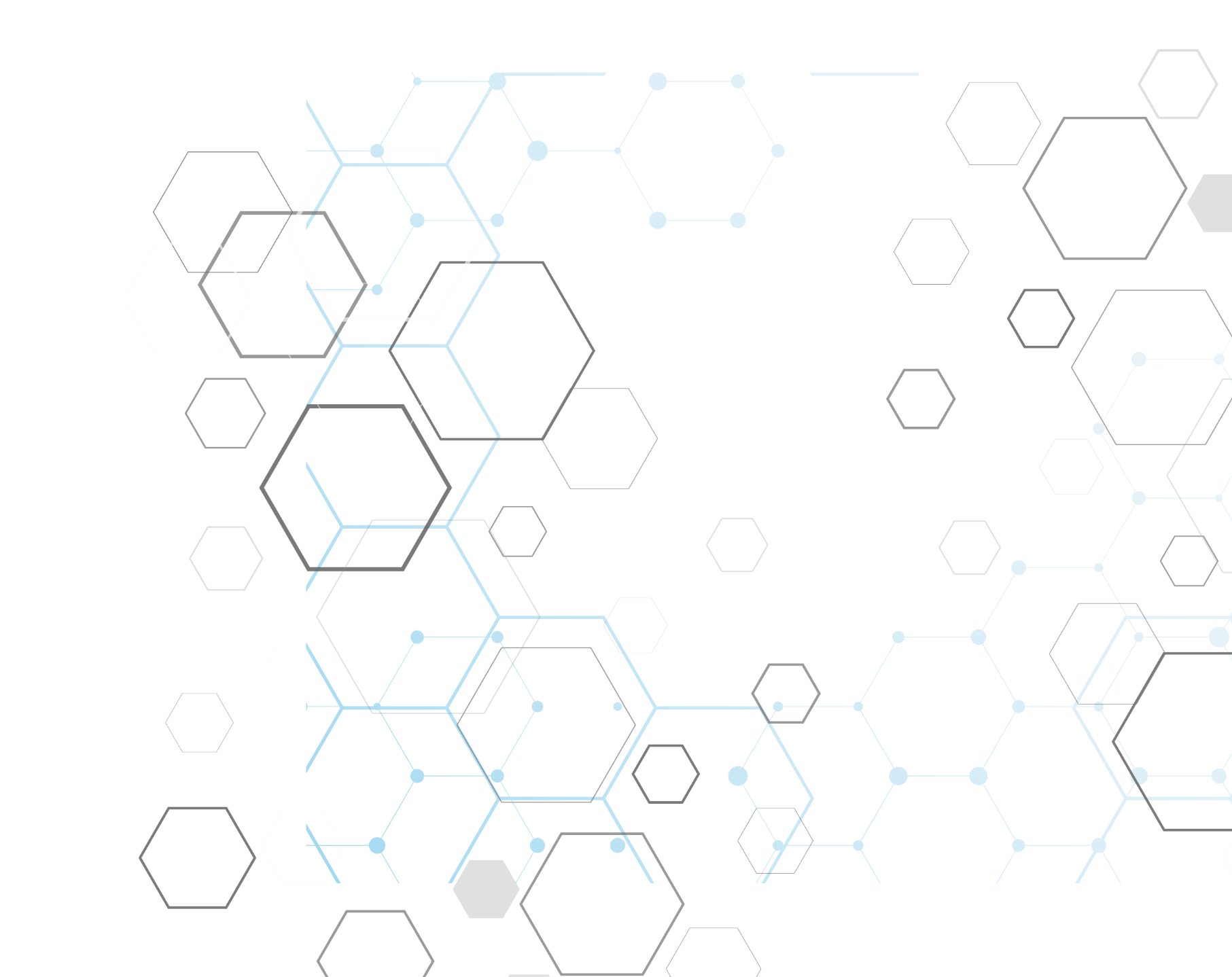
Background pattern

Description automatically generated with low confidence



Software Security Project

Part 1 & 2

Submitted by:

Shahad Alqarni 1808100

Sara Alharthi 1807648

Shadan Khan 1876267

Section: FF

**Part 1**

1. **Security Requirements:**

A security requirement is a statement of needed security functionality that ensures one of many different security properties of software is being satisfied.

The Requirement in our system:

* 1. The system must validate all user input to ensure it does not exceed the specified type.
  2. Parts of the system must be divided into smaller parts to restrict interaction between them.
  3. The system should be given only the necessary privileges to complete tasks for the user.
  4. The system must be easy to use and must output understandable messages for the user.
  5. The system should reject displaying the suggested programs if username or password were wrong, a message should be shown stating that the login failed, the login feature will be re-started.

1. **Design Principles for Secure Software:**
   1. **Compartmentalization**

We broke a system into small components (classes and functions) to facilitate control between the components of the system and increase security, using the following Classes:

* Programs
* Category
* SE
* CY
* DN
* CSAI
* CNE
  1. **Fail secure principle**

Fail secure principle involves when a system fails, it should not revert to an insecure behavior. We handled all failures securely and returned the system to its proper state and in some cases, we returned the system to condition before work.

* Using handling exception to deal with user input to ensure it does not exceed the specified type.
* If the user enters the password incorrectly three times, the system must be stopped and returned to its previous state before work.
  1. **Least Privilege**

In this principle, only the minimum access necessary to the system should be assigned to each subject that requests access to a resource, and that access should be granted only for the minimum amount of time necessary.

* We used “private” components on many parts of the system and customized “imports” to minimize access.

import java.util.Date;

import java.util.InputMismatchException;

import java.util.Scanner;

import java.util.Timer;

import java.util.TimerTask;

private rather than public on many parts of system.

* If the password and username are correct, the system will start the duration of using the program which are 10 minutes, which is enough to complete the task and access the suggested programs.
  1. **Complete Mediation:**

This principle requires that all access to objects shall be checked to ensure that they are allowed this access.

* In our system, whenever a user attempts to display the suggested programs, the system should mediate the action. We ask the user for the password before the programs display, if the password is correct, we display the suggested programs.
  1. **Minimize trust surface:**

In this principle, the user input should not be trusted, and subjected to strong input validation checks before being used.

* We use handling exception to deal with user input to be subjected input validation checks before being used.
  1. **Least Astonishment (Psychological Acceptability):**

Security requires that the error messages communicate no unnecessary information.

* We used this principle to deal with wrong passwords or usernames during displaying the suggested programs. If it said that the password or username was incorrect, the user would know that either one was legitimate. If the "user" was really an unauthorized attacker, they would know the name or password of said account for which they could try to guess the password.

In practice, the principle of psychological acceptability is interpreted to mean that the security mechanism may add some extra burden, but that burden must be both minimal and reasonable.

* In our system we placed a password as a security mechanism. To access the suggested programs, we ask the user for the password before displaying it. Although this mechanism violates the principle, as stated, it is considered sufficiently minimal to be acceptable. Where the pattern of suggested programs accesses is more frequent, this requirement would be too great of a burden to be acceptable.

**Part 2**

1. **Test cases:**

**1.1)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | 1 | **Test Case Description** | | Test the Login Functionality in the system. | | | | | | | | | | |
| **Date Tested** | | 11-Nov-2022 | **Test Case (Pass/Fail/Not Executed)** | | Pass | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | | | | | | |
| 1 | Access to System. | | |  | 1 | Username = Aleen | | | | | | | | | | |
| 2 | Enter the required information to show the suggested programs. | | |  | 2 | Password = 135634 | | | | | | | | | | |
| 3 | - | | |  | 3 |  | | | | | | | | | | |
| 4 | - | | |  | 4 |  | | | | | | | | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Test Scenario** | Verify on entering valid userName and password. | | | | | |  | | |  | |  | |  | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | | | **Pass / Fail / Not executed / Suspended** | | | | | |
|
| 1 | Navigate to System | | View the welcome message. | | As Expected | | | | | Pass | | | | | |
| 2 | Enter Username = “Aleen” | | Credential can be entered. | | As Expected | | | | | Pass | | | | | |
| 3 | Enter Password = “135634” | | Credential can be entered. | | As Expected | | | | | Pass | | | | | |
| 4 | Click Enter button | | Customer is logged in. | | As Expected | | | | | Pass | | | | | |

**1.2)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | 2 | **Test Case Description** | | Test the system when entering an incorrect password many times. | | | | | | | | | | |
| **Date Tested** | | 11-Nov-2022 | **Test Case (Pass/Fail/Not Executed)** | | Pass | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | | | | | | |
| 1 | After logging in. | | |  | 1 | Password = 135634 | | | | | | | | | | |
| 2 | Access to See the results. | | |  | 2 |  | | | | | | | | | | |
| 3 | Enter the required information to search. | | |  | 3 |  | | | | | | | | | | |
| 4 | Enter the password to see results. | | |  | 4 |  | | | | | | | | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Test Scenario** | Verify on entering incorrectly password 3 times. | | | | | |  | | |  | |  | |  | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | | | **Pass / Fail / Not executed / Suspended** | | | | | |
|
| 1 | After entering the search data. | | View the message to enter the password. | | As Expected | | | | | Pass | | | | | |
| 2 | Enter password = “111111” | | View the error message and message to enter the password again. | | As Expected | | | | | Pass | | | | | |
| 3 | Enter Password = “222222” | | View the error message and message to enter the password again. | | As Expected | | | | | Pass | | | | | |
| 4 | Enter Password = “333333” | | Shut down the system. | | As Expected | | | | | Pass | | | | | |

**1.3)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | 3 | **Test Case Description** | | Test the Search Functionality in the system | | | | | | | | | | |
| **Date Tested** | | 11-Nov-2022 | **Test Case (Pass/Fail/Not Executed)** | | Pass | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | | | | | | |
| 1 | After logging in. | | |  | 1 | Salary = 6000 | | | | | | | | | | |
| 2 | Enter the required information to search. | | |  | 2 | GPA = 5.0 | | | | | | | | | | |
| 3 | - | | |  | 3 | Interest = High | | | | | | | | | | |
| 4 | - | | |  | 4 | Password = 135634 | | | | | | | | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Test Scenario** | Verify on enter correct data to search. | | | | | |  | | |  | |  | |  | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | | | **Pass / Fail / Not executed / Suspended** | | | | | |
|
| 1 | After logging in. | | View the message to enter the salary. | | As Expected | | | | | Pass | | | | | |
| 2 | Enter Salary = “6000” | | View the message to enter the GPA. | | As Expected | | | | | Pass | | | | | |
| 3 | Enter GPA = “5.0” | | View the message to enter the Interest. | | As Expected | | | | | Pass | | | | | |
| 4 | Enter Interest = “High” | | View the message to enter the password. | | As Expected | | | | | Pass | | | | | |
| 5 | Enter password = “135634” | | View the results. | | As Expected | | | | | Pass | | | | | |

**1.4)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | 4 | **Test Case Description** | | Fuzz testing login input | | | | | | | | | | |
| **Date Tested** | | 11-Nov-2022 | **Test Case (Pass/Fail/Not Executed)** | | Pass | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | | | | | | |
| 1 | Correct input | | |  | 1 | Username = Marwa Loai | | | | | | | | | | |
| 2 | - | | |  | 2 | Password = 123 | | | | | | | | | | |
| 3 | - | | |  | 3 |  | | | | | | | | | | |
| 4 | - | | |  | 4 |  | | | | | | | | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Test Scenario** | Input incorrect login data | | | | | |  | | |  | |  | |  | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | | | **Pass / Fail / Not executed / Suspended** | | | | | |
|
| 1 | Navigate to System | | View the welcome message. | | As Expected | | | | | Pass | | | | | |
| 2 | Enter Username = “Marwa Loai” | | Credentials not acceptable | | As Expected | | | | | Pass | | | | | |
| 3 | Enter Password = “123” | | Credential can be entered. | | As Expected | | | | | Pass | | | | | |
| 4 | Click Enter button | | Customer is logged in. | | Not as Expected | | | | | Fail | | | | | |

**1.5)**

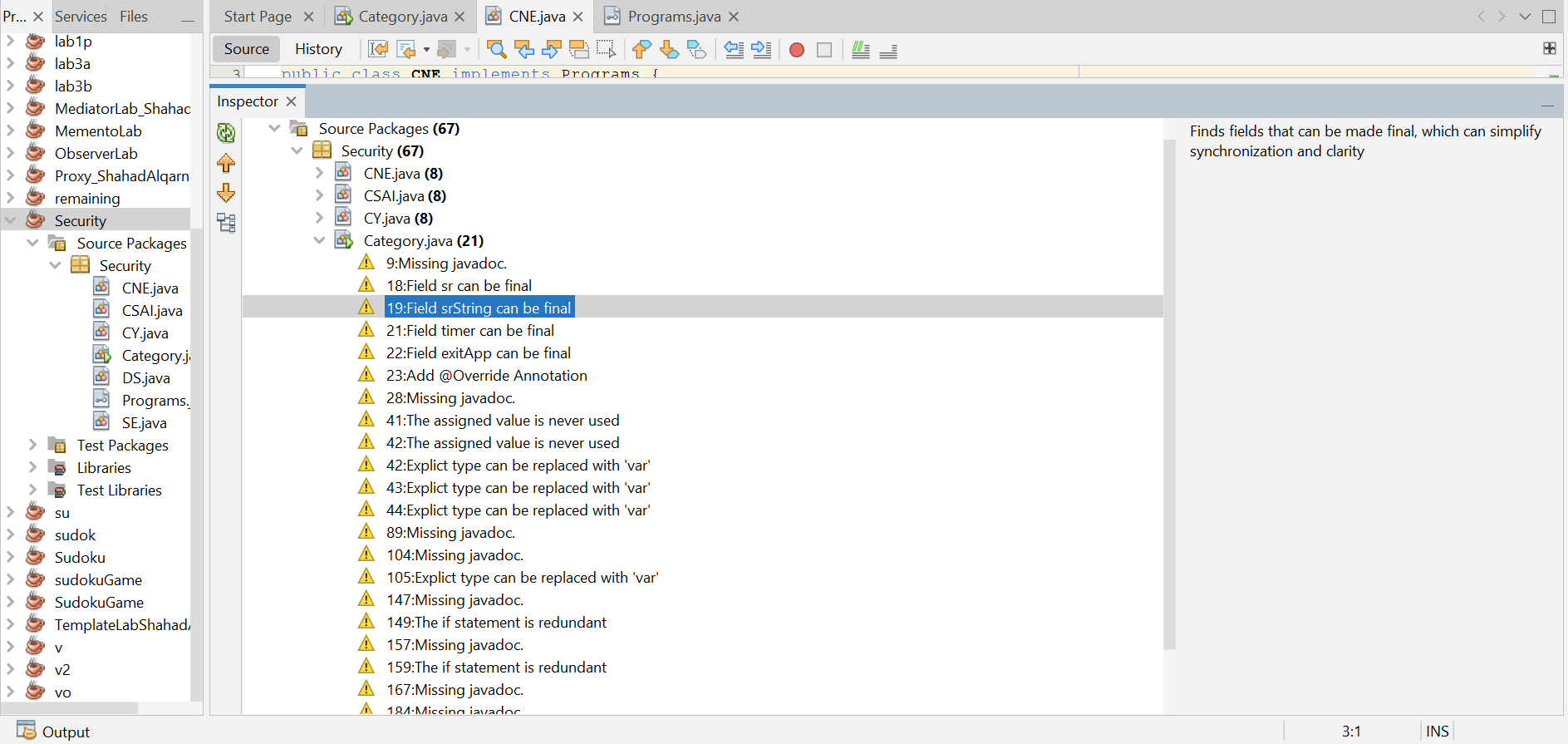
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | 5 | **Test Case Description** | | Fuzz testing system input – whitebox fuzzzing | | | | | | | | | | |
| **Date Tested** | | 11-Nov-2022 | **Test Case (Pass/Fail/Not Executed)** | | Pass | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | | | | | | |
| 1 | After logging in. | | |  | 1 | Salary: a12 | | | | | | | | | | |
| 2 | Access to See the results. | | |  | 2 |  | | | | | | | | | | |
| 3 | Enter the required information to search. | | |  | 3 |  | | | | | | | | | | |
| 4 | Enter the password to see results. | | |  | 4 |  | | | | | | | | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Test Scenario** | Input incorrect salary/gpa/interest data | | | | | |  | | |  | |  | |  | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | | | **Pass / Fail / Not executed / Suspended** | | | | | |
|
| 1 | Navigate to System | | View the welcome message. | | As Expected | | | | | Pass | | | | | |
| 2 | Enter Username = “Marwa” | | Credentials not acceptable | | As Expected | | | | | Pass | | | | | |
| 3 | Enter Password = “1234” | | Credential can be entered. | | As Expected | | | | | Pass | | | | | |
| 4 | Click Enter button | | Customer is logged in. | | As Expected | | | | | Pass | | | | | |
| 5 | Enter Salary = “a12” | | Salary is acceptable | | Not As Expected | | | | | Fail | | | | | |

**1.6)**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | 6 | **Test Case Description** | | Exit system failure – blackbox fuzzing | | | | | | | | | | |
| **Date Tested** | | 11-Nov-2022 | **Test Case (Pass/Fail/Not Executed)** | | Pass | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | | | | | | | |
| 1 | After logging in. | | |  | 1 | Exit system 0 or 1 = “3” | | | | | | | | | | |
| 2 | Access to See the results. | | |  | 2 |  | | | | | | | | | | |
| 3 | Enter the required information to search. | | |  | 3 |  | | | | | | | | | | |
| 4 | Enter the password to see results. | | |  | 4 |  | | | | | | | | | | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Test Scenario** | Input incorrect exit choice | | | | | |  | | |  | |  | |  | |
|  |  |  |  |  |  |  | |  | | |  | |  | |  | |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | | | **Pass / Fail / Not executed / Suspended** | | | | | |
|
| 1 | After logging in. | | View the message to enter the salary. | | As Expected | | | | | Pass | | | | | |
| 2 | Enter Salary = “3000” | | View the message to enter the GPA. | | As Expected | | | | | Pass | | | | | |
| 3 | Enter GPA = “3.0” | | View the message to enter the Interest. | | As Expected | | | | | Pass | | | | | |
| 4 | Enter Interest = “Medium” | | View the message to enter the password. | | As Expected | | | | | Pass | | | | | |
| 5 | Enter password = “12345” | | View the results. | | As Expected | | | | | Pass | | | | | |
| 6 | Enter exit system choice = “3” | | Exit system | | Not as expected | | | | | Fail | | | | | |

1. **Static** **analysis result:**

Static analysis is a process to inspect a program based on its source code. There is no need to compile or execute the program: only the source code is necessary. As only the source code is needed, static analysis tools are language specific. Such tools parse, interpret the program to analyze it, and find potential issues.

****The static analysis tool processes the source code and finds potential issues. We will use an Inspector from Apache NetBeans to report problems that can be severe and require immediate attention.

The largest number found is in the category class, which is equivalent to 21 issues, then we found 8 issues in (CBE, CSE, CY, DS and S class) and the smallest number found is in the Programs class which is equivalent to 6 issues.

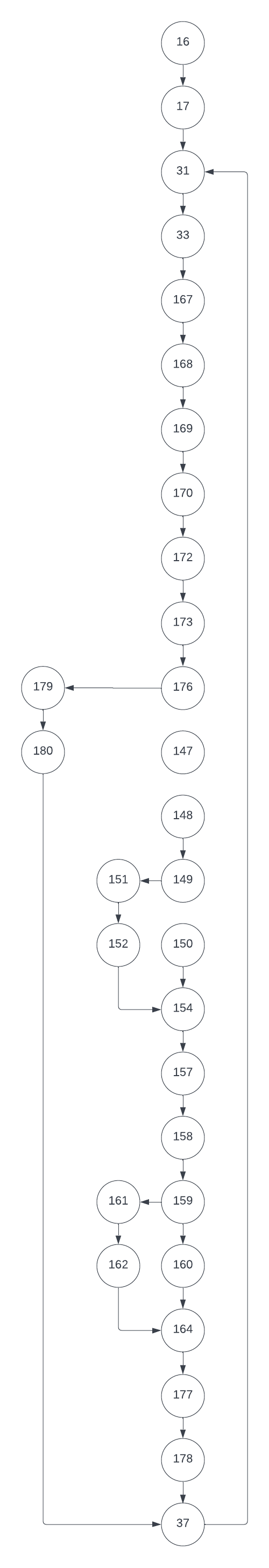
1. **Dynamic analysis result:**

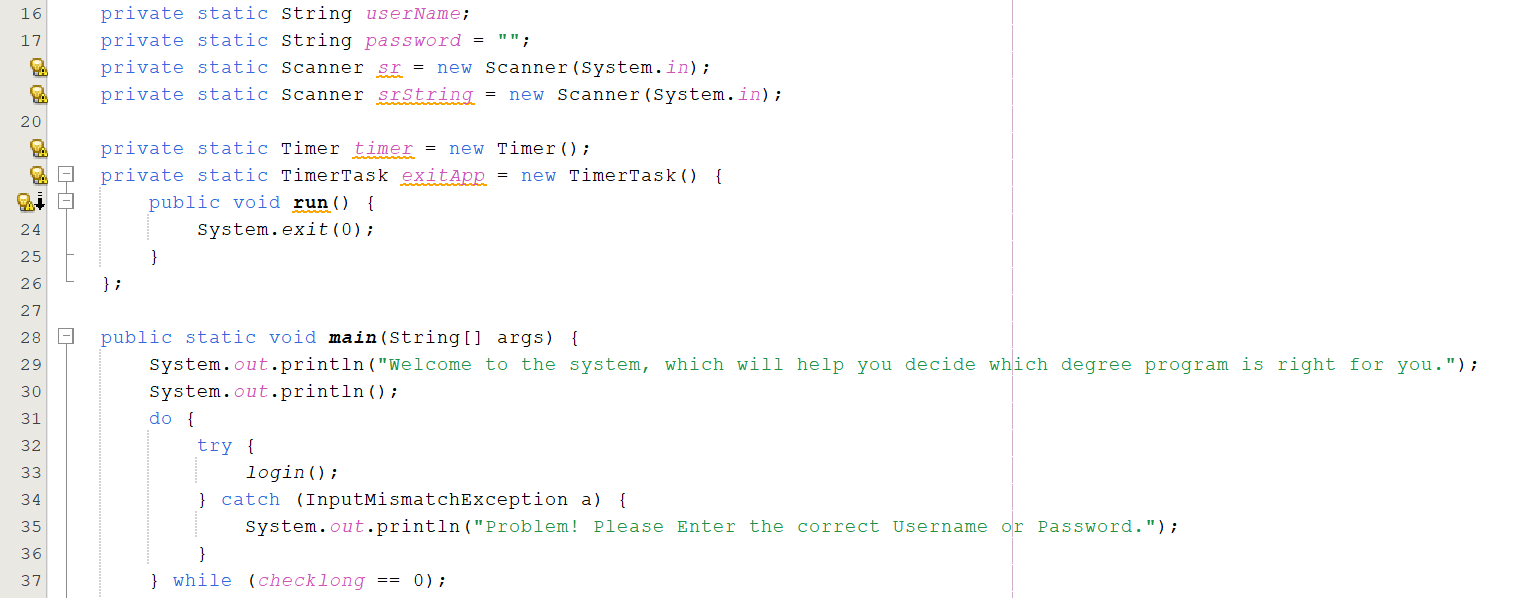
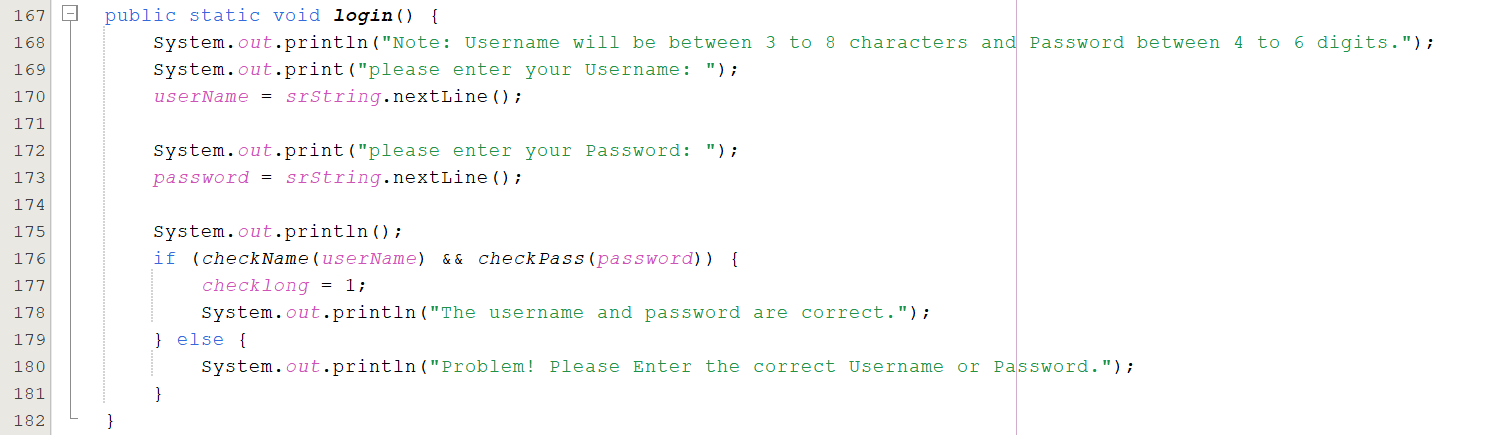
Dynamic analysis is done by executing the program and inspecting its behavior.

* 1. **Testing techniques:**

**4.1.1) White box testing:**

Control Flow Graphs (CFG) technique comes under white box testing. This type of testing method is used to test our code. Control flow graphs are graphical representations of control flow that is done during the execution of the program. After drawing the graphics, we can use cyclomatic complexity is a software metric used to describe the complexity of a program. It is computed using this equation M = E - N + 2P.

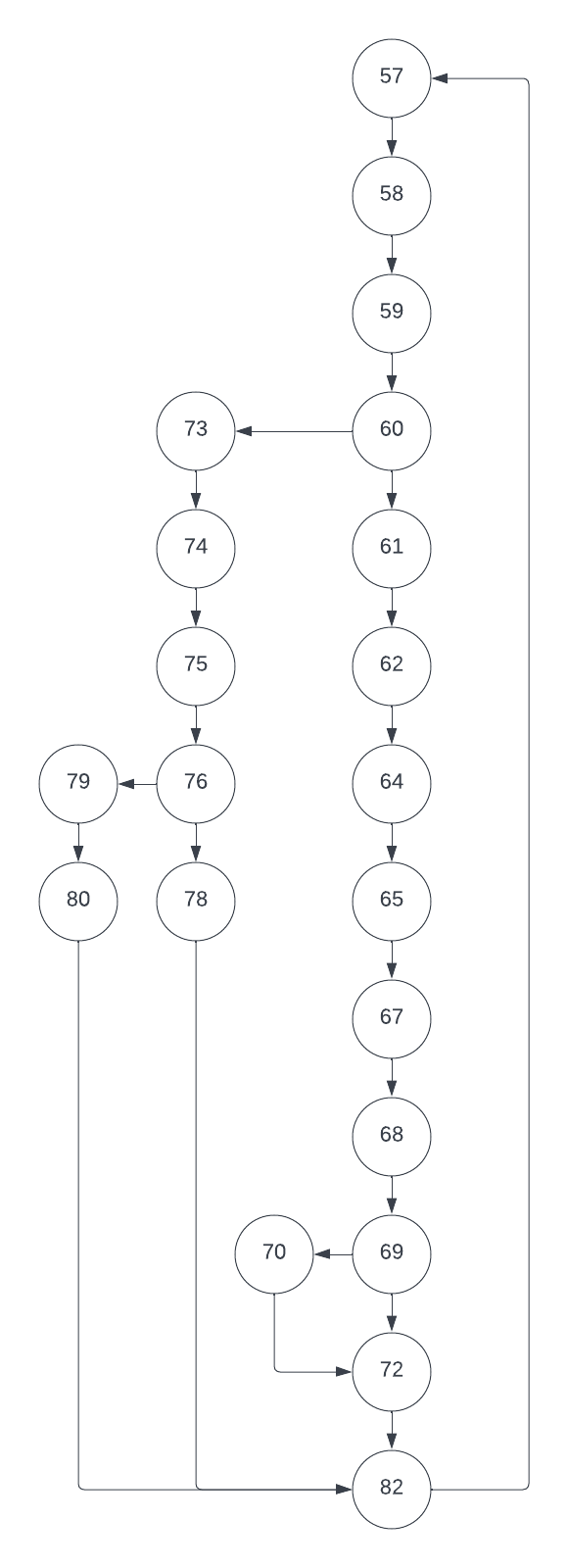
* **For test case number 1:**

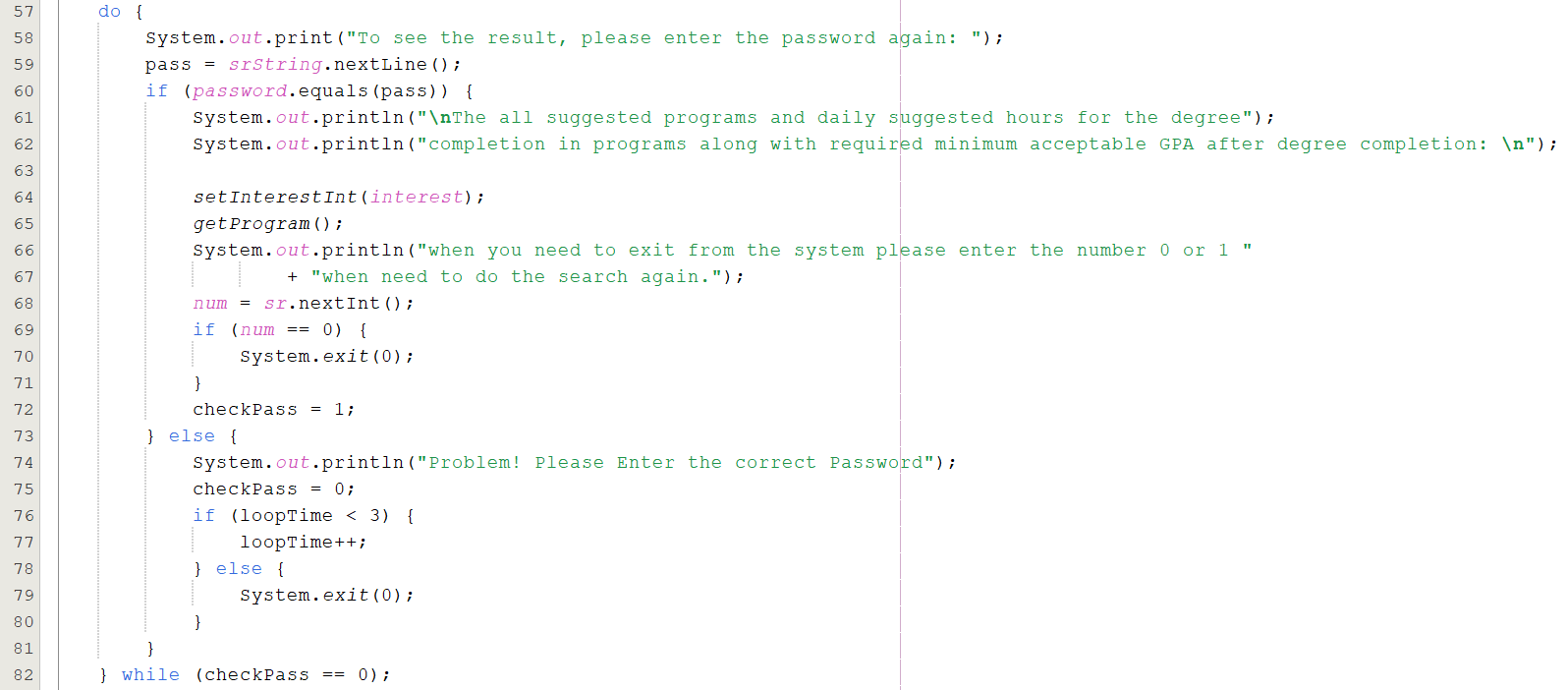
 In this graphic, we have shown the graphic for the Control Flow Graphs of the login process. In Test case 1, we tested if the username and password were correct, which will cover 80 of the graphics, but if we try to test when the username is incorrect, 83 of the graphics will be covered. This indicates that the coverage percentage will vary according to the inputs.

Graphical user interface, text, email

Description automatically generated

The Cyclomatic Complexity = 5

* **For test case number 2:**

 In this graphic, we have shown the graphic for the Control Flow Graphs of what happens when entering an incorrect password. In Test case 2, we tested when entering an incorrect password 3 times, which will cover 50 of the graphics, but if we try to test when entering an incorrect password 1 time, 45 of the graphics will be covered. This indicates that the coverage percentage will vary according to how many times will enter an incorrect password.

The Cyclomatic Complexity = 4

* For test case number 3:

**Diagram

Description automatically generated** In this graphic, we have shown the graphic for the Control Flow Graphs of Search Functionality. In Test case 3, we tested when entering the correct data to do the search, which will cover 100 of the graphics, but if we try to test when entering incorrect data, also 100 of the graphics will be covered. because the graphic is simple without details.

**A picture containing logo

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

The Cyclomatic Complexity = 2

* Summary of statement coverage:

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case** | **Input** | **Expected Output** | **Decision Coverage** |
| **1** | Username = Aleen  Password = 135634 | Log in successive. | 80% |
| **1** | Username = Aleen  Password = 112233 | Print error massage. | 83% |
| **2** | Password = 111111  (First time) | Print error massage. | 45% |
| **2** | Password = 111111  (Third time) | Shut down the system. | 50% |
| **3** | Salary = 6000  GPA = 5.0  Interesting = High | Request password from user. | 100% |

**4.1.2) Black box testing:**

This table contains boundary conditions for username, password, and Check password times.

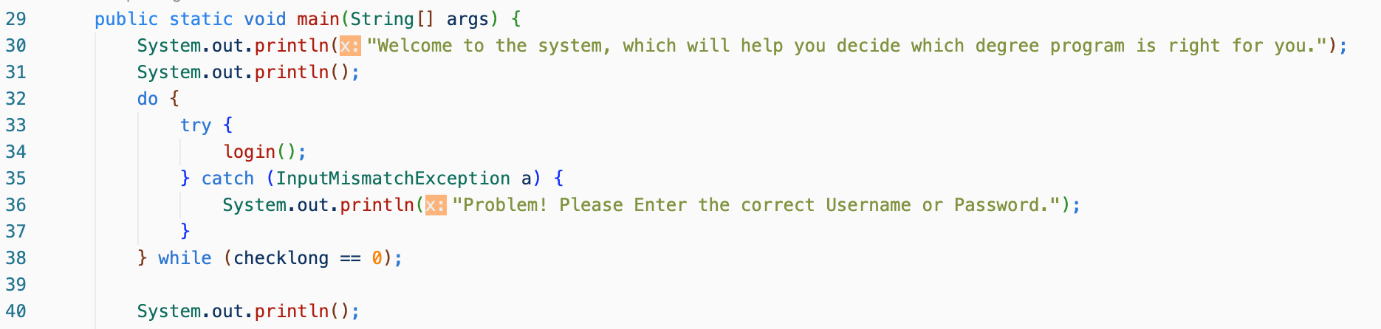
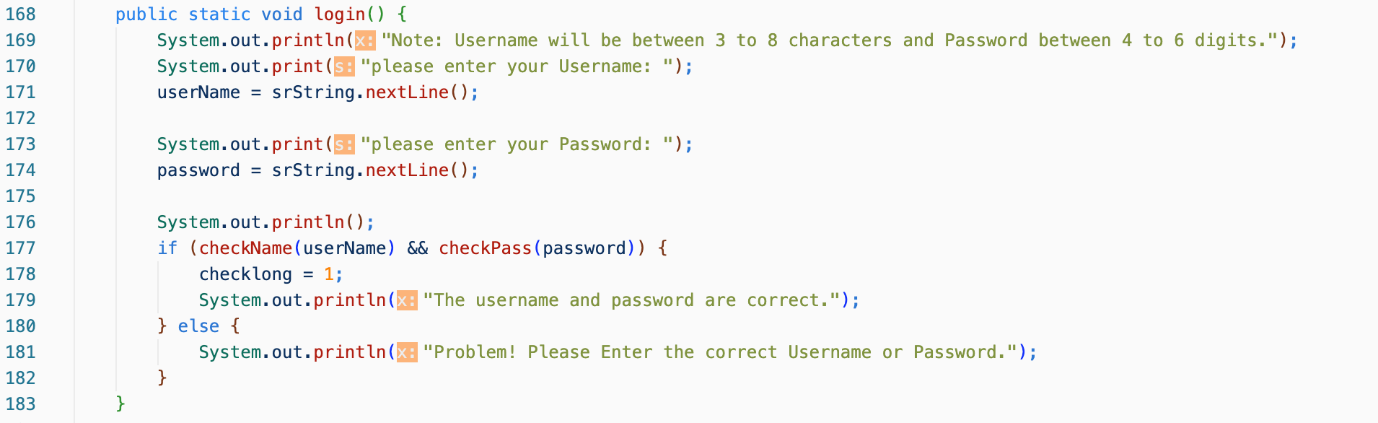
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **conditions** | **Valid Partitions** | **Invalid Partitions** | **Valid Boundaries** | **Invalid Boundaries** |
| **Name** | 3 to 8 chars | <3 chars | 3 chars | 2 chars |
| Valid chars | >8 chars | 8 chars | 9 chars |
| Invalid chars | 0 chars |
| **Password** | 4 to 6 chars | <4 chars | 4 chars | 3 chars |
| Valid chars | >6 chars | 6 chars | 7 chars |
| Invalid chars | 0 chars |
| **Check password times** | 1 to 3 times | < 1 time | 1 time | 0 time |
| Valid input | > 3 time | 3 time | 4 time |
| Invalid input |  |

**4.1.3) Fuzz testing:**

Fuzzing is an automated software testing technique that involves providing an invalid input to a program. It has many types, but in our software, we used the following types:

* **White box fuzzing:**

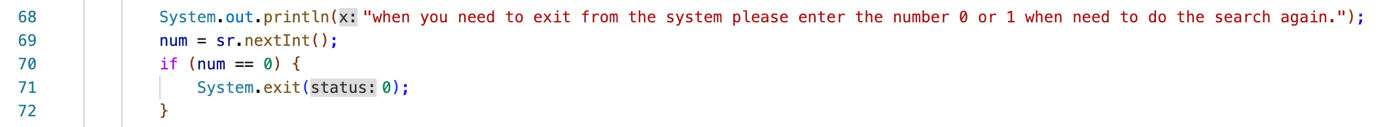
White box fuzzing is an alternative approach to testing a whole program dynamically, starting with a well-formed input.

* **For test case 4:**

In this test case we are trying to enter an invalid username and password, but our system verifies the type of entry so it will not become a system crash.

* **Black box fuzzing:**

Black box fuzzing is a form of black box testing which mutates well-formed inputs and tests the program with those inputs with hope of triggering a bug in the system

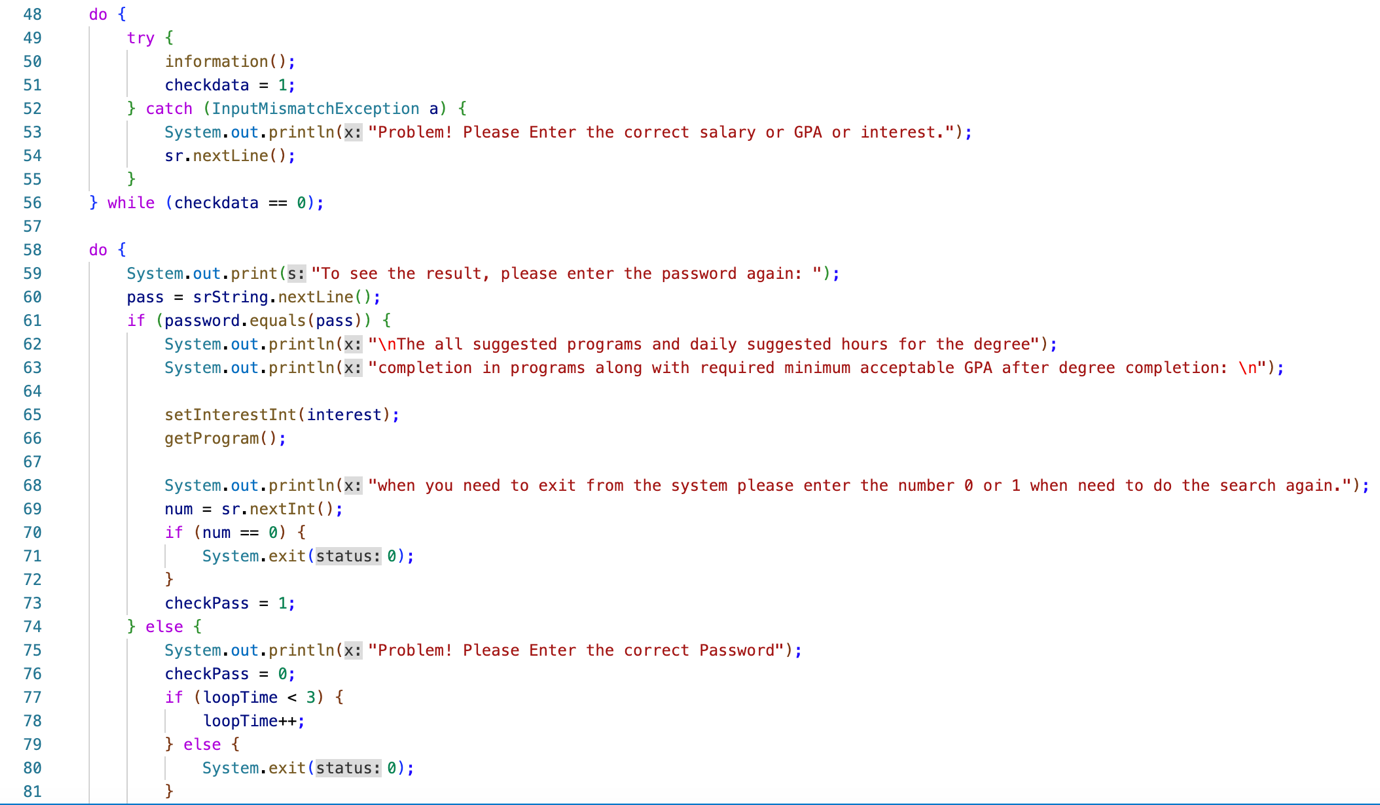
* **For test case 6:**

In this test case we try to enter an invalid number, but our system will not deal and will not respond to any input other than zero or one, and the system will remain open without interaction.

**Note: the defect report will be displayed in the section six.**

* **Grey box fuzzing:**

Grey box fuzzing is a test that combines both the white box fuzzing and the black box fuzzing. It’s an automated test-input technique that aims to uncover program errors.

* **For test case 5:**

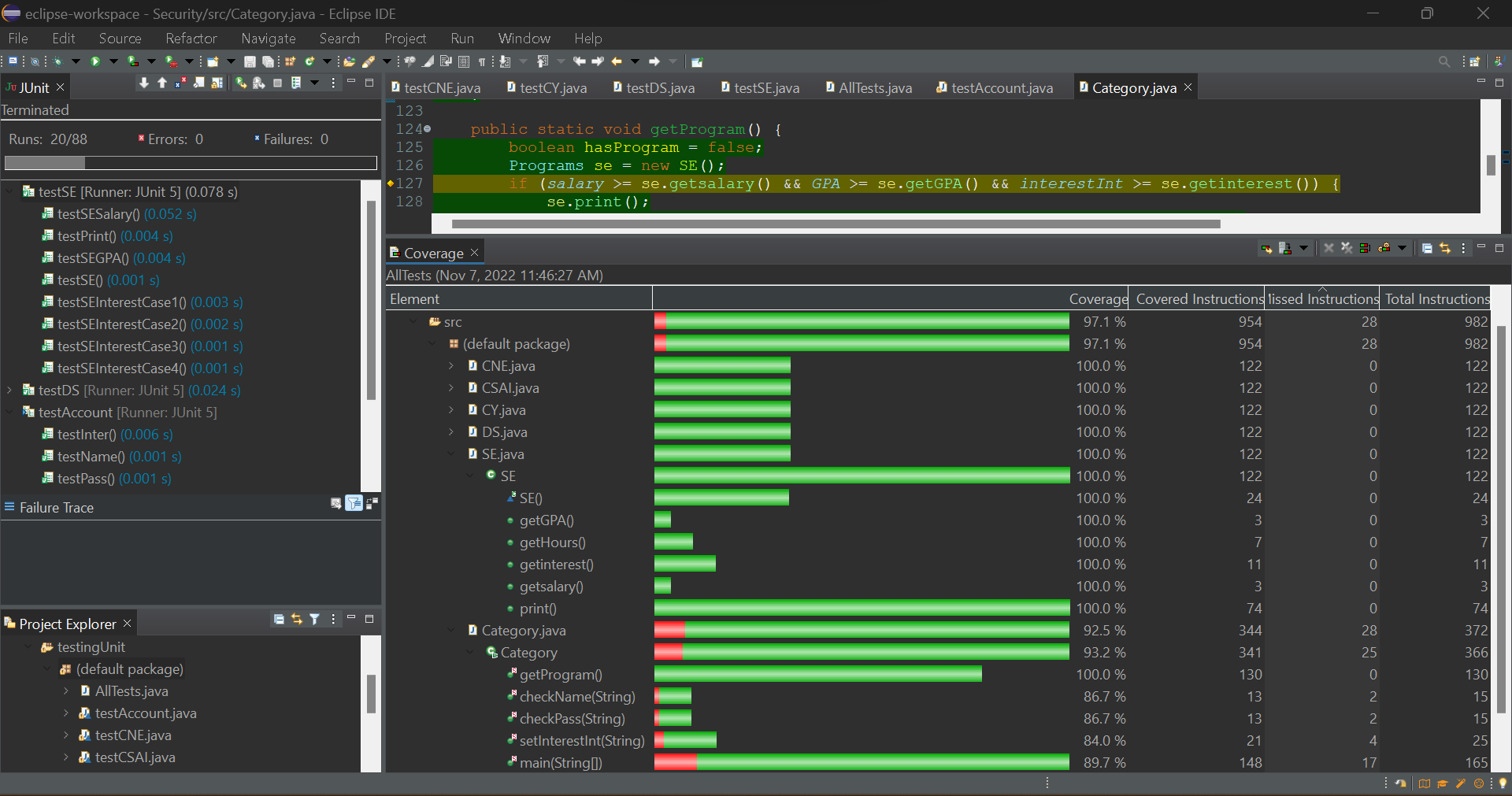
In this test case we are trying to enter an invalid salary, but our system verifies the type of entry so it will not become a system crash.

1. **Junit and EclEmma:**

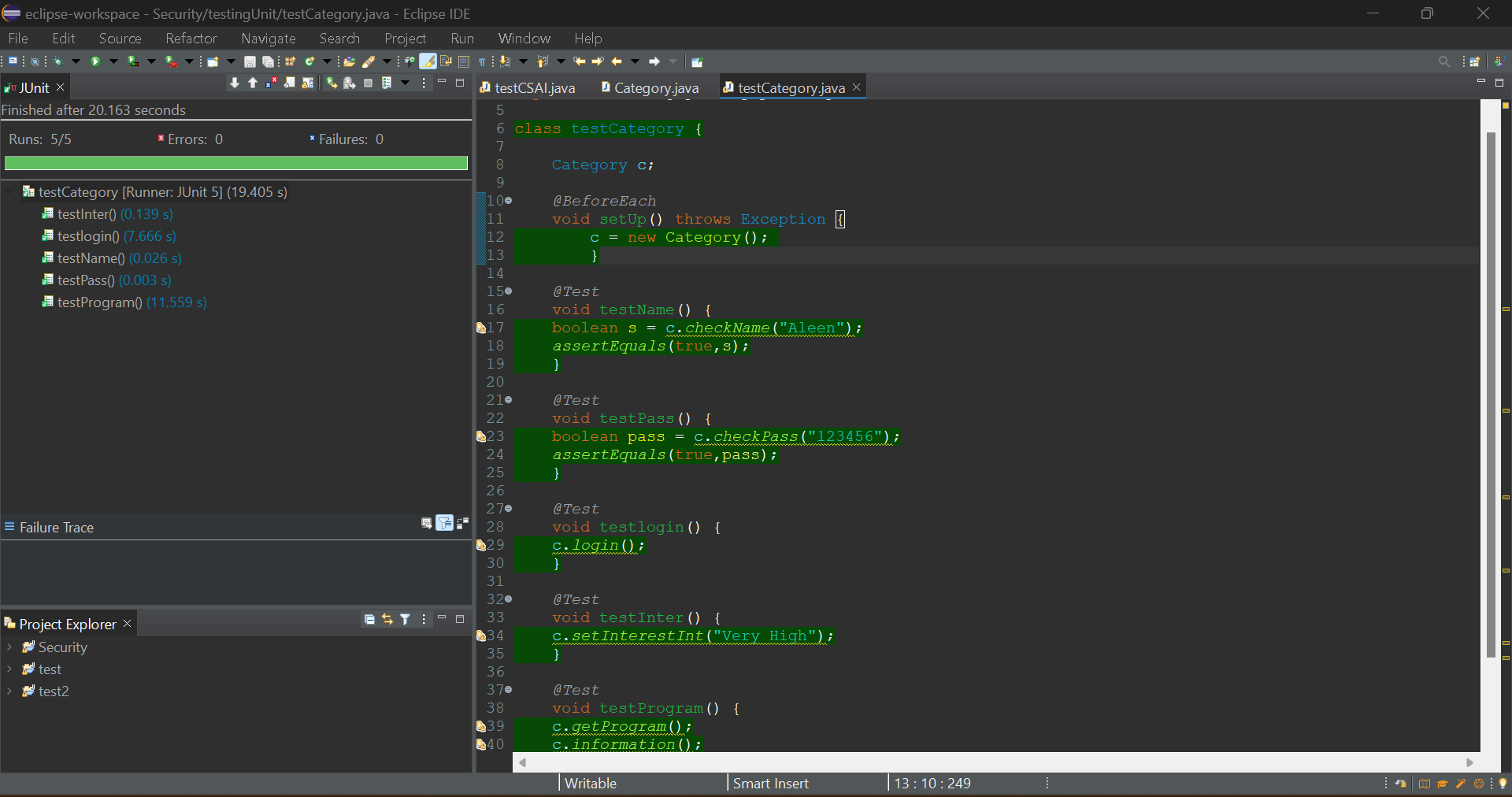
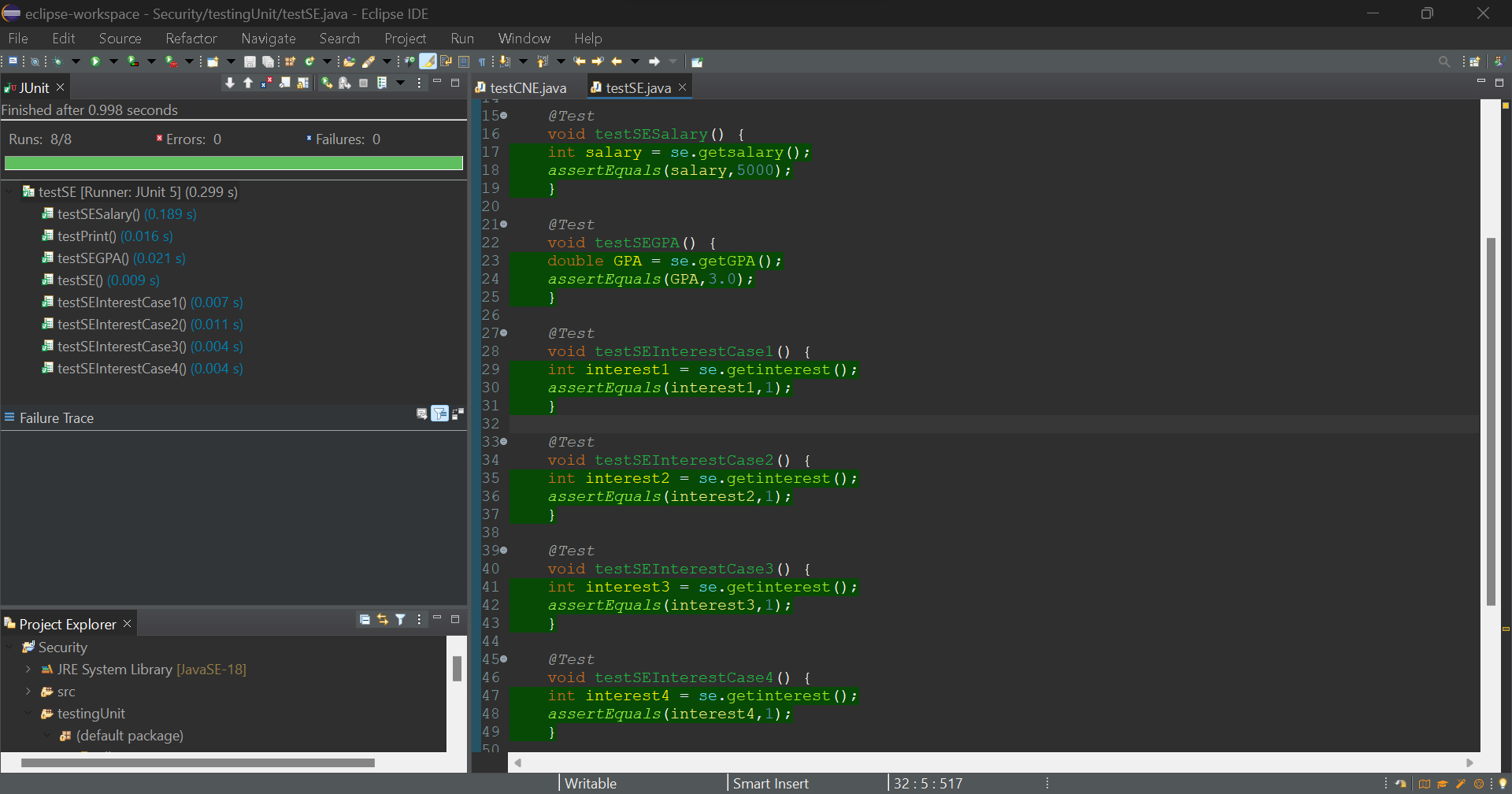
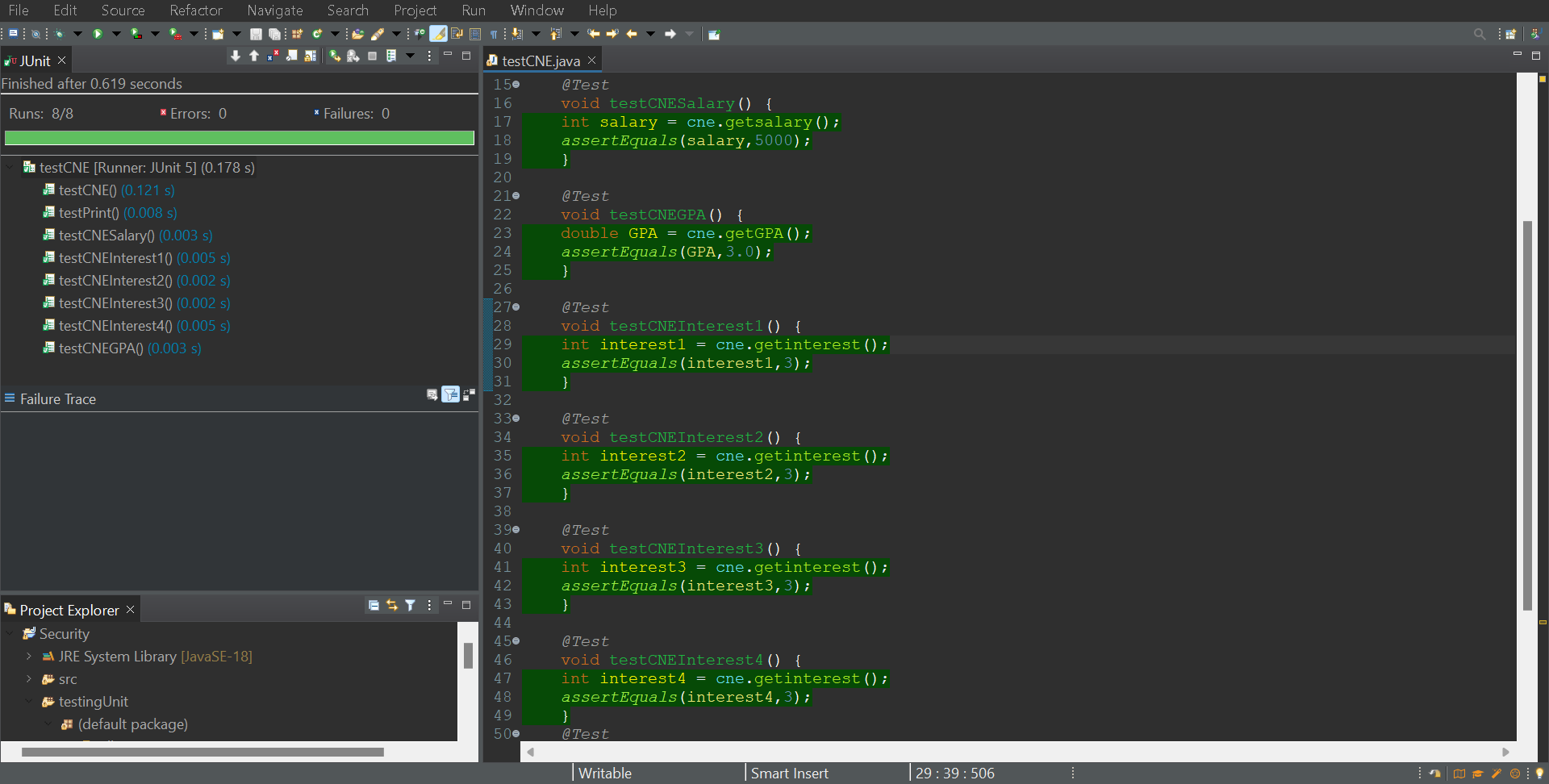
Eclipse EclEmma is a Java code coverage for the Eclipse IDE.

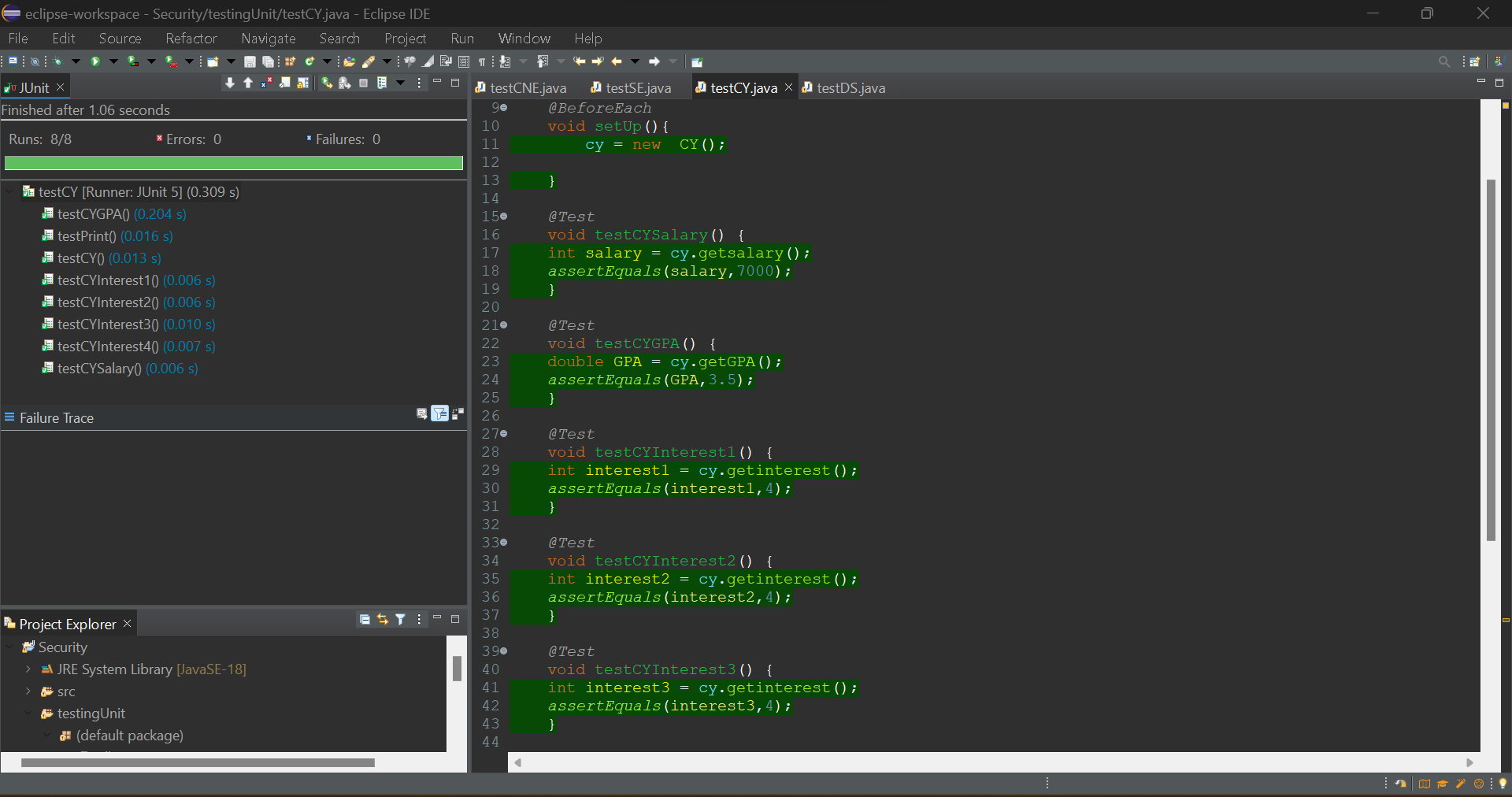
* 1. **Shows the EclEmma coverage Report as follows:**

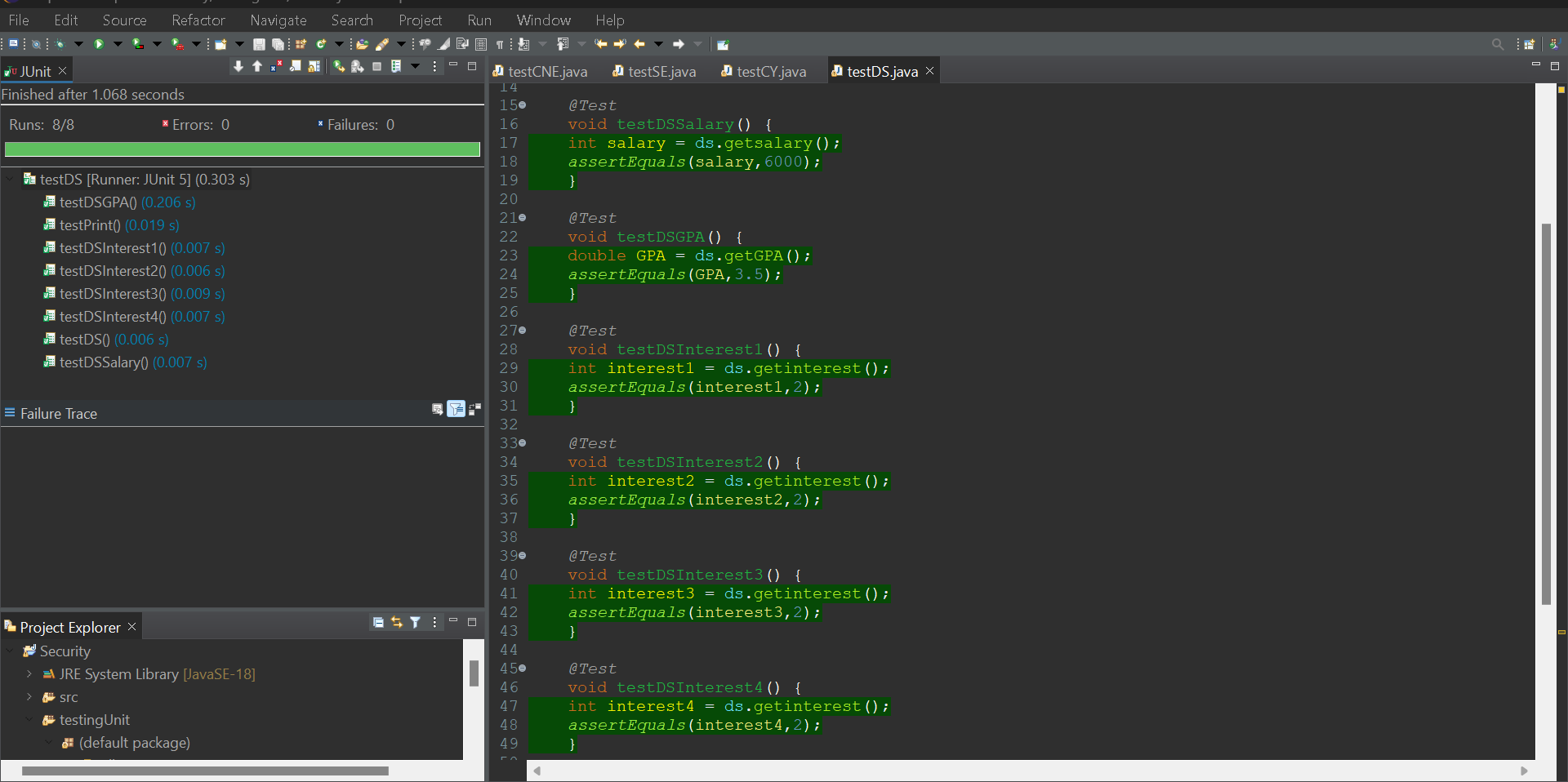
In Junit and EclEmma we can test Coverage with @BeforeEach and @Test annotations.

****

From the image above, we can deduce the coverage of the entire system covers 97.1%, and it varies from one class to another. For example in (CNE,CSAI,SY,DS and SE classes) the coverage rate was 100%, but in the main class (Category class) it 92.5% and the statement coverage rate varies for each functions in the category class.

* 1. **Junit test cases are as follows:**
* In category class we have 5 test cases in Junit testing, and most of test cases have real and expected outputs to compare between them.
* In SE class we have 8 test cases in Junit testing, and most of test cases have real and expected outputs to compare between them.
* In CNE class we have 8 test cases in Junit testing, and most of test cases have real and expected outputs to compare between them.
* In CY class we have 8 test cases in Junit testing, and most of test cases have real and expected outputs to compare between them.



* In DS class we have 8 test cases in Junit testing, and most of test cases have real and expected outputs to compare between them.
* A screenshot of a computer

  Description automatically generatedIn CSAI class we have 8 test cases in Junit testing, and most of test cases have real and expected outputs to compare between them.

**From the images above we see the green indicator bar in the upper-left corner for each Junit testing indicating that the test case has been executed successfully - the code under test passes the test case.**

1. **Test cases developed:**

Based on black box fuzzing testing and test case 6 we found a defect in our system, and we tried this defect in the Junit testing, but the system has become inoperable and will remain available, and this will weaken the system and make it more vulnerability.

**Defect Report:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | | **Label** | **Value** |  |
| Bug ID | | ID number | #100 |  |
|  | | Name | Unable to respond to user input. |  |
|  | | Reporter | Deema |  |
|  | | Submit Date | 8/11/2022 |  |
| Bug overview | | Summary | When our system finishes displaying the suggested programs, the user is given the option to complete the display again by entering a one or exit the system through entering zero, but when the user enters another number, the system will not respond and will remain open until the specified working time ends. |  |
|  | | Screenshot | Text  Description automatically generated |  |
| Bug details | | Expected result | Display an error message or alert. |  |
|  | | Actual result | Nothing happens the system will not respond and will remain open until the specified working time ends. |  |
| Bug tracking | | Severity | Major |  |
|  | | Priority | High |  |
|  |
|  | |  |  |  |