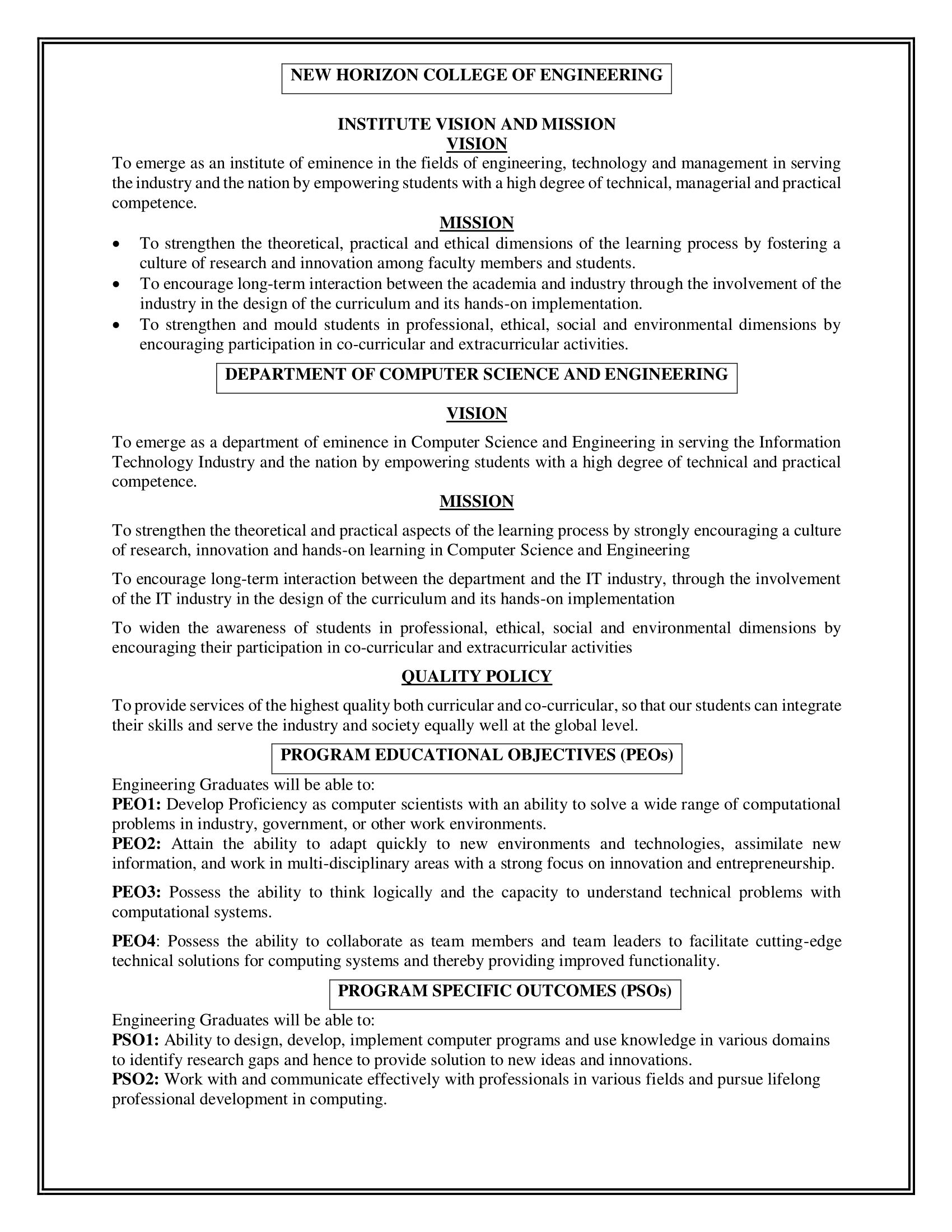
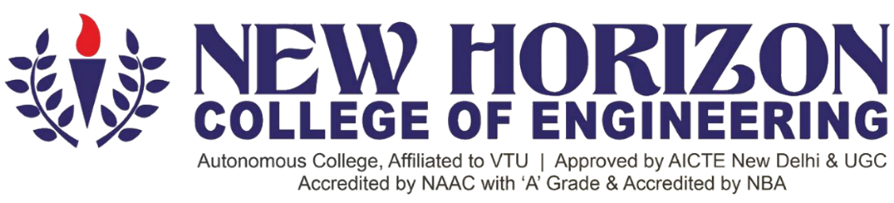




**PRACTICAL RECORD BOOK**

|  |
| --- |
| **Name**  RAJESH G  1NH19CS139  2022 - 2023  **USN Year**  B.E. in CSE  C  7  **Program Semester Section**  SOFTWARE TESTING LAB  20CSL75A  **Course Course Code** |





*Laboratory Certificate*

|  |  |  |
| --- | --- | --- |
| *This is to certify that*  *Mr. ........*Rajesh G*.........*  *has satisfactorily completed the experiments prescribed by*  *New Horizon College of Engineering, Bangalore Affiliated to*  *Visvesvaraya Technological University*  *in ..*Software Testing*.. Laboratory Course for the ..*7th*.. semester of*  *Computer Science and Engineering Program.*  *Academic Year: 2021 to 2022 (ODD Semester)*   |  | | --- | | Marks Obtained | | Max. Marks |   **Student Name:** Rajesh G  **USN:** 1NH19CS139  **Sem/Sec:** 7 - C  **Course Code:** 20CSL75A  Signature of Student  **Head of the Department**  **Signature of the Faculty In-charge** |



**LABORATORY PERFORMANCE EVALUATION SHEET**

**Name of Student: Rajesh G**

**USN:** 1NH19CS139

**Lab Course:** SOFTWARE TESTING LAB

**Course Code:** 20CSL75A

**Sem/Sec:** 7 - C

**Session:** ODD Sem 2022-23

**CIE - PART A - Record and Performance (Max Marks: 10)**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SN** | **Date of Evaluation** | **Name of Experiment/ Program** | **1** | **2** | **3** | **Total** | **Faculty Signature** |
| Write test cases for the following scenarios | | | | | | | |
| 1. | 06-09-2022 | ATM System |  |  |  |  |  |
| 2. | 06-09-2022 | The Triangle Problem |  |  |  |  |  |
| Demonstrate Black box testing techniques using open-source testing tool - JUnit | | | | | | | |
| 3. | 13-09-2022 | Boundary Value Analysis (BVA) for the  NextDate Function |  |  |  |  |  |
| 4. | 20-09-2022 | Equivalence Class Partitioning for the  NextDate Function |  |  |  |  |  |
| Demonstrate White box testing techniques using open-source testing tool - ECLemma | | | | | | | |
| 5. | 27-09-2022 | The Triangle Problem |  |  |  |  |  |
| 6. | 11-10-2022 | The NextDate Function |  |  |  |  |  |
| Demonstration of Selenium IDE & Webdriver for conducting test on websites | | | | | | | |
| 7. | 31-10-2022 | Using Selenium IDE to conduct a test for any web site |  |  |  |  |  |
| 8. | 07-11-2022 | Using Selenium Web driver, automate any web page using Java Script |  |  |  |  |  |
| **SN** | **Date of Evaluation** | **Name of Experiment / Program** | **1** | **2** | **3** | **Total** | **Faculty Signature** |
| 9. | 07-11-2022 | List the total number of objects present on a web page |  |  |  |  |  |
| 10. | 14-11-2022 | Demonstrate URL and title check point |  |  |  |  |  |
| 11. | 14-11-2022 | Demonstrate selecting and deselecting option from multi select dropdown |  |  |  |  |  |
| 12. | 21-11-2022 | Demonstrate Synchronization. |  |  |  |  |  |
| **Average (out of 10 Marks)** | | | | | |  |  |

1. **Conduction of Experiment / Writing the Program: 3 Marks**
2. **Specimen Calculation / Execution: 3 Marks**
3. **Result and Record Writing: 4 Marks**



**CIE - PART B - Lab Test (Max Marks: 50)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Date of Lab Test** | **Procedure and Write Up**  **(15 Marks)** | **Conduction and Results**  **(25 Marks)** | **Viva Voce**  **(10 Marks)** | **Total**  **(50 Marks)** | **Faculty**  **Signature** |
| **Test 1** |  |  |  |  |  |  |
| **Test 2** |  |  |  |  |  |  |
| **Average (out of 15 Marks)** | | | | |  |  |

**CIE - Marks Obtained**

|  |  |  |  |
| --- | --- | --- | --- |
| **CIE-Part A**  **Record and Performance**  **(10 Marks)** | **CIE-Part B**  **Lab Test**  **(15 Marks)** | **Total**  **(25 Marks)** | **Faculty Signature** |
|  |  |  |  |

**PROGRAM NO.: 1**

**Exp. No. :** 1

**Date :** 08-10-2021

**ATM SYSTEM**

Consider any ATM system, design and develop a program in a language of your choice for the same. Create the test cases for the following scenarios:

1. Unsuccessful operation due to enter wrong PIN number 3 times.
2. Unsuccessful operation due to invalid account type.
3. Successful selection of amount to be withdrawn.
4. Expected message due to amount to withdraw is greater than possible balance

**IMPLEMENTATION:**

import java.util.\*;

import java.lang.\*;

public class ATMmain {

private static int pin = 7563, lim = 0, upin, accountType, balance=5000, operation, wAmt, dAmt;

public static void main(String args[]){

Scanner in = new Scanner(System.in);

System.out.println("Welcome to the ICICI Bank ATM!");

System.out.println("Please insert your ATM card.");

while(lim<=3){

System.out.println("Please enter your PIN.");

upin = in.nextInt();

if (upin == pin){

while(true){

System.out.println("Select your account type.\n1. Savings\n2.

Current");

accountType = in.nextInt();

if (accountType == 1){

System.out.println("You have selected Savings Account.");

break;

}

else if(accountType == 2){

System.out.println("You have selected Current Account.");

break;

}

else{

System.out.println("You have entered an invalid input. Try

again.");

}

}

while(true){

System.out.println("Select an operation:\n1. Check Balance\n2.

Withdrawal\n3. Deposit\n4. Exit");

operation = in.nextInt();

if (operation == 1){

System.out.println("Your current balance is: "+balance);

}

else if(operation == 2){

System.out.println("Enter amount to be withdrawn: ");

wAmt = in.nextInt();

if(wAmt > balance){

System.out.println("The amount to be withdrawn is

greater than current balance.");

}

else if (wAmt > 0){

balance -= wAmt;

System.out.println("Please collect your cash. Thank

you.");

}

else{

System.out.println("Please enter an amount greater

than zero.");

}

}

else if(operation == 3){

System.out.println("Enter amount to be deposited: ");

dAmt = in.nextInt();

if (dAmt > 0){

balance += dAmt;

System.out.println("Thank you.");

}

else{

System.out.println("Please enter an amount greater

than zero.");

}

}

else if(operation == 4){

System.out.println("Thank you for banking with ICICI

Bank. Have a nice day!");

System.exit(0);

break;

}

else{

System.out.println("You have entered an invalid input. Try

again.");

}

}

}

else{

lim++;

if(lim==1){

System.out.println("Incorrect PIN. You have 2 more chances after

which your card will be blocked.");

}

if(lim==2){

System.out.println("Incorrect PIN. You have 1 more chance after

which your card will be blocked.");

}

if(lim==3){

System.out.println("Incorrect PIN. Your card has been blocked.");

System.out.println("Please take out your card. Thank you.");

System.exit(0);

}

}

}

}

}

**TEST CASES:**

**TEST** **CASE 1:** Validity of ATM PIN entered by the user

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | | Invalid PIN Number | |
| Project ID: | ATM\_01 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality with invalid pin number | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Insert valid card in the insertion point of ATM | Valid ATM card | | ATM displays the PIN number entry screen | ATM should display the PIN number entry screen | | Pass |
|  | Enter incorrect PIN  (1st Attempt) | Invalid PIN | | ATM does not validate PIN and prompts customer to reenter PIN. | ATM should not validate PIN and prompts customer to reenter PIN. | | Pass |
|  | Reenter incorrect PIN  (2nd Attempt) | Invalid PIN | | ATM does not validate PIN and prompts customer to reenter PIN | ATM should not validate PIN and prompts customer to reenter PIN | | Pass |
|  | Reenter incorrect PIN  (3rd Attempt) | Invalid PIN | | ATM does not validate PIN and blocks the card | ATM should not validate PIN and blocks the card | | Pass |
|  | Enter Correct PIN | Valid PIN | | ATM validates the PIN and displays the Account type selection module | ATM should validate the PIN and display the Account type selection module | | Pass |

**TEST CASE 2:** Validity of Account type selection choice

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | | Invalid Account Type | |
| Project ID: | ATM\_02 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality with invalid account type | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Check working of Account type selection module | Use entering a valid PIN | | The module displays the two Account types: Savings and Current | The module should display the two Account types: Savings and Current | | Pass |
|  | Check if valid input for Savings Account type works | User selects Savings Account by entering ‘1’ | | A message saying “You have selected Savings Account.” appears. | A message saying “You have selected Savings Account.” should appear. | | Pass |
|  | Check if valid input for Current Account type works | User selects Current Account by entering ‘2’ | | A message saying “You have selected Current Account.” appears. | A message saying “You have selected Current Account.” should appear. | | Pass |
|  | Check if invalid inputs are handled | User enters an invalid option | | A message saying “You have entered an invalid input. Try again.” appears. | A message saying “You have entered an invalid input. Try again.” should appear. | | Pass |
|  | Check working of module for valid input | User enters either ‘1’ or ‘2’ | | The ATM navigates the user to the Operation selection module. | The ATM should navigate the user to the Operation selection module. | | Pass |

**TEST CASE 3:** Successful selection of amount to be withdrawn.

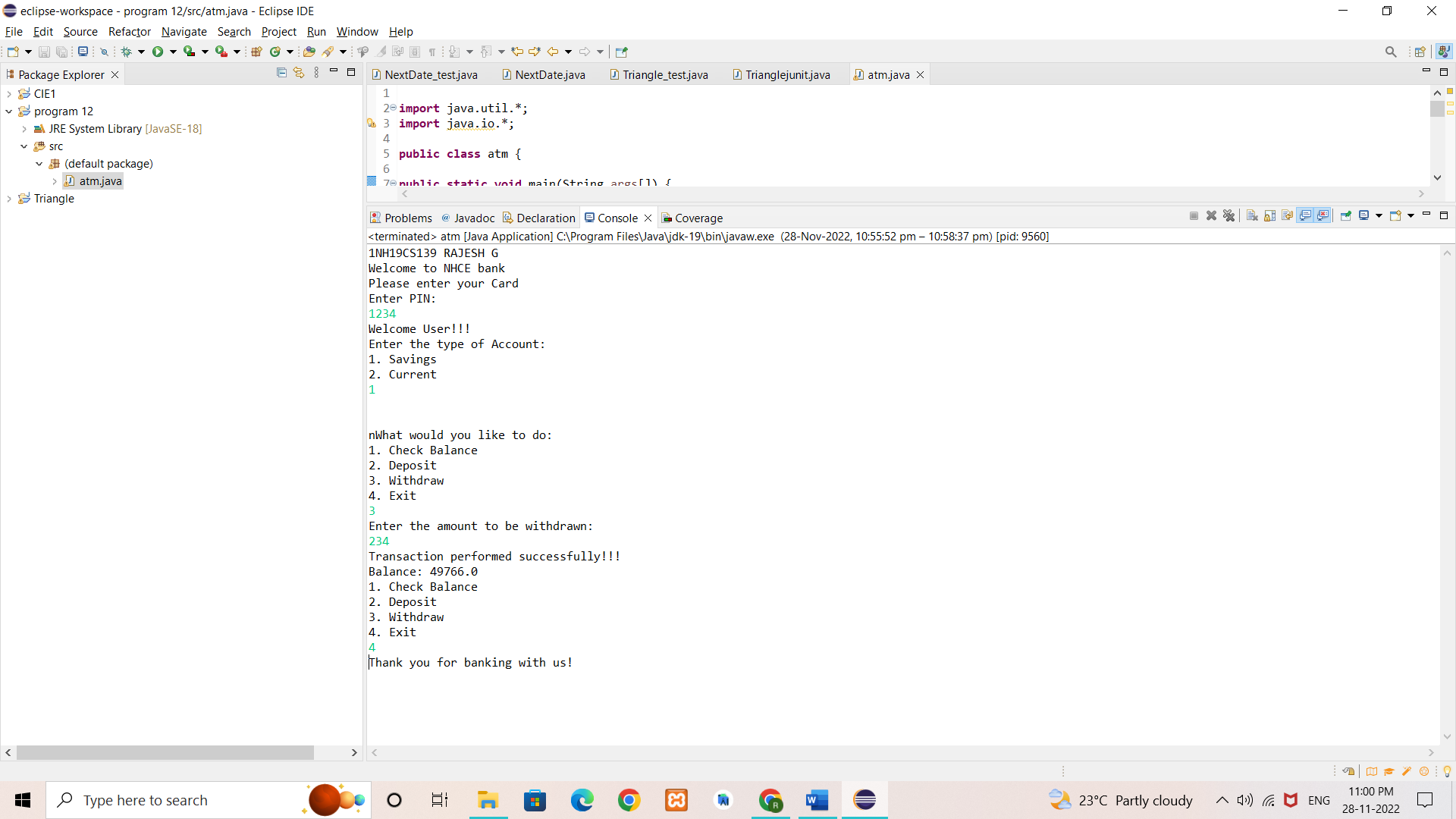
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | | Valid Withdrawal Amount | |
| Project ID: | ATM\_03 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality of the withdrawal operation using valid inputs | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Check working of Withdrawal module | User enters ‘2’ in Operation selection module | | A message saying ‘Enter amount to be withdrawn:’ appears. | A message saying ‘Enter amount to be withdrawn:’ should appear. | | Pass |
|  | Check for valid withdrawal amount | User enters ‘500’ as withdrawal amount | | A message saying ‘Please collect your cash. Thank you.’ appears. | A message saying ‘Please collect your cash. Thank you.’ should appear. | | Pass |
|  | Check balance for previous Test Case | User enters ‘1’ in Operation selection module | | A message saying ‘Your current balance is: 4500’ appears. | A message saying ‘Your current balance is: 4500’ should appear. | | Pass |
|  | Check for valid withdrawal amount | User enters ‘1000’ as withdrawal amount | | A message saying ‘Please collect your cash. Thank you.’ appears. | A message saying ‘Please collect your cash. Thank you.’ should appear. | | Pass |
|  | Check balance for previous Test Case | User enters ‘1’ in Operation selection module | | A message saying ‘Your current balance is: 3500’ appears. | A message saying ‘Your current balance is: 3500’ should appear. | | Pass |

**TEST CASE 4:** Expected message due to amount to withdraw is greater than possible balance or is a negative amount.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | | Invalid Withdrawal Amount | |
| Project ID: | ATM\_04 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality of the withdrawal operation using invalid inputs | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Check for invalid withdrawal amount greater than balance | User enters ‘6000’ as withdrawal amount | | A message saying ‘The amount to be withdrawn is greater than current balance.’ appears. | A message saying ‘The amount to be withdrawn is greater than current balance.’ should appear. | | Pass |
|  | Check for invalid negative withdrawal amount | User enters ‘-500’ as withdrawal amount | | A message saying ‘Please enter an amount greater than zero.’ appears. | A message saying ‘Please enter an amount greater than zero.’ should appear. | | Pass |
|  | Check for invalid withdrawal amount greater than balance | User enters ‘7000’ as withdrawal amount | | A message saying ‘The amount to be withdrawn is greater than current balance.’ appears. | A message saying ‘The amount to be withdrawn is greater than current balance.’ should appear. | | Pass |
|  | Check for invalid negative withdrawal amount | User enters ‘-100’ as withdrawal amount | | A message saying ‘Please enter an amount greater than zero.’ appears. | A message saying ‘Please enter an amount greater than zero.’ should appear. | | Pass |
|  | Check for invalid withdrawal amount greater than balance | User enters ‘8000’ as withdrawal amount | | A message saying ‘The amount to be withdrawn is greater than current balance.’ appears. | A message saying ‘The amount to be withdrawn is greater than current balance.’ should appear. | | Pass |

**EXECUTION**

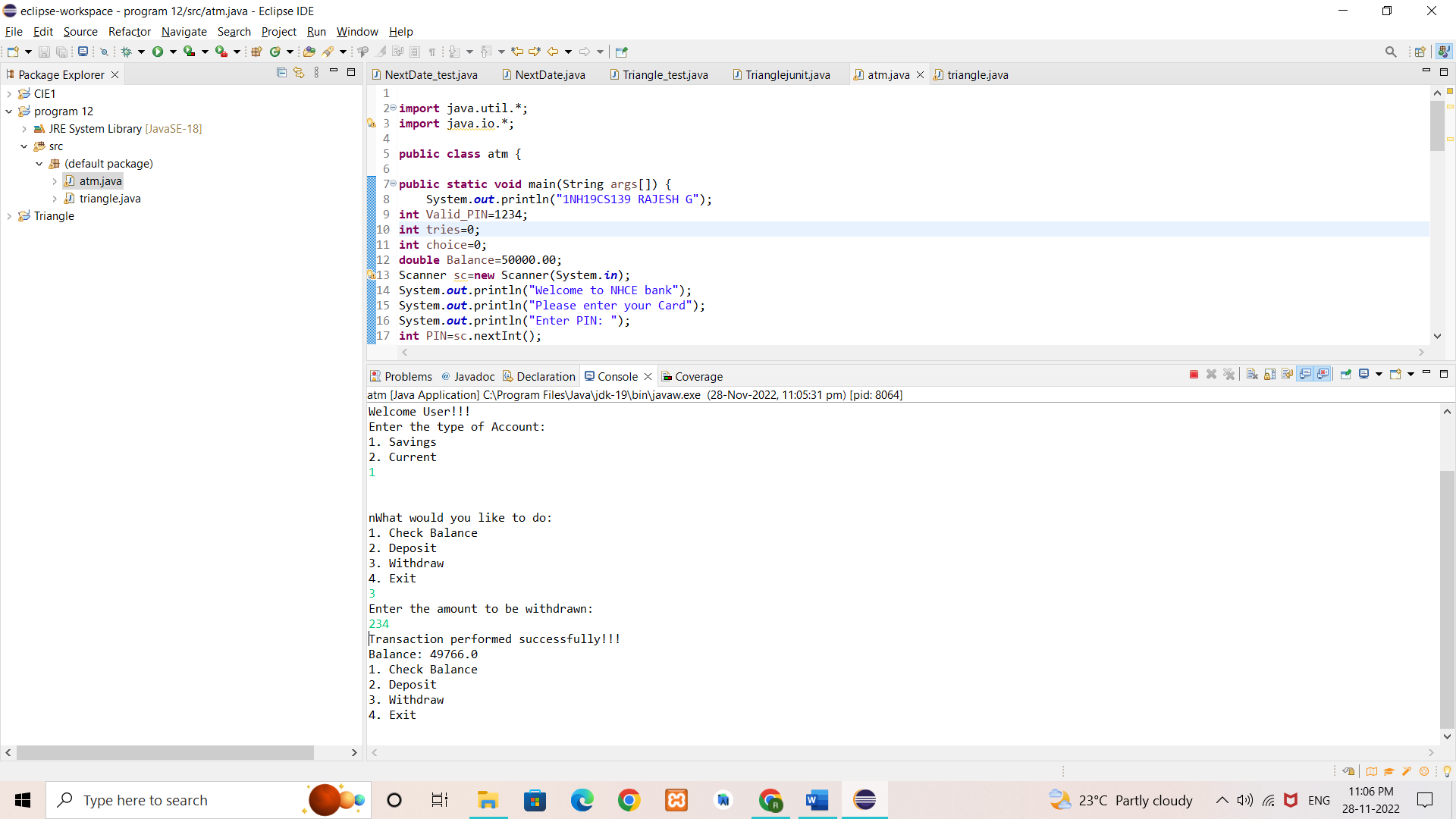
**1) Invalid ATM PIN**

****

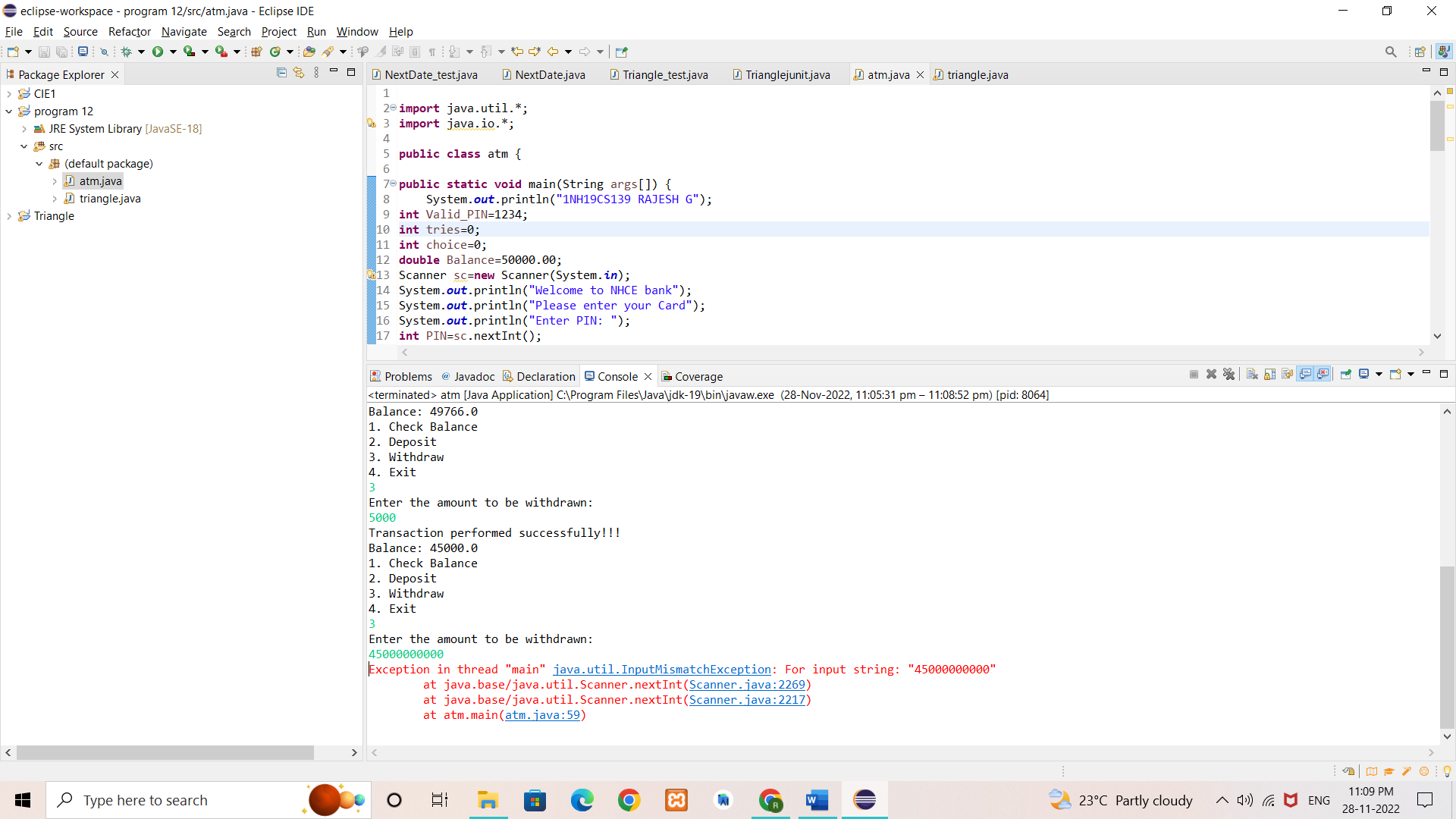
**2) Account Type**

****

**3) Valid Withdrawal Amount**

****

**4) Invalid Withdrawal Amount**

****

**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed : 20

2. Number of Test Cases Passed : 20

3. Number of Test Cases Failed : 0

**Exp. No. :** 2

**Date :** 22-10-2021

**TRIANGLE PROBLEM**

Design and develop a program in a language of your choice to solve the triangle problem defined as follows: Accept three integers which are supposed to be the three sides of triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all. Create the test cases for the following scenarios:

1. Represents not a triangle
2. Represents a valid scalene triangle
3. Represents a valid equilateral triangle
4. Represents a valid isosceles triangle

Execute the test cases manually and discuss the result.

**IMPLEMENTATION**

import java.util.\*;

public class triangle {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

int a, b, c;

while(true){

System.out.print("Enter value of 1st side: ");

a = in.nextInt();

System.out.print("Enter value of 2nd side: ");

b = in.nextInt();

System.out.print("Enter value of 3rd side: ");

c = in.nextInt();

if(a>=1 && a<=200 && b>=1 && b<=200 && c>=1 && c<=200){

if((a < b+c) && (b < a+c) && (c < b+a)){

if(a == b && b == c)

System.out.println("Given dimensions form an equilateral

triangle!");

else if(a==b || b==c || c==a)

System.out.println("Given dimensions form an isosceles

triangle!");

else

System.out.println("Given dimensions form a scalene

triangle!");

break;

}

else {

System.out.println("Given dimensions do not form a triangle!");

break;

}

}

else {

System.out.println("Enter a valid input!\n");

}

}

}

}

**TEST CASES**

**TEST CASE 1:** Represents not a triangle

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | Triangle | | Test Name: | | | Not a Triangle | |
| Project ID: | Triangle\_01 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality using inputs that do not form a triangle or are invalid | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Enter invalid inputs | Side a = -10  Side b = 20  Side c = -50 | | A message saying ‘Enter a valid input!’ appears. | A message saying ‘Enter a valid input!’ should appear. | | Pass |
|  | Enter invalid inputs | Side a = -10  Side b = 20  Side c = 100 | | A message saying ‘Enter a valid input!’ appears. | A message saying ‘Enter a valid input!’ should appear. | | Pass |
|  | Enter valid inputs that do not form a triangle | Side a = 10  Side b = 10  Side c = 30 | | A message saying ‘Given dimensions do not form a triangle!’ appears. | A message saying ‘Given dimensions do not form a triangle!’ should appear. | | Pass |
|  | Enter valid inputs that do not form a triangle | Side a = 199  Side b = 1  Side c = 200 | | A message saying ‘Given dimensions do not form a triangle!’ appears. | A message saying ‘Given dimensions do not form a triangle!’ should appear. | | Pass |
|  | Enter valid inputs that do not form a triangle | Side a = 3  Side b = 2  Side c = 7 | | A message saying ‘Given dimensions do not form a triangle!’ appears. | A message saying ‘Given dimensions do not form a triangle!’ should appear. | | Pass |

**TEST CASE 2:** Represents a valid scalene triangle

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | Triangle | | Test Name: | | | Scalene Triangle | |
| Project ID: | Triangle\_02 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality using inputs that form a scalene triangle | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Enter valid inputs | Side a = 10  Side b = 20  Side c = 15 | | A message saying ‘Given dimensions form a scalene triangle!’ appears. | A message saying ‘Given dimensions form a scalene triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 150  Side b = 140  Side c = 130 | | A message saying ‘Given dimensions form a scalene triangle!’ appears. | A message saying ‘Given dimensions form a scalene triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 90  Side b = 91  Side c = 92 | | A message saying ‘Given dimensions form a scalene triangle!’ appears. | A message saying ‘Given dimensions form a scalene triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 199  Side b = 198  Side c = 200 | | A message saying ‘Given dimensions form a scalene triangle!’ appears. | A message saying ‘Given dimensions form a scalene triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 3  Side b = 5  Side c = 7 | | A message saying ‘Given dimensions form a scalene triangle!’ appears. | A message saying ‘Given dimensions form a scalene triangle!’ should appear. | | Pass |

**TEST CASE 3:** Represents a valid equilateral triangle

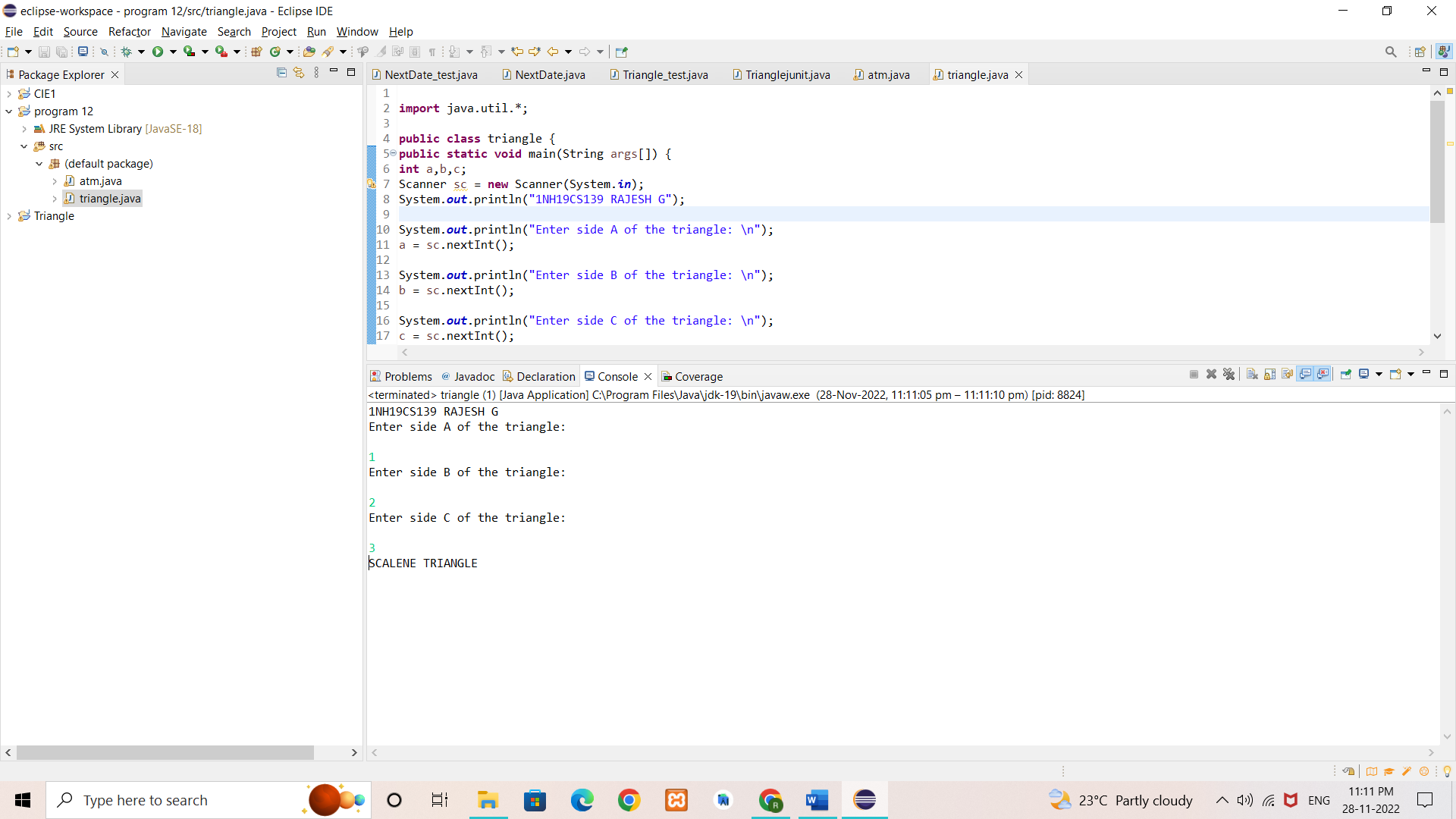
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | Triangle | | Test Name: | | | Equilateral Triangle | |
| Project ID: | Triangle\_03 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality using inputs that form an equilateral triangle | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Enter valid inputs | Side a = 10  Side b = 10  Side c = 10 | | A message saying ‘Given dimensions form an equilateral triangle!’ appears. | A message saying ‘Given dimensions form an equilateral triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 100  Side b = 100  Side c = 100 | | A message saying ‘Given dimensions form an equilateral triangle!’ appears. | A message saying ‘Given dimensions form an equilateral triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 200  Side b = 200  Side c = 200 | | A message saying ‘Given dimensions form an equilateral triangle!’ appears. | A message saying ‘Given dimensions form an equilateral triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 1  Side b = 1  Side c = 1 | | A message saying ‘Given dimensions form an equilateral triangle!’ appears. | A message saying ‘Given dimensions form an equilateral triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 75  Side b = 75  Side c = 75 | | A message saying ‘Given dimensions form an equilateral triangle!’ appears. | A message saying ‘Given dimensions form an equilateral triangle!’ should appear. | | Pass |

**TEST CASE 4:** Represents a valid isosceles triangle

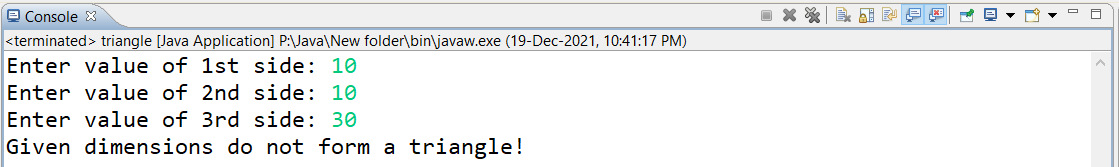
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | Triangle | | Test Name: | | | Isosceles Triangle | |
| Project ID: | Triangle\_03 | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | This test case is to verify the functionality using inputs that form an isosceles triangle | | | | | | |
| **Case No.** | **Test Case Description** | **Test Data** | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
|  | Enter valid inputs | Side a = 10  Side b = 20  Side c = 20 | | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 50  Side b = 60  Side c = 50 | | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 200  Side b = 100  Side c = 200 | | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 199  Side b = 199  Side c = 200 | | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |
|  | Enter valid inputs | Side a = 1  Side b = 2  Side c = 2 | | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |

**EXECUTION**

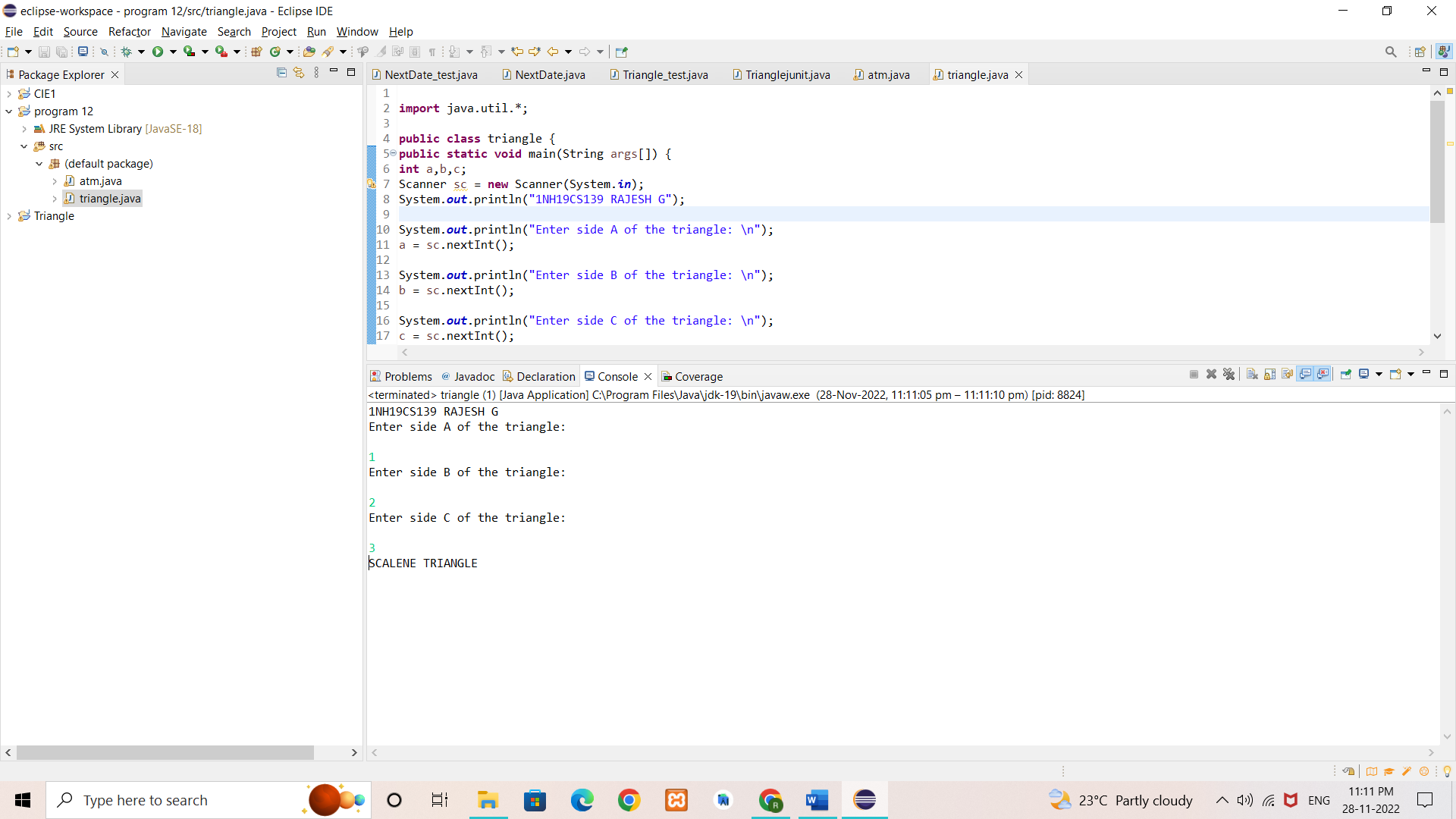
**1) Invalid case: Invalid input**

****

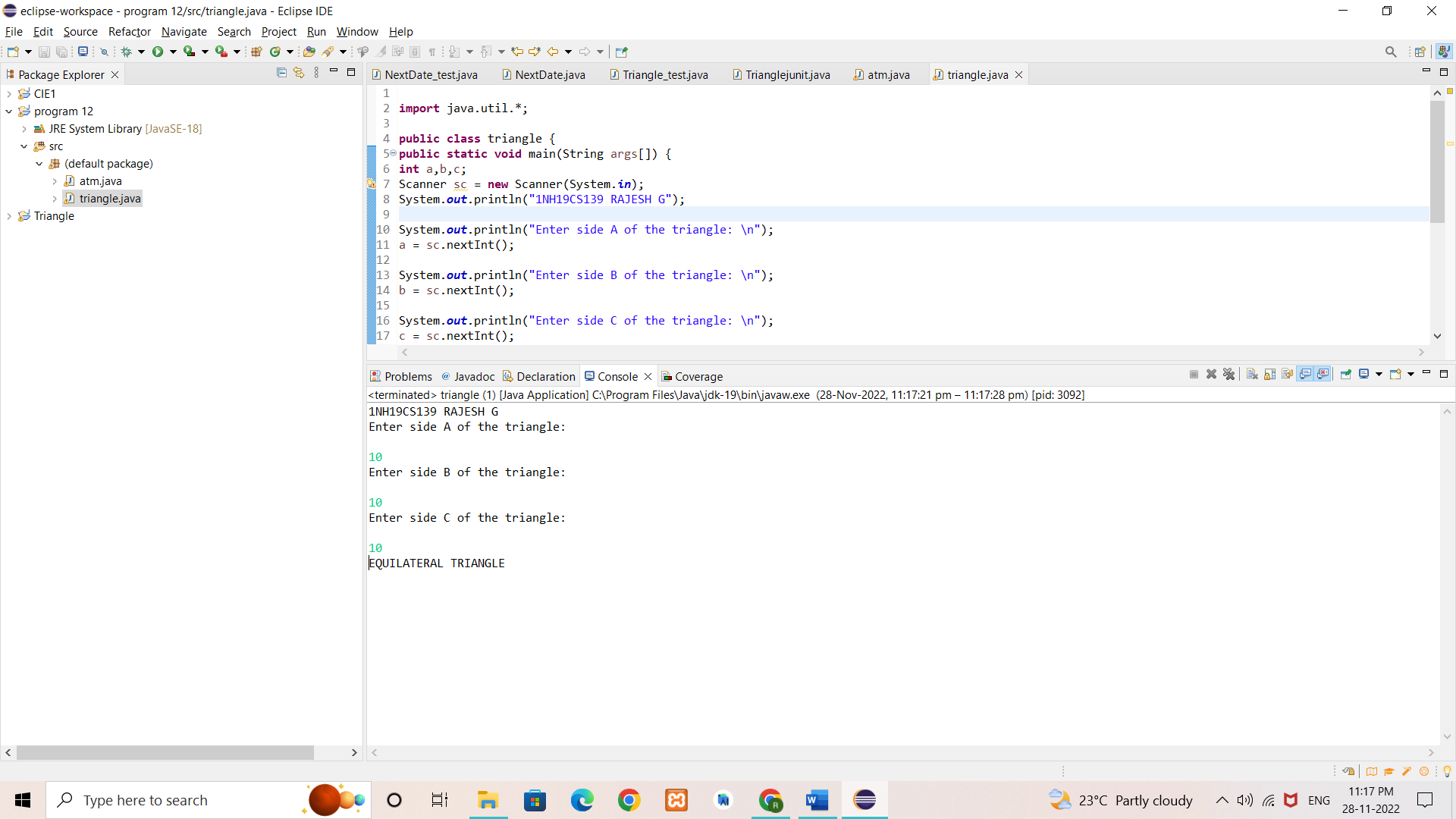
**2) Invalid case: Not a Triangle**

****

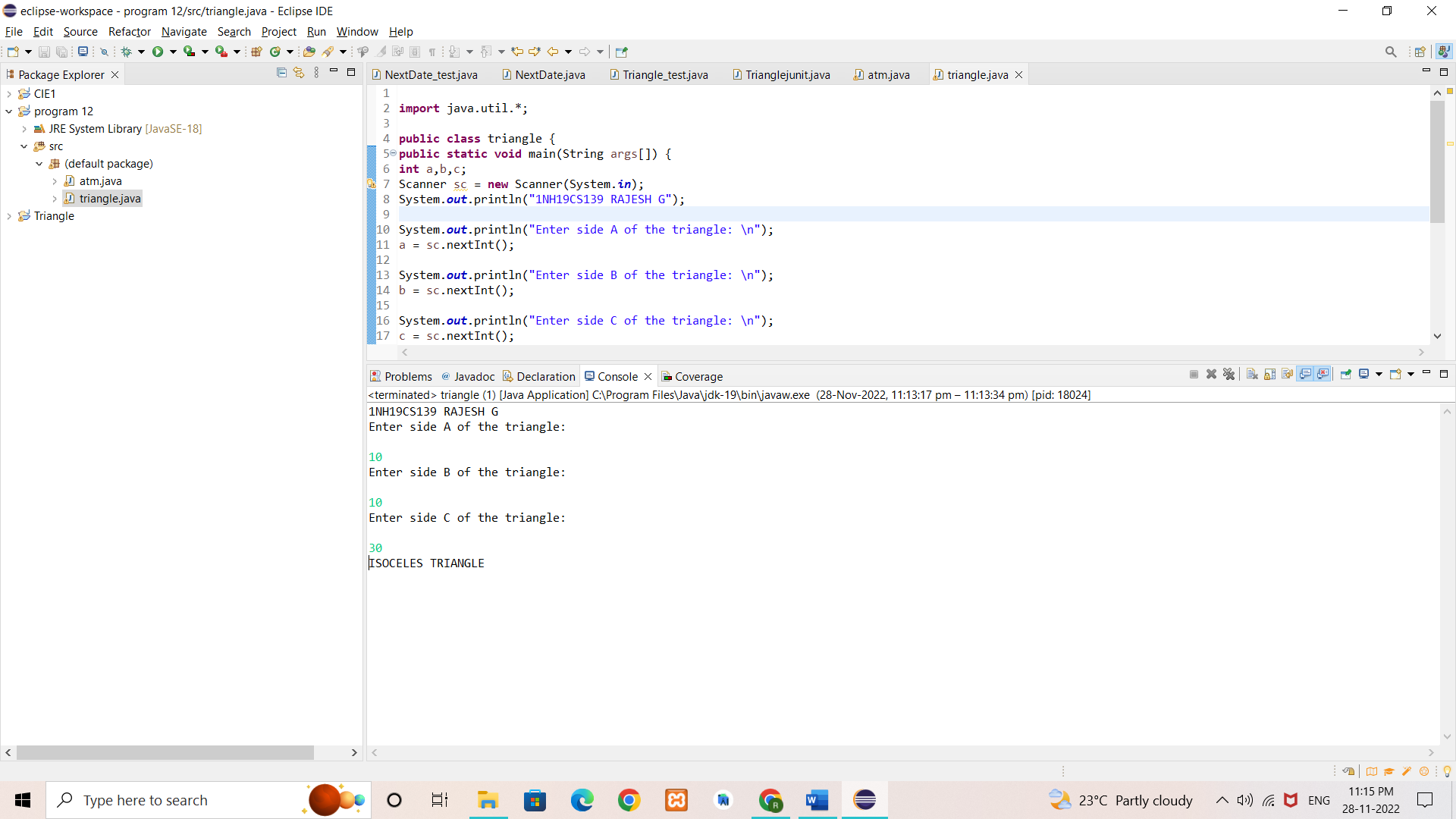
**3) Scalene Triangle**

****

**4) Equilateral Triangle**

****

**5) Isosceles Triangle**

****

**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed : 20

2. Number of Test Cases Passed : 20

3. Number of Test Cases Failed : 0

**Exp. No. :** 3

**Date :** 29-10-2021

**BOUNDARY VALUE ANALYSIS (BVA) FOR NEXTDATE FUNCTION**

Design, develop, code and run the program in any suitable language to implement the NextDate function. Analyse it from the perspective boundary value testing. Create different test cases based on the following variants, execute the test cases by using Junit and discuss the test results.

1. Normal Boundary Value Testing
2. Robust Boundary Value Testing
3. Worst-Case Boundary Value Testing
4. Robust Worst-Case Boundary Value Testing

**IMPLEMENTATION**

**1. NextDate function**

import java.util.\*;

public class date {

public String nextDate (int d, int m, int y) {

int nd, nm, ny;

if(d>31 || d<1 || m>12 || m <1 || y<1821 || y>2021){

return ("Invalid date!");

}

else if(m==2 || m==4 || m==6 || m==9 || m==11 ){

if(d==31)

return("Invalid date!");

else if(m==2){

if(checkLeapYear(y)){

if(d>29){

return("Invalid date!");

}

if(d == 29) {

nd = 1;

nm = 3;

}

else {

nd = ++d;

nm = 2;

}

}

else {

if(d>28){

return("Invalid date!");

}

if(d == 28) {

nd = 1;

nm = 3;

}

else {

nd = ++d;

nm = 2;

}

}

ny = y;

}

else {

if (d == 30){

nd = 1;

nm = ++m;

}

else{

nd = ++d;

nm = m;

}

ny = y;

}

}

else {

if (d == 31 && m != 12){

nd = 1;

nm = ++m;

ny = y;

}

else if (d == 31 && m == 12){

nd = 1;

nm = 1;

ny = ++y;

}

else{

nd = ++d;

nm = m;

ny = y;

}

}

return("The next date is: "+nd+"-"+nm+"-"+ny);

}

public static boolean checkLeapYear(int year){

if(year % 400 == 0)

return true;

else if(year % 100 == 0)

return false;

else if(year % 4 == 0)

return true;

else

return false;

}

}

**2. Normal BVA - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class NormalBVT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 1821), "The next date is: 16-6-1821");

}

@Test

public void test2() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 1822), "The next date is: 16-6-1822");

}

@Test

public void test3() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 1921), "The next date is: 16-6-1921");

}

@Test

public void test4() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2020), "The next date is: 16-6-2020");

}

@Test

public void test5() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2021), "The next date is: 16-6-2021");

}

@Test

public void test6() {

date d1 = new date();

assertEquals(d1.nextDate(15, 1, 1921), "The next date is: 16-1-1921");

}

@Test

public void test7() {

date d1 = new date();

assertEquals(d1.nextDate(15, 2, 1921), "The next date is: 16-2-1921");

}

@Test

public void test8() {

date d1 = new date();

assertEquals(d1.nextDate(15, 11, 1921), "The next date is: 16-11-1921");

}

@Test

public void test9() {

date d1 = new date();

assertEquals(d1.nextDate(15, 12, 1921), "The next date is: 16-12-1921");

}

@Test

public void test10() {

date d1 = new date();

assertEquals(d1.nextDate(1, 6, 1921), "The next date is: 2-6-1921");

}

@Test

public void test11() {

date d1 = new date();

assertEquals(d1.nextDate(2, 6, 1921), "The next date is: 3-6-1921");

}

@Test

public void test12() {

date d1 = new date();

assertEquals(d1.nextDate(29, 6, 1921), "The next date is: 30-6-1921");

}

@Test

public void test13() {

date d1 = new date();

assertEquals(d1.nextDate(30, 6, 1921), "The next date is: 1-7-1921");

}

}

**3. Worst Case BVA - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class WC\_BVT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 1821), "The next date is: 2-1-1821");

}

@Test

public void test2() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 1822), "The next date is: 2-1-1822");

}

@Test

public void test3() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 1921), "The next date is: 2-1-1921");

}

@Test

public void test4() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 2020), "The next date is: 2-1-2020");

}

@Test

public void test5() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 2021), "The next date is: 2-1-2021");

}

@Test

public void test6() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 1821), "The next date is: 3-1-1821");

}

@Test

public void test7() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 1822), "The next date is: 3-1-1822");

}

@Test

public void test8() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 1921), "The next date is: 3-1-1921");

}

@Test

public void test9() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 2020), "The next date is: 3-1-2020");

}

@Test

public void test10() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 2021), "The next date is: 3-1-2021");

}

@Test

public void test11() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 1821), "The next date is: 7-1-1821");

}

@Test

public void test12() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 1822), "The next date is: 7-1-1822");

}

@Test

public void test13() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 1921), "The next date is: 7-1-1921");

}

@Test

public void test14() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 2020), "The next date is: 7-1-2020");

}

@Test

public void test15() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 2021), "The next date is: 7-1-2021");

}

@Test

public void test16() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 1821), "The next date is: 31-1-1821");

}

@Test

public void test17() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 1822), "The next date is: 31-1-1822");

}

@Test

public void test18() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 1921), "The next date is: 31-1-1921");

}

@Test

public void test19() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 2020), "The next date is: 31-1-2020");

}

@Test

public void test20() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 2021), "The next date is: 31-1-2021");

}

@Test

public void test21() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 1821), "The next date is: 1-2-1821");

}

@Test

public void test22() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 1822), "The next date is: 1-2-1822");

}

@Test

public void test23() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 1921), "The next date is: 1-2-1921");

}

@Test

public void test24() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 2020), "The next date is: 1-2-2020");

}

@Test

public void test25() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 2021), "The next date is: 1-2-2021");

}

}

**4. Robust BVA - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class RobustBVT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 1820), "Invalid date!");

}

@Test

public void test2() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 1821), "The next date is: 16-6-1821");

}

@Test

public void test3() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 1822), "The next date is: 16-6-1822");

}

@Test

public void test4() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 1921), "The next date is: 16-6-1921");

}

@Test

public void test5() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2020), "The next date is: 16-6-2020");

}

@Test

public void test6() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2021), "The next date is: 16-6-2021");

}

@Test

public void test7() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2022), "Invalid date!");

}

@Test

public void test8() {

date d1 = new date();

assertEquals(d1.nextDate(15, 0, 1921), "Invalid date!");

}

@Test

public void test9() {

date d1 = new date();

assertEquals(d1.nextDate(15, 1, 1921), "The next date is: 16-1-1921");

}

@Test

public void test10() {

date d1 = new date();

assertEquals(d1.nextDate(15, 2, 1921), "The next date is: 16-2-1921");

}

@Test

public void test11() {

date d1 = new date();

assertEquals(d1.nextDate(15, 11, 1921), "The next date is: 16-11-1921");

}

@Test

public void test12() {

date d1 = new date();

assertEquals(d1.nextDate(15, 12, 1921), "The next date is: 16-12-1921");

}

@Test

public void test13() {

date d1 = new date();

assertEquals(d1.nextDate(15, 13, 1921), "Invalid date!");

}

@Test

public void test14() {

date d1 = new date();

assertEquals(d1.nextDate(0, 6, 1921), "Invalid date!");

}

@Test

public void test15() {

date d1 = new date();

assertEquals(d1.nextDate(1, 6, 1921), "The next date is: 2-6-1921");

}

@Test

public void test16() {

date d1 = new date();

assertEquals(d1.nextDate(2, 6, 1921), "The next date is: 3-6-1921");

}

@Test

public void test17() {

date d1 = new date();

assertEquals(d1.nextDate(29, 6, 1921), "The next date is: 30-6-1921");

}

@Test

public void test18() {

date d1 = new date();

assertEquals(d1.nextDate(30, 6, 1921), "The next date is: 1-7-1921");

}

@Test

public void test19() {

date d1 = new date();

assertEquals(d1.nextDate(31, 6, 1921), "Invalid date!");

}

}

**5. Robust Worst Case - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class Robust\_WC\_BVT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(0, 1, 1821), "Invalid date!");

}

@Test

public void test2() {

date d1 = new date();

assertEquals(d1.nextDate(0, 1, 1822), "Invalid date!");

}

@Test

public void test3() {

date d1 = new date();

assertEquals(d1.nextDate(0, 1, 1921), "Invalid date!");

}

@Test

public void test4() {

date d1 = new date();

assertEquals(d1.nextDate(0, 1, 2020), "Invalid date!");

}

@Test

public void test5() {

date d1 = new date();

assertEquals(d1.nextDate(0, 1, 2021), "Invalid date!");

}

@Test

public void test6() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 1821), "The next date is: 2-1-1821");

}

@Test

public void test7() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 1822), "The next date is: 2-1-1822");

}

@Test

public void test8() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 1921), "The next date is: 2-1-1921");

}

@Test

public void test9() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 2020), "The next date is: 2-1-2020");

}

@Test

public void test10() {

date d1 = new date();

assertEquals(d1.nextDate(1, 1, 2021), "The next date is: 2-1-2021");

}

@Test

public void test11() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 1821), "The next date is: 3-1-1821");

}

@Test

public void test12() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 1822), "The next date is: 3-1-1822");

}

@Test

public void test13() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 1921), "The next date is: 3-1-1921");

}

@Test

public void test14() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 2020), "The next date is: 3-1-2020");

}

@Test

public void test15() {

date d1 = new date();

assertEquals(d1.nextDate(2, 1, 2021), "The next date is: 3-1-2021");

}

@Test

public void test16() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 1821), "The next date is: 7-1-1821");

}

@Test

public void test17() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 1822), "The next date is: 7-1-1822");

}

@Test

public void test18() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 1921), "The next date is: 7-1-1921");

}

@Test

public void test19() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 2020), "The next date is: 7-1-2020");

}

@Test

public void test20() {

date d1 = new date();

assertEquals(d1.nextDate(6, 1, 2021), "The next date is: 7-1-2021");

}

@Test

public void test21() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 1821), "The next date is: 31-1-1821");

}

@Test

public void test22() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 1822), "The next date is: 31-1-1822");

}

@Test

public void test23() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 1921), "The next date is: 31-1-1921");

}

@Test

public void test24() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 2020), "The next date is: 31-1-2020");

}

@Test

public void test25() {

date d1 = new date();

assertEquals(d1.nextDate(30, 1, 2021), "The next date is: 31-1-2021");

}

@Test

public void test26() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 1821), "The next date is: 1-2-1821");

}

@Test

public void test27() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 1822), "The next date is: 1-2-1822");

}

@Test

public void test28() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 1921), "The next date is: 1-2-1921");

}

@Test

public void test29() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 2020), "The next date is: 1-2-2020");

}

@Test

public void test30() {

date d1 = new date();

assertEquals(d1.nextDate(31, 1, 2021), "The next date is: 1-2-2021");

}

@Test

public void test31() {

date d1 = new date();

assertEquals(d1.nextDate(32, 1, 1821), "Invalid date!");

}

@Test

public void test32() {

date d1 = new date();

assertEquals(d1.nextDate(32, 1, 1822), "Invalid date!");

}

@Test

public void test33() {

date d1 = new date();

assertEquals(d1.nextDate(32, 1, 1921), "Invalid date!");

}

@Test

public void test34() {

date d1 = new date();

assertEquals(d1.nextDate(32, 1, 2020), "Invalid date!");

}

@Test

public void test35() {

date d1 = new date();

assertEquals(d1.nextDate(32, 1, 2021), "Invalid date!");

}

}

**TEST CASES**

**Test Case Name** : BVA testing for NextDate function

**Test Data** : Day, Month and Year

**Pre-condition** : Day {1 <= d <= 31}, Month {1 <= m <= 12}, Year {1821 <= y <= 2021}

**Test Objective** : To find the next date for the given valid date.

1. **TEST CASES FOR NORMAL BOUNDARY VALUE TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_BVA | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Normal BVA. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_BVA\_01 | Date: nom  Month: nom  Year: min | 15 | 6 | 1821 | | A message is displayed as  “The next date is: 16-6-1821” | A message must be displayed as  “The next date is: 16-6-1821” | | Pass |
| ND\_BVA\_02 | Date: nom  Month: nom  Year: min+ | 15 | 6 | 1822 | | A message is displayed as  “The next date is: 16-6-1822” | A message must be displayed as  “The next date is: 16-6-1822” | | Pass |
| ND\_BVA\_03 | Date: nom  Month: nom  Year: nom | 15 | 6 | 1921 | | A message is displayed as  “The next date is: 16-6-1921” | A message must be displayed as  “The next date is: 16-6-1921” | | Pass |
| ND\_BVA\_04 | Date: nom  Month: nom  Year: max- | 15 | 6 | 2020 | | A message is displayed as  “The next date is: 16-6-2020” | A message must be displayed as  “The next date is: 16-6-2020” | | Pass |
| ND\_BVA\_05 | Date: nom  Month: nom  Year: max | 15 | 6 | 2021 | | A message is displayed as  “The next date is: 16-6-2021” | A message must be displayed as  “The next date is: 16-6-2021” | | Pass |
| ND\_BVA\_06 | Date: nom  Month: min  Year: nom | 15 | 1 | 1921 | | A message is displayed as  “The next date is: 16-1-1921” | A message must be displayed as  “The next date is: 16-1-1921” | | Pass |
| ND\_BVA\_07 | Date: nom  Month: min+  Year: nom | 15 | 2 | 1921 | | A message is displayed as  “The next date is: 16-2-1921” | A message must be displayed as  “The next date is: 16-2-1921” | | Pass |
| ND\_BVA\_08 | Date: nom  Month: max-  Year: nom | 15 | 11 | 1921 | | A message is displayed as  “The next date is: 16-11-1921” | A message must be displayed as  “The next date is: 16-11-1921” | | Pass |
| ND\_BVA\_09 | Date: nom  Month: max  Year: nom | 15 | 12 | 1921 | | A message is displayed as  “The next date is: 16-12-1921” | A message must be displayed as  “The next date is: 16-12-1921” | | Pass |
| ND\_BVA\_10 | Date: min  Month: nom  Year: nom | 1 | 6 | 1921 | | A message is displayed as  “The next date is: 2-6-1921” | A message must be displayed as  “The next date is: 2-6-1921” | | Pass |
| ND\_BVA\_11 | Date: min+  Month: nom  Year: nom. | 2 | 6 | 1921 | | A message is displayed as  “The next date is: 3-6-1921” | A message must be displayed as  “The next date is: 3-6-1921” | | Pass |
| ND\_BVA\_12 | Date: max-  Month: nom  Year: nom | 30 | 6 | 1921 | | A message is displayed as  “The next date is: 31-6-1921” | A message must be displayed as  “The next date is: 31-6-1921” | | Pass |
| ND\_BVA\_13 | Date: max  Month: nom  Year: nom | 31 | 6 | 1921 | | A message is displayed as  “The next date is: 1-7-1921” | A message must be displayed as  “The next date is: 1-7-1921” | | Pass |

1. **TEST CASES FOR ROBUST BOUNDARY VALUE TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_R\_BVA | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Robust BVA. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_R\_BVA\_01 | Date: nom  Month: nom  Year: min- | 15 | 6 | 1820 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_R\_BVA\_02 | Date: nom  Month: nom  Year: min | 15 | 6 | 1821 | | A message is displayed as  “The next date is: 16-6-1821” | A message must be displayed as  “The next date is: 16-6-1821” | | Pass |
| ND\_R\_BVA\_03 | Date: nom  Month: nom  Year: min+ | 15 | 6 | 1822 | | A message is displayed as  “The next date is: 16-6-1822” | A message must be displayed as  “The next date is: 16-6-1822” | | Pass |
| ND\_R\_BVA\_04 | Date: nom  Month: nom  Year: nom | 15 | 6 | 1921 | | A message is displayed as  “The next date is: 16-6-1921” | A message must be displayed as  “The next date is: 16-6-1921” | | Pass |
| ND\_R\_BVA\_05 | Date: nom  Month: nom  Year: max- | 15 | 6 | 2020 | | A message is displayed as  “The next date is: 16-6-2020” | A message must be displayed as  “The next date is: 16-6-2020” | | Pass |
| ND\_R\_BVA\_06 | Date: nom  Month: nom  Year: max | 15 | 6 | 2021 | | A message is displayed as  “The next date is: 16-6-2021” | A message must be displayed as  “The next date is: 16-6-2021” | | Pass |
| ND\_R\_BVA\_07 | Date: nom  Month: nom  Year: max+ | 15 | 6 | 2022 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_R\_BVA\_08 | Date: nom  Month: min-  Year: nom | 15 | 0 | 1921 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_R\_BVA\_09 | Date: nom  Month: min  Year: nom | 15 | 1 | 1921 | | A message is displayed as  “The next date is: 16-1-1921” | A message must be displayed as  “The next date is: 16-1-1921” | | Pass |
| ND\_R\_BVA\_10 | Date: nom  Month: min+  Year: nom | 15 | 2 | 1921 | | A message is displayed as  “The next date is: 16-2-1921” | A message must be displayed as  “The next date is: 16-2-1921” | | Pass |
| ND\_R\_BVA\_11 | Date: nom  Month: max-  Year: nom | 15 | 11 | 1921 | | A message is displayed as  “The next date is: 16-11-1921” | A message must be displayed as  “The next date is: 16-11-1921” | | Pass |
| ND\_R\_BVA\_12 | Date: nom  Month: max  Year: nom | 15 | 12 | 1921 | | A message is displayed as  “The next date is: 16-12-1921” | A message must be displayed as  “The next date is: 16-12-1921” | | Pass |
| ND\_R\_BVA\_13 | Date: nom  Month: max+  Year: nom | 15 | 13 | 1820 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_R\_BVA\_14 | Date: min-  Month: nom  Year: nom | 0 | 6 | 1921 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_R\_BVA\_15 | Date: min  Month: nom  Year: nom | 1 | 6 | 1921 | | A message is displayed as  “The next date is: 2-6-1921” | A message must be displayed as  “The next date is: 2-6-1921” | | Pass |
| ND\_R\_BVA\_16 | Date: min+  Month: nom  Year: nom. | 2 | 6 | 1921 | | A message is displayed as  “The next date is: 3-6-1921” | A message must be displayed as  “The next date is: 3-6-1921” | | Pass |
| ND\_R\_BVA\_17 | Date: max-  Month: nom  Year: nom | 30 | 6 | 1921 | | A message is displayed as  “The next date is: 31-6-1921” | A message must be displayed as  “The next date is: 31-6-1921” | | Pass |
| ND\_R\_BVA\_18 | Date: max  Month: nom  Year: nom | 31 | 6 | 1921 | | A message is displayed as  “The next date is: 1-7-1921” | A message must be displayed as  “The next date is: 1-7-1921” | | Pass |
| ND\_R\_BVA\_19 | Date: max+  Month: nom  Year: nom | 32 | 6 | 1921 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |

1. **TEST CASES FOR WORST CASE BOUNDARY VALUE TESTING**

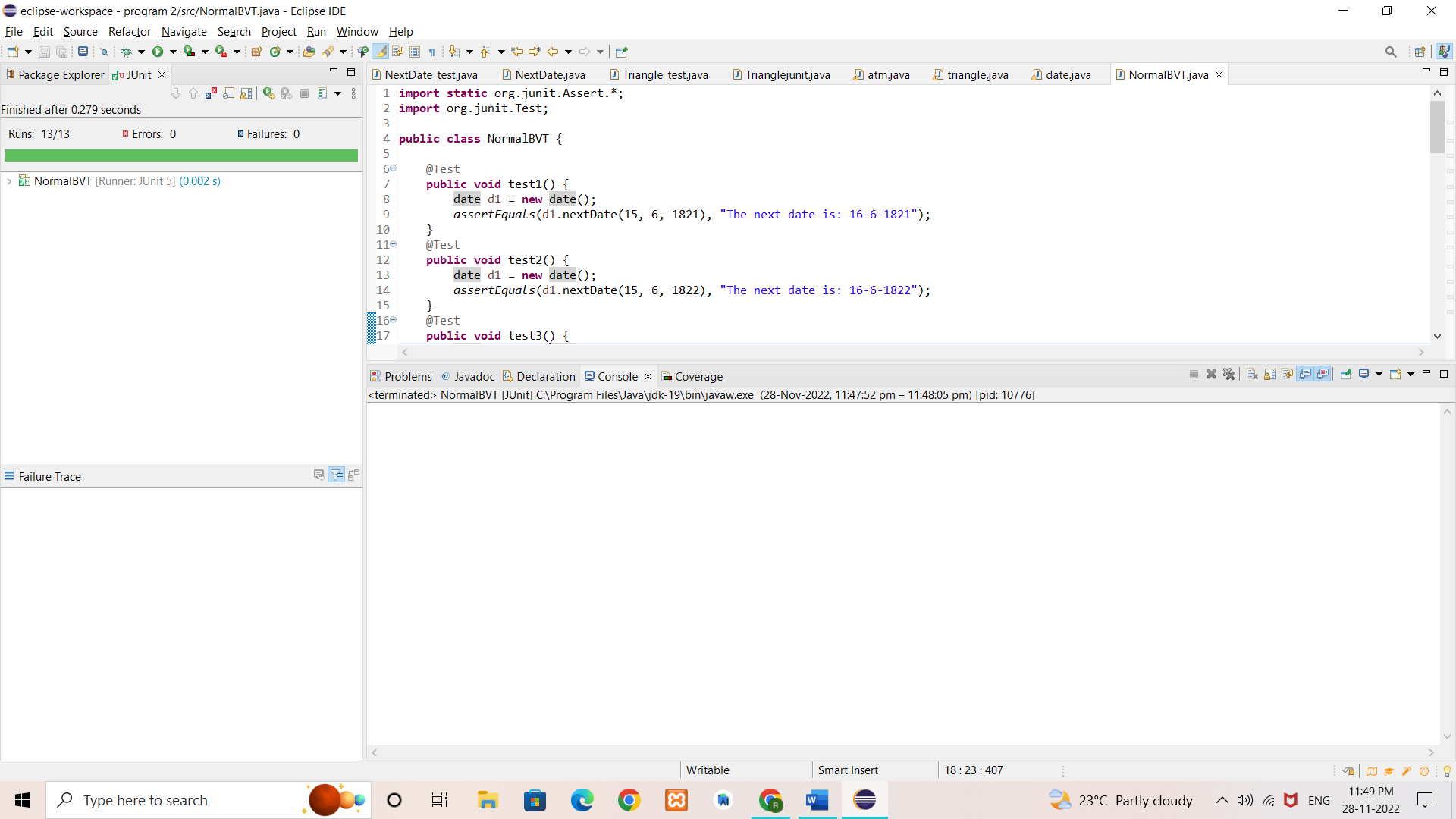
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_WC\_BVA | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Worst Case BVA. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_WC\_BVA\_01 | Date: min  Month: min  Year: min | 1 | 1 | 1821 | | A message is displayed as  “The next date is: 2-1-1821” | A message must be displayed as  “The next date is: 2-1-1821” | | Pass |
| ND\_WC\_BVA\_02 | Date: min  Month: min  Year: min+ | 1 | 1 | 1822 | | A message is displayed as  “The next date is: 2-1-1822” | A message must be displayed as  “The next date is: 2-1-1822” | | Pass |
| ND\_WC\_BVA\_03 | Date: min  Month: min  Year: nom | 1 | 1 | 1921 | | A message is displayed as  “The next date is: 2-1-1921” | A message must be displayed as  “The next date is: 2-1-1921” | | Pass |
| ND\_WC\_BVA\_04 | Date: min  Month: min  Year: max- | 1 | 1 | 2020 | | A message is displayed as  “The next date is: 2-1-2020” | A message must be displayed as  “The next date is: 2-1-2020” | | Pass |
| ND\_WC\_BVA\_05 | Date: min  Month: min  Year: max | 1 | 1 | 2021 | | A message is displayed as  “The next date is: 2-1-2021” | A message must be displayed as  “The next date is: 2-1-2021” | | Pass |
| ND\_WC\_BVA\_06 | Date: min+  Month: min  Year: min | 2 | 1 | 1821 | | A message is displayed as  “The next date is: 3-1-1821” | A message must be displayed as  “The next date is: 3-1-1821” | | Pass |
| ND\_WC\_BVA\_07 | Date: min+  Month: min  Year: min+ | 2 | 1 | 1822 | | A message is displayed as  “The next date is: 3-1-1822” | A message must be displayed as  “The next date is: 3-1-1822” | | Pass |
| ND\_WC\_BVA\_08 | Date: min+  Month: min  Year: nom | 2 | 1 | 1921 | | A message is displayed as  “The next date is: 3-1-1921” | A message must be displayed as  “The next date is: 3-1-1921” | | Pass |
| ND\_WC\_BVA\_09 | Date: min+  Month: min  Year: max- | 2 | 1 | 2020 | | A message is displayed as  “The next date is: 3-1-2020” | A message must be displayed as  “The next date is: 3-1-2020” | | Pass |
| ND\_WC\_BVA\_10 | Date: min+  Month: min  Year: max | 2 | 1 | 2021 | | A message is displayed as  “The next date is: 3-1-2021” | A message must be displayed as  “The next date is: 3-1-2021” | | Pass |
| ND\_WC\_BVA\_11 | Date: nom  Month: min  Year: min | 15 | 1 | 1821 | | A message is displayed as  “The next date is: 16-1-1821” | A message must be displayed as  “The next date is: 16-1-1821” | | Pass |
| ND\_WC\_BVA\_12 | Date: nom  Month: min  Year: min+ | 15 | 1 | 1822 | | A message is displayed as  “The next date is: 16-1-1822” | A message must be displayed as  “The next date is: 16-1-1822” | | Pass |
| ND\_WC\_BVA\_13 | Date: nom  Month: min  Year: nom | 15 | 1 | 1921 | | A message is displayed as  “The next date is: 16-1-1921” | A message must be displayed as  “The next date is: 16-1-1921” | | Pass |
| ND\_WC\_BVA\_14 | Date: nom  Month: min  Year: max- | 15 | 1 | 2020 | | A message is displayed as  “The next date is: 16-1-2020” | A message must be displayed as  “The next date is: 16-1-2020” | | Pass |
| ND\_WC\_BVA\_15 | Date: nom  Month: min  Year: max | 15 | 1 | 2021 | | A message is displayed as  “The next date is: 16-1-2021” | A message must be displayed as  “The next date is: 16-1-2021” | | Pass |
| ND\_WC\_BVA\_16 | Date: max-  Month: min  Year: min | 30 | 1 | 1821 | | A message is displayed as  “The next date is: 31-1-1821” | A message must be displayed as  “The next date is: 31-1-1821” | | Pass |
| ND\_WC\_BVA\_17 | Date: max-  Month: min  Year: min+ | 30 | 1 | 1822 | | A message is displayed as  “The next date is: 31-1-1822” | A message must be displayed as  “The next date is: 31-1-1822” | | Pass |
| ND\_WC\_BVA\_18 | Date: max-  Month: min  Year: nom | 30 | 1 | 1921 | | A message is displayed as  “The next date is: 31-1-1921” | A message must be displayed as  “The next date is: 31-1-1921” | | Pass |
| ND\_WC\_BVA\_19 | Date: max-  Month: min  Year: max- | 30 | 1 | 2020 | | A message is displayed as  “The next date is: 31-1-2020” | A message must be displayed as  “The next date is: 31-1-2020” | | Pass |
| ND\_WC\_BVA\_20 | Date: max-  Month: min  Year: max | 30 | 1 | 2021 | | A message is displayed as  “The next date is: 31-1-2021” | A message must be displayed as  “The next date is: 31-1-2021” | | Pass |

1. **TEST CASES FOR ROBUST WORST CASE BOUNDARY VALUE TESTING**

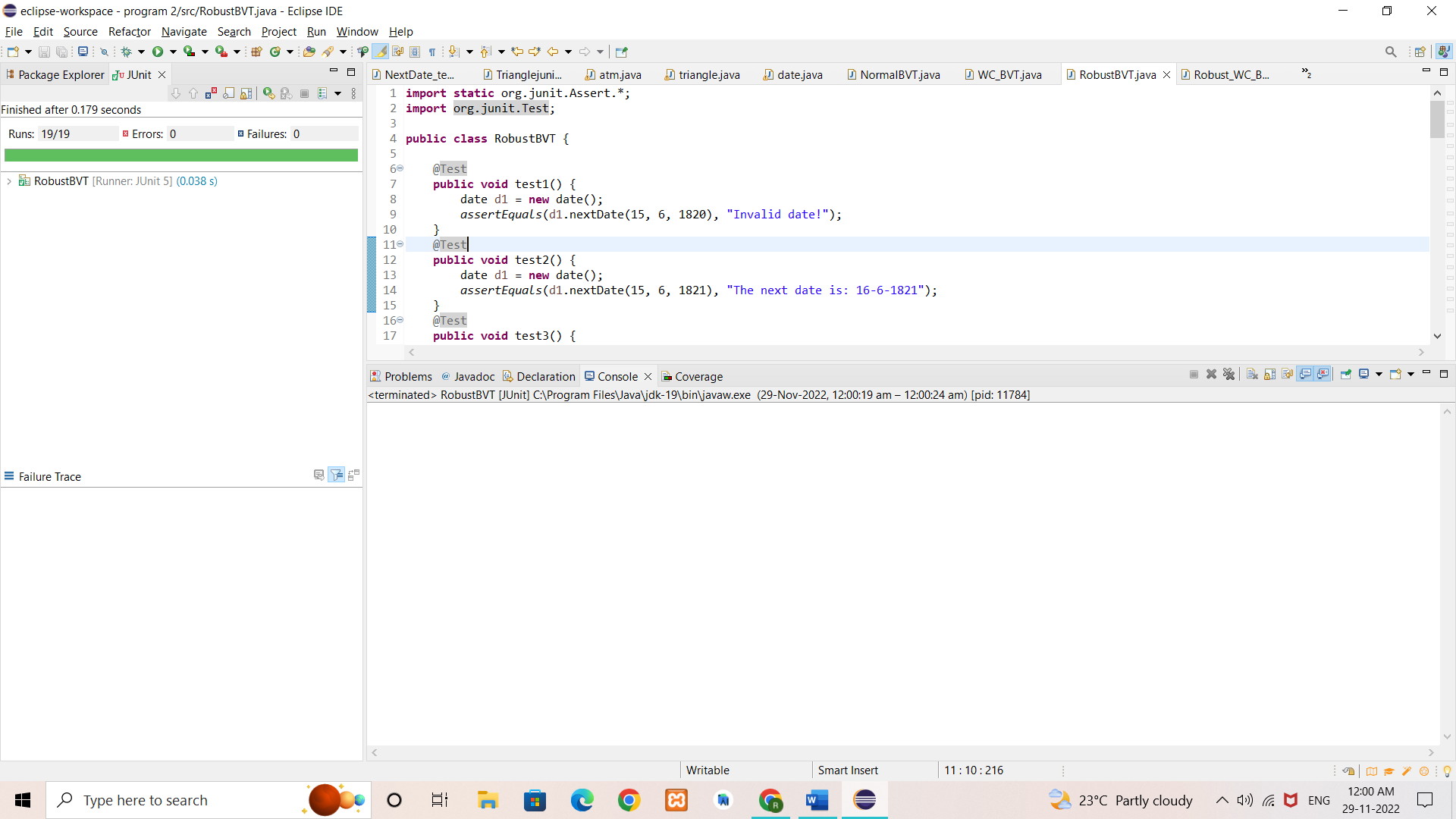
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_RWC\_BVA | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Robust Worst Case BVA. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_RWC\_  BVA\_01 | Date: min-  Month: min  Year: min | 0 | 1 | 1821 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_RWC\_  BVA\_02 | Date: min-  Month: min  Year: min+ | 0 | 1 | 1822 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_RWC\_  BVA\_03 | Date: min-  Month: min  Year: nom | 0 | 1 | 1921 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_RWC\_  BVA\_04 | Date: min-  Month: min  Year: max- | 0 | 1 | 2020 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_RWC\_  BVA\_05 | Date: min-  Month: min  Year: max | 0 | 1 | 2021 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_RWC\_  BVA\_06 | Date: min  Month: min  Year: min | 1 | 1 | 1821 | | A message is displayed as  “The next date is: 2-1-1821” | A message must be displayed as  “The next date is: 2-1-1821” | | Pass |
| ND\_RWC\_  BVA\_07 | Date: min  Month: min  Year: min+ | 1 | 1 | 1822 | | A message is displayed as  “The next date is: 2-1-1822” | A message must be displayed as  “The next date is: 2-1-1822” | | Pass |
| ND\_RWC\_  BVA\_08 | Date: min  Month: min  Year: nom | 1 | 1 | 1921 | | A message is displayed as  “The next date is: 2-1-1921” | A message must be displayed as  “The next date is: 2-1-1921” | | Pass |
| ND\_RWC\_  BVA\_09 | Date: min  Month: min  Year: max- | 1 | 1 | 2020 | | A message is displayed as  “The next date is: 2-1-2020” | A message must be displayed as  “The next date is: 2-1-2020” | | Pass |
| ND\_RWC\_  BVA\_10 | Date: min  Month: min  Year: max | 1 | 1 | 2021 | | A message is displayed as  “The next date is: 2-1-2021” | A message must be displayed as  “The next date is: 2-1-2021” | | Pass |
| ND\_RWC\_  BVA\_11 | Date: min+  Month: min  Year: min | 2 | 1 | 1821 | | A message is displayed as  “The next date is: 3-1-1821” | A message must be displayed as  “The next date is: 3-1-1821” | | Pass |
| ND\_RWC\_  BVA\_12 | Date: min+  Month: min  Year: min+ | 2 | 1 | 1822 | | A message is displayed as  “The next date is: 3-1-1822” | A message must be displayed as  “The next date is: 3-1-1822” | | Pass |
| ND\_RWC\_  BVA\_13 | Date: min+  Month: min  Year: nom | 2 | 1 | 1921 | | A message is displayed as  “The next date is: 3-1-1921” | A message must be displayed as  “The next date is: 3-1-1921” | | Pass |
| ND\_RWC\_  BVA\_14 | Date: min+  Month: min  Year: max- | 2 | 1 | 2020 | | A message is displayed as  “The next date is: 3-1-2020” | A message must be displayed as  “The next date is: 3-1-2020” | | Pass |
| ND\_RWC\_  BVA\_15 | Date: min+  Month: min  Year: max | 2 | 1 | 2021 | | A message is displayed as  “The next date is: 3-1-2021” | A message must be displayed as  “The next date is: 3-1-2021” | | Pass |
| ND\_RWC\_  BVA\_16 | Date: nom  Month: min  Year: min | 15 | 1 | 1821 | | A message is displayed as  “The next date is: 16-1-1821” | A message must be displayed as  “The next date is: 16-1-1821” | | Pass |
| ND\_RWC\_  BVA\_17 | Date: nom  Month: min  Year: min+ | 15 | 1 | 1822 | | A message is displayed as  “The next date is: 16-1-1822” | A message must be displayed as  “The next date is: 16-1-1822” | | Pass |
| ND\_RWC\_  BVA\_18 | Date: nom  Month: min  Year: nom | 15 | 1 | 1921 | | A message is displayed as  “The next date is: 16-1-1921” | A message must be displayed as  “The next date is: 16-1-1921” | | Pass |
| ND\_RWC\_  BVA\_19 | Date: nom  Month: min  Year: max- | 15 | 1 | 2020 | | A message is displayed as  “The next date is: 16-1-2020” | A message must be displayed as  “The next date is: 16-1-2020” | | Pass |
| ND\_RWC\_  BVA\_20 | Date: nom  Month: min  Year: max | 15 | 1 | 2021 | | A message is displayed as  “The next date is: 16-1-2021” | A message must be displayed as  “The next date is: 16-1-2021” | | Pass |

**EXECUTION**

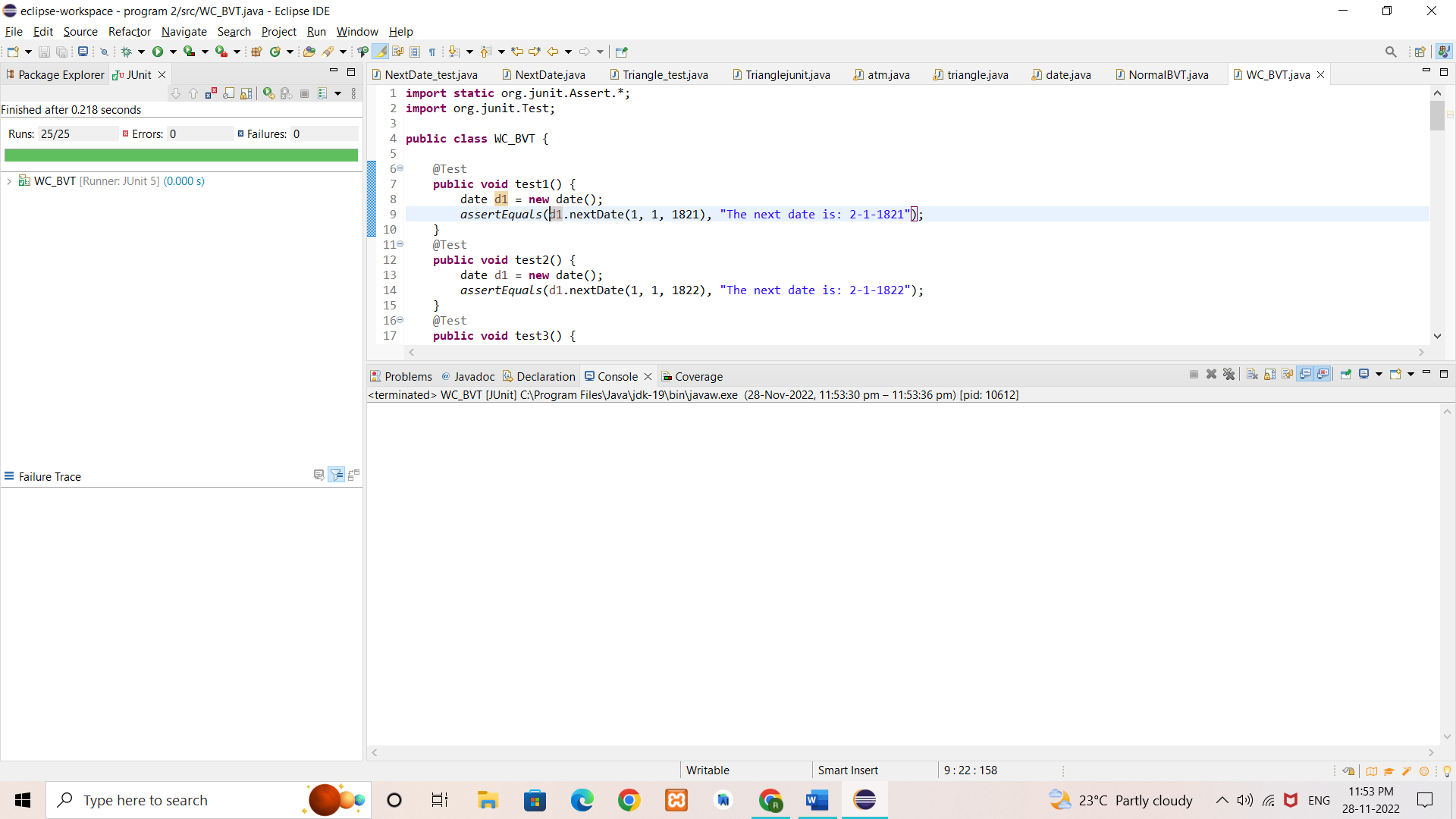
**1. Normal BVA - JUnit Test Cases Result**

****

**2. Robust BVA - JUnit Test Cases Result**

****

**3. Worst Case BVA - JUnit Test Cases Result**

****

**4. Robust Worst Case BVA - JUnit Test Cases Result**

****

**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed : 92

2. Number of Test Cases Passed : 92

3. Number of Test Cases Failed : 0

**Exp. No. :** 4

**Date :** 12-11-2021

**EQUIVALENCE CLASS PARTITIONING (ECP) FOR NEXTDATE FUNCTION**

Design, develop, code and run the program in any suitable language to implement the NextDate function. Analyse it from the perspective equivalence class testing. Create different test cases, execute these test cases by using JUnit and discuss the test results.

1. Weak Normal Equivalence Class Testing
2. Strong Normal Equivalence Class Testing
3. Weak Robust Equivalence Class Testing
4. Strong Robust Equivalence Class Testing

**IMPLEMENTATION**

**1. NextDate function**

import java.util.\*;

public class date {

public String nextDate (int d, int m, int y) {

int nd, nm, ny;

if(d>31 || d<1 || m>12 || m <1 || y<1821 || y>2021){

return ("Invalid date!");

}

else if(m==2 || m==4 || m==6 || m==9 || m==11 ){

if(d==31)

return("Invalid date!");

else if(m==2){

if(checkLeapYear(y)){

if(d>29){

return("Invalid date!");

}

if(d == 29) {

nd = 1;

nm = 3;

}

else {

nd = ++d;

nm = 2;

}

}

else {

if(d>28){

return("Invalid date!");

}

if(d == 28) {

nd = 1;

nm = 3;

}

else {

nd = ++d;

nm = 2;

}

}

ny = y;

}

else {

if (d == 30){

nd = 1;

nm = ++m;

}

else{

nd = ++d;

nm = m;

}

ny = y;

}

}

else {

if (d == 31 && m != 12){

nd = 1;

nm = ++m;

ny = y;

}

else if (d == 31 && m == 12){

nd = 1;

nm = 1;

ny = ++y;

}

else{

nd = ++d;

nm = m;

ny = y;

}

}

return("The next date is: "+nd+"-"+nm+"-"+ny);

}

public static boolean checkLeapYear(int year){

if(year % 400 == 0)

return true;

else if(year % 100 == 0)

return false;

else if(year % 4 == 0)

return true;

else

return false;

}

}

**2. Weak Normal Equivalence Class - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class WN\_ECT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(14, 6, 2000), "The next date is: 15-6-2000");

}

}

**3.** **Strong Normal Equivalence Class - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class SN\_ECT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(14, 6, 2000), "The next date is: 15-6-2000");

}

}

**4.** **Weak Robust Equivalence Class - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class WR\_ECT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2001), "The next date is: 16-6-2001");

}

@Test

public void test2() {

date d1 = new date();

assertEquals(d1.nextDate(0, 6, 1822), "Invalid date!");

}

@Test

public void test3() {

date d1 = new date();

assertEquals(d1.nextDate(15, 13, 1921), "Invalid date!");

}

@Test

public void test4() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2028), "Invalid date!");

}

}

**5.** **Strong Robust Equivalence Class - JUnit Test Cases**

import static org.junit.Assert.\*;

import org.junit.Test;

public class SR\_ECT {

@Test

public void test1() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2001), "The next date is: 16-6-2001");

}

@Test

public void test2() {

date d1 = new date();

assertEquals(d1.nextDate(0, 6, 1822), "Invalid date!");

}

@Test

public void test3() {

date d1 = new date();

assertEquals(d1.nextDate(15, 13, 1921), "Invalid date!");

}

@Test

public void test4() {

date d1 = new date();

assertEquals(d1.nextDate(15, 6, 2028), "Invalid date!");

}

@Test

public void test5() {

date d1 = new date();

assertEquals(d1.nextDate(0, 13, 2021), "Invalid date!");

}

@Test

public void test6() {

date d1 = new date();

assertEquals(d1.nextDate(41, 1, 1785), "Invalid date!");

}

@Test

public void test7() {

date d1 = new date();

assertEquals(d1.nextDate(5, 15, 2112), "Invalid date!");

}

@Test

public void test8() {

date d1 = new date();

assertEquals(d1.nextDate(46, 19, 1512), "Invalid date!");

}

}

**TEST CASES**

**Test Case Name** : Equivalence Class Testing for NextDate function

**Test Data** : Day, Month and Year

**Pre-condition** : Day {1 <= d <= 31}, Month {1 <= m <= 12}, Year {1821 <= y <= 2021}

**Test Objective** : To find the next date for the given valid date.

1. **TEST CASES FOR WEAK NORMAL ECP TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_WN\_ECP | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Weak Normal ECP. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_WN\_ECP\_01 | A date that contains 1 valid input each from date, month and year. | 14 | 6 | 2000 | | A message is displayed as  “The next date is: 15-6-2000” | A message must be displayed as  “The next date is: 15-6-2000” | | Pass |

1. **TEST CASES FOR STRONG NORMAL ECP TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_SN\_ECP | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Strong Normal ECP. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_SN\_ECP\_01 | A date that contains 1 valid input each from date, month and year. | 14 | 6 | 2000 | | A message is displayed as  “The next date is: 15-6-2000” | A message must be displayed as  “The next date is: 15-6-2000” | | Pass |

1. **TEST CASES FOR WEAK ROBUST ECP TESTING**

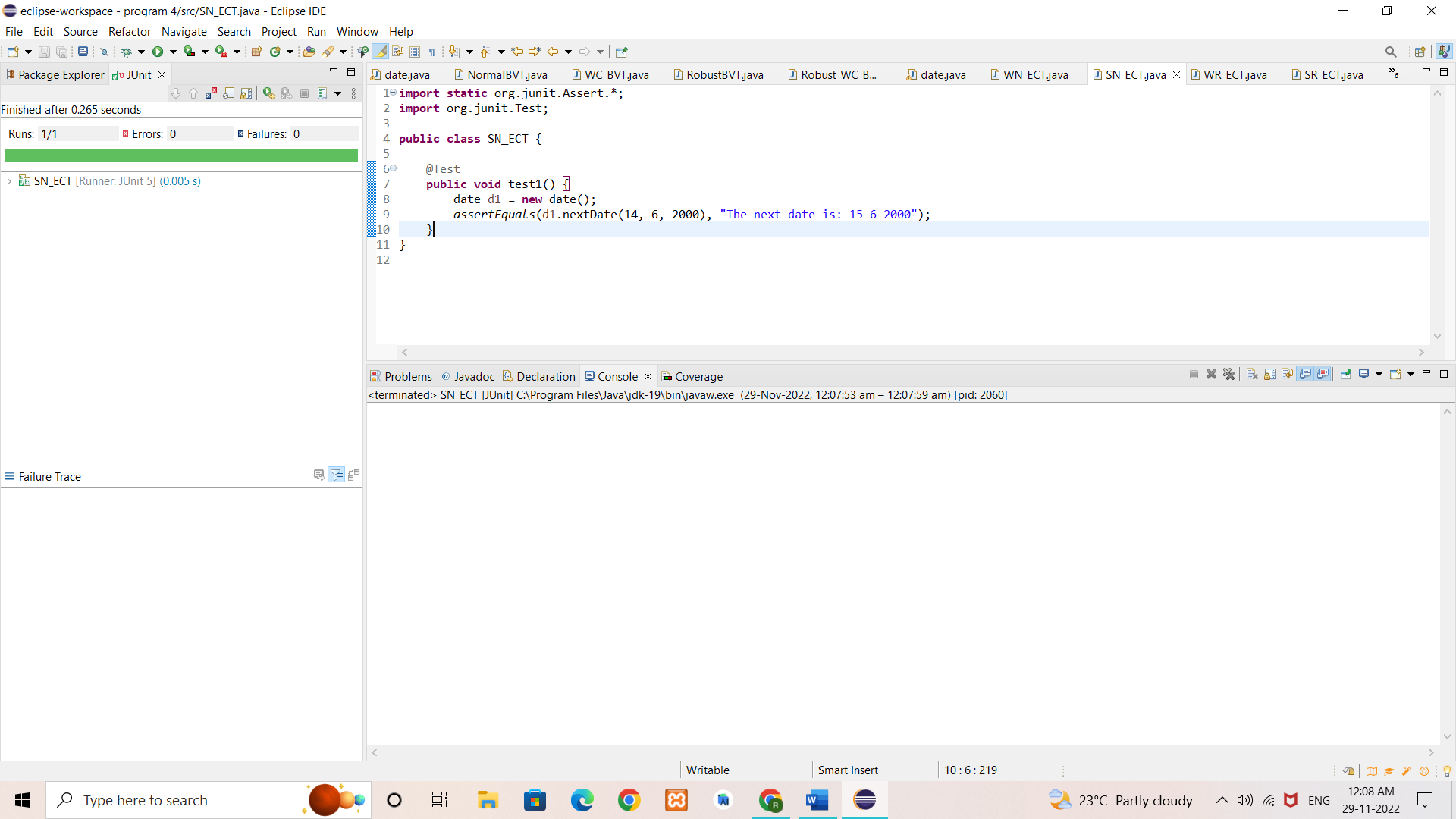
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_WR\_ECP | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Weak Robust ECP. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_WR\_ECP\_01 | A date that contains 1 valid input each from date, month and year. | 15 | 6 | 2001 | | A message is displayed as  “The next date is: 16-6-2001” | A message must be displayed as  “The next date is: 16-6-2001” | | Pass |
| ND\_WR\_ECP\_02 | Invalid day. | 0 | 6 | 1822 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_WR\_ECP\_03 | Invalid month. | 15 | 13 | 1921 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_WR\_ECP\_04 | Invalid year. | 15 | 6 | 2028 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |

1. **TEST CASES FOR STRONG ROBUST ECP TESTING**

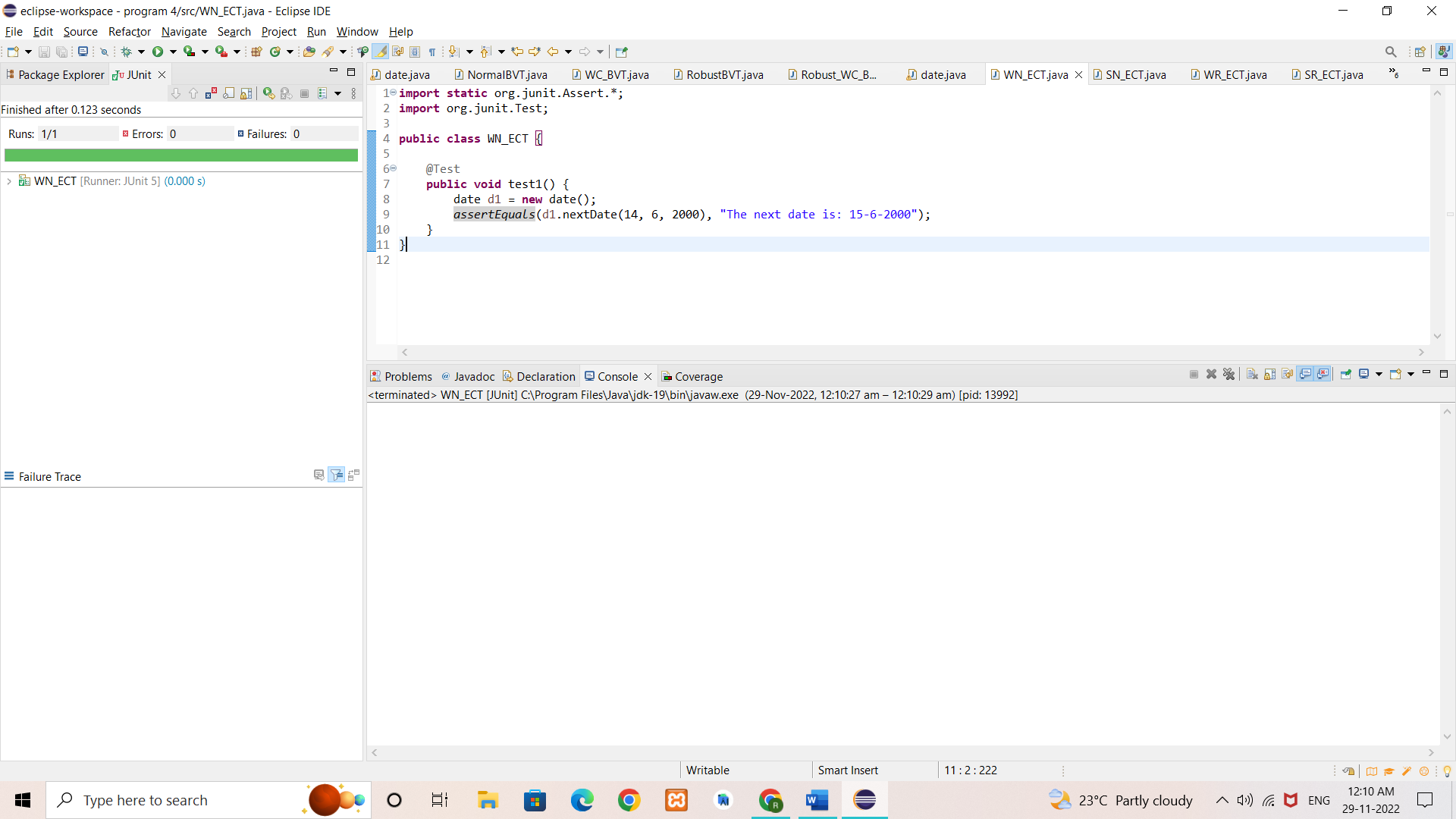
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NextDate | | | | Test Name: | | | ND\_SR\_ECP | |
| Project ID: | NextDate\_01 | | | | Original Author: | | | Rahul M Dinesh | |
| Test Objective: | To find the next date for the given date using Strong Robust ECP. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | **Test Data** | | | | **Observed Result** | **Expected Result** | | **Status (Pass/Fail)** |
| **d** | **m** | **y** | |
| ND\_SR\_ECP\_01 | A date that contains 1 valid input each from date, month and year. | 15 | 6 | 2001 | | A message is displayed as  “The next date is: 16-6-2001” | A message must be displayed as  “The next date is: 16-6-2001” | | Pass |
| ND\_SR\_ECP\_02 | Invalid day. | 0 | 6 | 1822 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_SR\_ECP\_03 | Invalid month. | 15 | 13 | 1921 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_SR\_ECP\_04 | Invalid year. | 15 | 6 | 2028 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_SR\_ECP\_05 | Invalid day and month. | 0 | 13 | 2021 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_SR\_ECP\_06 | Invalid day and year. | 41 | 1 | 1785 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_SR\_ECP\_07 | Invalid month and year. | 5 | 15 | 2112 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |
| ND\_SR\_ECP\_08 | Invalid day, month and year. | 46 | 19 | 1512 | | A message is displayed as  “Invalid date!” | A message must be displayed as  “Invalid date!” | | Pass |

**EXECUTION**

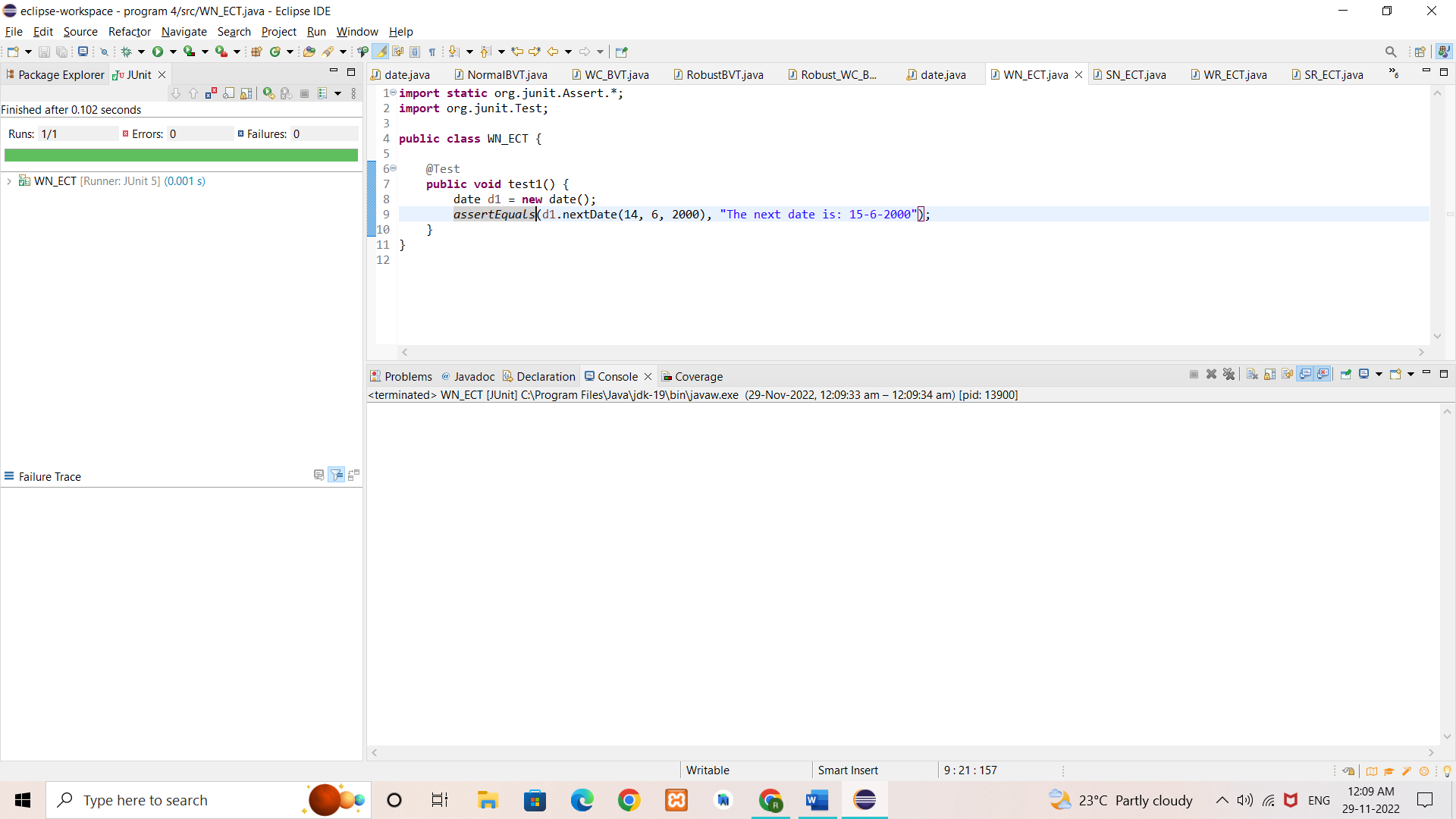
**1. Weak Normal ECT - JUnit Test Cases Result**

****

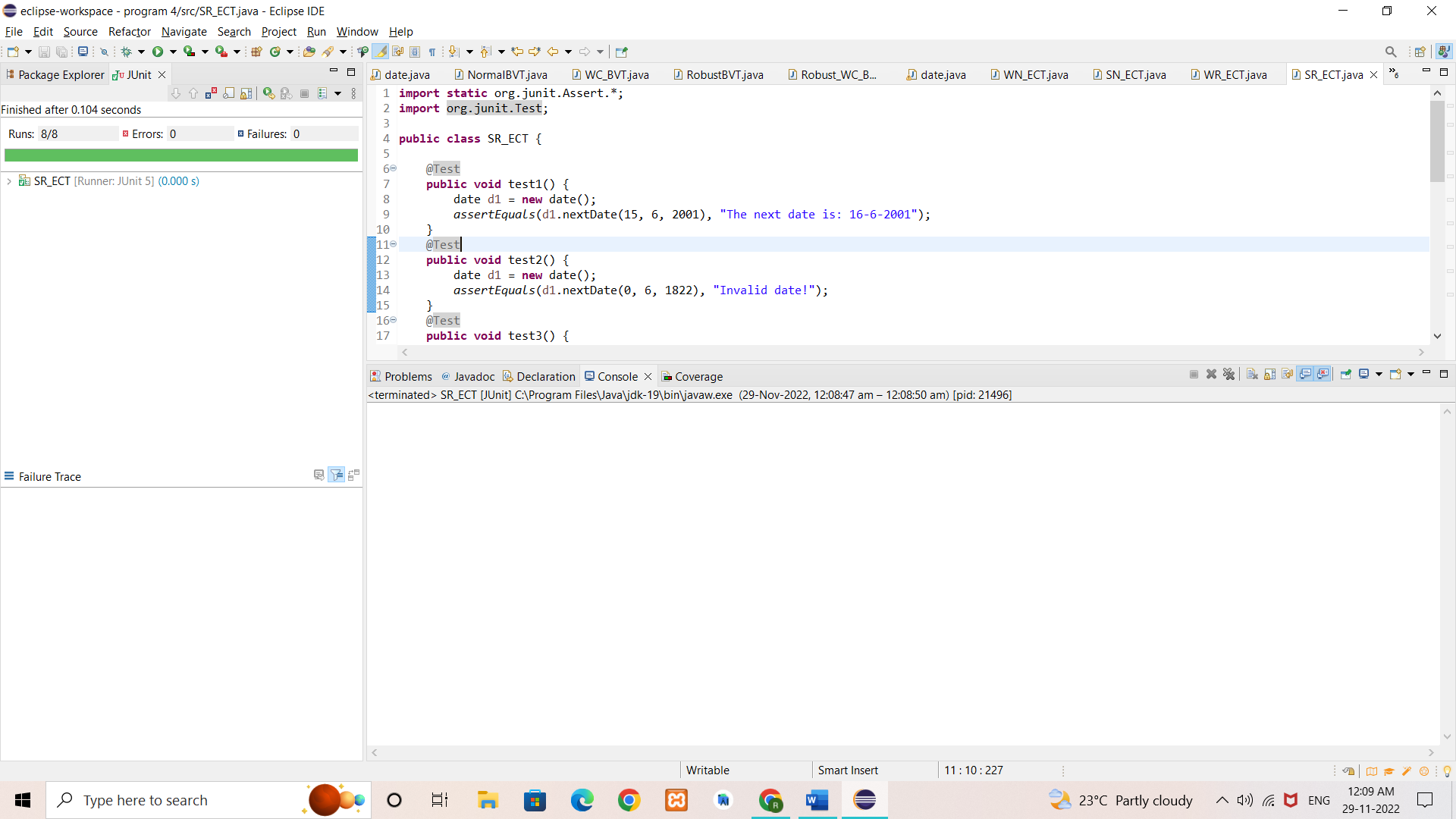
**2. Strong Normal ECT - JUnit Test Cases Result**

****

**3. Weak Robust ECT - JUnit Test Cases Result**

****

**4. Strong Robust ECT - JUnit Test Cases Result**

****

**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed : 14

2. Number of Test Cases Passed : 14

3. Number of Test Cases Failed : 0

**Exp. No. :** 5

**Date :** 19-11-2021

**DEMONSTRATION OF WHITE BOX TESTING TECHNIQUE USING ECLEMMA**

Demonstrate white box testing techniques using open-source testing tool JUnit and ECLEMMA. Implement and execute test cases for achieving full statement coverage, decision/branch coverage and condition coverage for the triangle problem.

**IMPLEMENTATION**

**1. Triangle problem**

import java.util.\*;

public class triangle {

public String check (int a, int b, int c){

while(true){

if(a>=1 && a<=200 && b>=1 && b<=200 && c>=1 && c<=200){

if((a < b+c) && (b < a+c) && (c < b+a)){

if(a == b && b == c)

return ("Given dimensions form an equilateral triangle!");

else if(a==b || b==c || c==a)

return("Given dimensions form an isosceles triangle!");

else

return("Given dimensions form a scalene triangle!");

}

else {

return("Given dimensions do not form a triangle!");

}

}

else{

return("Enter a valid input!");

}

}

}

}

**2. JUnit Test Cases for Complete Coverage**

import static org.junit.Assert.\*;

import org.junit.Test;

import org.junit.Test;

public class triangleTest {

@Test

public void test1() {

triangle t1 = new triangle();

assertEquals(t1.check(1, 2, 3), "Given dimensions do not form a triangle!");

}

@Test

public void test2() {

triangle t1 = new triangle();

assertEquals(t1.check(2, 2, 2), "Given dimensions form an equilateral triangle!");

}

@Test

public void test3() {

triangle t1 = new triangle();

assertEquals(t1.check(2, 2, 3), "Given dimensions form an isosceles triangle!");

}

@Test

public void test4() {

triangle t1 = new triangle();

assertEquals(t1.check(4, 5, 6), "Given dimensions form a scalene triangle!");

}

@Test

public void test5() {

triangle t1 = new triangle();

assertEquals(t1.check(-4, 5, 6), "Enter a valid input!");

}

@Test

public void test6() {

triangle t1 = new triangle();

assertEquals(t1.check(4, 5, 4), "Given dimensions form an isosceles triangle!");

}

@Test

public void test7() {

triangle t1 = new triangle();

assertEquals(t1.check(5, 4, 4), "Given dimensions form an isosceles triangle!");

}

@Test

public void test8() {

triangle t1 = new triangle();

assertEquals(t1.check(7, 4, 2), "Given dimensions do not form a triangle!");

}

@Test

public void test9() {

triangle t1 = new triangle();

assertEquals(t1.check(4, 7, 2), "Given dimensions do not form a triangle!");

}

@Test

public void test10() {

triangle t1 = new triangle();

assertEquals(t1.check(4, -5, 6), "Enter a valid input!");

}

@Test

public void test11() {

triangle t1 = new triangle();

assertEquals(t1.check(4, 5, -6), "Enter a valid input!");

}

@Test

public void test12() {

triangle t1 = new triangle();

assertEquals(t1.check(4, 205, -6), "Enter a valid input!");

}

@Test

public void test13() {

triangle t1 = new triangle();

assertEquals(t1.check(204, 205, 209), "Enter a valid input!");

}

@Test

public void test14() {

triangle t1 = new triangle();

assertEquals(t1.check(5, 5, 209), "Enter a valid input!");

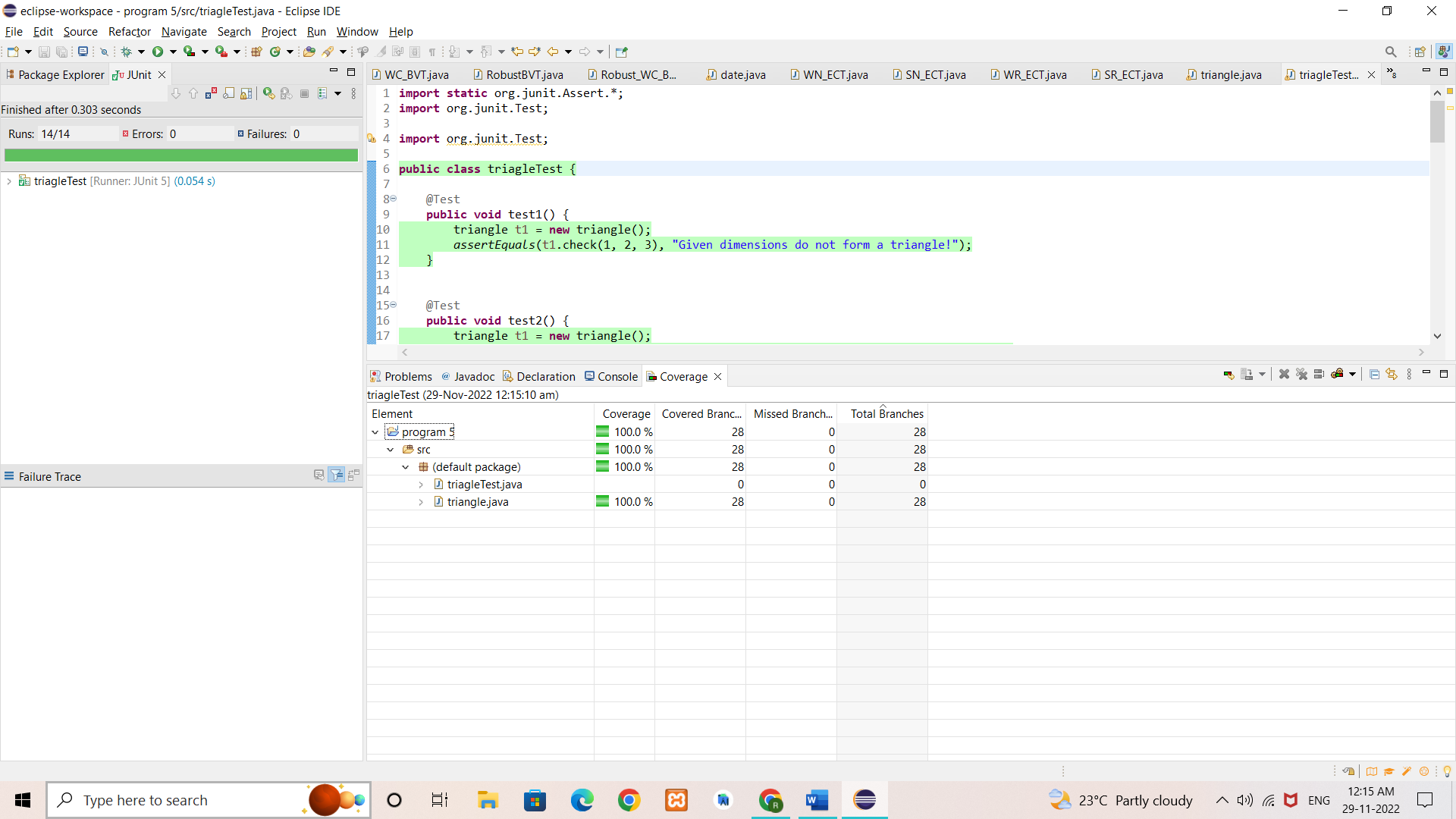
}

}

**TEST CASES FOR TRIANGLE PROGRAM**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | | | **Test Information** | | | |
| Project Name: | | Triangle | | | | | Test Name: | | Triangle Code Coverage | |
| Project ID: | | Triangle\_02 | | | | | Original Author: | | Rahul M Dinesh | |
| Test Objective: | | To check whether three given sides forms a scalene or equilateral or isosceles triangle or if given sides do not form a triangle or if given sides are invalid inputs. | | | | | | | | |
| **Case No.** | **Test Case**  **Description** | | **Test Data** | | | **Observed Result** | | **Expected Result** | | **Status (Pass/Fail)** |
| **a** | **b** | **c** |
| Triangle\_Cov\_1 | Check for dimensions that do not form a triangle. | | 1 | 2 | 3 | A message saying ‘Given dimensions do not form a triangle!’ appears. | | A message saying ‘Given dimensions do not form a triangle!’ should appear. | | Pass |
| Triangle\_Cov\_2 | Check for dimensions that form an equilateral triangle. | | 2 | 2 | 2 | A message saying ‘Given dimensions form an equilateral triangle!’ appears. | | A message saying ‘Given dimensions form an equilateral triangle!’ should appear. | | Pass |
| Triangle\_Cov\_3 | Check for dimensions that form an isosceles triangle. | | 4 | 4 | 5 | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |
| Triangle\_Cov\_4 | Check for dimensions that form a scalene triangle. | | 4 | 5 | 6 | A message saying ‘Given dimensions form a scalene triangle!’ appears. | | A message saying ‘Given dimensions form a scalene triangle!’ should appear. | | Pass |
| Triangle\_Cov\_5 | Check for dimensions that are not valid inputs. | | -4 | 5 | 6 | A message is displayed as “Enter a valid input!” | | A message must be displayed as “Enter a valid input!” | | Pass |
| Triangle\_Cov\_6 | Check for dimensions that form an isosceles triangle. | | 4 | 5 | 4 | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |
| Triangle\_Cov\_7 | Check for dimensions that form an isosceles triangle. | | 5 | 4 | 4 | A message saying ‘Given dimensions form an isosceles triangle!’ appears. | | A message saying ‘Given dimensions form an isosceles triangle!’ should appear. | | Pass |
| Triangle\_Cov\_8 | Check for dimensions that do not form a triangle. | | 7 | 4 | 2 | A message saying ‘Given dimensions do not form a triangle!’ appears. | | A message saying ‘Given dimensions do not form a triangle!’ should appear. | | Pass |
| Triangle\_Cov\_9 | Check for dimensions that do not form a triangle. | | 4 | 7 | 2 | A message saying ‘Given dimensions do not form a triangle!’ appears. | | A message saying ‘Given dimensions do not form a triangle!’ should appear. | | Pass |
| Triangle\_Cov\_10 | Check for dimensions that are not valid inputs. | | 4 | -5 | 6 | A message is displayed as “Enter a valid input!” | | A message must be displayed as “Enter a valid input!” | | Pass |
| Triangle\_Cov\_11 | Check for dimensions that are not valid inputs. | | 4 | 5 | -6 | A message is displayed as “Enter a valid input!” | | A message must be displayed as “Enter a valid input!” | | Pass |
| Triangle\_Cov\_12 | Check for dimensions that are not valid inputs. | | 4 | 205 | -6 | A message is displayed as “Enter a valid input!” | | A message must be displayed as “Enter a valid input!” | | Pass |
| Triangle\_Cov\_13 | Check for dimensions that are not valid inputs. | | 204 | 205 | 209 | A message is displayed as “Enter a valid input!” | | A message must be displayed as “Enter a valid input!” | | Pass |
| Triangle\_Cov\_14 | Check for dimensions that are not valid inputs. | | 5 | 5 | 209 | A message is displayed as “Enter a valid input!” | | A message must be displayed as “Enter a valid input!” | | Pass |

**EXECUTION**

****

**RESULT & DISCUSSION:** Thus, the above programs are written and executed using JUnit and ECLEMMA, and 100% coverage is achieved.

**Exp. No. :** 6

**Date :** 26-11-2021

**DEMONSTRATION OF WHITE BOX TESTING TECHNIQUE USING ECLEMMA**

Demonstrate white box testing techniques using open-source testing tool JUnit and ECLEMMA. Implement and execute test cases for achieving full statement coverage, decision/branch coverage and condition coverage for the NextDate problem.

**IMPLEMENTATION**

**1. NextDate function**

import java.util.Scanner;

public class date {

public String nextDate (int d, int m, int y) {

int nd, nm, ny;

if(d>31 || d<1 || m>12 || m <1 || y<1821 || y>2021){

return ("Invalid date!");

}

else if(m==2 || m==4 || m==6 || m==9 || m==11 ){

if(d==31){

return("Invalid date!");

}

else if(m==2){

if(checkLeapYear(y)){

if(d>29){

return("Invalid date!");

}

if(d == 29) {

nd = 1;

nm = 3;

}

else {

nd = ++d;

nm = 2;

}

}

else {

if(d>28){

return("Invalid date!");

}

if(d == 28) {

nd = 1;

nm = 3;

}

else {

nd = ++d;

nm = 2;

}

}

ny = y;

}

else {

if (d == 30){

nd = 1;

nm = ++m;

}

else{

nd = ++d;

nm = m;

}

ny = y;

}

}

else {

if (d == 31){

if (m!=12){

nd = 1;

nm = ++m;

ny = y;

}

else {

nd = 1;

nm = 1;

ny = ++y;

}

}

else{

nd = ++d;

nm = m;

ny = y;

}

}

return("The next date is: "+nd+"-"+nm+"-"+ny);

}

public static boolean checkLeapYear(int year){

if(year % 400 == 0)

return true;

else if(year % 100 == 0)

return false;

else if(year % 4 == 0)

return true;

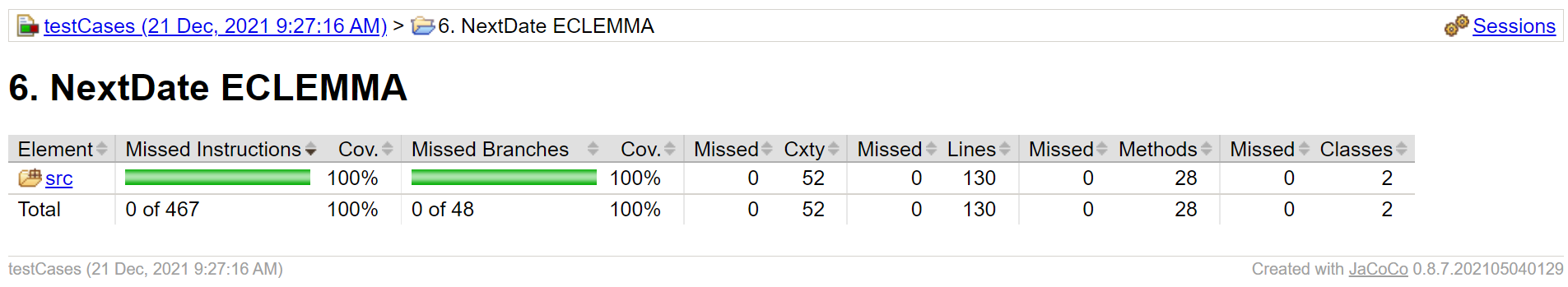
else

return false;

}

}

**EXECUTION**

****

**RESULT & DISCUSSION**: Thus, the above programs are written and executed using JUnit and ECLEMMA, and 100% coverage is achieved.

**Exp. No. :** 7

**Date :** 10-12-2021

**DEMONSTRATION OF SELENIUM IDE FOR CONDUCTING TEST ON WEBSITE(S)**

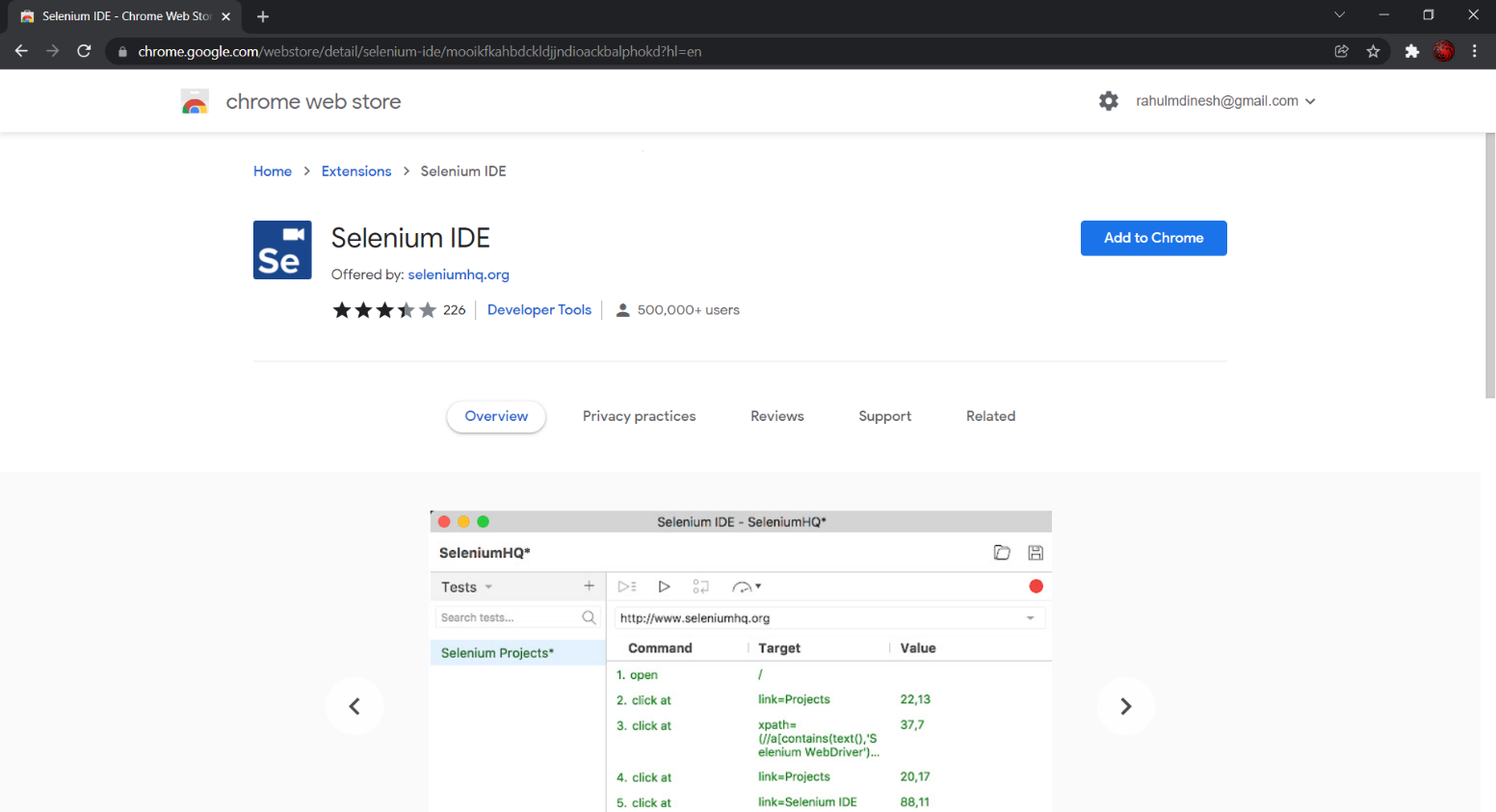
Designing Test Cases using Selenium IDE.

**IMPLEMENTATION**

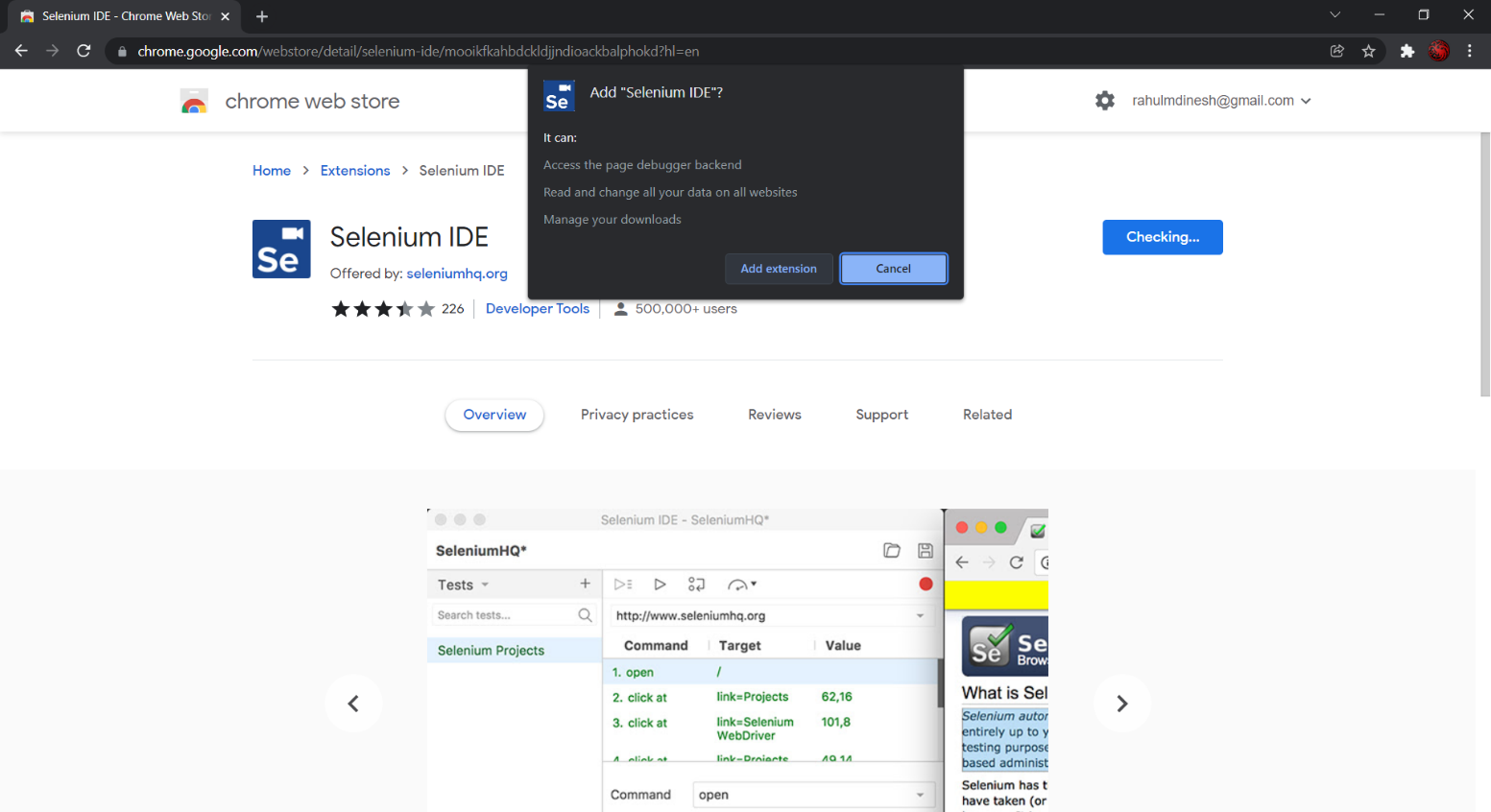
[**https://chrome.google.com/webstore/detail/selenium-ide/mooikfkahbdckldjjndioackbalphokd**](https://chrome.google.com/webstore/detail/selenium-ide/mooikfkahbdckldjjndioackbalphokd)

**Installing Selenium IDE**

**Step 1**: Using Chrome, first, download the Selenium IDE extension from the Chrome Web Store by clicking on the ‘Add to Chrome’ button.

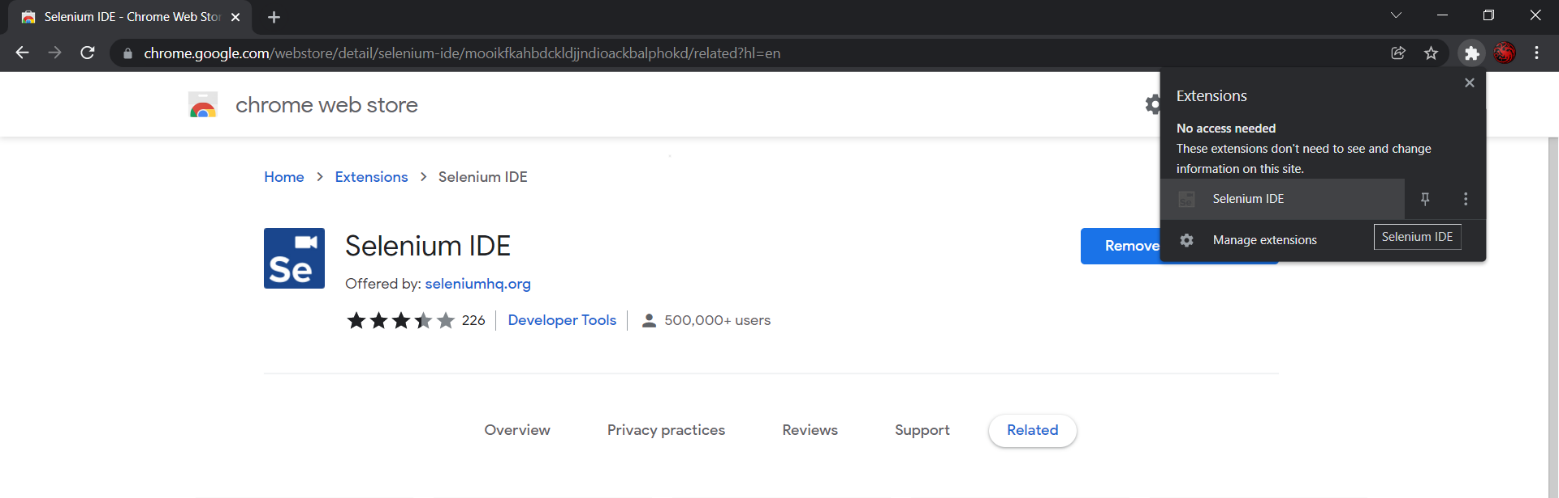


**Step 2**: This will lead to a dialog box appearing at the top of the screen. Read the permissions that the extension will have and click on ‘Add extension’.

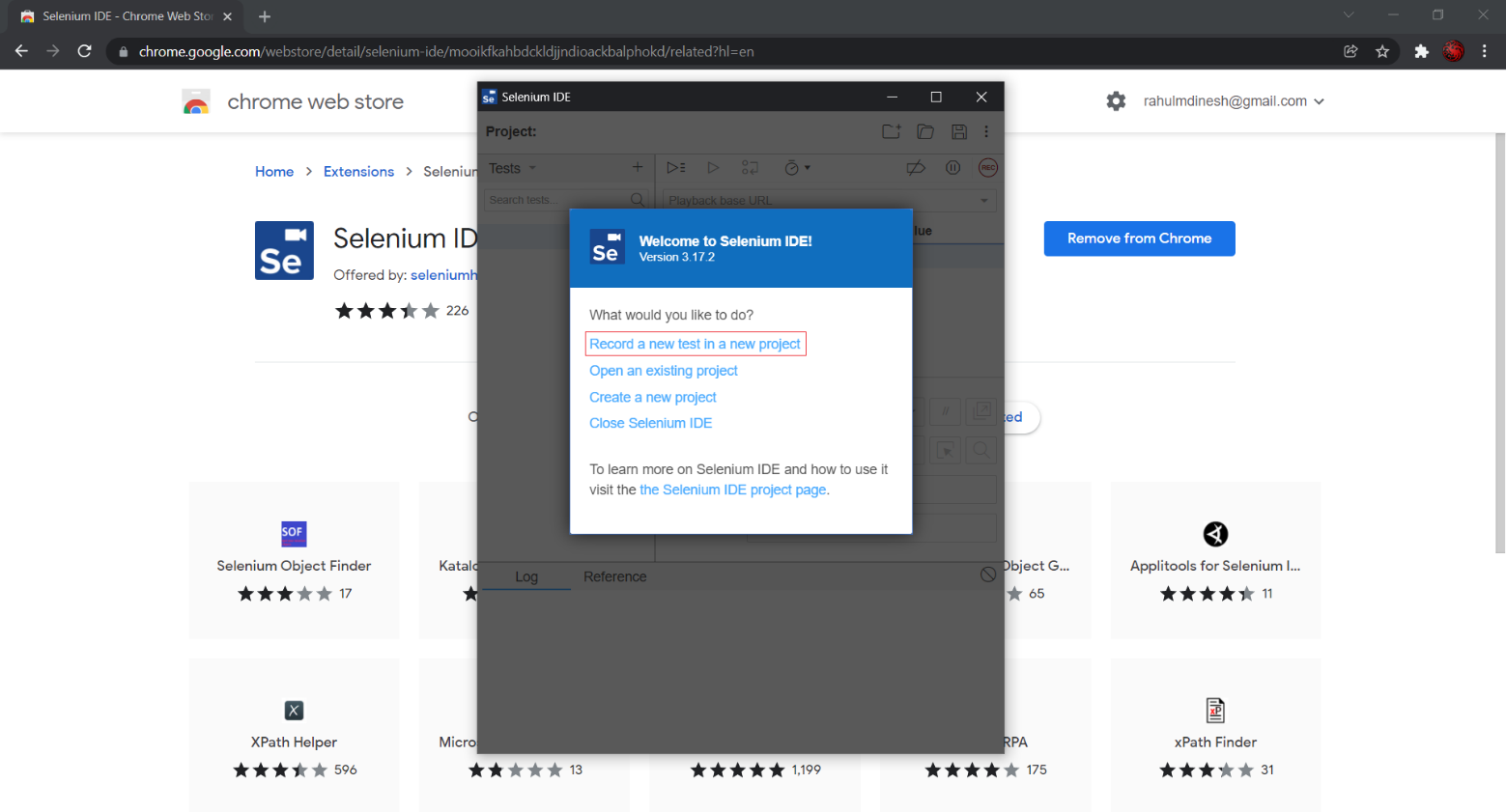
****

**Launching Selenium IDE and Creating a new Project**

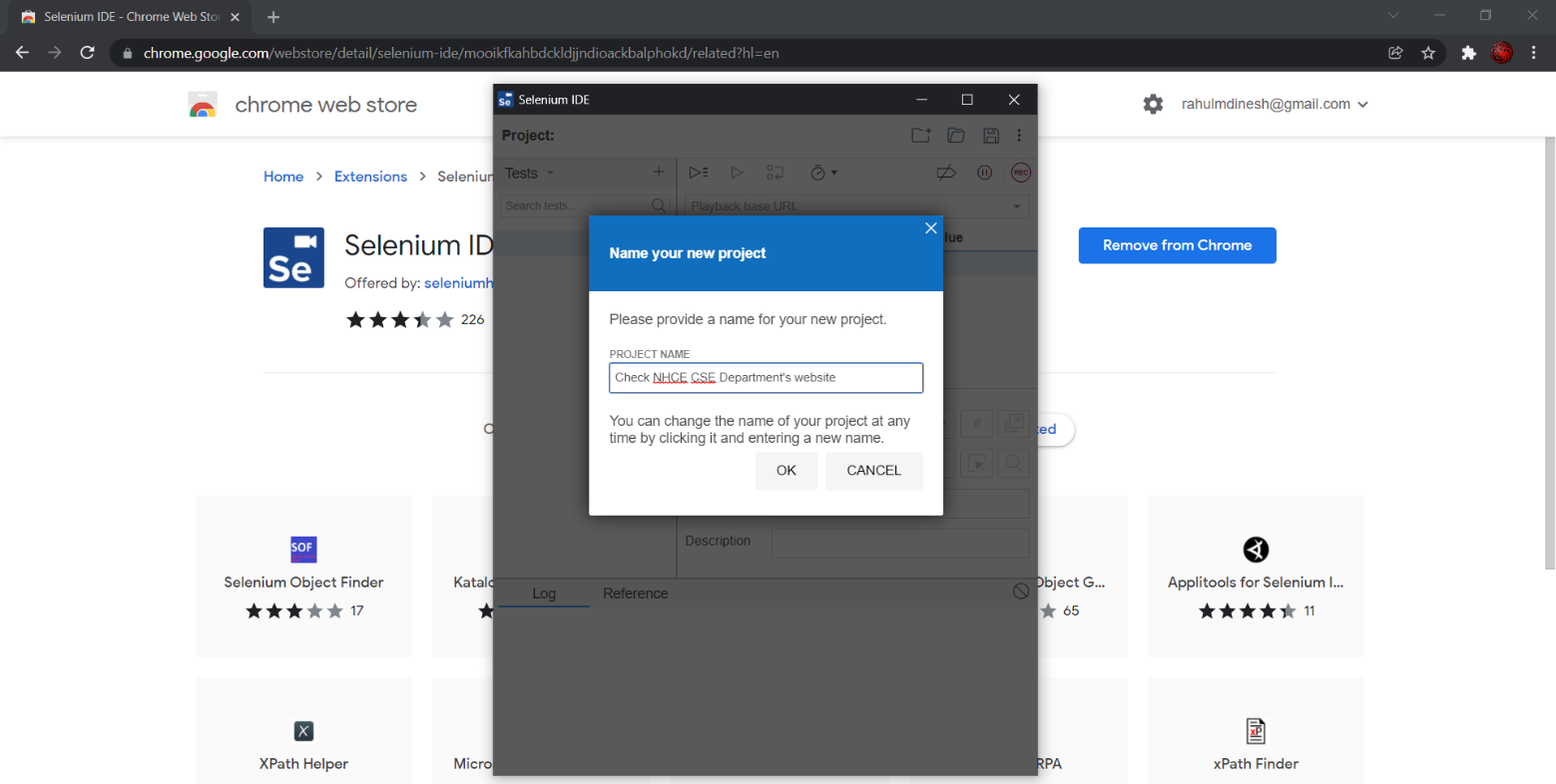
**Step 3**: Click on the Extension icon which looks like a puzzle piece, located in the top-right corner of the screen. Then click on ‘Selenium IDE’.



**Step 4**: After the Selenium IDE extension launches, select ‘Record a new test in a new project’.

****

**Step 5**: Give a suitable project name.



**Recording and Running an Automated Test**

**Step 6**: Provide the base URL. Then click ‘Start Recording’.

**Step 7**: Now, it will take you to the base URL. Perform some action like googling something or visiting a website. These actions are being recorded by Selenium IDE. Once you are done, go to the Selenium IDE window and click on ‘Stop Recording’. Then enter a suitable test name.

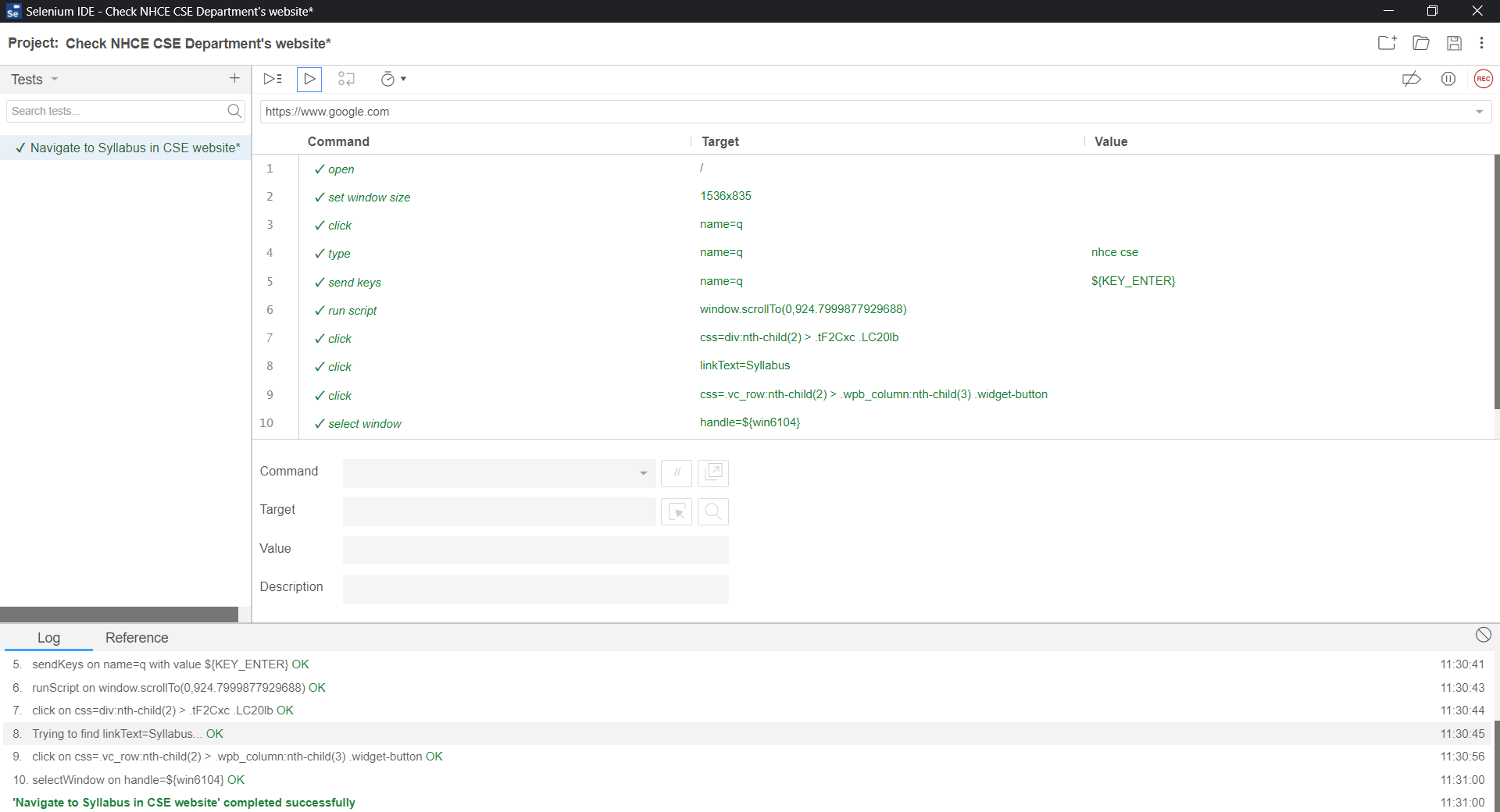
**Step 8**:Run the recorded steps as an automated test by clicking on the ‘Run current test’ button ( ▷ )

**TEST CASES**

Test Case 1 - Manual Steps:

* Open (http://www.google.com)
* Type “nhce cse” in the Google Search Input Box
* Hit the ‘Enter’ key or click on ‘Search’ button
* Scroll down to find the intended website and click on it.
* Once the website loads, click on an link in the website (here, ‘Syllabus’)
* Click on any button in the newly opened tab.

**EXECUTION**



**RESULT**: Thus, the demonstration of Selenium IDE for conducting test on a website is done successfully.

**Exp. No. :** 8

**Date :** 10-12-2021

**DEMONSTRATION OF SELENIUM WEBDRIVER FOR CONDUCTING TEST ON WEBSITE(S)**

Write an automated selenium script to login into a web page by using Selenium Web driver, automate any website using Java based Selenium Script.

**IMPLEMENTATION**

**Installation**

Step 1:

1. Go to the Chromium Driver website: <https://chromedriver.chromium.org>
2. Download the latest stable release
3. Ensure that your Chrome browser is updated to the latest version

Step 2:

1. Go to the Selenium website’s download page: <https://www.selenium.dev/downloads/>
2. Download the latest stable version of Selenium Server

Step 3:

1. Extract the jar file of Selenium Server Standalone and add it to the Eclipse project
2. Right click on the Project in the Project Explorer ⟶ Build Path ⟶ Configure Build Path ⟶ ‘Libraries’ tab ⟶ Add External Jar
3. Now, navigate to the Selenium Server Standalone jar, downloaded earlier and add it.

**Java based Selenium Script**

import org.openqa.selenium.By;

import org.openqa.selenium.chrome.ChromeDriver;

public class exp8 {

public static void main(String args[]){

System.setProperty("webdriver.chrome.driver", "R:\\ST Lab \\Jar\\chromedriver.exe");

ChromeDriver driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("file:///R:\\ST Lab\\login738.html");

driver.findElement(By.name("username")).sendKeys("rahulmd");

driver.findElement(By.name("password")).sendKeys("12345");

driver.findElement(By.name("submit")).click();

}

}

**HTML code for Login Page**

<form action="loginSuccess.html">

<label><b>Username</b></label>

<input name="username" type="text" required>

<br/>

<label><b>Password</b></label>

<input name="password" type="password" required>

<br/>

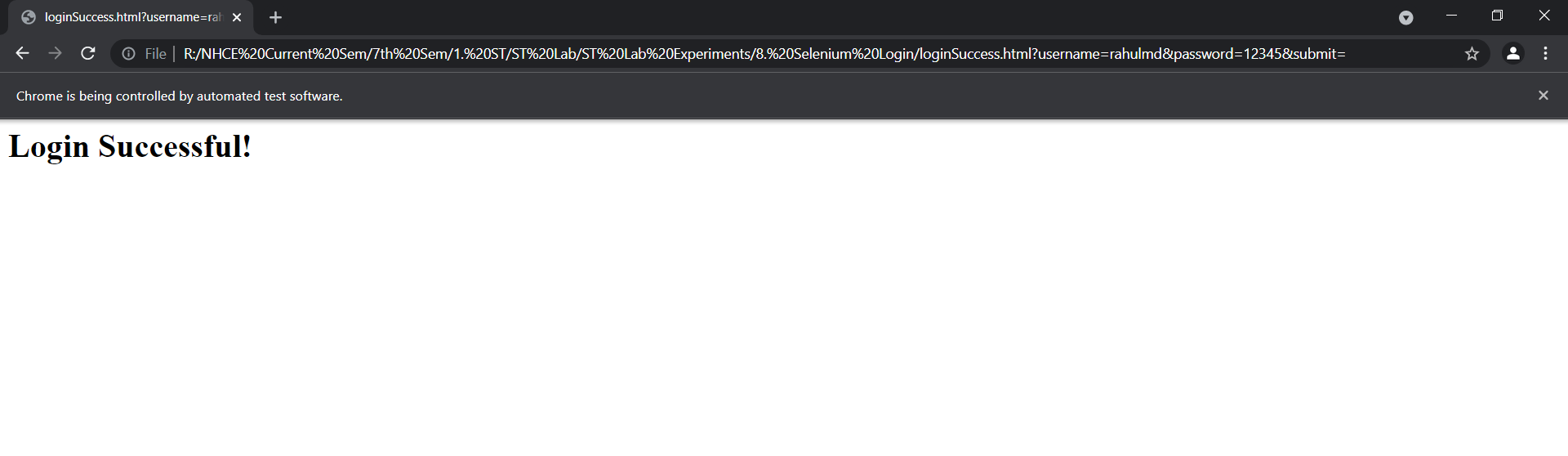
<button name="submit" type="submit">Login</button>

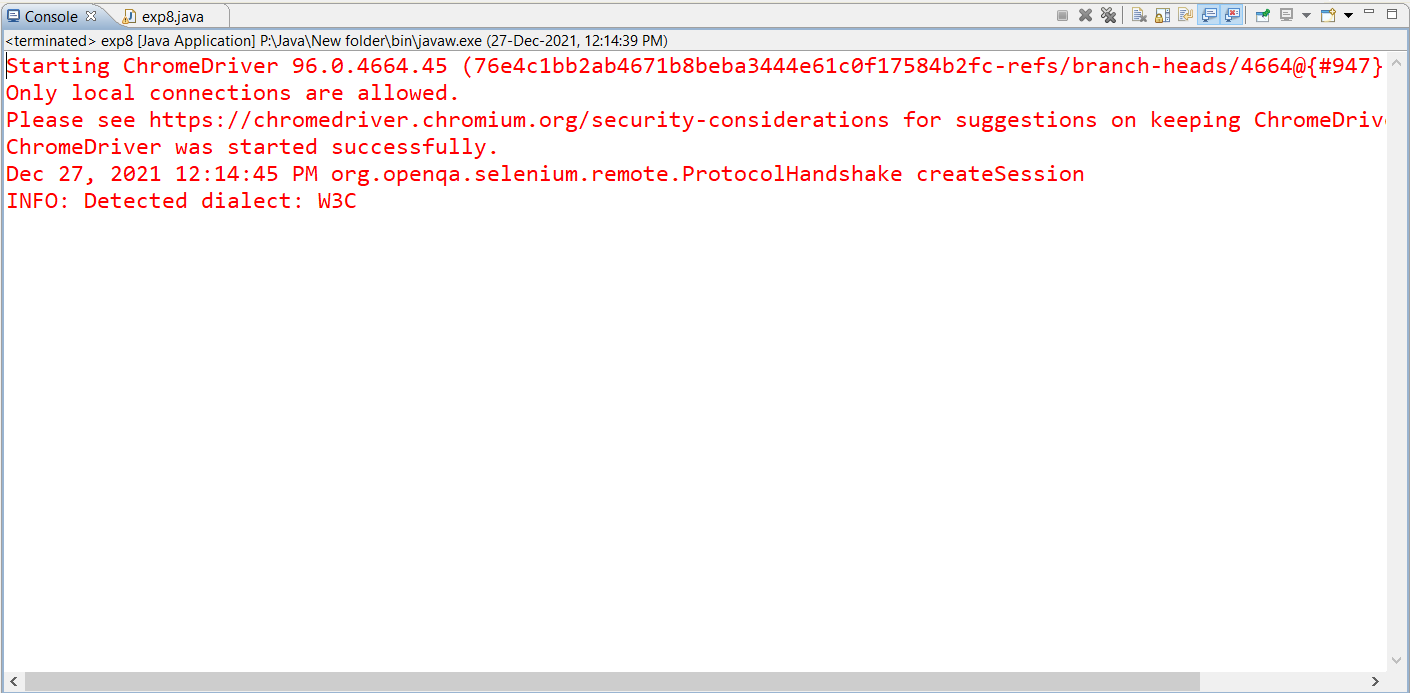
</form>

**HTML code for Login Sucess Page**

<h1>Login Successful!</h1>

**EXECUTION**





**RESULT:** Thus, the above program, written and executed using selenium web driver has successfully tested the login functionality of a sample web page.

**Exp. No. :** 9

**Date :** 24-12-2021

**DEMONSTRATION OF SELENIUM WEBDRIVER FOR CONDUCTING TEST ON WEBSITE(S)**

Write a test program to list the total number of objects present on a web page

**IMPLEMENTATION**

**Installation**

Step 1:

1. Go to the Chromium Driver website: <https://chromedriver.chromium.org>
2. Download the latest stable release
3. Ensure that your Chrome browser is updated to the latest version

Step 2:

1. Go to the Selenium website’s download page: <https://www.selenium.dev/downloads/>
2. Download the latest stable version of Selenium Server

Step 3:

1. Extract the jar file of Selenium Server Standalone and add it to the Eclipse project
2. Right click on the Project in the Project Explorer ⟶ Build Path ⟶ Configure Build Path ⟶ ‘Libraries’ tab ⟶ Add External Jar
3. Now, navigate to the Selenium Server Standalone jar, downloaded earlier and add it.

**Java based Selenium Script**

import org.openqa.selenium.By;

import org.openqa.selenium.chrome.ChromeDriver;

public class exp9 {

public static void main(String args[]){

System.setProperty("webdriver.chrome.driver", "R:\\ST Lab \\Jar\\chromedriver.exe");

ChromeDriver driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("https://www.msn.com");

List<WebElement> linksList = driver.findElements(By.xpath("//a"));

int linkCount = linksList.size();

System.out.println("Total number of links in the webpage: "+linkCount);

List<WebElement> elementsList = driver.findElements(By.xpath("//\*"));

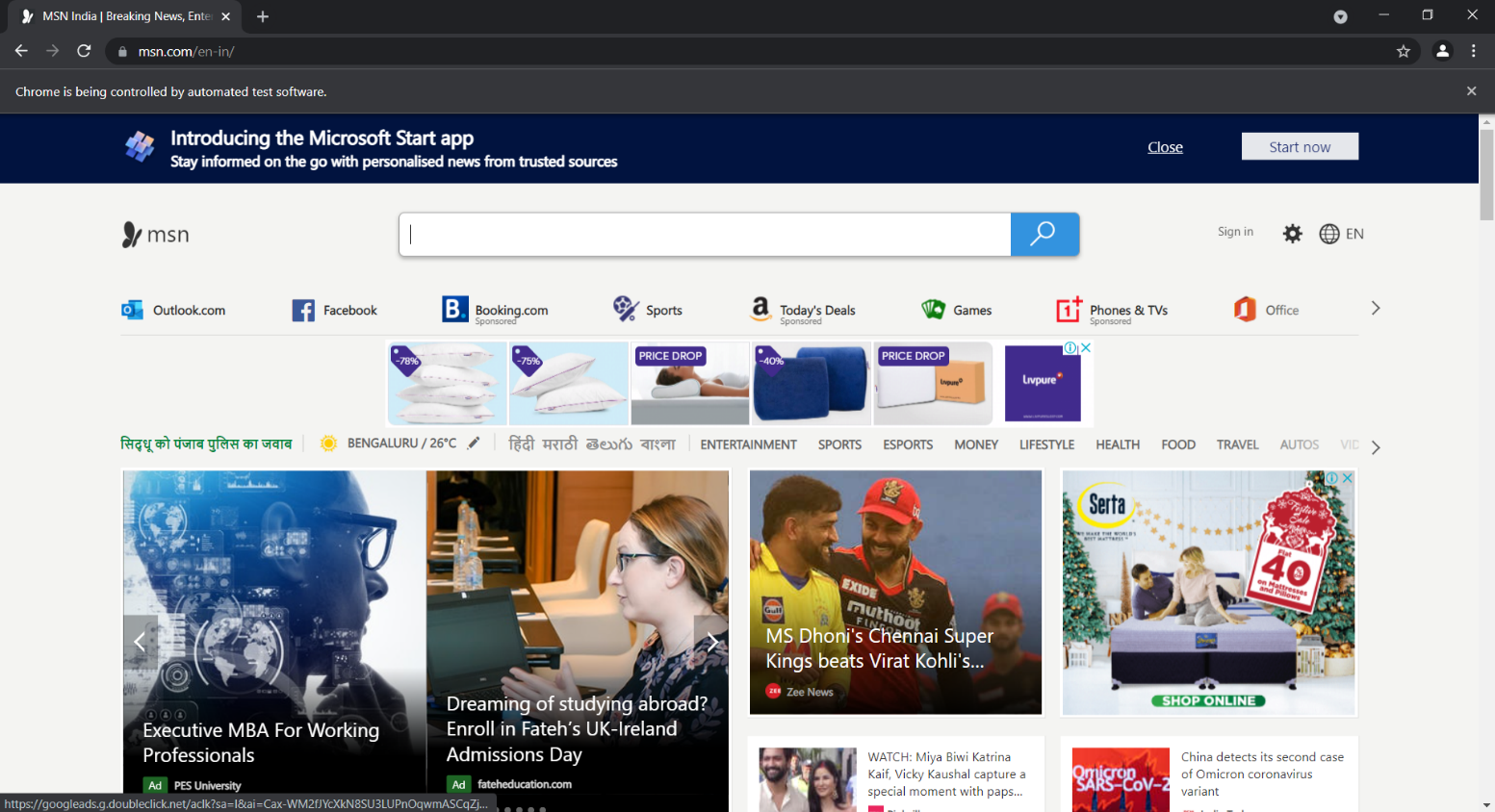
int elementsCount = elementsList.size();

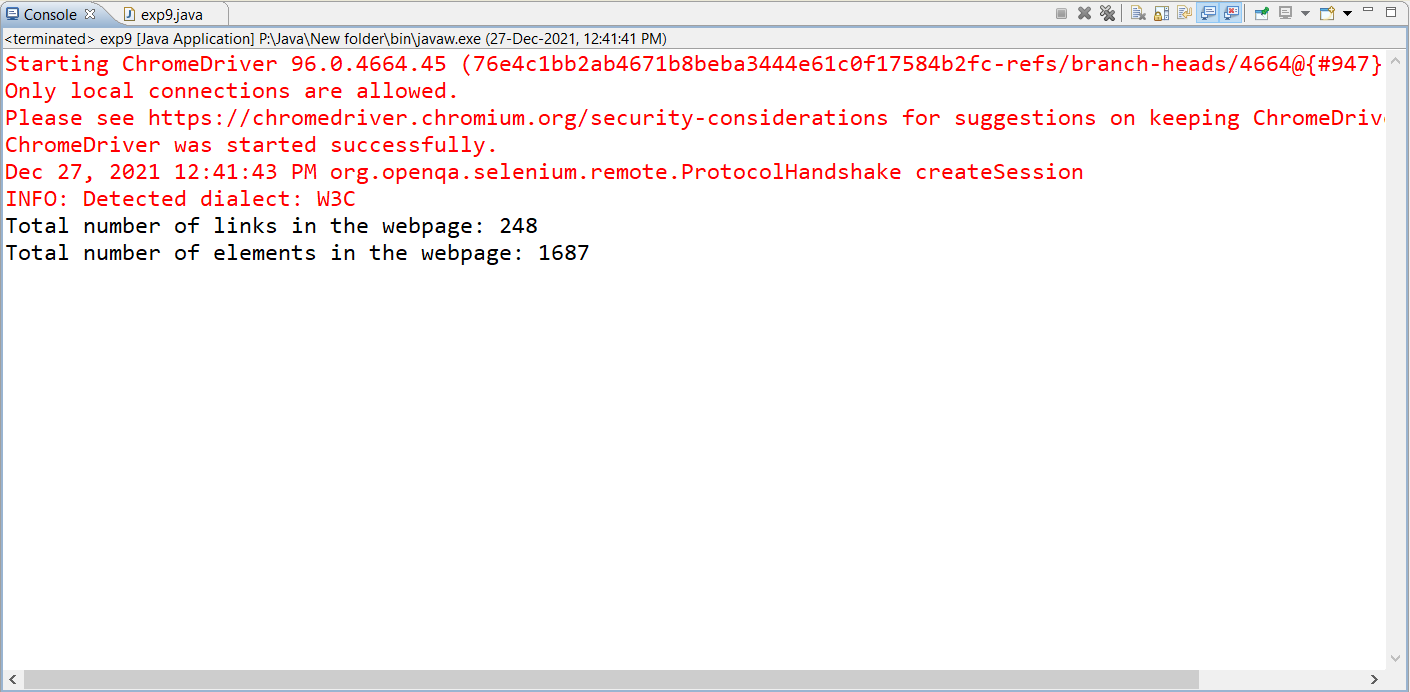
System.out.println("Total number of elements in the webpage: "+elementsCount);

}

}

**EXECUTION**

****

****

**RESULT:** Thus, the above program, written and executed using selenium web driver has successfully displayed the total number of links and elements in the given webpage.

**Exp. No. :** 10

**Date :** 31-12-2021

**DEMONSTRATION OF SELENIUM WEBDRIVER FOR CONDUCTING TEST ON WEBSITE(S)**

Write a test program to demonstrate URL and title check point

**IMPLEMENTATION**

**Installation**

Step 1:

1. Go to the Chromium Driver website: <https://chromedriver.chromium.org>
2. Download the latest stable release
3. Ensure that your Chrome browser is updated to the latest version

Step 2:

1. Go to the Selenium website’s download page: <https://www.selenium.dev/downloads/>
2. Download the latest stable version of Selenium Server

Step 3:

1. Extract the jar file of Selenium Server Standalone and add it to the Eclipse project
2. Right click on the Project in the Project Explorer ⟶ Build Path ⟶ Configure Build Path ⟶ ‘Libraries’ tab ⟶ Add External Jar
3. Now, navigate to the Selenium Server Standalone jar, downloaded earlier and add it.

**Java based Selenium Script**

import org.openqa.selenium.By;

import org.openqa.selenium.chrome.ChromeDriver;

public class exp10 {

public static void main(String args[]){

System.setProperty("webdriver.chrome.driver", "R:\\ST Lab \\Jar\\chromedriver.exe");

ChromeDriver driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("https://en.wikipedia.org/wiki/Wikipedia");

String url = driver.getCurrentUrl();

System.out.println("Current URL: " + url);

if(url.equals("https://en.wikipedia.org/wiki/Wikipedia"))

System.out.println("URL matches!");

else

System.out.println("URL doesn't match.");

driver.get("https://www.google.com");

String title = driver.getTitle();

System.out.println("Current Title: " + title);

if(title.equals("Google"))

System.out.println("Title matches!");

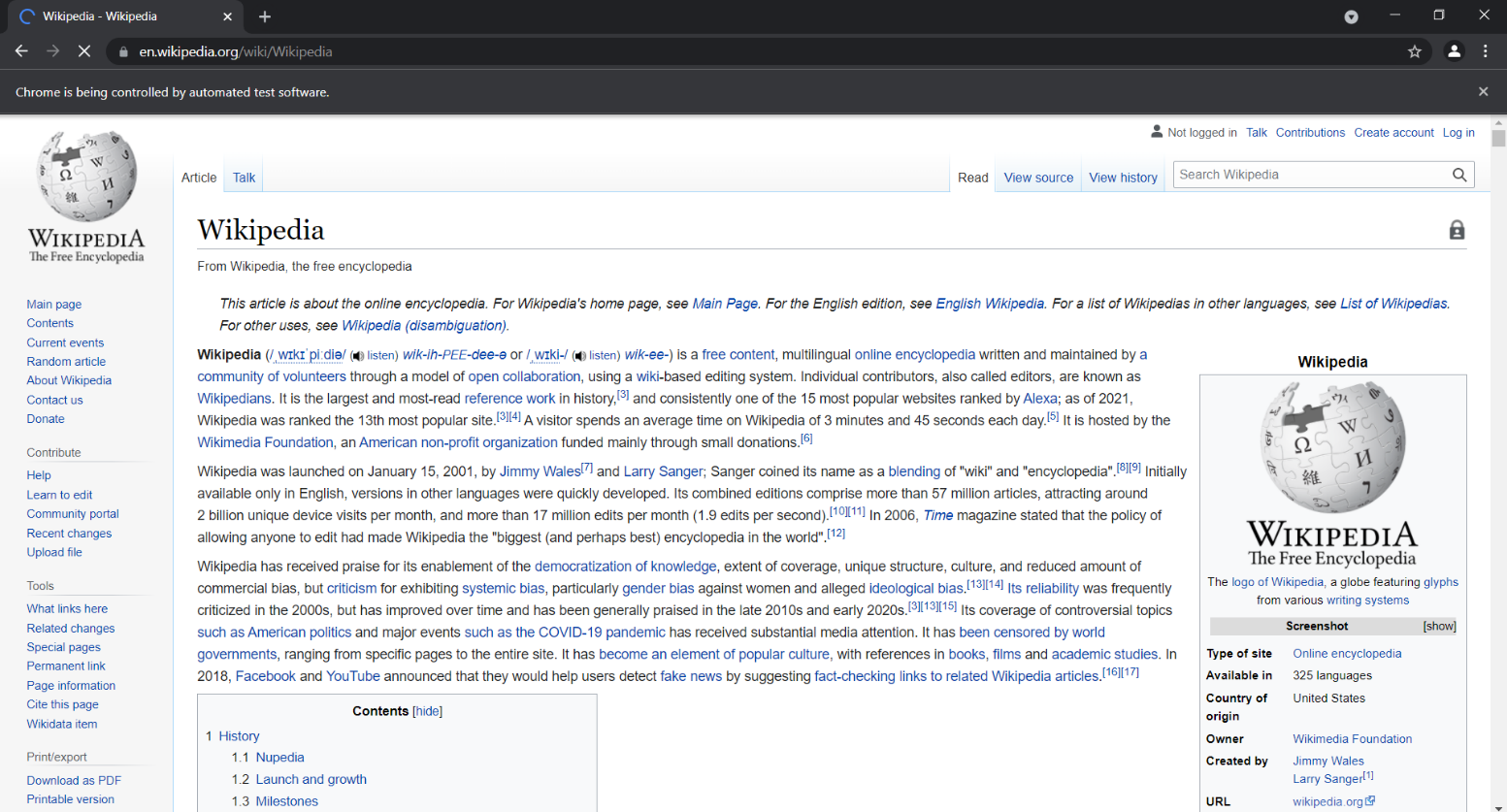
else

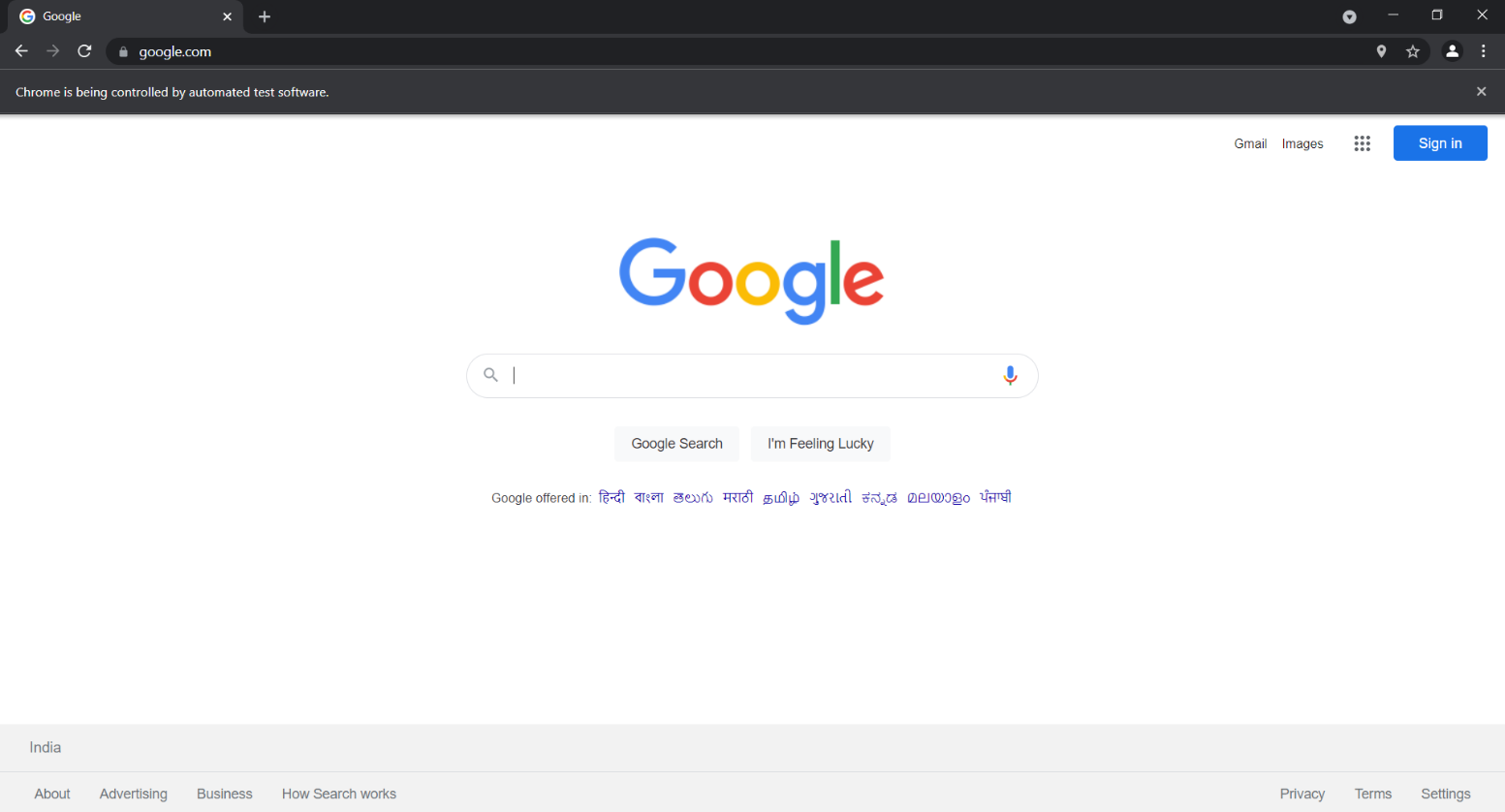
System.out.println("Title doesn't match.");

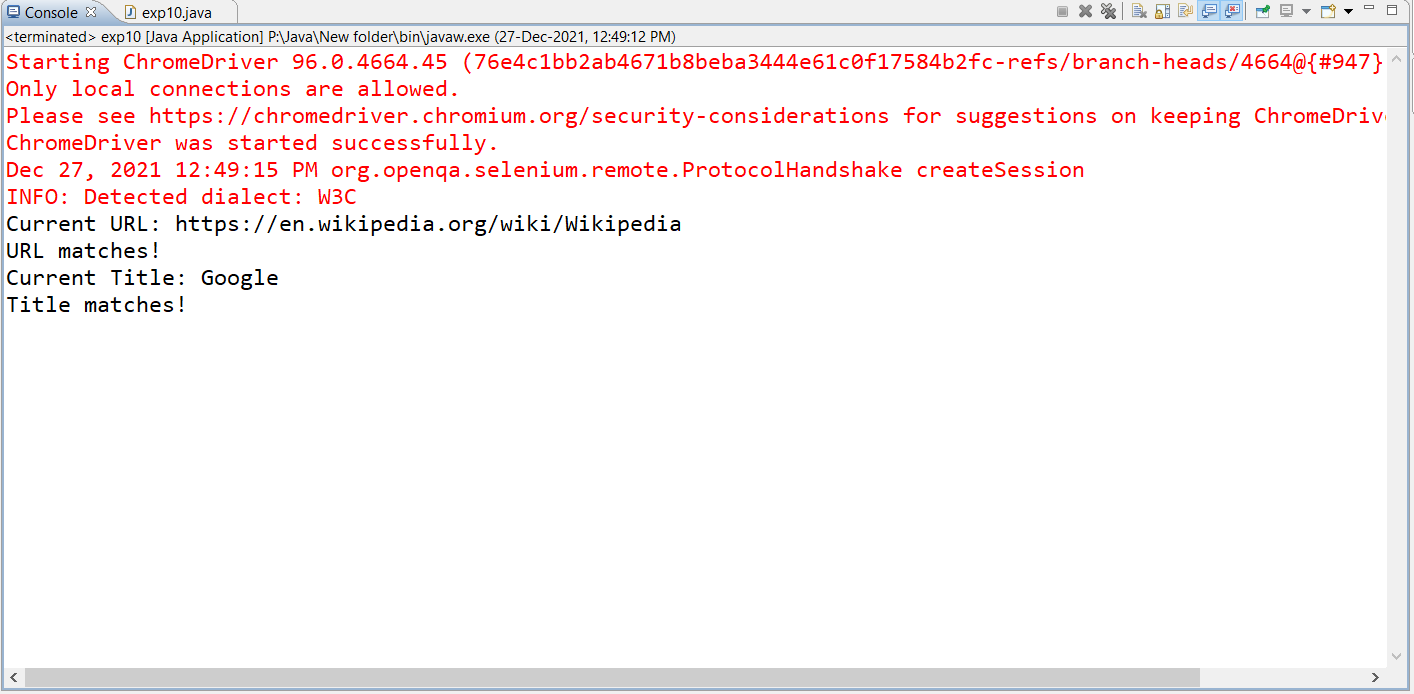
}

}

**EXECUTION**

****

****

****

**RESULT:** Thus, the above program, written and executed using selenium web driver has successfully displayed and verified the URL and title of two different websites using check point.

**Exp. No. :** 11

**Date :** 31-12-2021

**DEMONSTRATION OF SELENIUM IDE & WEBDRIVER FOR CONDUCTING TEST ON WEBSITE(S)**

Write a test program to demonstrate selecting and deselecting option from multi select dropdown

**IMPLEMENTATION**

**Installation**

Step 1:

1. Go to the Chromium Driver website: <https://chromedriver.chromium.org>
2. Download the latest stable release
3. Ensure that your Chrome browser is updated to the latest version

Step 2:

1. Go to the Selenium website’s download page: <https://www.selenium.dev/downloads/>
2. Download the latest stable version of Selenium Server

Step 3:

1. Extract the jar file of Selenium Server Standalone and add it to the Eclipse project
2. Right click on the Project in the Project Explorer ⟶ Build Path ⟶ Configure Build Path ⟶ ‘Libraries’ tab ⟶ Add External Jar
3. Now, navigate to the Selenium Server Standalone jar, downloaded earlier and add it.

**Java based Selenium Script**

import java.util.List;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.support.ui.Select;

public class exp11 {

public static void main(String[] args) throws InterruptedException {

System.setProperty("webdriver.chrome.driver", "R:\\ST Lab \\Jar\\chromedriver.exe");

ChromeDriver driver = new ChromeDriver();

driver.manage().window().maximize();

driver.get("file:///R:\\ST Lab\\multiselect.html");

Select select = new Select(driver.findElement(By.id("depts")));

System.out.println("The multiselect options are: ");

List<WebElement> options = select.getOptions();

for(WebElement option: options)

System.out.println(option.getText());

System.out.println("Is the selected element a multiselect element?: "+ select.isMultiple());

if(select.isMultiple()){

System.out.println("Selecting option ECE using its index.");

select.selectByIndex(2);

Thread.sleep(4000);

System.out.println("Selecting option ISE using its value.");

select.selectByValue("ise");

Thread.sleep(4000);

System.out.println("Selecting option CSE using its visible text.");

select.selectByVisibleText("CSE");

Thread.sleep(4000);

System.out.println("The selected options are: ");

options = select.getAllSelectedOptions();

for(WebElement option: options)

System.out.println(option.getText());

System.out.println("Deselecting option ECE using its index.");

select.deselectByIndex(2);

Thread.sleep(4000);

System.out.println("Deselecting option ISE using its value.");

select.deselectByValue("ise");

Thread.sleep(4000);

System.out.println("The selected values after deselecting some options are: ");

options = select.getAllSelectedOptions();

for(WebElement option: options)

System.out.println(option.getText());

System.out.println("Deselecting all options.");

select.deselectAll();

}

}

}

**HTML code for Multiselect Page**

<h1>

Select and Deselect Departments

</h1>

<form>

<select multiple name=*"depts"* id=*"depts"*>

<option value=*"cse"*>CSE</option>

<option value=*"ise"*>ISE</option>

<option value=*"ece"*>ECE</option>

<option value=*"eee"*>EEE</option>

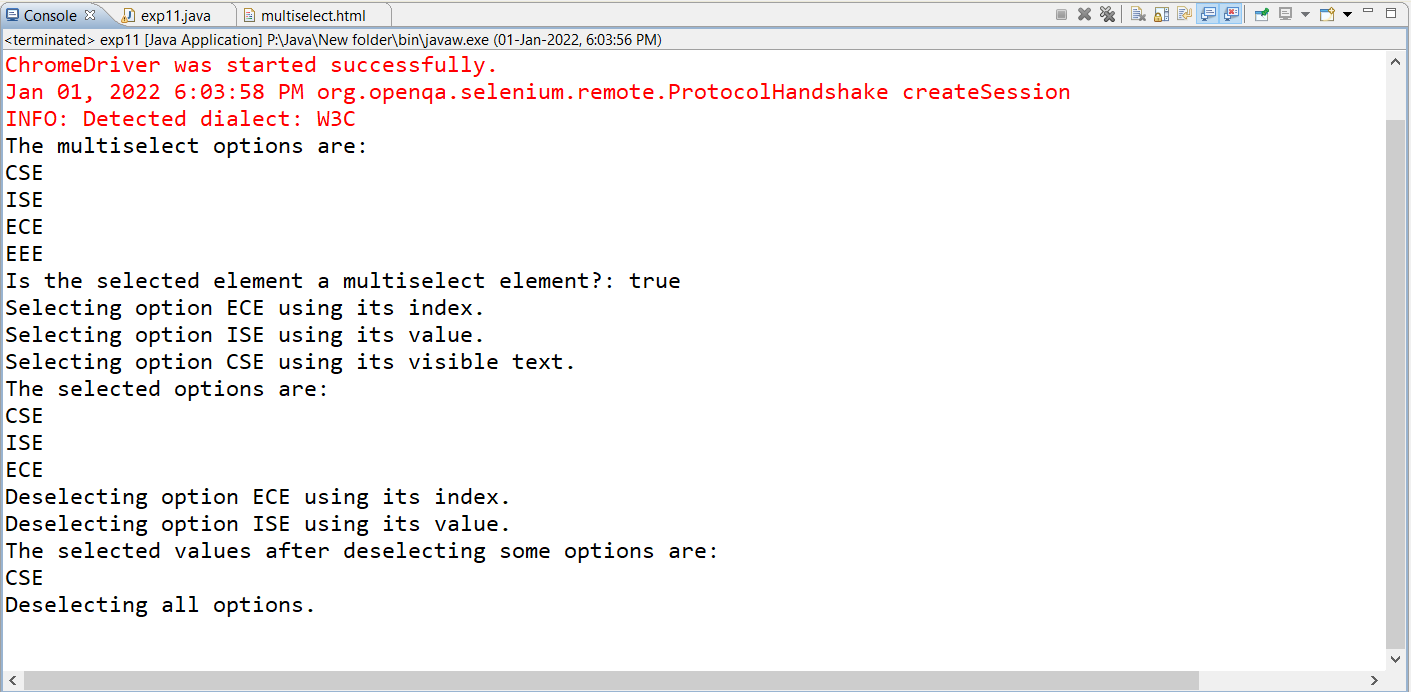
</select>

</form>

**EXECUTION**

**The HTML page having a Multiselect Element**

****

****

**RESULT:** Thus, the above program, written and executed using selenium web driver has successfully tested the functionality of a multiselect element by selecting and deselecting various options.

**Exp. No. :** 12

**Date :** 07-01-2022

**DEMONSTRATION OF SELENIUM IDE & WEBDRIVER FOR CONDUCTING TEST ON WEBSITE(S)**

Write a test program to demonstrate Synchronization

**IMPLEMENTATION**

**Installation**

Step 1:

1. Go to the Chromium Driver website: <https://chromedriver.chromium.org>
2. Download the latest stable release
3. Ensure that your Chrome browser is updated to the latest version

Step 2:

1. Go to the Selenium website’s download page: <https://www.selenium.dev/downloads/>
2. Download the latest stable version of Selenium Server

Step 3:

1. Extract the jar file of Selenium Server Standalone and add it to the Eclipse project
2. Right click on the Project in the Project Explorer ⟶ Build Path ⟶ Configure Build Path ⟶ ‘Libraries’ tab ⟶ Add External Jar
3. Now, navigate to the Selenium Server Standalone jar, downloaded earlier and add it.

**Java based Selenium Script (Implicit Wait)**

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.NoSuchElementException;

import org.openqa.selenium.By;

import org.openqa.selenium.chrome.ChromeDriver;

public class exp12\_implicit {

public static void main(String[] args){

System.setProperty("webdriver.chrome.driver", "R:\\ST Lab \\Jar\\chromedriver.exe");

ChromeDriver driver = new ChromeDriver();

driver.manage().window().maximize();

String eText = "Welcome"; //Expected Text

String aText\_1=""; //Actual Text (Test 1)

String aText\_2=""; //Actual Text (Test 2)

//\*\*\*Using Implicit Wait\*\*\*

driver.get("file:///R:\\\\ST Lab\\sample.html");

driver.manage().timeouts().implicitlyWait(10, TimeUnit.SECONDS);

//\*\*\*Test 1 - Will Pass as element exists\*\*\*

try{

aText\_1 = driver.findElement(By.id("welcome")).getText();

if (aText\_1.equals(eText))

System.out.println("Test 1 Passed using Implicit Wait");

}

catch (NoSuchElementException e){

System.out.println("Test 1 Failed using Implicit Wait");

}

//\*\*\*Test 2 - Will Fail after waiting for 10s as element doesn't exist\*\*\*\*

try{

aText\_2 = driver.findElement(By.id("abcd")).getText();

if (aText\_2.equals(eText))

System.out.println("Test 2 Passed using Implicit Wait");

}

catch (NoSuchElementException e){

System.out.println("Test 2 Failed using Implicit Wait");

}

}

}

**Java based Selenium Script (Explicit Wait)**

import org.openqa.selenium.By;

import org.openqa.selenium.TimeoutException;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.support.ui.ExpectedConditions;

import org.openqa.selenium.support.ui.WebDriverWait;

public class exp12\_explicit {

public static void main(String[] args){

System.setProperty("webdriver.chrome.driver", "R:\\ST Lab \\Jar\\chromedriver.exe");

ChromeDriver driver = new ChromeDriver();

driver.manage().window().maximize();

String eText = "Welcome"; //Expected Text

String aText\_1=""; //Actual Text (Test 1)

String aText\_2=""; //Actual Text (Test 2)

//\*\*\*Using Explicit Wait\*\*\*

driver.get("file:///R:\\ST Lab\\sample.html");

WebDriverWait wait = new WebDriverWait(driver, 10);

//\*\*\*Test 1 - Will Pass as element exists\*\*\*

try{

WebElement text\_1 =

wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("welcome")));

aText\_1 = text\_1.getText();

if (aText\_1.equals(eText))

System.out.println("Test 1 Passed using Explicit Wait");

}

catch (TimeoutException e){

System.out.println("Test 1 Failed using Explicit Wait");

}

//\*\*\*Test 2 - Will Fail after waiting for 10s as element doesn't exist\*\*\*\*

try{

WebElement text\_2 =

wait.until(ExpectedConditions.visibilityOfElementLocated(By.id("abcd")));

aText\_2 = text\_2.getText();

if (aText\_2.equals(eText))

System.out.println("Test 2 Passed using Explicit Wait");

}

catch (TimeoutException e){

System.out.println("Test 2 Failed using Explicit Wait");

}

}

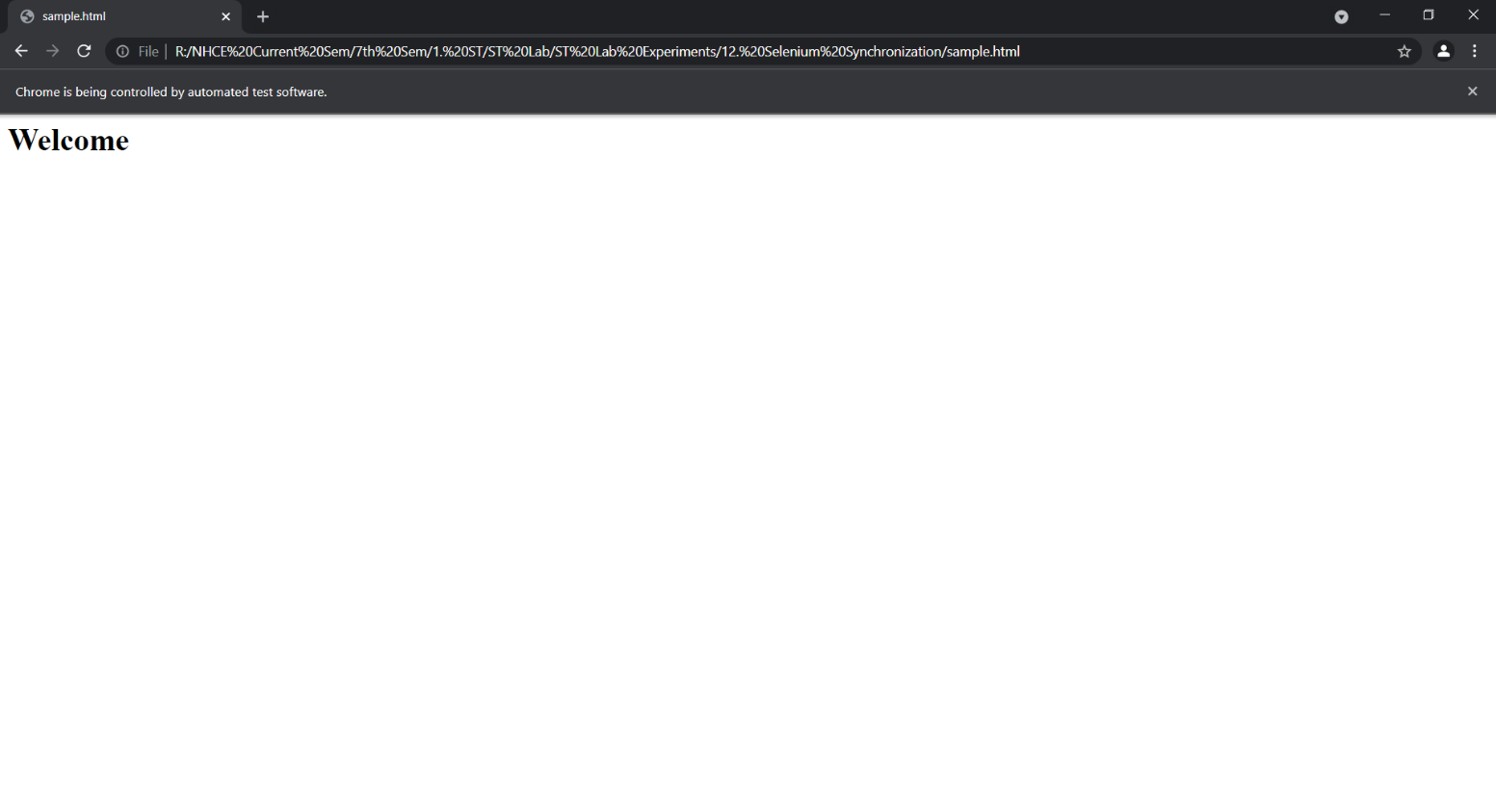
}

**HTML code for Sample Page**

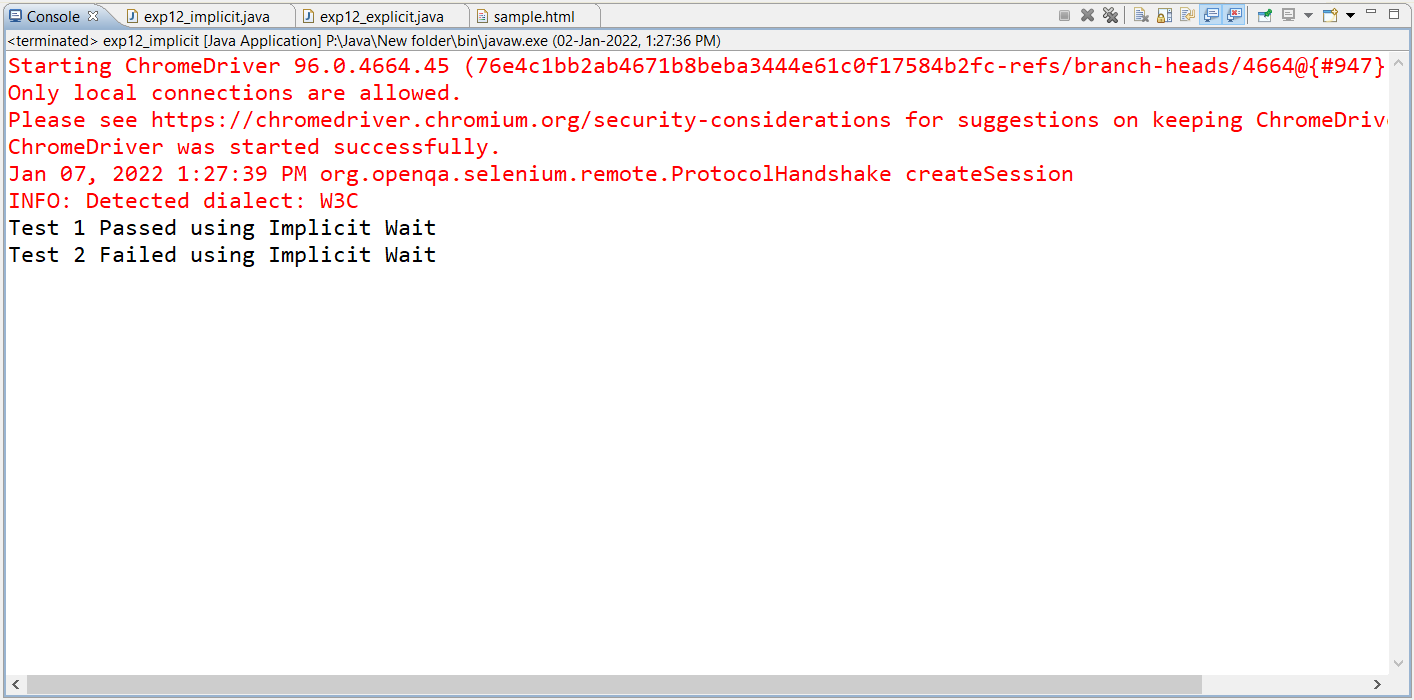
<h1 id="welcome">Welcome</h1>

**EXECUTION**

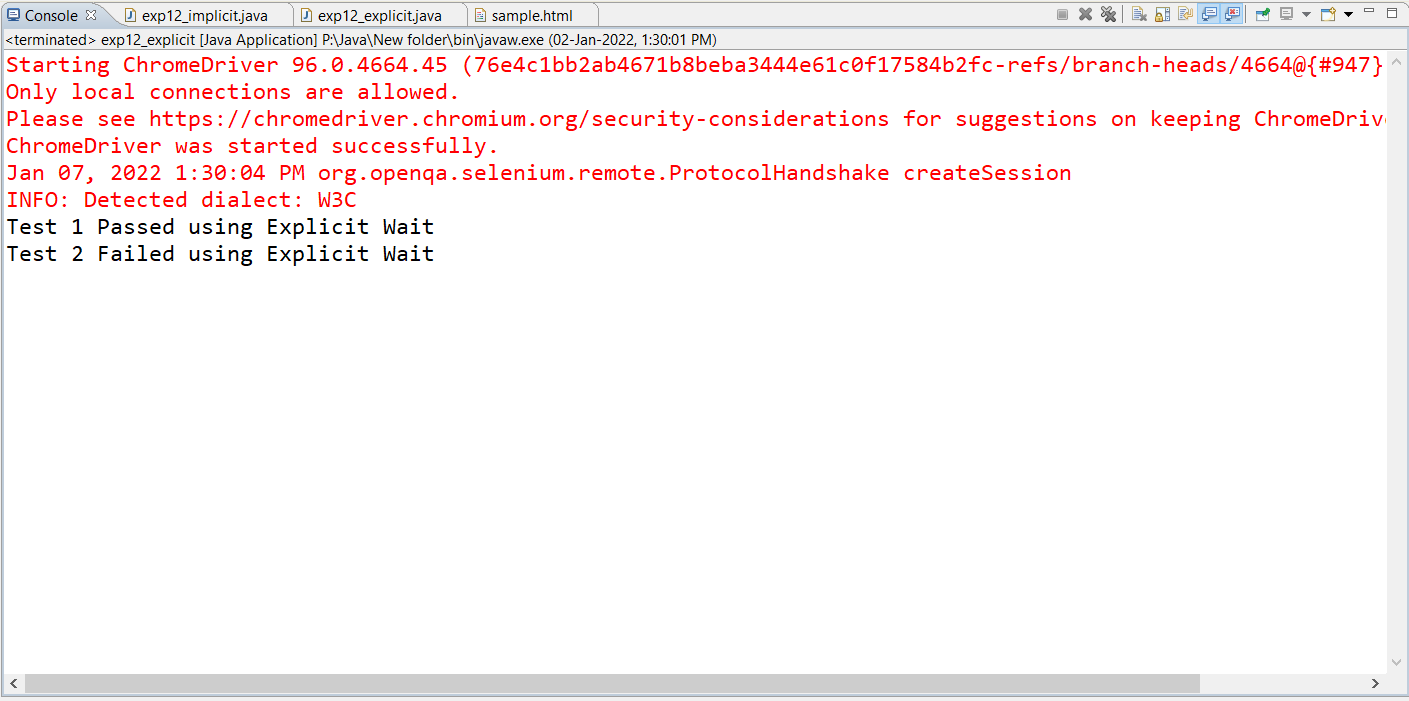
**The Sample HTML page**



**Implicit Wait**

****

**Explicit Wait**



**RESULT:** Thus, the above program, written and executed using selenium web driver has successfully demonstrated synchronization using implicit and explicit waits.