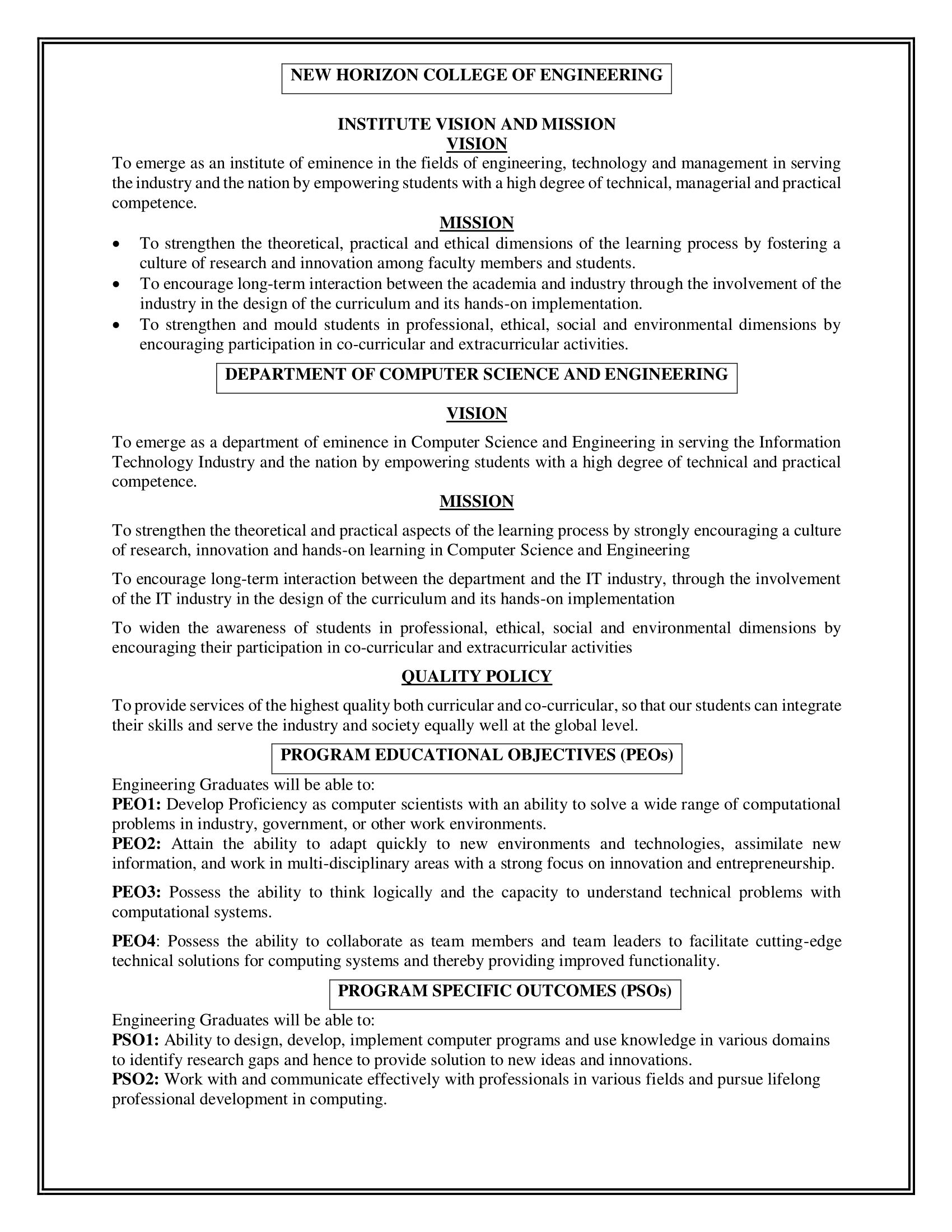
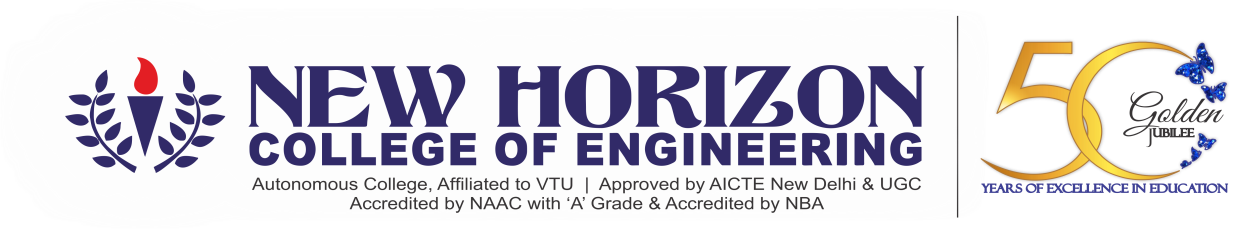




**PRACTICAL RECORD BOOK**

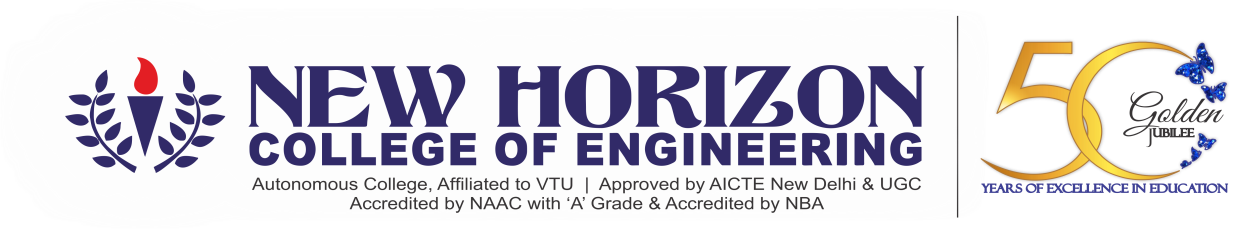
|  |
| --- |
| **Name**  GOURAV RUDRAWAR  1NH18CS067  2021 - 2022  **USN Year**  B.E. in CSE  A  7  **Program Semester Section**  SOFTWARE TESTING LAB  20CSL75A  **Course Course Code** |





*Laboratory Certificate*

|  |  |  |
| --- | --- | --- |
| *This is to certify that*  *Mr. ........*GOURAV RUDRAWAR*.........*  *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:** GOURAV RUDRAWAR  **USN:** 1NH18CS067  **Sem/Sec:** 7 - A  **Course Code:** 20CSL75A  Signature of Student  **Head of the Department**  **Signature of the Faculty In-charge** |



**LABORATORY PERFORMANCE EVALUATION SHEET**

**Name of Student:** GOURAV RUDRAWAR

**USN:** 1NH18CS067

**Lab Course:** SOFTWARE TESTING LAB

**Course Code:** 20CSL75A

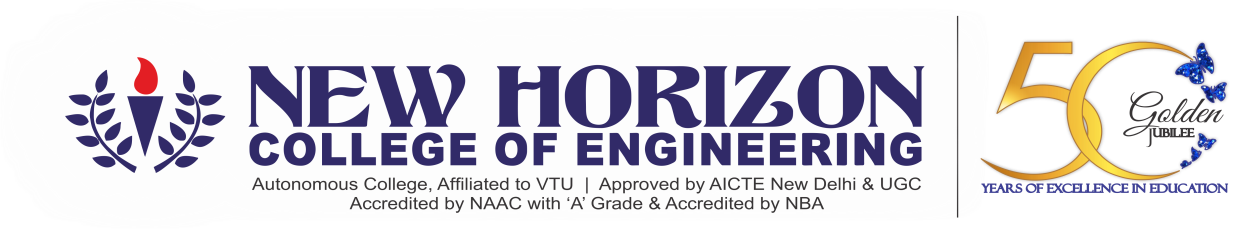
**Sem/Sec:** 7 - A

**Session:** ODD Sem 2021-22

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

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

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)** |
| **Test 1** |  |  |  |  |  |
| **Test 2** |  |  |  |  |  |

**CIE - Marks Obtained**

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

|  |
| --- |
|  |

**PROGRAM NO.: 1**

**Exp. No. : 1**

**Date :**

**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.\*;

**public** **class** Atm\_ST {

**public** **static** **void** main(String args[]){

Scanner sc=**new** Scanner(System.***in***);

**int** balance=10000, pin=1234, time=0, amount;

**boolean** deposit=**true**, flag=**true**, act=**true**;

System.***out***.println("Welcome to The Himalayan Bank.\n");

**while**(flag==**true**){

System.***out***.println("Enter Pin Number: ");

**int** userpin=sc.nextInt();

**if**(userpin==pin){

**while**(act==**true**){

System.***out***.println("Enter the Account type: \n1-Savings\n2-Current\n");

**int** actype=sc.nextInt();

**if**(actype!=1 && actype!=2)

{System.***out***.println("Invalid Account Type");

System.***out***.println("Do you want to try again? 1-Yes 2-No");

**int** c=sc.nextInt();

**if**(c==1) act=**true**;

**else** act=**false**;

}

**else**{

System.***out***.println("Press 1 for Withdrawal\nPress 2 for Deposition");

**int** x=sc.nextInt();

**while**(x==1){

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

amount=sc.nextInt();

**if**(amount>balance)

{

System.***out***.println("Account balance is lesser than withdrawal amount.");

System.***out***.println("Do you want to try again? 1-Yes 2-No");

**int** ch=sc.nextInt();

**if**(ch==1) x=1;

**else** x=0;

System.***out***.println("\n");

}

**else** {

System.***out***.println("Transaction is successful.");

System.***out***.println("Available balance is: "+(balance-amount)+"\n\n");

x=0;

act=**false**;

}

}

**if**(x==2){

System.***out***.println("Kindly place the amount in the ATM.");

**if**(deposit==**true**) System.***out***.println("Transaction is successful.");

**else** System.***out***.println("Transaction is unsuccessful.");

act=**false**;

}}

flag=**false**;

}

}

**else**{

**if**(time<3)System.***out***.println("Invalid pin. Please enter correct pin.\n\n");

**if**(time==3) flag=**false**;

time++;

}

}

}

}

**TEST CASES:**

**Example:**

**TEST** **CASE 1:** Unsuccessful operation due to enter wrong PIN number 3 times.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Invalid PIN Number | | |
| Project ID: | ATM\_01 | | Original Author: | | GOURAV | | |
| Test Objective: | This test case is to verify the functionality with invalid pin number | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert valid card in the insertion point of ATM | Valid ATM card | | ATM should display language page with following objects English, Kannada, Hindi | | Pass |  |
| 2 | Select the preferred language | language | | ATM should display the PIN number entry screen in selected language | | Pass |  |
| 3 | Enter the invalid pin number | Invalid PIN number | | ATM does not validate PIN and prompts customer to reenter PIN. | | Pass |  |
| 4 | Reenter incorrect PIN | Invalid PIN number | | ATM does not validate PIN and prompts customer to reenter PIN | | Pass |  |
| 5 | Reenter incorrect PIN | Invalid PIN number | | ATM does not validate PIN | | Pass |  |

**TEST CASE 2:** Unsuccessful operation due to invalid account type.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | INVALID ACC TYPE | | |
| Project ID: | ATM\_02 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify unsuccessful operation due to invalid account type. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert  valid card | Valid card | | Enter PIN | | Pass |  |
| 2 | Enter the PIN no. | Valid PIN | | Select account type | | Pass |  |
| 3 | Enter Invalid account type | Invalid entered | | Invalid account type entered | | Pass |  |

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Valid withdrawn. | | |
| Project ID: | ATM\_03 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify successful selection of withdrawn amount. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert valid card in the insertion point of ATM | Valid ATM card | | ATM should display language page with following objects English, Kannada, Hindi | | Pass |  |
| 2 | Enter valid PIN | Valid PIN | | Select account type | | Pass |  |
| 3 | Select savings account | Savings account | | Choose.  1.balance  2.withdraw  3.deposit  4.exit | | Pass |  |
| 4 | Select withdrawal | Withdraw | | Enter amount. | | Pass |  |
| 5 | Enter valid amount | Valid amount | | Amount withdrawn. | | pass |  |

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | WITHDRAWN AMOUNT GREATER | | |
| Project ID: | ATM\_04 | | Original Author: | | GOURAV | | |
| Test Objective: | TO VERIFY SELECTED MESSAGE AS AMOUNT IS GREATER THAN AVAILABLE BALANCE. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Valid card | Insert valid card | | Enter PIN | | Pass |  |
| 2 | Valid PIN, savings account | Enter valid PIN and select savings account | | 1.balance  2.withdraw  3.deposit  4.exit | | Pass |  |
| 3 | withdrawn | Enter withdrawal amount | | Enter amount to be withdrawn | | Pass |  |
| 4 | Invalid withdrawn amount | Amount entered is greater than balance. | | Amount entered is invalid. | | pass |  |

**TEST CASE 5:** Machine is accepting ATM card

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | ATM card accepted. | | |
| Project ID: | ATM\_05 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify the machine is accepting card. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert valid card in the insertion point of ATM | Valid ATM card | | Enter the PIN no. | | Pass |  |

**TEST CASE 6:** Machine is rejecting expired card.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Reject expired. | | |
| Project ID: | ATM\_06 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify the rejecting expired ATM card. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert expired card | Expired card | | Invalid card | | Pass |  |

**TEST CASE 7:** Successful entry of PIN no.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Valid withdrawn. | | |
| Project ID: | ATM\_07 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify successful entry of PIN | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert valid card in the insertion point of ATM | Valid ATM card | | Enter PIN. | | Pass |  |
| 2 | Enter valid PIN | Valid PIN | | Select account type | | Pass |  |

**TEST CASE 8:** Successful selection of language.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Successful language selection. | | |
| Project ID: | ATM\_08 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify the functionality with invalid pin number | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert valid card in the insertion point of ATM | Valid ATM card | | Enter PIN. | | Pass |  |
| 2 | Enter valid PIN | Valid PIN | | ATM should display language page with following objects English, Kannada, Hindi | | Pass |  |
| 3 | Enter language | Valid language. | | Select amount. | | Pass |  |

**TEST CASE 9:** Successful selection of account type.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Successful account selection. | | |
| Project ID: | ATM\_09 | | Original Author: | | GOURAV | | |
| Test Objective: | Successful selection of account type. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Insert valid card in the insertion point of ATM | Valid ATM card | | ATM should display language page with following objects English, Kannada, Hindi | | Pass |  |
| 2 | Enter valid PIN | Valid PIN | | Select account type | | Pass |  |
| 3 | Select savings account | Valid account | | Account selected | | Pass |  |

**TEST CASE 10:** Selected message due to amount greater than day limit.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Display message. | | |
| Project ID: | ATM\_10 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify successful selected message as amount greater than day limit. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Identify expected message | Enter amount above limit | | Withdrawal limit exceeded. | | Pass |  |

**TEST CASE 11:** unsuccessful withdraw operation due to lack of money.

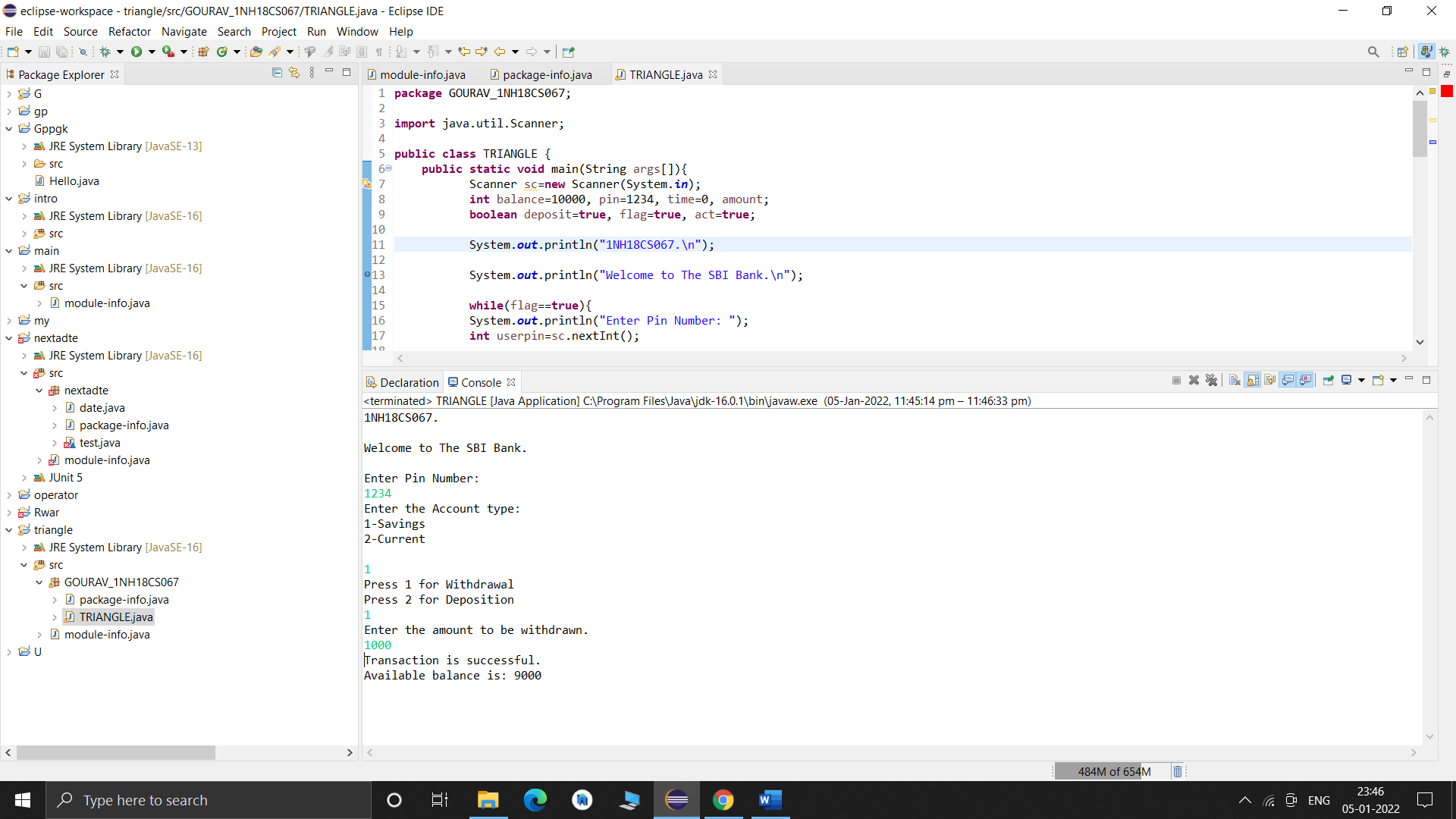
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Unsuccessful withdraw. | | |
| Project ID: | ATM\_11 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify unsuccessful withdraw operation due to lack of money. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Unsuccessful withdraw operation | Invalid withdraw amount. | | ATM doesn’t support this withdrawal and balance is displayed. | | Pass |  |

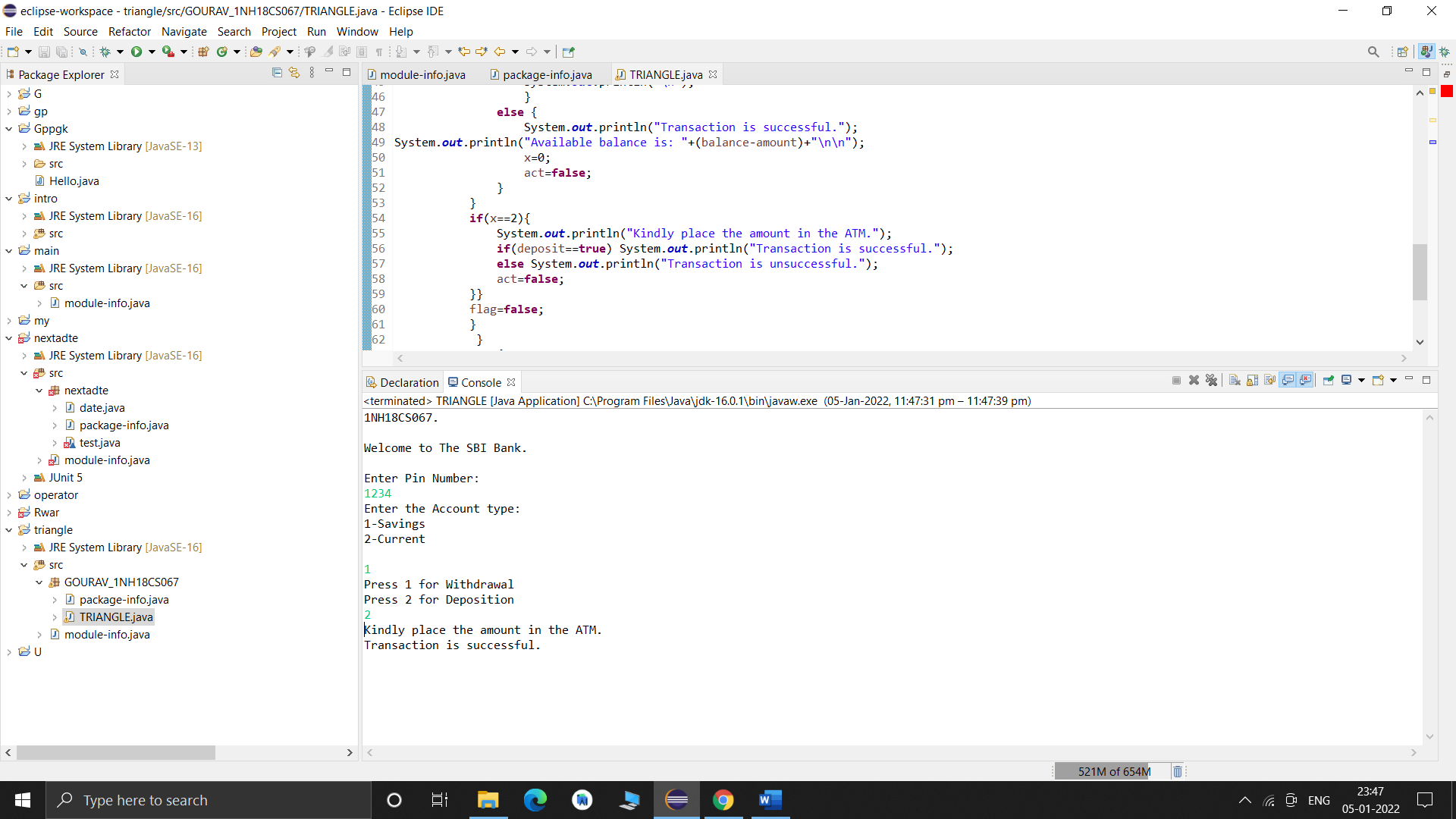
**TEST CASE 12:** unsuccessful withdraw operation due to click cancel after insert card.

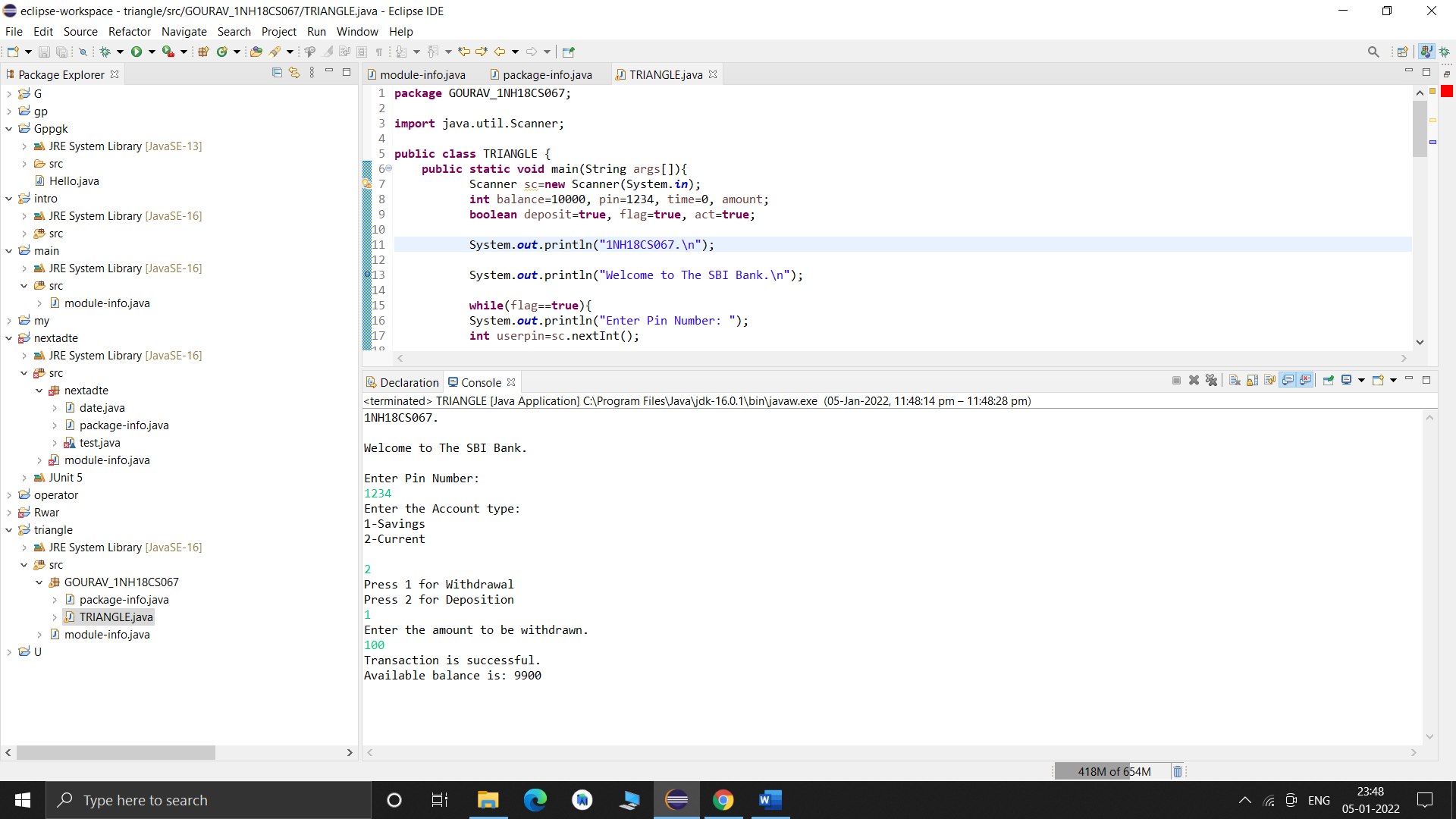
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | ATM | | Test Name: | | Cancel operation. | | |
| Project ID: | ATM\_12 | | Original Author: | | GOURAV | | |
| Test Objective: | To verify unsuccessful withdraw operation due to click cancel after insert card. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Unsuccessful withdraw operation | Click on cancel after card insertion. | | Displaying relevant option message. | | Pass |  |

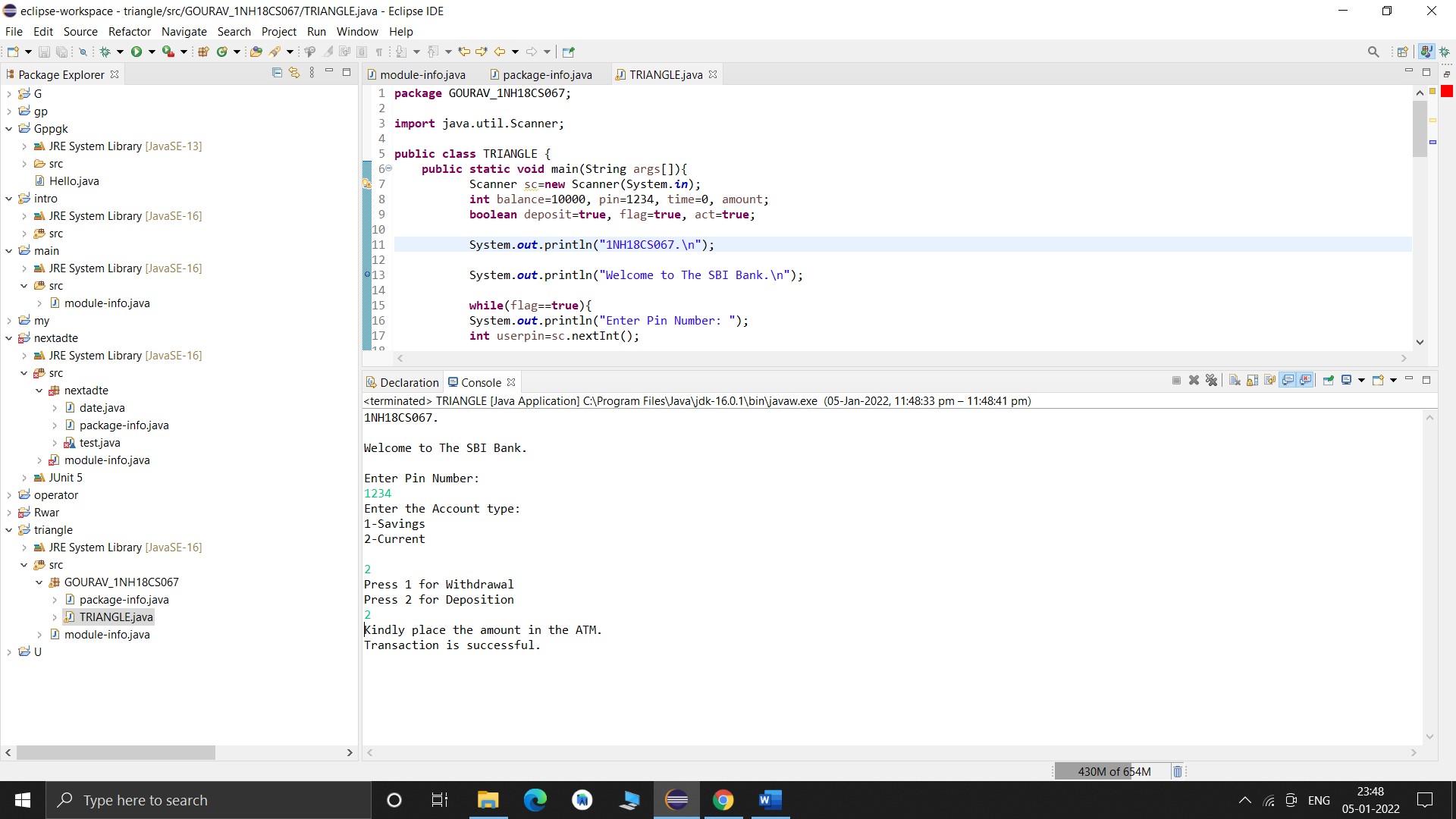
**EXECUTION**

**Instruction to be followed: Execute the above scenarios, take a screen shot of it (all four shots in one page, if possible, else can be made into two pages), paste it.**









**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed :

2. Number of Test Cases Passed :

3. Number of Test Cases Failed :

**Exp. No. : 2**

**Date :**

**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.Scanner;

**public** **class** triangle {

**public** **static** **void** main(String[] args){

Scanner s=**new** Scanner(System.***in***);

**int** O=1;

**do**{

System.***out***.println("Enter 3 inputs which are the sides of a triangle");

**int** a=s.nextInt();

**int** b=s.nextInt();

**int** c=s.nextInt();

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

{

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

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

{

System.***out***.println("It is an equilateral triangle\n");

}

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

{

System.***out***.println("It is an isoceles triangle\n");

}

**else**

{

System.***out***.println("It is a scalene triangle\n");

}

}

**else**

System.***out***.println("It is not a triangle\n");

}

**else**

System.***out***.println("Invalid input\nEnter sides within the range 1-200\n");

System.***out***.println("1. To enter input\n 2.to exit\nEnter your choice ");

O=s.nextInt();

}**while**(O!=2);

s.close();

}

}

**TEST CASES**

**Example:**

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | TRIANGLE | | Test Name: | | NOT A TRIANGLE | | |
| Project ID: | TRI\_01 | | Original Author: | | GOURAV | | |
| Test Objective: | TO VERIFY THAT IT IS NOT A TRIANGLE | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data**  **A B C** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | Not a triangle | 1 2 3 | | Not a traingle | | Pass |  |
| 2 | Not a triangle | 2 2 4 | | Not a traingle | | Pass |  |
| 3 | Not a triangle | 3 3 6 | | Not a traingle | | Pass |  |
| 4 | Not a triangle | 4 8 4 | | Not a traingle | | Pass |  |
| 5 | Not a triangle | 5 6 11 | | Not a traingle | | Pass |  |

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | TRIANGLE | | Test Name: | | EQUILATERAL TRIANGLE | | |
| Project ID: | TRI\_02 | | Original Author: | | GOURAV | | |
| Test Objective: | TO VERIFY IT IS A EQUILATERAL TRIANGLE | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data**  **A B C** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | It is a equilateral triangle | 100 100 100 | | Equilateral triangle | | Pass |  |
| 2 | It is a equilateral triangle | 1 1 1 | | Equilateral triangle | | Pass |  |
| 3 | It is a equilateral triangle | 10 10 10 | | Equilateral triangle | | Pass |  |
| 4 | It is a equilateral triangle | 50 50 50 | | Equilateral triangle | | Pass |  |
| 5 | It is a equilateral triangle | 110 110 110 | | Equilateral triangle | | Pass |  |

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

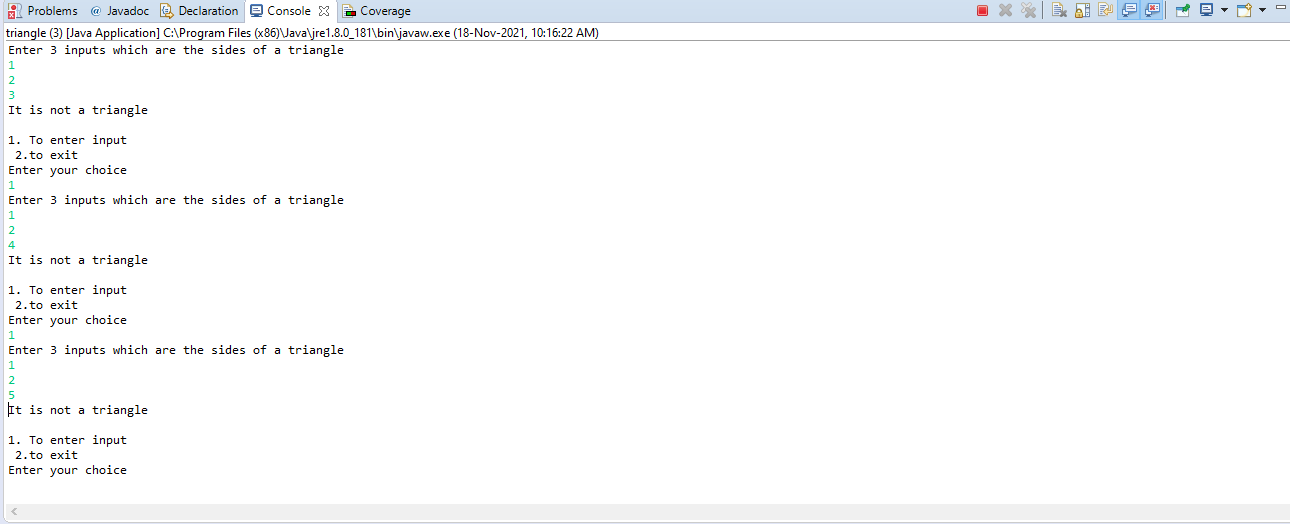
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | TRIANGLE | | Test Name: | | SCALENE TRIANGLE | | |
| Project ID: | TRI\_03 | | Original Author: | | GOURAV | | |
| Test Objective: | TO VERIFY SCALENE TRIANGLE | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data**  **A B C** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | It is a scalene triangle | 4 5 6 | | Scalene triangle. | | Pass |  |
| 2 | It is a scalene triangle | 5 6 7 | | Scalene triangle. | | Pass |  |
| 3 | It is a scalene triangle | 10 11 12 | | Scalene triangle. | | Pass |  |
| 4 | It is a scalene triangle | 100 110 120 | | Scalene triangle. | | Pass |  |
| 5 | It is a scalene triangle | 14 15 16 | | Scalene triangle. | | Pass |  |

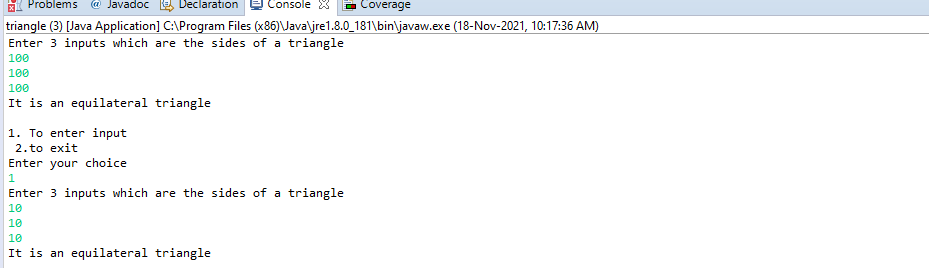
**TEST CASE 4:** Represents a valid isosceles triangle

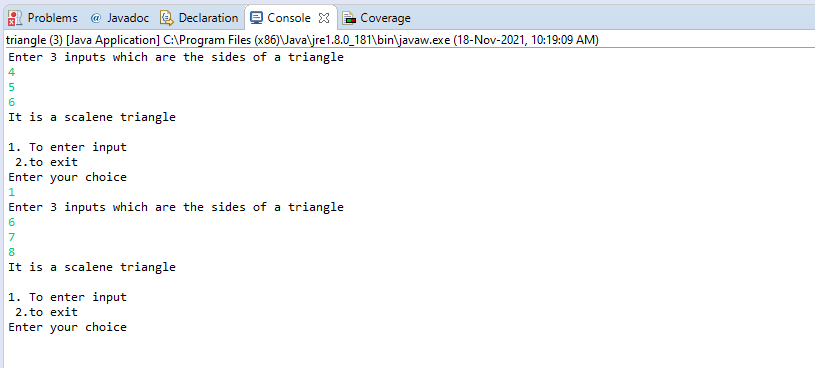
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | **Test Information** | | | | |
| Project Name: | TRIANGLE | | Test Name: | | ISOSCELES TRIANGLE | | |
| Project ID: | TRI\_04 | | Original Author: | | GOURAV | | |
| Test Objective: | TO VERIFY ISOSCELES TRIANGLE. | | | | | | |
| **Step No.** | **Test Case Description** | **Test Data**  **A B C** | | **Expected Result** | | **Status (Pass/Fail)** | **Remarks** |
| 1 | It is a isosceles | 4 6 6 | | Isosceles triangle | | Pass |  |
| 2 | It is a isosceles | 4 4 6 | | Isosceles triangle | | Pass |  |
| 3 | It is a isosceles | 5 6 6 | | Isosceles triangle | | Pass |  |
| 4 | It is a isosceles | 10 15 10 | | Isosceles triangle | | Pass |  |
| 5 | It is a isosceles | 100 50 100 | | Isosceles triangle | | Pass |  |

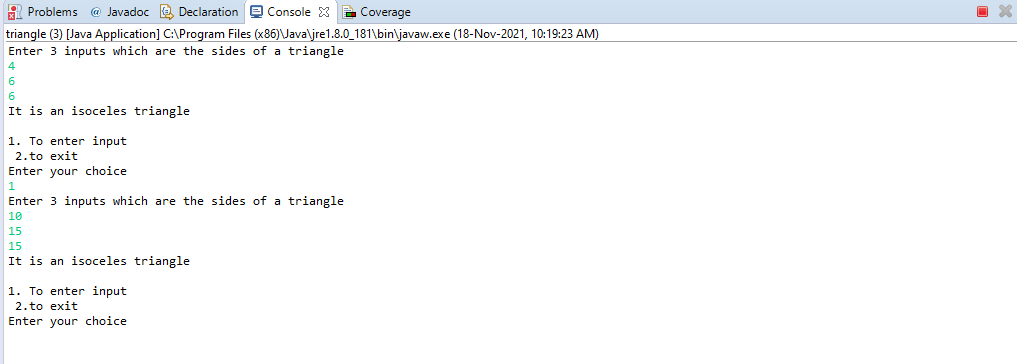
**EXECUTION**

**Instruction to be followed: Execute the above scenarios, take a screen shot of it (all four shots in one page, if possible, else can be made into two pages), paste it.**

****

****

****

****

**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed :

2. Number of Test Cases Passed :

3. Number of Test Cases Failed :

**Exp. No. : 3**

**Date :**

**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**

**JAVA CODE**

**import** java.util.\*;

**public** **class** Next {

**public** String nextd(**int** day,**int** month, **int** year) {

**if**((month>12)||((year<1812)||(year>2020))||(day>31))

{

**return**("Enter valid dates");

}

**else**

{

**if**((day==31 && month%2!=1 && month<8)||(day==31 && month>7 && month%2==1))

{

**return**("Enter valid dates");

}

**else**

{

**if**((month%2==1)||((month>7)&&(month%2==0)))

{

**if**(day==31)

{

**if**(month==12){

day=1;

month=1;

year+=1;

}

**else**

{

day=1;

month+=1;

}

}

**else**

{

day+=1;

}

}

**else**

{

**if**(month==2 && day==28)

{

**if**((year%4==0 && year%100!=0)||(year%400==0))

{

day+=1;

}

**else**

{

month+=1;

day=1;

}

}

**else** **if**(day==30)

{

**if**(month==12){

day=1;

month=1;

year+=1;

}

**else**

{

day=1;

month+=1;

}

}

**else**

{

day+=1;

}

}

}

}

**return**(day+"/"+month+"/"+year);

}

}

**Junit Code**

\*Normal BVA

import static org.junit.Assert.\*;

import org.junit.Test;

public class Normalbva {

@Test

public void test1()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1812),"13/3/1812");

}

@Test

public void test2()

{

Next d1 = new Next();

assertEquals(d1.nextd(30,3,1813),"31/3/1813");

}

@Test

public void test3()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,1912),"1/1/1913");

}

@Test

public void test4()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,2019),"13/3/2019");

}

@Test

public void test5()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,2020),"13/3/2020");

}

@Test

public void test6()

{

Next d1 = new Next();

assertEquals(d1.nextd(15,1,2020),"16/1/2020");

}

@Test

public void test7()

{

Next d1 = new Next();

assertEquals(d1.nextd(15,2,2020),"16/2/2020");

}

@Test

public void test8()

{

Next d1 = new Next();

assertEquals(d1.nextd(15,11,2020),"16/11/2020");

}

@Test

public void test9()

{

Next d1 = new Next();

assertEquals(d1.nextd(15,12,2020),"16/12/2020");

}

@Test

public void test10()

{

Next d1 = new Next();

assertEquals(d1.nextd(15,6,2020),"16/6/2020");

}

@Test

public void test11()

{

Next d1 = new Next();

assertEquals(d1.nextd(1,6,2020),"2/6/2020");

}

@Test

public void test12()

{

Next d1 = new Next();

assertEquals(d1.nextd(2,6,2020),"3/6/2020");

}

@Test

public void test13()

{

Next d1 = new Next();

assertEquals(d1.nextd(15,6,2020),"16/6/2020");

}

@Test

public void test14()

{

Next d1 = new Next();

assertEquals(d1.nextd(30,6,2020),"1/7/2020");

}

@Test

public void test15()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,3,2020),"1/4/2020");

}

}

\*Robust BVA

import static org.junit.Assert.\*;

import org.junit.Test;

public class robustbva {

@Test

public void test()

{

Next d1 = new Next();

assertEquals(d1.nextd(25,3,2019),"26/3/2019");

}

@Test

public void test1()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1950),"13/3/1950");

}

@Test

public void test3()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,1915),"1/1/1916");

}

@Test

public void test6()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1915),"13/3/1915");

}

@Test

public void test4()

{

Next d1 = new Next();

assertEquals(d1.nextd(32,3,1914),"Enter valid dates");

}

@Test

public void test5()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,13,2021),"Enter valid dates");

}

@Test

public void test7()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,2020),"13/3/2020");

}

}

\*Worst-case BVA

import static org.junit.Assert.\*;

import org.junit.Test;

public class worstcase {

@Test

public void test()

{

Next d1 = new Next();

assertEquals(d1.nextd(25,3,2012),"26/3/2012");

}

@Test

public void test1()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1925),"13/3/1925");

}

@Test

public void test2()

{

Next d1 = new Next();

assertEquals(d1.nextd(30,3,1950),"31/3/1950");

}

@Test

public void test3()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2010),"1/1/2011");

}

public void test4()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2010),"1/1/2010");

}

public void test5()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2010),"1/1/2010");

}

@Test

public void test6()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1915),"13/3/1915");

}

@Test

public void test7()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1920),"13/3/1920");

}

@Test

public void test8()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2009),"1/1/2010");

}

@Test

public void test9()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2000),"1/1/2001");

}

}

\*Robust worst-case BVA

import static org.junit.Assert.\*;

import org.junit.Test;

public class robustworstcase {

@Test

public void test()

{

Next d1 = new Next();

assertEquals(d1.nextd(25,3,2012),"26/3/2012");

}

@Test

public void test1()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1925),"13/3/1925");

}

@Test

public void test2()

{

Next d1 = new Next();

assertEquals(d1.nextd(30,3,1950),"31/3/1950");

}

@Test

public void test3()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2010),"1/1/2011");

}

public void test4()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2010),"1/1/2010");

}

public void test5()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2010),"1/1/2010");

}

@Test

public void test6()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1915),"13/3/1915");

}

@Test

public void test7()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,3,1920),"13/3/1920");

}

@Test

public void test8()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2009),"1/1/2010");

}

@Test

public void test9()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2000),"1/1/2001");

}

public void test12()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,2019),"1/1/2020");

}

@Test

public void test13()

{

Next d1 = new Next();

assertEquals(d1.nextd(31,12,1999),"1/1/2000");

}

@Test

public void test10()

{

Next d1 = new Next();

assertEquals(d1.nextd(32,3,1914),"Enter valid dates");

}

@Test

public void test11()

{

Next d1 = new Next();

assertEquals(d1.nextd(12,13,2021),"Enter valid dates");

}

}

**TEST CASES**

**/\* MUST BE WRITTEN (for all 4 variants) AS PER THE GIVEN TEMPLATE WRITE SIMILAR TEST CASE AS MENTIONED BELOW FOR NEXTDATE ALSO \*/**

**\*/**

Test Case Name: Equivalence Class testing for next problem

Test Data: Enter the 3 Integer Value (m, d and y)

Pre-condition: month {1<=m<=12}, day {1<=d<=31}, year {1812<=y<=2012} Test Objective: To find the next date to the given valid date.

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

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXT DATE | |
| Project ID: | NEXTDATE\_01 | | | | Original Author: | | | GOURAV | |
| Test  Objective: | Find out the next date for a given date (Normal BVA) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **a** | **b** | **c** | |
| NXTDATE2b  \_n1 | Enter the nominal values for m& d, y changes | 6 | 15 | 1812 | | Message must be displayed as “16.6.1812” | Pass | |  |
| NXTDATE2b  \_n2 | Enter the nominal values for m& d, y changes | 6 | 15 | 1813 | | Message must be displayed as “16.6.1813” | Pass | |  |
| NXTDATE2b  \_n3 | Enter the nominal values for m& d, y changes | 6 | 15 | 1912 | | Message must be displayed as “16.6.1912” | Pass | |  |
| NXTDATE2b  \_n4 | Enter the nominal values for m& d, y changes | 6 | 15 | 2011 | | Message must be displayed as “16.6.2011” | Pass | |  |
| NXTDATE2b  \_n5 | Enter the nominal values for m& d, y changes | 6 | 15 | 2012 | | Message must be displayed as “16.6.2012” | Pass | |  |
| NXTDATE2b  \_n6 | Enter the nominal values for m& y, dchanges | 6 | 1 | 1912 | | Message must be displayed as “2.6.1912” | Pass | |  |
| NXTDATE2b  \_n7 | Enter the nominal values for m& y, dchanges | 6 | 2 | 1912 | | Message must be displayed as “3.6.1912” | Pass | |  |
| NXTDATE2b  \_n8 | Enter the nominal values for m& y, dchanges | 6 | 30 | 1912 | | Message must be displayed as “1.7.1912” | Pass | |  |
| NXTDATE2b  \_n9 | Enter the nominal values for m& y, dchanges | 6 | 31 | 1912 | | Message must be displayed as “Invalid values” | Pass | |  |
| NXTDATE2b  \_n10 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1912 | | Message must be displayed as “16.1.1912” | Pass | |  |
| NXTDATE2b  \_n11 | Enter the nominal values for m changes, d,&y | 2 | 15 | 1912 | | Message must be displayed as “16.2.1912” | Pass | |  |
| NXTDATE2b  \_n12 | Enter the nominal values for m changes, d,&y | 11 | 15 | 1912 | | Message must be displayed as “16.11.1912” | Pass | |  |
| NXTDATE2b  \_n13 | Enter the nominal values for m changes, d,&y | 12 | 15 | 1912 | | Message must be displayed as “16.12.2012” | Pass | |  |

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

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXT DATE | |
| Project ID: | NEXTDATE\_02 | | | | Original Author: | | | GOURAV | |
| Test  Objective: | Find out the next date for a given date (ROUST BVA) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **a** | **b** | **c** | |
| NXTDATE2b  \_n1 | Enter the nominal values for m& d, y changes | 6 | 15 | 1812 | | Message must be displayed as “16.6.1812” | Pass | |  |
| NXTDATE2b  \_n2 | Enter the nominal values for m& d, y changes | 6 | 15 | 1813 | | Message must be displayed as “16.6.1813” | Pass | |  |
| NXTDATE2b  \_n3 | Enter the nominal values for m& d, y changes | 6 | 15 | 1912 | | Message must be displayed as “16.6.1912” | Pass | |  |
| NXTDATE2b  \_n4 | Enter the nominal values for m& d, y changes | 6 | 15 | 2011 | | Message must be displayed as “16.6.2011” | Pass | |  |
| NXTDATE2b  \_n5 | Enter the nominal values for m& d, y changes | 6 | 15 | 2012 | | Message must be displayed as “16.6.2012” | Pass | |  |
| NXTDATE2b  \_n6 | Enter the nominal values for m& y, dchanges | 6 | 1 | 1912 | | Message must be displayed as “2.6.1912” | Pass | |  |
| NXTDATE2b  \_n7 | Enter the nominal values for m& y, dchanges | 6 | 2 | 1912 | | Message must be displayed as “3.6.1912” | Pass | |  |
| NXTDATE2b  \_n8 | Enter the nominal values for m& y, dchanges | 6 | 30 | 1912 | | Message must be displayed as “1.7.1912” | Pass | |  |
| NXTDATE2b  \_n9 | Enter the nominal values for m& y, dchanges | 6 | 31 | 1912 | | Message must be displayed as “Invalid values” | Pass | |  |
| NXTDATE2b  \_n10 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1912 | | Message must be displayed as “16.1.1912” | Pass | |  |
| NXTDATE2b  \_n11 | Enter the nominal values for m changes, d,&y | 2 | 15 | 1912 | | Message must be displayed as “16.2.1912” | Pass | |  |
| NXTDATE2b  \_n12 | Enter the nominal values for m changes, d,&y | 11 | 15 | 1912 | | Message must be displayed as “16.11.1912” | Pass | |  |
| NXTDATE2b  \_n13 | Enter the nominal values for m changes, d,&y | 12 | 15 | 1912 | | Message must be displayed as “16.12.2012” | Pass | |  |
| NXTDATE2b  \_n14 | Enter the nominal values for m changes, d,&y | 6 | 15 | 1811 | |  | Pass | |  |
| NXTDATE2b  \_n15 | Enter the nominal values for m changes, d,&y | 6 | 15 | 2013 | |  | Pass | |  |
| NXTDATE2b  \_n16 | Enter the nominal values for m changes, d,&y | 6 | 0 | 1912 | |  | Pass | |  |
| NXTDATE2b  \_n17 | Enter the nominal values for m changes, d,&y | 6 | 32 | 1912 | |  | Pass | |  |
| NXTDATE2b  \_n18 | Enter the nominal values for m changes, d,&y | 0 | 15 | 1912 | |  | Pass | |  |
| NXTDATE2b  \_n19 |  | 13 | 15 | 1912 | |  |  | |  |

1. **TEST CASES FOR WORST-CASE BOUNDAR VALUE TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXT DATE | |
| Project ID: | NEXTDATE\_03 | | | | Original Author: | | | GOURAV | |
| Test  Objective: | Find out the next date for a given date (ROUST BVA) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **a** | **b** | **c** | |
| NXTDATE2b  \_n1 | Enter the nominal values for m& d, y changes | 1 | 1 | 1811 | | Message must be displayed as “16.6.1812” | Pass | |  |
| NXTDATE2b  \_n1 | Enter the nominal values for m& d, y changes | 1 | 1 | 1812 | | Message must be displayed as “16.6.1812” | Pass | |  |
| NXTDATE2b  \_n2 | Enter the nominal values for m& d, y changes | 1 | 1 | 1813 | | Message must be displayed as “16.6.1813” | Pass | |  |
| NXTDATE2b  \_n3 | Enter the nominal values for m& d, y changes | 1 | 1 | 1912 | | Message must be displayed as “16.6.1912” | Pass | |  |
| NXTDATE2b  \_n4 | Enter the nominal values for m& d, y changes | 1 | 1 | 2011 | | Message must be displayed as “16.6.2011” | Pass | |  |
| NXTDATE2b  \_n5 | Enter the nominal values for m& d, y changes | 1 | 1 | 2012 | | Message must be displayed as “16.6.2012” | Pass | |  |
| NXTDATE2b  \_n5 | Enter the nominal values for m& d, y changes | 1 | 1 | 2013 | | Message must be displayed as “16.6.2012” | Pass | |  |
| NXTDATE2b  \_n5 | Enter the nominal values for m& d, y changes | 1 | 2 | 1811 | | Message must be displayed as “16.6.2012” | Pass | |  |
| NXTDATE2b  \_n6 | Enter the nominal values for m& y, dchanges | 1 | 2 | 1812 | | Message must be displayed as “2.6.1912” | Pass | |  |
| NXTDATE2b  \_n7 | Enter the nominal values for m& y, dchanges | 1 | 2 | 1813 | | Message must be displayed as “3.6.1912” | Pass | |  |
| NXTDATE2b  \_n8 | Enter the nominal values for m& y, dchanges | 1 | 2 | 1912 | | Message must be displayed as “1.7.1912” | Pass | |  |
| NXTDATE2b  \_n9 | Enter the nominal values for m& y, dchanges | 1 | 2 | 2011 | | Message must be displayed as “Invalid values” | Pass | |  |
| NXTDATE2b  \_n10 | Enter the nominal values for m changes, d,&y | 1 | 2 | 2012 | | Message must be displayed as “16.1.1912” | Pass | |  |
| NXTDATE2b  \_n10 | Enter the nominal values for m changes, d,&y | 1 | 2 | 2013 | | Message must be displayed as “16.1.1912” | Pass | |  |
| NXTDATE2b  \_n11 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1811 | | Message must be displayed as “16.2.1912” | Pass | |  |
| NXTDATE2b  \_n11 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1812 | | Message must be displayed as “16.2.1912” | Pass | |  |
| NXTDATE2b  \_n12 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1813 | | Message must be displayed as “16.11.1912” | Pass | |  |
| NXTDATE2b  \_n13 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1912 | | Message must be displayed as “16.12.2012” | Pass | |  |
| NXTDATE2b  \_n14 | Enter the nominal values for m changes, d,&y | 1 | 15 | 2011 | |  | Pass | |  |
| NXTDATE2b  \_n15 | Enter the nominal values for m changes, d,&y | 1 | 15 | 2012 | |  | Pass | |  |
| NXTDATE2b  \_n15 | Enter the nominal values for m changes, d,&y | 1 | 15 | 2013 | |  | Pass | |  |
| NXTDATE2b  \_n16 | Enter the nominal values for m changes, d,&y | 1 | 30 | 1811 | |  | Pass | |  |
| NXTDATE2b  \_n16 | Enter the nominal values for m changes, d,&y | 1 | 30 | 1812 | |  | Pass | |  |
| NXTDATE2b  \_n17 | Enter the nominal values for m changes, d,&y | 1 | 30 | 1813 | |  | Pass | |  |
| NXTDATE2b  \_n18 | Enter the nominal values for m changes, d,&y | 1 | 30 | 1912 | |  | Pass | |  |
| NXTDATE2b  \_n19 |  | 1 | 30 | 2011 | |  | Pass | |  |
| NXTDATE2b  \_n18 | Enter the nominal values for m changes, d,&y | 1 | 30 | 2012 | |  | Pass | |  |
| NXTDATE2b  \_n18 | Enter the nominal values for m changes, d,&y | 1 | 30 | 2013 | |  | Pass | |  |

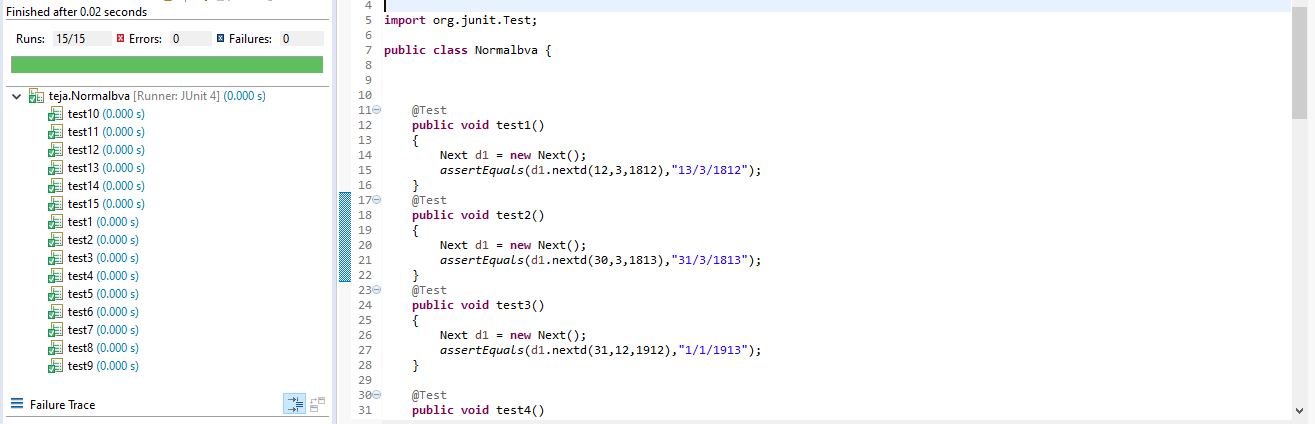
1. **TEST CASES FOR ROBUST WORST-CASE BOUNDARY VALUE TESTING**

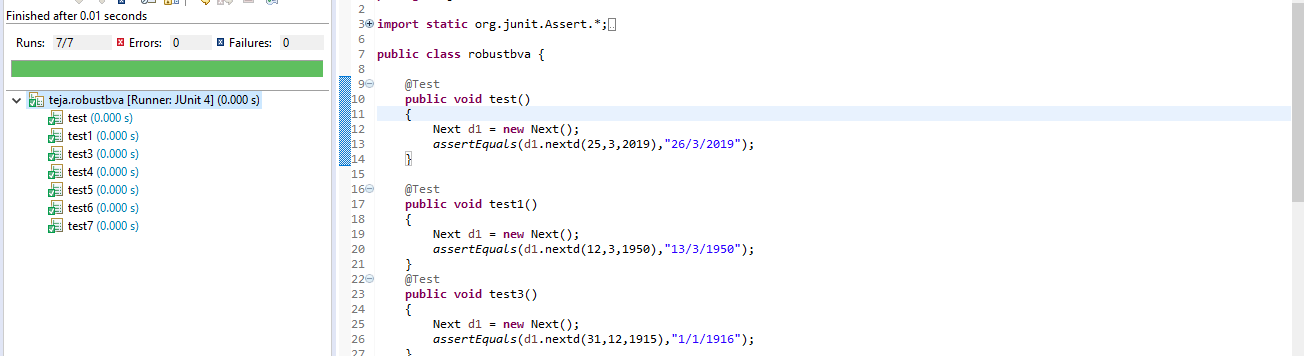
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXT DATE | |
| Project ID: | NEXTDATE\_03 | | | | Original Author: | | | GOURAV | |
| Test  Objective: | Find out the next date for a given date (ROUST BVA) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **a** | **b** | **c** | |
| NXTDATE2b  \_n1 | Enter the nominal values for m& d, y changes | 1 | 1 | 1812 | | Message must be displayed as “16.6.1812” | Pass | |  |
| NXTDATE2b  \_n2 | Enter the nominal values for m& d, y changes | 1 | 1 | 1813 | | Message must be displayed as “16.6.1813” | Pass | |  |
| NXTDATE2b  \_n3 | Enter the nominal values for m& d, y changes | 1 | 1 | 1912 | | Message must be displayed as “16.6.1912” | Pass | |  |
| NXTDATE2b  \_n4 | Enter the nominal values for m& d, y changes | 1 | 1 | 2011 | | Message must be displayed as “16.6.2011” | Pass | |  |
| NXTDATE2b  \_n5 | Enter the nominal values for m& d, y changes | 1 | 1 | 2012 | | Message must be displayed as “16.6.2012” | Pass | |  |
| NXTDATE2b  \_n6 | Enter the nominal values for m& y, dchanges | 1 | 2 | 1812 | | Message must be displayed as “2.6.1912” | Pass | |  |
| NXTDATE2b  \_n7 | Enter the nominal values for m& y, dchanges | 1 | 2 | 1813 | | Message must be displayed as “3.6.1912” | Pass | |  |
| NXTDATE2b  \_n8 | Enter the nominal values for m& y, dchanges | 1 | 2 | 1912 | | Message must be displayed as “1.7.1912” | Pass | |  |
| NXTDATE2b  \_n9 | Enter the nominal values for m& y, dchanges | 1 | 2 | 2011 | | Message must be displayed as “Invalid values” | Pass | |  |
| NXTDATE2b  \_n10 | Enter the nominal values for m changes, d,&y | 1 | 2 | 2012 | | Message must be displayed as “16.1.1912” | Pass | |  |
| NXTDATE2b  \_n11 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1812 | | Message must be displayed as “16.2.1912” | Pass | |  |
| NXTDATE2b  \_n12 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1813 | | Message must be displayed as “16.11.1912” | Pass | |  |
| NXTDATE2b  \_n13 | Enter the nominal values for m changes, d,&y | 1 | 15 | 1912 | | Message must be displayed as “16.12.2012” | Pass | |  |
| NXTDATE2b  \_n14 | Enter the nominal values for m changes, d,&y | 1 | 15 | 2011 | |  | Pass | |  |
| NXTDATE2b  \_n15 | Enter the nominal values for m changes, d,&y | 1 | 15 | 2012 | |  | Pass | |  |
| NXTDATE2b  \_n16 | Enter the nominal values for m changes, d,&y | 1 | 30 | 1812 | |  | Pass | |  |
| NXTDATE2b  \_n17 | Enter the nominal values for m changes, d,&y | 1 | 30 | 1813 | |  | Pass | |  |
| NXTDATE2b  \_n18 | Enter the nominal values for m changes, d,&y | 1 | 30 | 1912 | |  | Pass | |  |
| NXTDATE2b  \_n19 |  | 1 | 30 | 2011 | |  | Pass | |  |

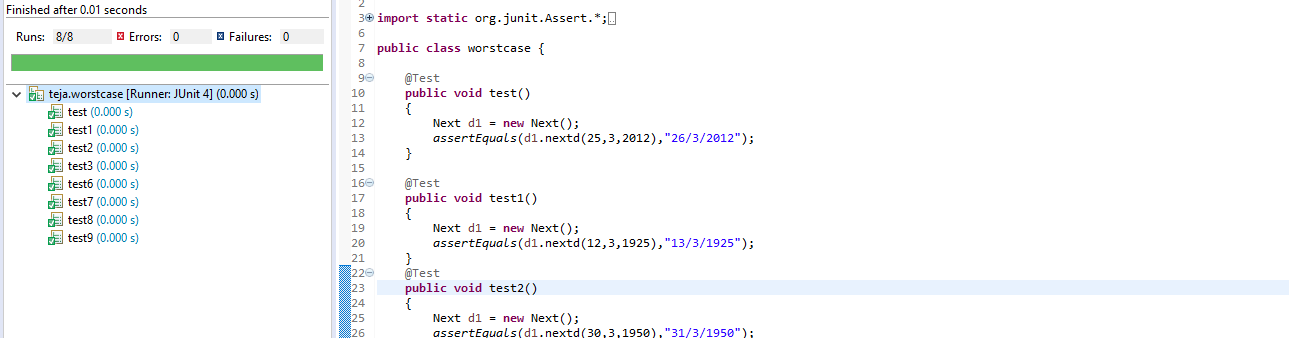
**/\* PREPARE SIMILAR TABLE WITH SAMPLE TEST CASES (AS ABOVE) FOR ROBUST-CASE BVA (min 20 test cases) \*/**

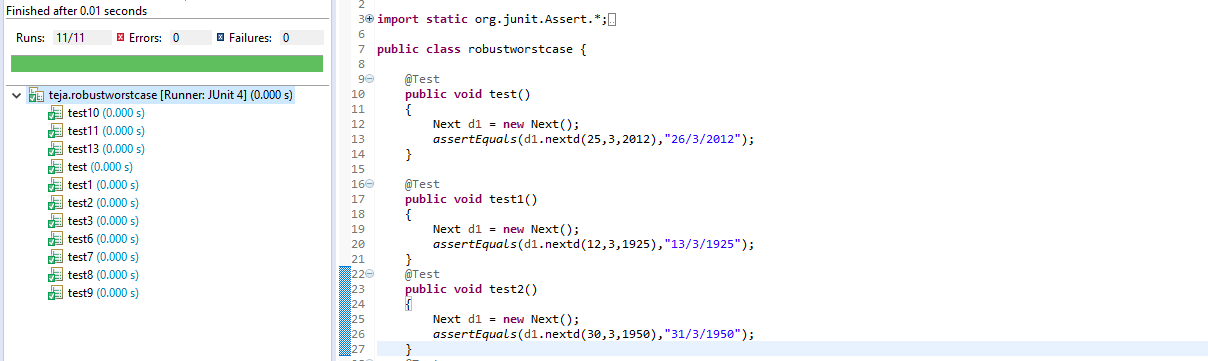
**EXECUTION**

**Instruction to be followed: Execute the above scenarios using JUnit, take screen shots of it, paste it.**

****

****

****

****

**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed :

2. Number of Test Cases Passed :

3. Number of Test Cases Failed :

**Exp. No. : 4**

**Date :**

**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**

**\*JAVA CODE**

**package** nd2;

//import java.util.Scanner;

**public** **class** nextdate

{

**public** **static** String next(**int** d, **int** m, **int** y, **int** cc)

{

**if**(d==cc)

{

d=1;

**if**(m==12)

{

y++;

m=1;

}

**else**

{

m++;

}

}

**else**

{

d++;

}

**return**(String.*valueOf*(d)+"/"+String.*valueOf*(m)+"/"+String.*valueOf*(y));

}

**public** String nextday(**int** d, **int** m, **int** y)

{

**if**(d>=1 && d<=31 && m>=1 && m<=12 && y>=1812 && y<=2012)

{

**switch**(m)

{

**case** 1:

**case** 3:**return**(*next*(d,m,y,31));

**case** 5:**return**(*next*(d,m,y,31));

**case** 7:**return**(*next*(d,m,y,31));

**case** 8:**return**(*next*(d,m,y,31));

**case** 10:**return**(*next*(d,m,y,31));

**case** 12: **return**(*next*(d,m,y,31));

**case** 4: **return**(*next*(d,m,y,30));

**case** 6: **return**(*next*(d,m,y,30));

**case** 9: **return**(*next*(d,m,y,30));

**case** 11: **return**(*next*(d,m,y,30));

**default**: **return**(*next*(d,m,y,((y%4==0 && y%100!=0) || y%400==0)?29:28));

}

}

**return** "Invalid Values";

}

}

\***Junit code**

package nd2;

import static org.junit.Assert.\*;

import org.junit.Test;

public class equind2pgm {

//weak and strong normal test case

@Test

public void test\_1()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(15,6,1912),"16/6/1912");

}

@Test

public void test\_2()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(10,6,1912),"11/6/1912");

}

@Test

public void test\_3()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(10,6,1900),"11/6/1900");

}

@Test

public void test\_4()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(10,5,1912),"11/5/1912");

}

@Test

public void test\_5()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(20,10,2010),"21/10/2010");

}

//weak robust test cases

@Test

public void test3()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(-1,10,1912),"Invalid Values");

}

@Test

public void test31()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(12,7,1912),"13/7/1912");

}

@Test

public void test32()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(12,8,1912),"13/8/1912");

}

@Test

public void test33()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(12,4,1912),"13/4/1912");

}

@Test

public void test34()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(12,9,1912),"13/9/1912");

}

@Test

public void test35()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(12,1,1912),"13/1/1912");

}

@Test

public void test36()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(12,2,1912),"13/2/1912");

}

@Test

public void test37()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(12,3,1912),"13/3/1912");

}

@Test

public void test30()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(10,3,1912),"11/3/1912");

}

@Test

public void test4()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(15,13,1912),"Invalid Values");

}

@Test

public void test5()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(1,6,2200),"Invalid Values");

}

@Test

public void test6()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(32,6,1912),"Invalid Values");

}

@Test

public void test7()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(15,6,1811),"Invalid Values");

}

@Test

public void test8()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(15,6,2013),"Invalid Values");

}

//strong robust test cases

@Test

public void test9()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(2,1,1912),"3/1/1912");

}

@Test

public void test10()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(-1,3,1900),"Invalid Values");

}

@Test

public void test11()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(15,0,1811),"Invalid Values");

}

@Test

public void test12()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(33,12,1912),"Invalid Values");

}

@Test

public void test13()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(15,-1,-1),"Invalid Values");

}

@Test

public void test14()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(-1,6,-1),"Invalid Values");

}

@Test

public void test15()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(-1,-1,-1),"Invalid Values");

}

@Test

public void test16()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(31,12,2010),"1/1/2011");

}

@Test

public void test17()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(30,11,2010),"1/12/2010");

}

//////leap

@Test

public void test18()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(3,2,2010),"4/2/2010");

}

@Test

public void test19()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(28,2,2010),"1/3/2010");

}

@Test

public void test20()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(20,2,2008),"21/2/2008");

}

@Test

public void test21()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(29,2,2000),"1/3/2000");

}

@Test

public void test22()

{

nextdate ob1=new nextdate();

assertEquals(ob1.nextday(28,2,1900),"1/3/1900");

}

}

**TEST CASES**

**/\* MUST BE WRITTEN (for all 4 variants) AS PER THE GIVEN TEMPLATE WRITE SIMILAR TEST CASE AS MENTIONED BELOW FOR NEXTDATE ALSO \*/**

Test Case Name: Equivalence Class testing for next problem

Test Data: Enter the 3 Integer Value (m, d and y)

Pre-condition: month{1<=m<=12}, day{1<=d<=31}, year{1812<=y<=2012}

Test Objective: To find the next date to the given valid date.

**I) TEST CASES FOR WEAK NORMAL EQUIVALENCE CLASS TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXTDATE | |
| Project ID: | NEXTDATE\_01 | | | | Original Author: | | | Gouarv | |
| Test  Objective: | Check if valid date input gives next date (Weak normal equivalence class testing) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **d** | **m** | **y** | |
| TEST2d\_wn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 3 | 2000 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 4 | **1912** | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 16 | 4 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 3 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 10 | 11 | 1920 | | Message must be displayed as “15.6.2000” | Pass | |  |

**(ii) TEST CASES FOR STRONG NORMAL EQUIVALENCE CLASS TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXTDATE | |
| Project ID: | NEXTDATE\_02 | | | | Original Author: | | | Gourav | |
| Test  Objective: | Check if valid date input gives next date (Strong normal equivalence class testing) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **d** | **m** | **y** | |
| TEST2d\_sn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 3 | 2000 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 4 | **1912** | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 16 | 4 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 3 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sn1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 10 | 11 | 1920 | | Message must be displayed as “15.6.2000” | Pass | |  |

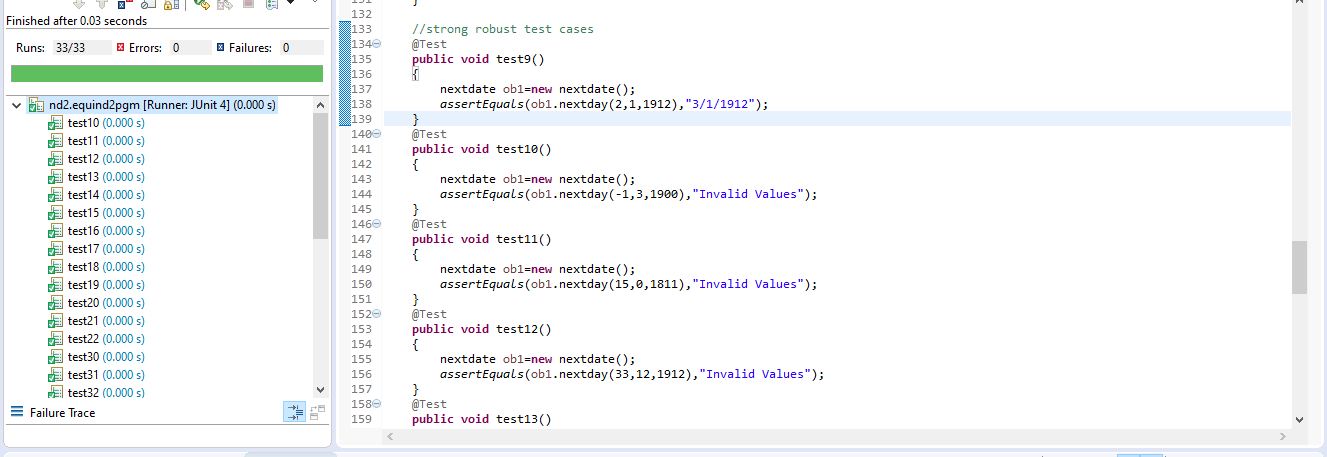
**(iii) TEST CASES FOR WEAK ROBUST EQUIVALENCE CLASS TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXTDATE | |
| Project ID: | NEXTDATE\_03 | | | | Original Author: | | | Gourav | |
| Test  Objective: | Check if valid date input gives next date (Robust equivalence class testing) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **d** | **m** | **y** | |
| TEST2d\_wr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 3 | 2000 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 4 | **1912** | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 16 | 4 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 15 | 3 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_wr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 10 | 11 | 1920 | | Message must be displayed as “15.6.2000” | Pass | |  |

**(iv) TEST CASES FOR STRONG ROBUST EQUIVALENCE CLASS TESTING**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | | |
| Project Name: | NEXTDATE | | | | Project Name: | | | NEXTDATE | |
| Project ID: | NEXTDATE\_03 | | | | Original Author: | | | Gourav | |
| Test  Objective: | Check if valid date input gives next date (Robust equivalence class testing) | | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status (Pass/ Fail)** | | **Remark** |
| **d** | **m** | **y** | |
| TEST2d\_sr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | -1 | 15 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 6 | -1 | **1810** | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 32 | 10 | 1810 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 1 | 2 | 1912 | | Message must be displayed as “15.6.2000” | Pass | |  |
| TEST2d\_sr1 | Enter the values for m, d, y arbitrarily chosen from equivalence class | 5 | 6 | 2000 | | Message must be displayed as “15.6.2000” | Pass | |  |

**EXECUTION**

****

**RESULT & DISCUSSION**

Test Report:

1. Number of Test Cases Executed :

2. Number of Test Cases Passed :

3. Number of Test Cases Failed :

**Exp. No. : 5**

**Date :**

**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**

**\*JAVA CODE**

**package** cs067;

**public** **class** triangle {

**public** String op(**int** a,**int** b,**int** c)

{

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

{

**if**(a+b>c && b+c>a && c+a>b)

{

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

{

**return** "Equilateral Triangle";

}

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

{

**return** "Isosceles Triangle";

}

**else**

{

**return** "Scalen Triangle";

}

}

**else**

{

**return** "Not a Triangle";

}

}

**else**

{

**return** "Invalid";

}

}

}

\***Junit code**

package cs067;

import static org.junit.Assert.\*;

import org.junit.Test;

import cs067.triangle;

public class triangleTest {

@Test

public void test() {

triangle t1=new triangle();

assertEquals(t1.op(1, 2, 3),"Not a Triangle");

}

@Test

public void test12() {

triangle t1=new triangle();

assertEquals(t1.op(2, 1, 1),"Not a Triangle");

}

@Test

public void test13() {

triangle t1=new triangle();

assertEquals(t1.op(2, 4, 2),"Not a Triangle");

}

@Test

public void test1() {

triangle t1=new triangle();

assertEquals(t1.op(100, 100, 100),"Equilateral Triangle");

}

@Test

public void test2() {

triangle t1=new triangle();

assertEquals(t1.op(4, 5, 6),"Scalen Triangle");

}

@Test

public void test3() {

triangle t1=new triangle();

assertEquals(t1.op(4, 6, 6),"Isosceles Triangle");

}

@Test

public void test4() {

triangle t1=new triangle();

assertEquals(t1.op(201, 201, 201),"Invalid");

}

@Test

public void test5() {

triangle t1=new triangle();

assertEquals(t1.op(6, 6, 4),"Isosceles Triangle");

}

@Test

public void test6() {

triangle t1=new triangle();

assertEquals(t1.op(4, 201, 7),"Invalid");

}

@Test

public void test7() {

triangle t1=new triangle();

assertEquals(t1.op(4, 7, 201),"Invalid");

}

@Test

public void test8() {

triangle t1=new triangle();

assertEquals(t1.op(0, 7, 201),"Invalid");

}

@Test

public void test9() {

triangle t1=new triangle();

assertEquals(t1.op(7, 0, 201),"Invalid");

}

@Test

public void test11() {

triangle t1=new triangle();

assertEquals(t1.op(7, 9, 0),"Invalid");

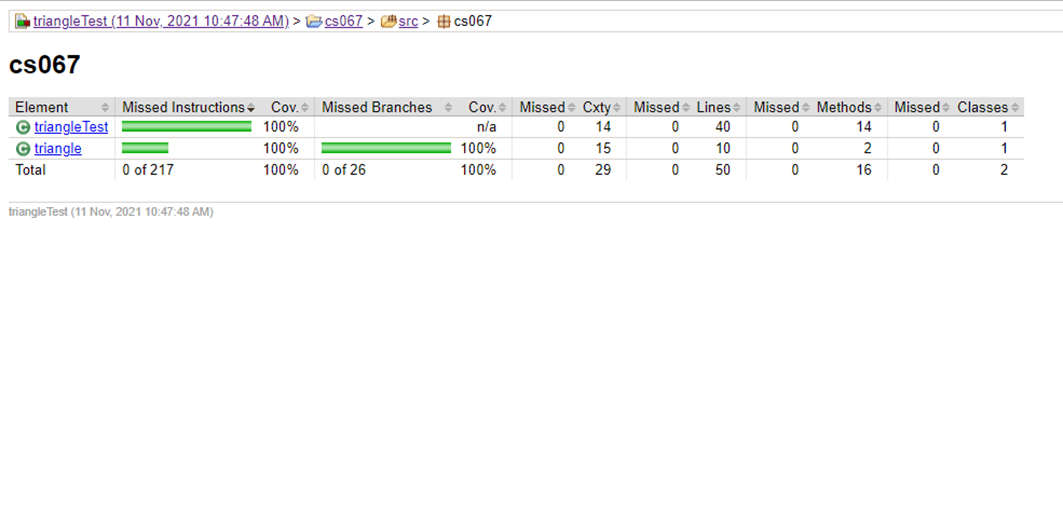
}

}

**EXECUTION**

**Instruction to be followed: Execute the above scenarios using JUnit, and use ECLEMMA for coverage and copy and paste the coverage report. Attach it here.**

**SAMPLE**

****

**TEST CASES FOR TRIANGLE PROGRAM**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Project Information** | | | | | **Test Information** | | | |
| Project Name: | TRIANGLE | | | | Project Name: | | TRIANGLE | |
| Project ID: | TRIANGLE\_01 | | | | Original Author: | |  | |
| Test  Objective: | Check whether given value for a equilateral, isosceles, Scalene triangle or can't from a  triangle | | | | | | | |
| **Test Case ID** | **Test Case Description** | **Test Data** | | | | **Expected Result** | **Status**  **(Pass/ Fail)** | **Remark** |
| **a** | **b** | **c** | |
| TEST2c\_1 | Enter the values for a, b, c arbitrarily chosen from equivalenceclass | 5 | 5 | 5 | | Message must be displayed as “the triangle is Equilateral” | Pass |  |
| TEST2c\_2 | Enter the values for a, b, c arbitrarily chosen from equivalenceclass | 2 | 2 | 3 | | Message must be displayed as “the triangle is Isosceles” | Pass |  |
| TEST2c\_3 | Enter the values for a, b, c arbitrarily chosen from equivalenceclass | 3 | 4 | 5 | | Message must be displayed as “the triangle is Scalene” | Pass |  |
| TEST2c\_4 | Enter the values for a, b, c arbitrarily chosen from equivalenceclass | 4 | 1 | 3 | | Message must be displayed as “Not a Triangle” | Pass |  |

**RESULT & DISCUSSION**

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

**Exp. No. : 6**

**Date :**

**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**

**\*JAVA CODE**

**public** **class** nextdate {

**public** **static** String next(**int** d,**int** m,**int** y,**int** cc){

**if**(d==cc){

d=1;

**if**(m==12){

y++;

m=1;

}

**else**{

m++;

}

}

**else** {

d++;

}

**return**(String.*valueOf*(d)+"/"+String.*valueOf*(m)+"/"+String.*valueOf*(y));

}

**public** String nextday(**int** d,**int** m,**int** y){

**if**(d>=1 && d<=31 && m>=1 && m<=12 && y>=1812 && y<=2012){

**switch**(m){

**case** 1:

**case** 3:

**case** 5:

**case** 8:

**case** 10:

**case** 12:**return**(*next*(d,m,y,31));

**case** 4:

**case** 6:

**case** 9:

**case** 11:**return**(*next*(d,m,y,30));

**default**:**return**(*next*(d,m,y,((y%4==0 && y%100!=0) || y%400==0)?29:28));

}

}

**return** "Invalid inputs";

}

}

\***Junit Code**

import static org.junit.Assert.\*;

import org.junit.Test;

public class test {

//weak and strong normal test cases

@Test

public void test1()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(15,3,1912),"16/3/1912");

}

@Test

public void test2()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(15,4,1912),"16/4/1912");

}

@Test

public void test3()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(16,4,1912),"17/4/1912");

}

@Test

public void test4()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(15,3,1912),"16/3/1912");

}

@Test

public void test5()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(10,11,1920),"11/11/1920");

}

@Test

public void test6()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(13,15,1912),"Invalid inputs");

}

@Test

public void test7()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(32,1,1813),"Invalid inputs");

}

@Test

public void test8()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(7,1,1810),"Invalid inputs");

}

@Test

public void test9()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(7,10,1912),"8/10/1912");

}

@Test

public void test10()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(6,11,2011),"7/11/2011");

}

@Test

public void test11()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(18,8,2012),"19/8/2012");

}

@Test

public void test12()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(-1,15,1912),"Invalid inputs");

}

@Test

public void test13()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(6,-1,1810),"Invalid inputs");

}

@Test

public void test14()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(32,10,1811),"Invalid inputs");

}

@Test

public void test15()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(1,2,1912),"2/2/1912");

}

@Test

public void test16()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(5,6,2000),"6/6/2000");

}

@Test

public void test17()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(21,6,2000),"22/6/2000");

}

@Test

public void test18()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(-1,-1,-1),"Invalid inputs");

}

@Test

public void test19()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(31,1,2001),"1/2/2001");

}

@Test

public void test20()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(31,12,2001),"1/1/2002");

}

@Test

public void test21()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(0,0,2013),"Invalid inputs");

}

@Test

public void test22()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(28,2,2011),"1/3/2011");

}

@Test

public void test23()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(28,13,2012),"Invalid inputs");

}

@Test

public void test24()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(28,2,2012),"29/2/2012");

}

@Test

public void test25()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(28,2,2000),"29/2/2000");

}

@Test

public void test26()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(31,1,1812),"1/2/1812");

}

@Test

public void test27()

{

nextdate d1 = new nextdate();

assertEquals(d1.nextday(31,12,2012),"1/1/2013");

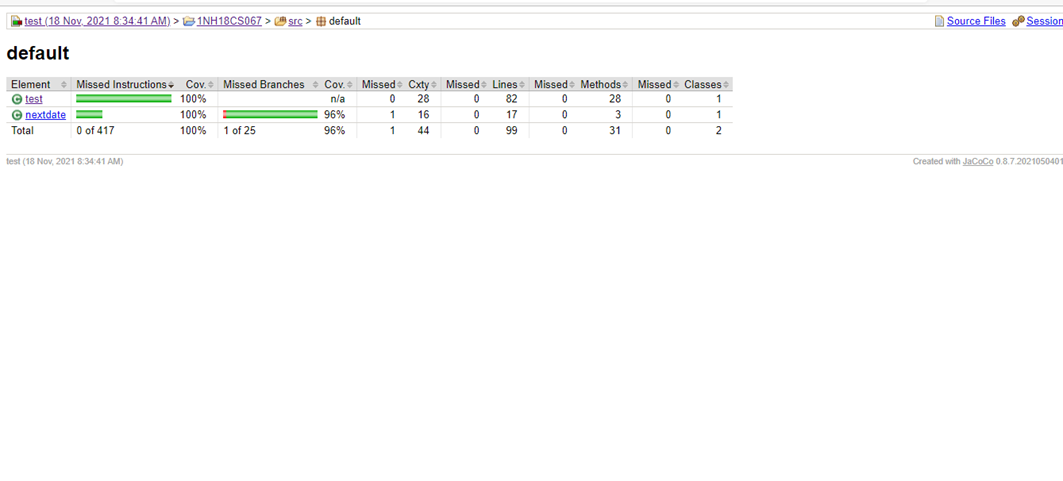
}

}

**EXECUTION**

**/\*Instruction to be followed: Execute the above scenarios using JUnit, and use ECLEMMA for coverage, copy and paste here. \*/**

**SAMPLE**

****

**RESULT & DISCUSSION /\*MUST BE HAND WRITTEN\*/**

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

**Exp. No. : 7**

**Date :**

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

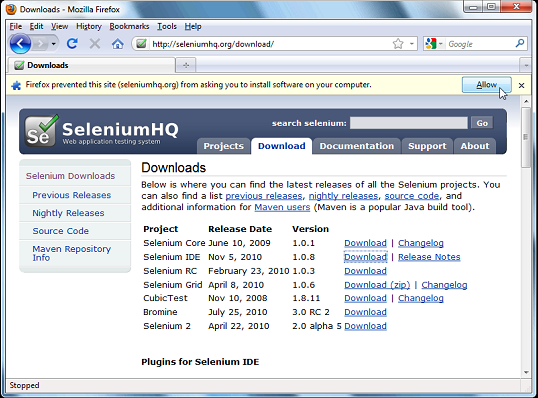
Designing Test Cases using Selenium IDE.

**IMPLEMENTATION**

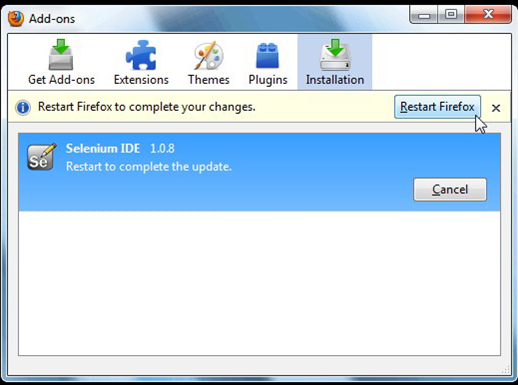
**Installing Selenium IDE**

Step 1: Using Firefox, first, download the IDE from the SeleniumHQ downloads page.

Step 2: Firefox will protect you from installing add-ons from unfamiliar locations, so you will need to click ‘Allow’ to proceed with the installation, as shown in the following screenshot.

****

Step 3: Select Install Now. The Firefox Add-ons window pops up, first showing a progress bar, and when the download is complete, displays the following.



Step 4: Restart Firefox. After Firefox reboots you will find the Selenium-IDE listed under the Firefox Tools menu.

**TEST CASES**

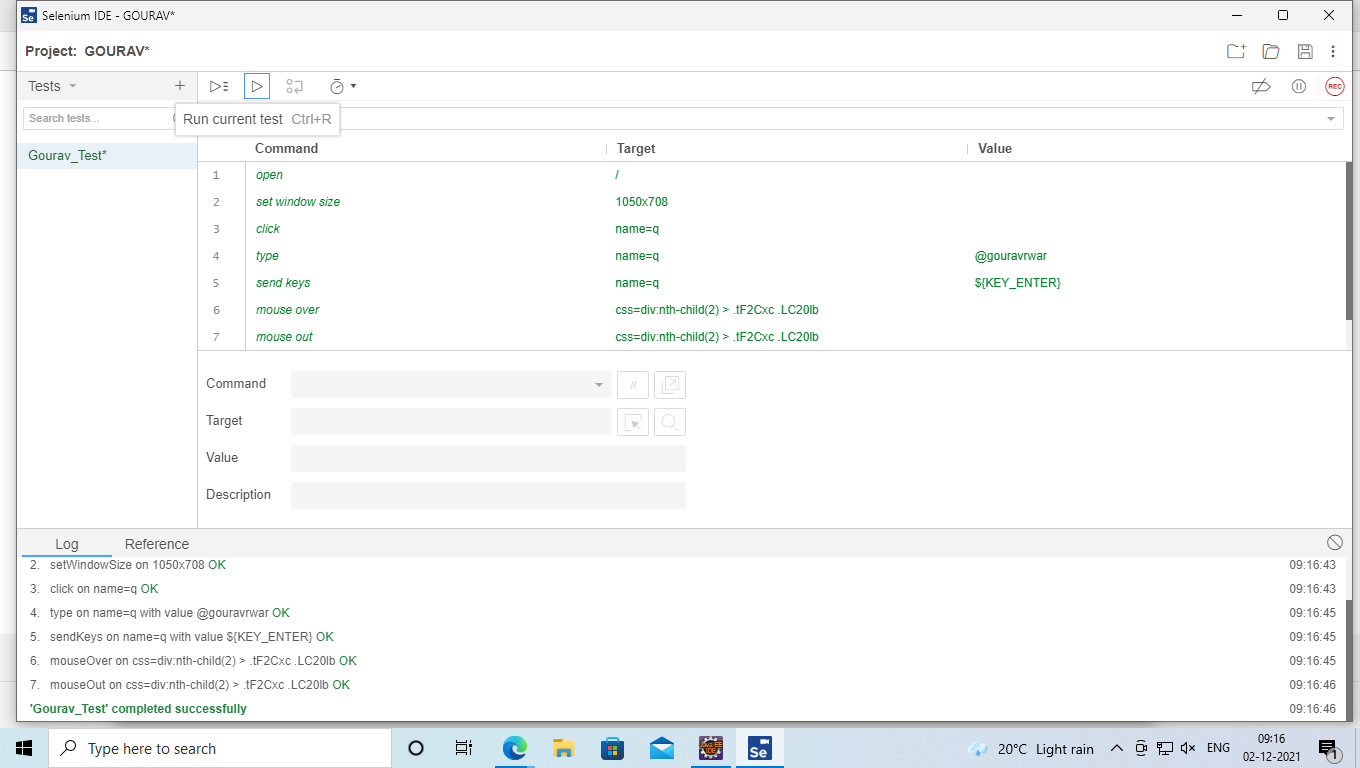
TC’S #1: Manual Steps:

* Open (Example: Type www.google.com)
* Type “Software Te sting” in the Google Search Input Box
* Click outside on an empty spot
* Click Search Button
* Verify the Text Present as “Software Testing”
* Assert the Title as “Software Testing”
* Save the test case with .HTML Extension.

**EXECUTION**

**/\*Copy and Paste Screenshot\*/**

**SAMPLE:**



**RESULT:**

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

**Exp. No. : 8**

**Date :**

**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 Script.

**IMPLEMENTATION**

**INSTALLATION**

Step 1: Download the Selenium Server Standalone as follows:

https://www.seleniumhq.org/download/ 🡪 Latest Release: ChromeDriver 2.43 🡪

Selenium Server Standalone.

Step 2: Download Selenium Web Driver from https://www.seleniumhq.org/download/ 🡪

Third Party Browser Drivers not developed by seleniumhq 🡪 Google Chrome Driver

Step 3: Extract the jar file of Selenium Server Standalone and add it to the project (eclipse)

created as follows: Right Click on the Project 🡪 Build Path 🡪 Configure Build Path 🡪

Library (tab) 🡪 Add External Jar 🡪 Add the Selenium Server Standalone jar.

**JAVA SCRIPT**

import org.openqa.selenium.By;

import org.openqa.selenium.chrome.ChromeDriver;

public class Demo1 {

public static void main(String[] args)

{

System.setProperty("webdriver.chrome.driver",

C:\\Users\\User\\Downloads\\chromedriver.exe");

ChromeDriver driver = new ChromeDriver();

driver.get("http://www.newtours.demoaut.com");

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

driver.findElement(By.name("userName")).sendKeys("mercury");

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

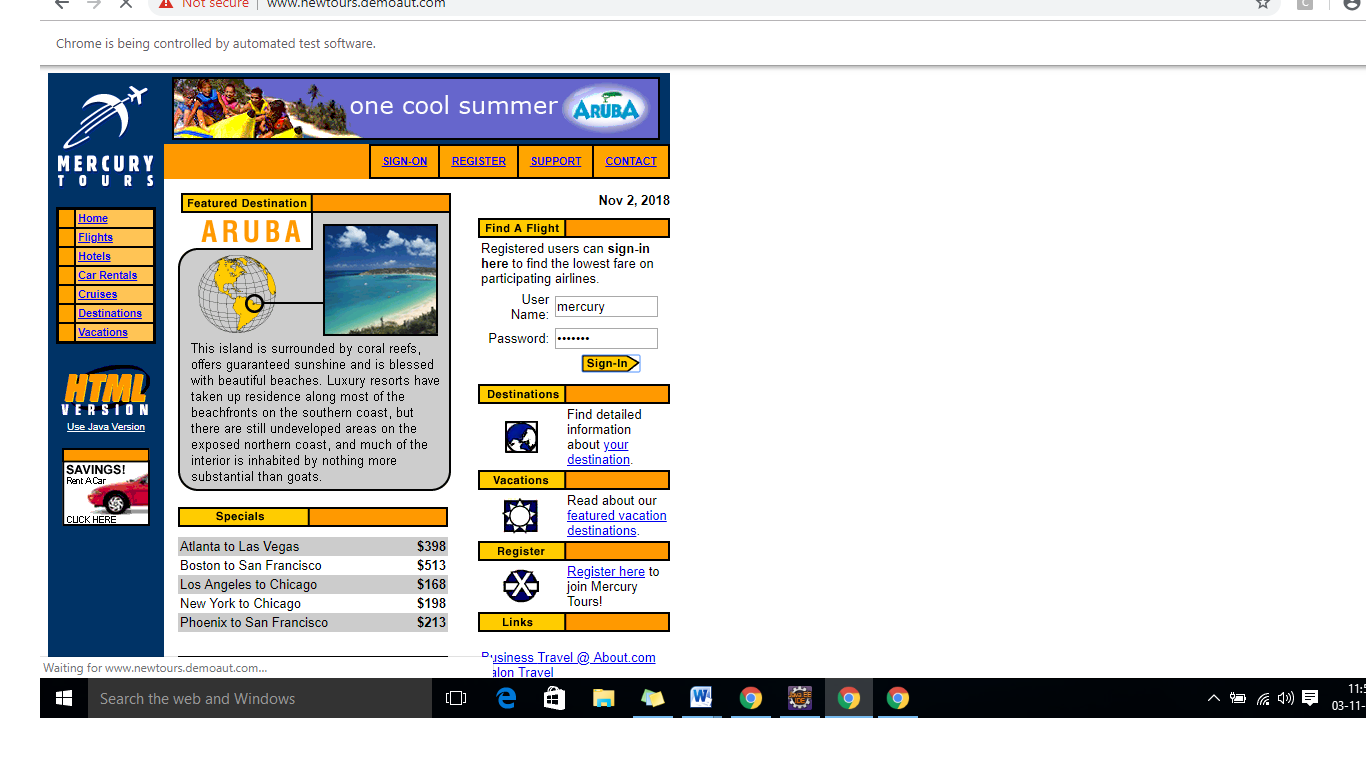
driver.findElement(By.name("login")).click();

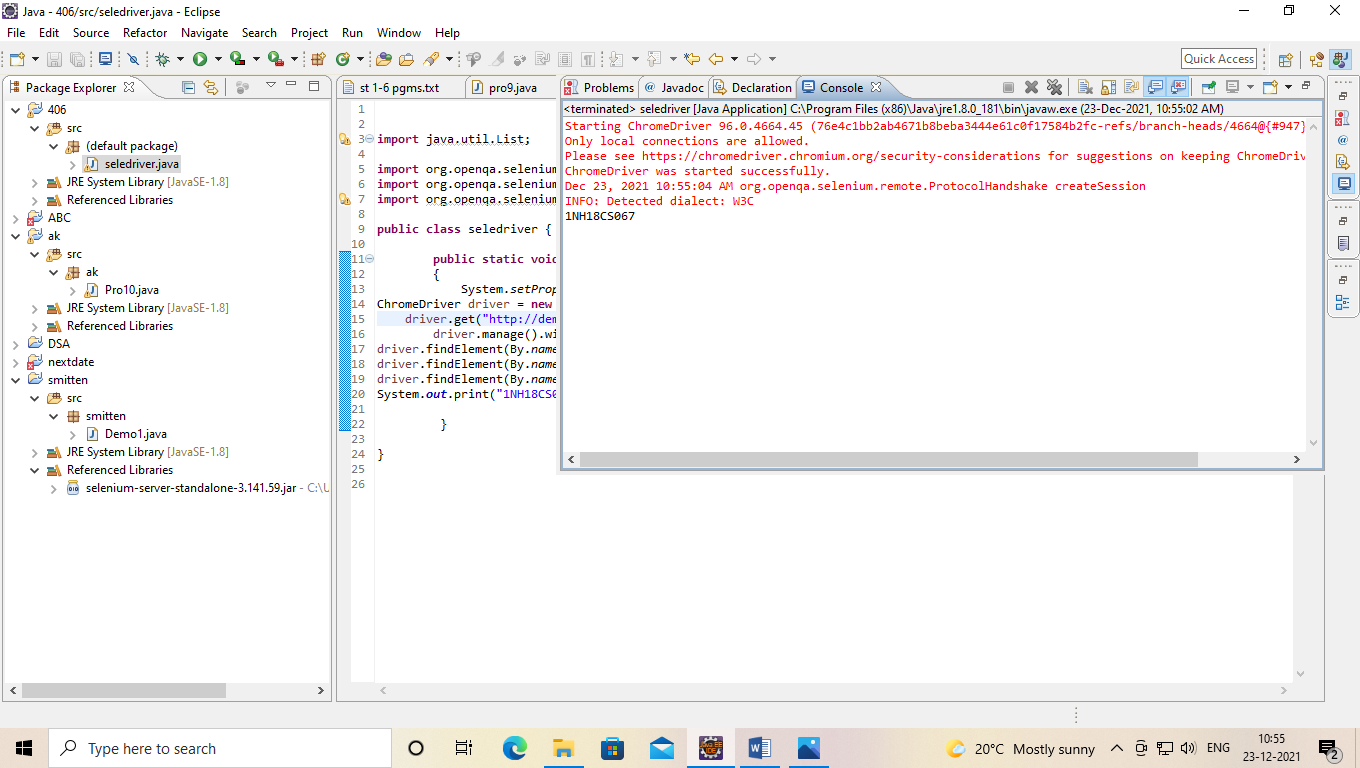
}

}

**EXECUTION**

**SAMPLE**





**RESULT:**

Thus, the above program is written and executed using selenium web driver.

**Exp. No. : 9**

**Date :**

**DEMONSTRATION OF SELENIUM IDE & 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: Download the Selenium Server Standalone as follows:

https://www.seleniumhq.org/download/ 🡪 Latest Release: ChromeDriver 2.43 🡪

Selenium Server Standalone.

Step 2: Download Selenium Web Driver from https://www.seleniumhq.org/download/ 🡪

Third Party Browser Drivers not developed by seleniumhq 🡪 Google Chrome Driver

Step 3: Extract the jar file of Selenium Server Standalone and add it to the project (eclipse)

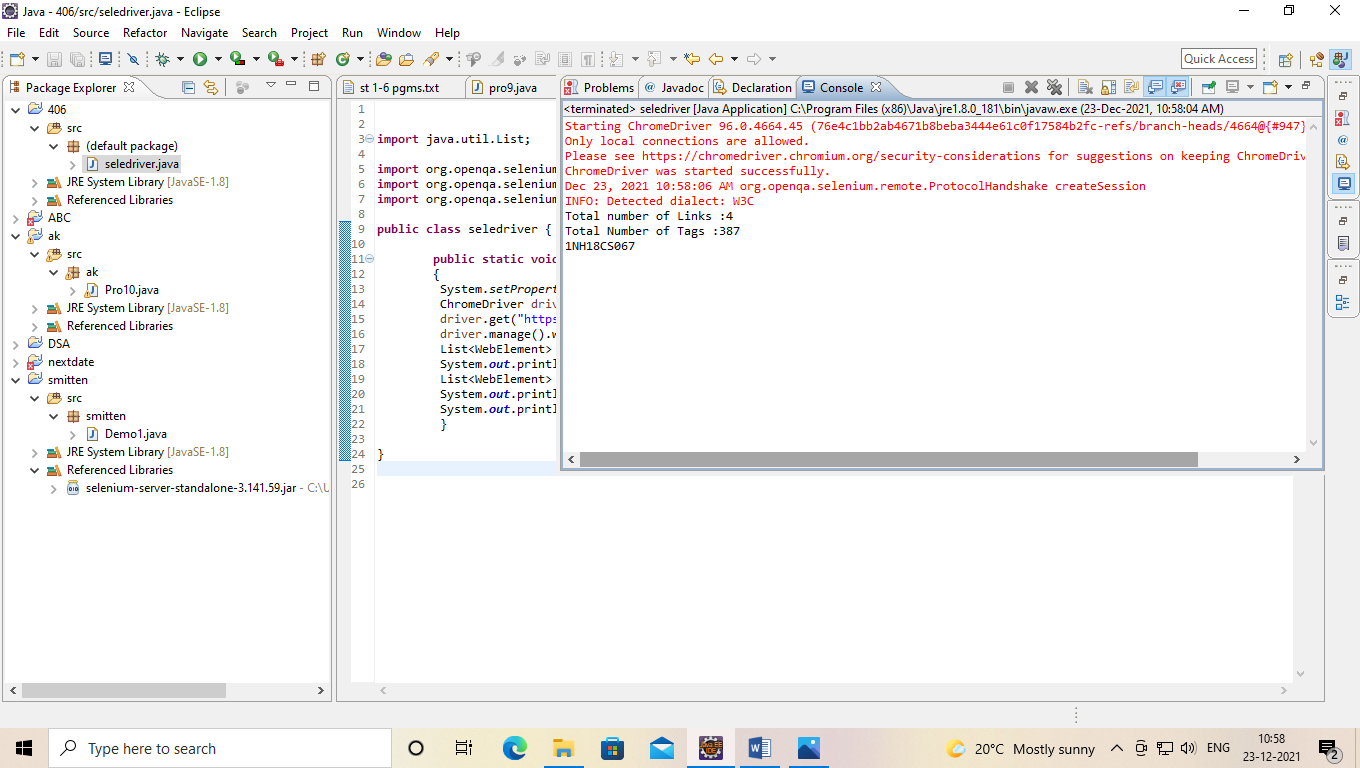
created as follows: Right Click on the Project 🡪 Build Path 🡪 Configure Build Path 🡪

Library (tab) 🡪 Add External Jar 🡪 Add the Selenium Server Standalone jar.

**PROGRAM**

package ex9;  
import [org.openqa.selenium.By](http://org.openqa.selenium.by/);  
import org.openqa.selenium.chrome.ChromeDriver;  
import org.openqa.selenium.WebElement;  
import java.util.List;  
  
public class links {  
public static void main(String[] args){  
System.setProperty("webdriver.chrome.driver","C:\\Users\\Student\\Downloads\\chromedriver\_win32 (1)\\chromedriver.exe");  
ChromeDriver d=new ChromeDriver();  
d.get("C:\\Users\\Student\\Desktop\\image.html");  
List <WebElement> a=d.findElements(By.xpath("//select"));  
int linkcount=a.size();  
System.out.println("total no of links ="+linkcount);  
  
List <WebElement> b=d.findElements(By.xpath("//\*"));  
int elements=b.size();  
System.out.println("total no of elements ="+elements);  
  
}  
  
}

**EXECUTION**

****

**RESULT**

Thus, the above program is written and executed using selenium web driver.

**Exp. No. : 10**

**Date :**

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

Write a test program to demonstrate URL and title check point

**IMPLEMENTATION**

**INSTALLATION**

Step 1: Download the Selenium Server Standalone as follows:

https://www.seleniumhq.org/download/ 🡪 Latest Release: ChromeDriver 2.43 🡪

Selenium Server Standalone.

Step 2: Download Selenium Web Driver from https://www.seleniumhq.org/download/ 🡪

Third Party Browser Drivers not developed by seleniumhq 🡪 Google Chrome Driver

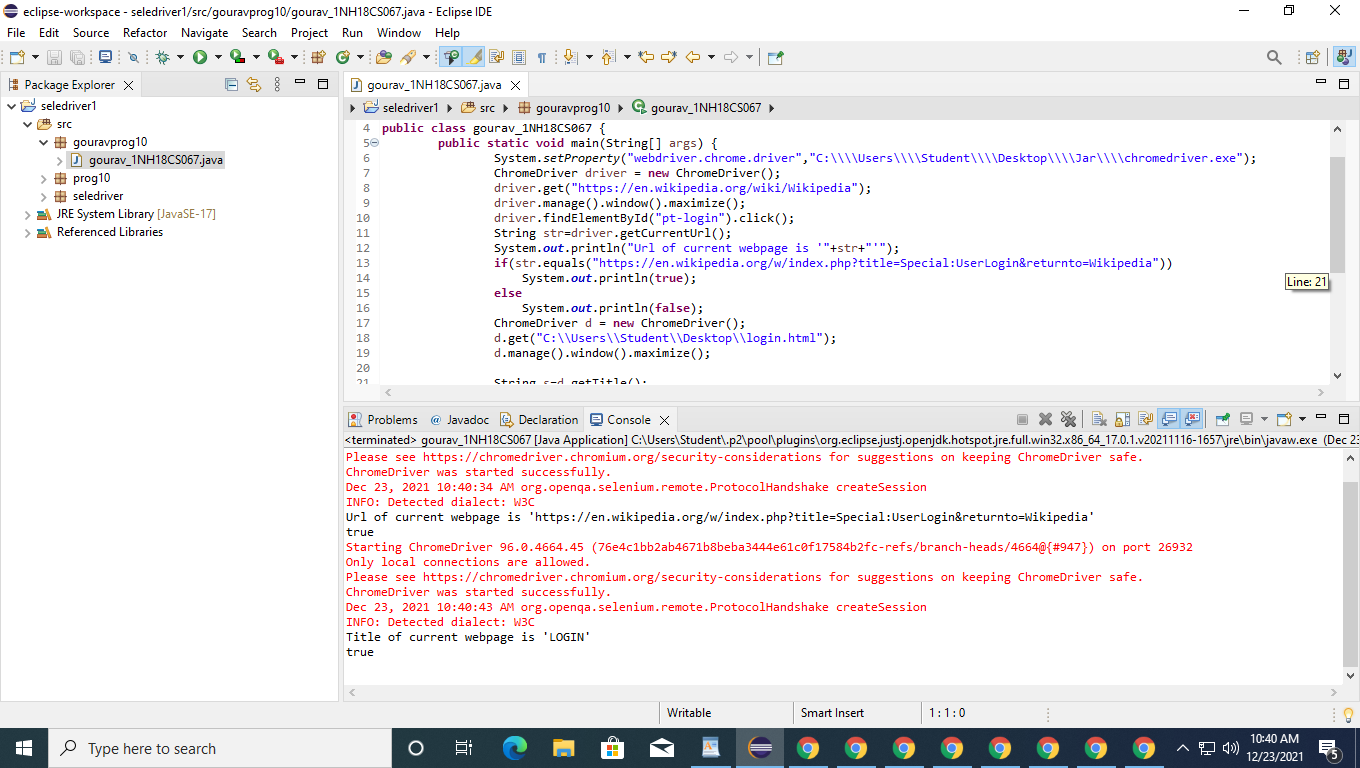
Step 3: Extract the jar file of Selenium Server Standalone and add it to the project (eclipse)

created as follows: Right Click on the Project 🡪 Build Path 🡪 Configure Build Path 🡪

Library (tab) 🡪 Add External Jar 🡪 Add the Selenium Server Standalone jar.

**PROGRAM**  
package progten;  
import org.openqa.selenium.chrome.ChromeDriver;  
public class gourav\_1NH18CS067{  
public static void main(String[] args) {  
System.setProperty("webdriver.chrome.driver","C:\\\\Users\\\\Student\\\\Desktop\\\\Jar\\\\chromedriver.exe");  
ChromeDriver driver = new ChromeDriver();  
driver.get("<https://en.wikipedia.org/wiki/Wikipedia>");  
driver.manage().window().maximize();  
driver.findElementById("pt-login").click();  
String str=driver.getCurrentUrl();  
System.out.println("Url of current webpage is '"+str+"'");  
if(str.equals("<https://en.wikipedia.org/w/index.php?title=Special:UserLogin&returnto=Wikipedia>"))  
System.out.println(true);  
else  
System.out.println(false);  
ChromeDriver d = new ChromeDriver();  
d.get("C:\\Users\\Student\\Desktop\\login.html");  
d.manage().window().maximize();  
  
String s=d.getTitle();  
System.out.println("Title of current webpage is '"+s+"'");  
if(s.equals("LOGIN"))  
System.out.println(true);  
else  
System.out.println(false);  
}  
}

**EXECUTION**



**RESULT**

Thus, the above program is written and executed using selenium web driver.

**Exp. No. : 11**

**Date :**

**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: Download the Selenium Server Standalone as follows:

https://www.seleniumhq.org/download/ 🡪 Latest Release: ChromeDriver 2.43 🡪

Selenium Server Standalone.

Step 2: Download Selenium Web Driver from https://www.seleniumhq.org/download/ 🡪

Third Party Browser Drivers not developed by seleniumhq 🡪 Google Chrome Driver

Step 3: Extract the jar file of Selenium Server Standalone and add it to the project (eclipse)

created as follows: Right Click on the Project 🡪 Build Path 🡪 Configure Build Path 🡪

Library (tab) 🡪 Add External Jar 🡪 Add the Selenium Server Standalone jar.

**PROGRAM**

package seledriver;

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 dropdown3 {

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

//pgm 11\_Write a test program to demonstrate selecting and

//deselecting option from multi select dropdown

//Creating instance of Chrome driver

System.setProperty("webdriver.chrome.driver",

"D:\\Software\\Eclipse&JAR\\Jar\\chromedriver\_win32\\chromedriver.exe");

WebDriver driver = new ChromeDriver();

// Navigate to the URL

driver.get("https://demoqa.com/select-menu");

//driver.get("file:///D:/NHCE/academic%20files/Academic%20files%20ODD%2021-22/ST/st%20lab/LAB-Checked/dropdown.html");

//Maximizing window

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

//Selecting the multi-select element by locating its id

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

//Get the list of all the options

System.out.println("The dropdown options are -");

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

for(WebElement option: options)

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

//Using isMultiple() method to verify if the element is multi-select,

//if yes go onto next steps else exit

if(select.isMultiple()){

//Selecting option as 'Opel'-- ByIndex

System.out.println("Select option Opel by Index");

select.selectByIndex(2);

Thread.sleep(5000);

//Selecting the option as 'Saab'-- ByValue

System.out.println("Select option saab by Value");

select.selectByValue("saab");

Thread.sleep(5000);

// Selecting the option by text

System.out.println("Select option Audi by Text");

select.selectByVisibleText("Audi");

Thread.sleep(5000);

//Get the list of selected options

System.out.println("The selected values in the dropdown options are -");

List<WebElement> selectedOptions = select.getAllSelectedOptions();

for(WebElement selectedOption: selectedOptions)

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

// Deselect the value "Audi" by Index

System.out.println("DeSelect option Audi by Index");

select.deselectByIndex(3);

Thread.sleep(10000);

//Deselect the value "Opel" by visible text

System.out.println("Select option Opel by Text");

select.deselectByVisibleText("Opel");

//Thread.sleep(10000);

//Validate that both the values are deselected

System.out.println("The selected values after deselect in the dropdown options are -");

List<WebElement> selectedOptionsAfterDeselect = select.getAllSelectedOptions();

for(WebElement selectedOptionAfterDeselect: selectedOptionsAfterDeselect)

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

//Step#8- Deselect all values

select.deselectAll();

}

driver.quit();

}

}

**RESULT**

**Exp. No. : 12**

**Date :**

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

Write a test program to demonstrate Synchronization

**IMPLEMENTATION**

**INSTALLATION**

Step 1: Download the Selenium Server Standalone as follows:

https://www.seleniumhq.org/download/ 🡪 Latest Release: ChromeDriver 2.43 🡪

Selenium Server Standalone.

Step 2: Download Selenium Web Driver from https://www.seleniumhq.org/download/ 🡪

Third Party Browser Drivers not developed by seleniumhq 🡪 Google Chrome Driver

Step 3: Extract the jar file of Selenium Server Standalone and add it to the project (eclipse)

created as follows: Right Click on the Project 🡪 Build Path 🡪 Configure Build Path 🡪

Library (tab) 🡪 Add External Jar 🡪 Add the Selenium Server Standalone jar.

**PROGRAM**

**IMPLICIT**

package seledriver;

import java.util.concurrent.TimeUnit;

import org.openqa.selenium.By;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeDriver;

//import org.testng.annotations.Test;

public class Implicit12\_final {

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

System.setProperty ("webdriver.chrome.driver",

"D:\\Software\\Eclipse&JAR\\Jar\\chromedriver\_win32\\chromedriver.exe" );

ChromeDriver driver = new ChromeDriver();

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

String eTitle = "Demo Guru99 Page";

String aTitle = "" ;

// launch Chrome and redirect it to the Base URL

driver.get("http://demo.guru99.com/test/guru99home/" );

//Maximizes the browser window

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

//get the actual value of the title

aTitle = driver.getTitle();

//compare the actual title with the expected title

if (aTitle.equals(eTitle))

{

System.out.println( "Test Passed") ;

}

else {

System.out.println( "Test Failed" );

}

//close browser

driver.close();

}

}

**EXPLICIT**

Package seledriver;

import java.util.List;

import java.util.concurrent.TimeUnit;

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.ExpectedConditions;

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

public class Explicit12\_final {

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

System.setProperty ("webdriver.chrome.driver",

"D:\\Software\\Eclipse&JAR\\Jar\\chromedriver\_win32\\chromedriver.exe" );

ChromeDriver driver = new ChromeDriver();

WebDriverWait wait=new WebDriverWait(driver, 10);

String eTitle = "Demo Guru99 Page";

String aTitle = "" ;

// launch Chrome and redirect it to the Base URL

driver.get("http://demo.guru99.com/test/guru99home/" );

//Maximizes the browser window

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

//get the actual value of the title

aTitle = driver.getTitle();

//compare the actual title with the expected title

if (aTitle.contentEquals(eTitle))

{

System.out.println( "Test Passed") ;

}

else {

System.out.println( "Test Failed" );

}

//driver.close();

WebElement guru99=wait.until(ExpectedConditions.visibilityOfElementLocated(By.xpath( "//a")));

guru99.click();

}

}

**RESULT**