**Name:Anisha M**

**Superset ID:6371860**

**WEEK-2 HANDS-ON SOLUTION**

**JUnit, Mockito and SL4J**

**Exercise 3: Assertions in junit**

**Code:**

public class Main {

public static void main(String[] args) {

System.out.println("Running simulated assertion tests...");

if (2 + 3 == 5) {

System.out.println("assertEquals passed");

} else {

System.out.println("assertEquals failed");

}

if (5 > 3) {

System.out.println("assertTrue passed");

} else {

System.out.println("assertTrue failed");

}

if (!(5 < 3)) {

System.out.println("assertFalse passed");

} else {

System.out.println("assertFalse failed");

}

Object obj1 = null;

if (obj1 == null) {

System.out.println("assertNull passed");

} else {

System.out.println("assertNull failed");

}

Object obj2 = new Object();

if (obj2 != null) {

System.out.println("assertNotNull passed");

} else {

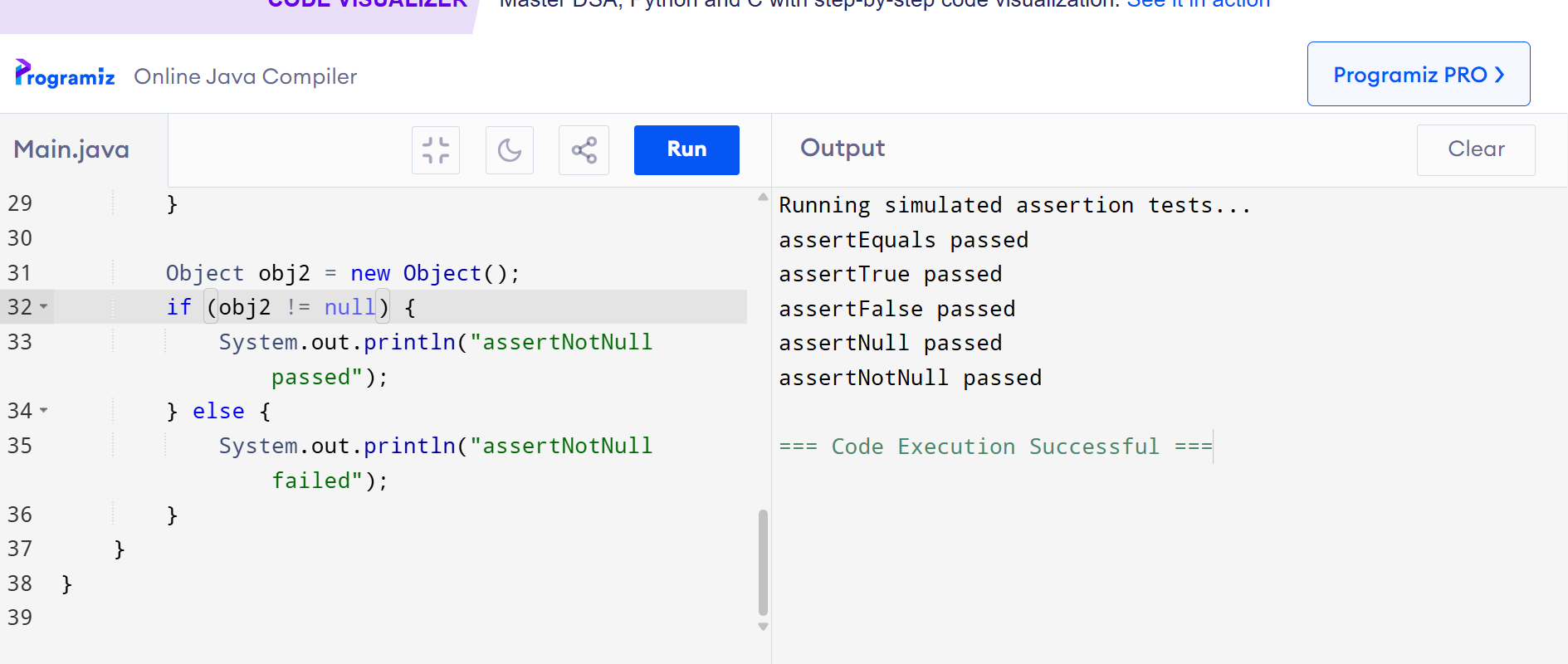
System.out.println("assertNotNull failed");

}

}

}

**Output:**



**Exercise 4: Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit**

**Code:**

class Calculator {

public int add(int a, int b) {

return a + b;

}

public int subtract(int a, int b) {

return a - b;

}

}

public class Main {

static Calculator calculator;

public static void main(String[] args) {

setUp();

System.out.println("Running testAdd:");

int a = 4;

int b = 6;

int addResult = calculator.add(a, b);

if (addResult == 10) {

System.out.println("testAdd passed");

} else {

System.out.println("testAdd failed: Expected 10 but got " + addResult);

}

tearDown();

setUp();

System.out.println("\nRunning testSubtract:");

int subResult = calculator.subtract(10, 4);

if (subResult == 6) {

System.out.println("testSubtract passed");

} else {

System.out.println("testSubtract failed: Expected 6 but got " + subResult);

}

tearDown();

}

public static void setUp() {

calculator = new Calculator();

System.out.println("\nSetup before test");

}

public static void tearDown() {

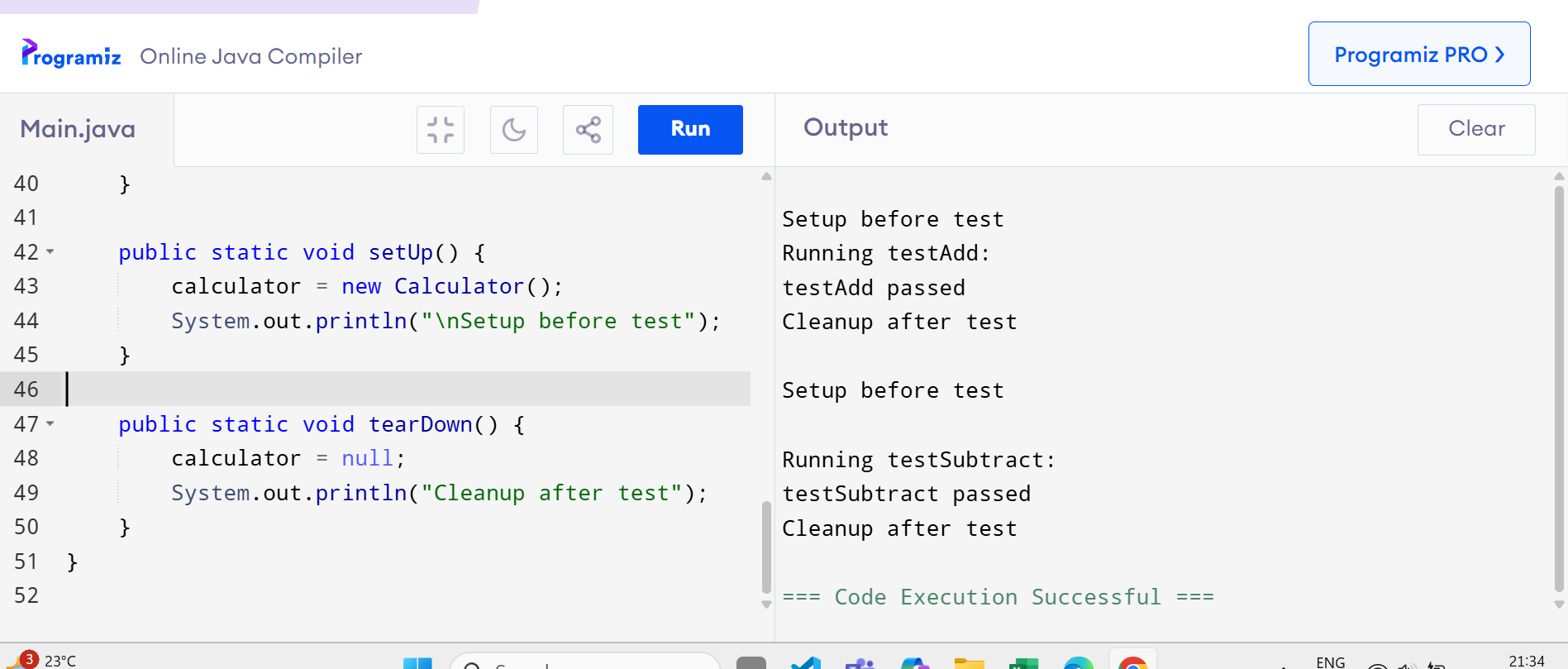
calculator = null;

System.out.println("Cleanup after test");

}

}

**Output:**

****

**Exercise 1: Mocking and Stubbing**

**Code:**

interface ExternalApi {

String getData();

}

class MockExternalApi implements ExternalApi {

public String getData() {

return "Mock Data";

}

}

class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public String fetchData() {

return api.getData();

}

}

public class Main {

public static void main(String[] args) {

ExternalApi mockApi = new MockExternalApi();

MyService service = new MyService(mockApi);

String result = service.fetchData();

if ("Mock Data".equals(result)) {

System.out.println("testExternalApi passed");

} else {

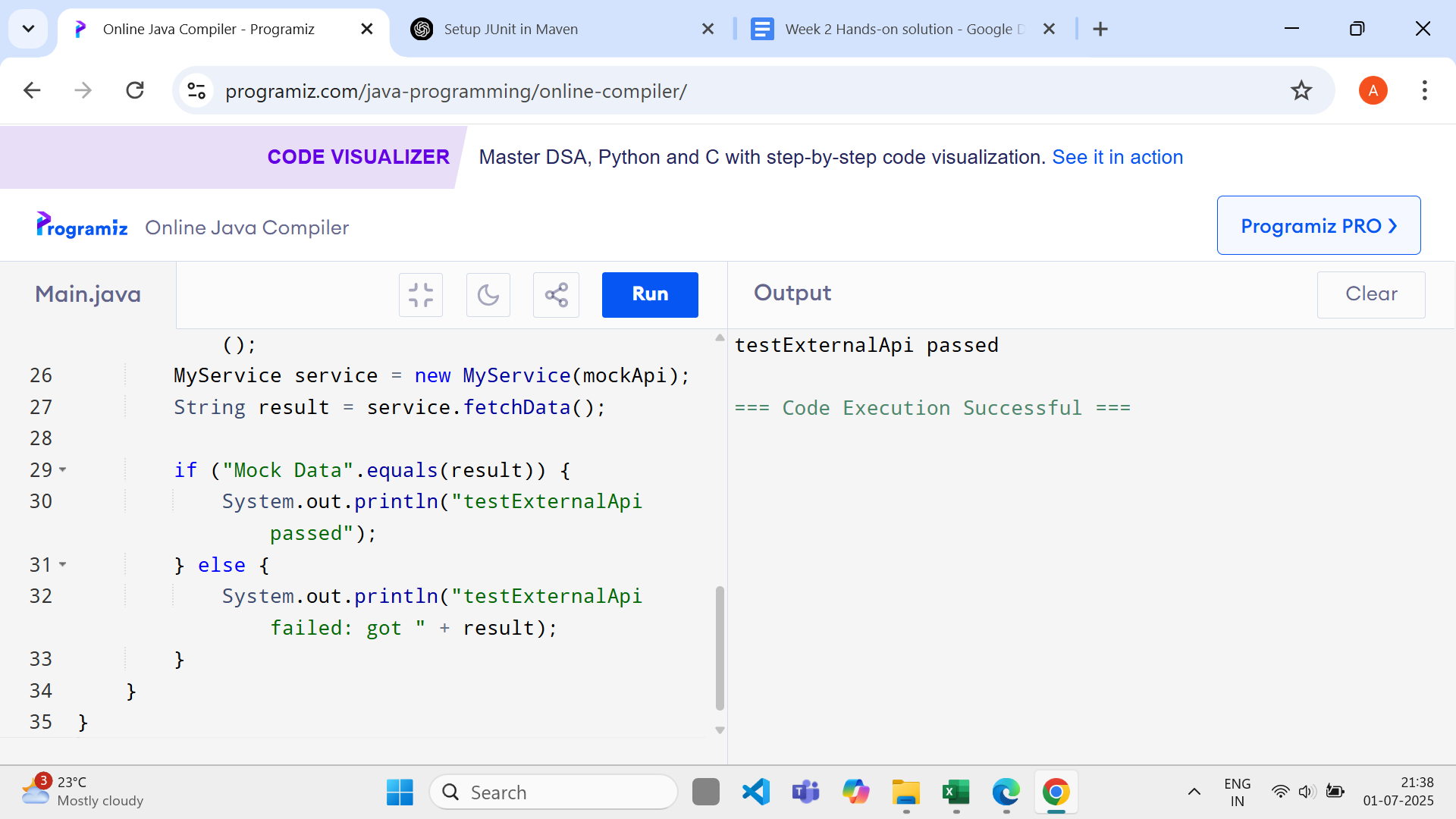
System.out.println("testExternalApi failed: got " + result);

}

}

}

**Output:**



**Exercise 2:**

**Code:**

interface ExternalApi {

String getData();

}

class MockExternalApi implements ExternalApi {

boolean called = false;

public String getData() {

called = true;

return "Mock Data";

}

public boolean wasCalled() {

return called;

}

}

class MyService {

private ExternalApi api;

public MyService(ExternalApi api) {

this.api = api;

}

public void fetchData() {

api.getData();

}

}

public class Main {

public static void main(String[] args) {

MockExternalApi mockApi = new MockExternalApi();

MyService service = new MyService(mockApi);

service.fetchData();

if (mockApi.wasCalled()) {

System.out.println("testVerifyInteraction passed");

} else {

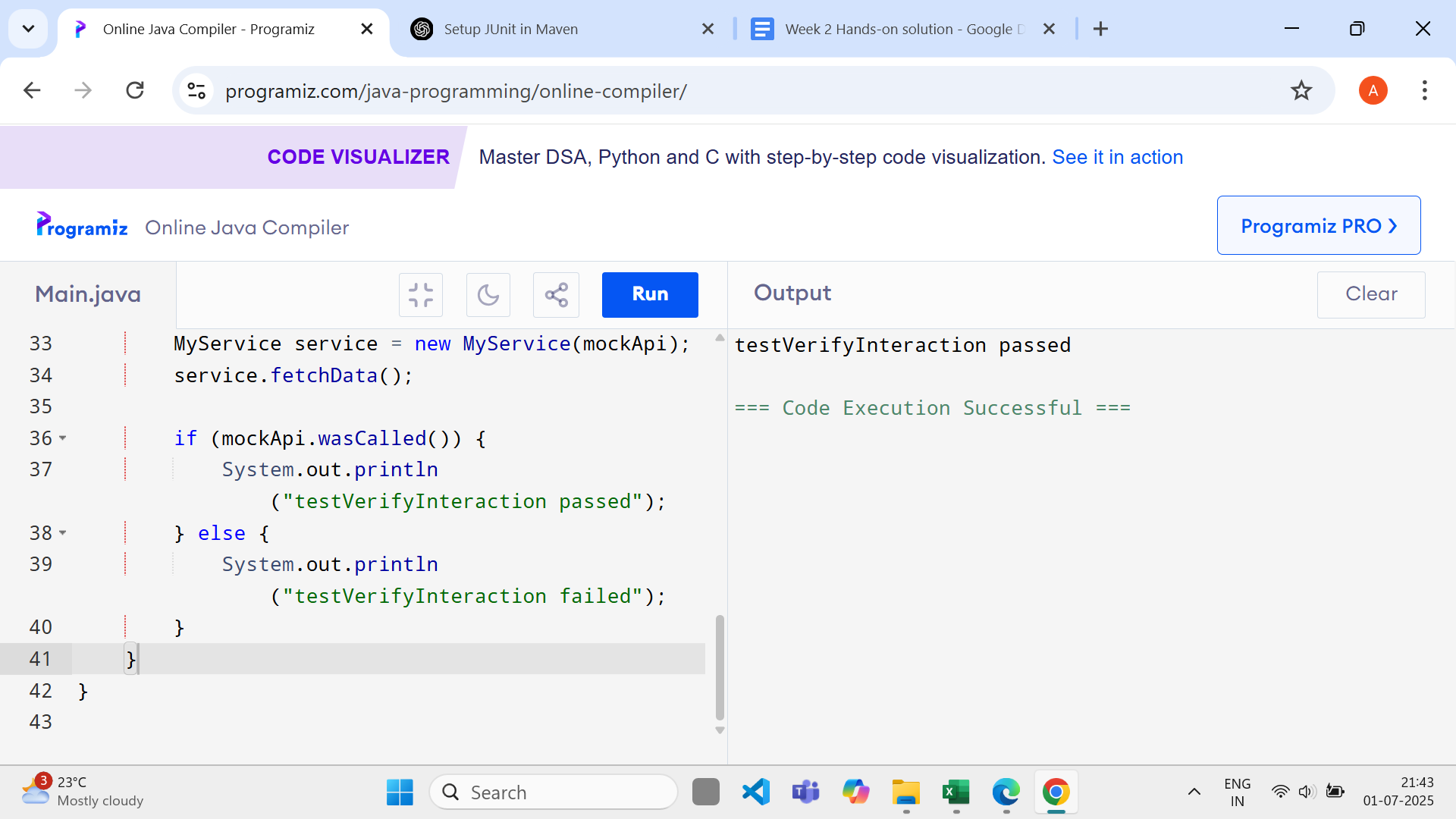
System.out.println("testVerifyInteraction failed");

}

}

}

**Output:**

****

**Exercise 1:**

**Code:**

public class Main {

public static void main(String[] args) {

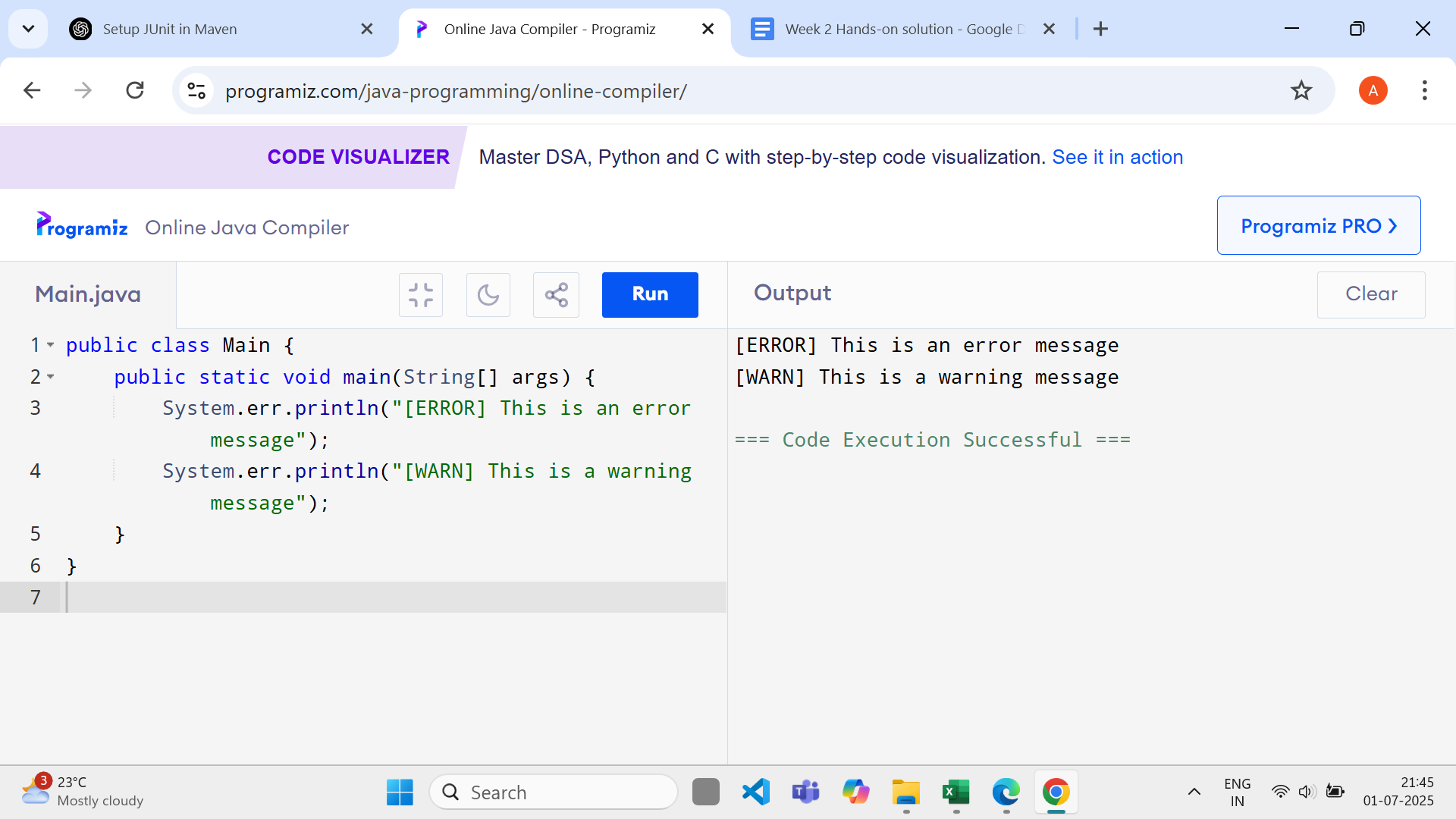
System.err.println("[ERROR] This is an error message");

System.err.println("[WARN] This is a warning message");

}

}

**Output:**

****