**Exercise 1: Setting Up JUnit**

**CODE**

**Main.java**

package com.example;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

PrimeUtils utils = new PrimeUtils();

Scanner scanner = new Scanner(System.in);

System.out.print("Enter a number to check if it is prime: ");

int number = scanner.nextInt();

boolean result = utils.isPrime(number);

if (result) {System.out.println(number + " is a prime number."); }

else {System.out.println(number + " is NOT a prime number."); }

}}

**PrimeUtils.java**

package com.example;

public class PrimeUtils {

public boolean isPrime(int n) {

if (n < 2) return false;

if (n == 2) return true;

if (n % 2 == 0) return false;

for (int i = 3; i \* i <= n; i += 2) {

if (n % i == 0) return false;

}

return true;

}

}

**PrimeUtilsTest.java**

package com.example;

import org.junit.Test;

import com.example.PrimeUtils;

import static org.junit.Assert.\*;

public class PrimeUtilsTest {

private final PrimeUtils utils = new PrimeUtils();

@Test

public void testSmallPrimes() {

assertTrue(utils.isPrime(2));

assertTrue(utils.isPrime(3));

assertTrue(utils.isPrime(5));

assertTrue(utils.isPrime(7));

}

@Test

public void testSmallComposites() {

assertFalse(utils.isPrime(0));

assertFalse(utils.isPrime(1));

assertFalse(utils.isPrime(4));

assertFalse(utils.isPrime(6));

assertFalse(utils.isPrime(9));

}

@Test

public void testLargeNumbers() {

assertTrue(utils.isPrime(97));

assertFalse(utils.isPrime(100));

}

@Test

public void testNegativeNumbers() {

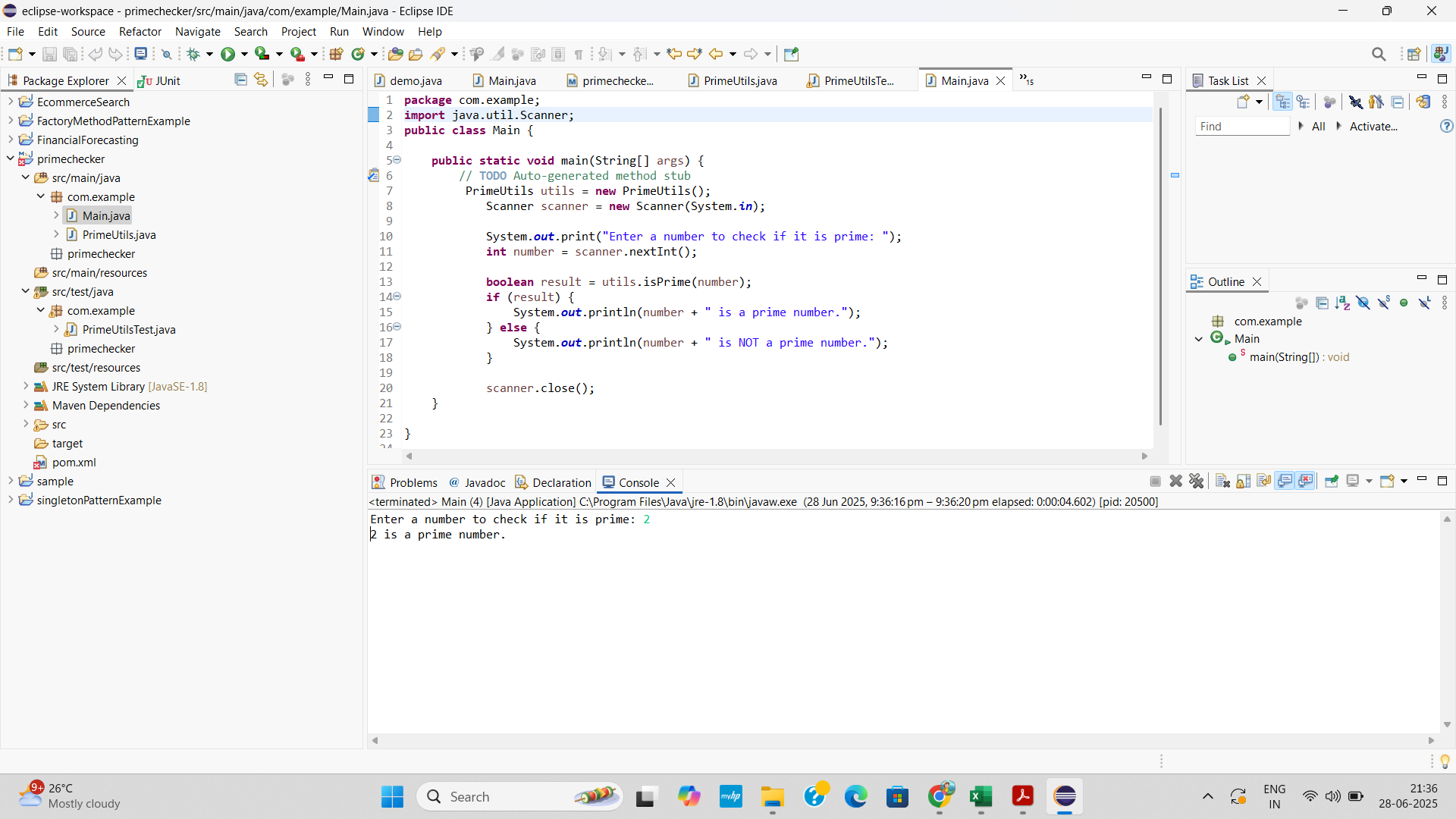
assertFalse(utils.isPrime(-11));

assertFalse(utils.isPrime(-2));

}

}

**OUTPUT**

****