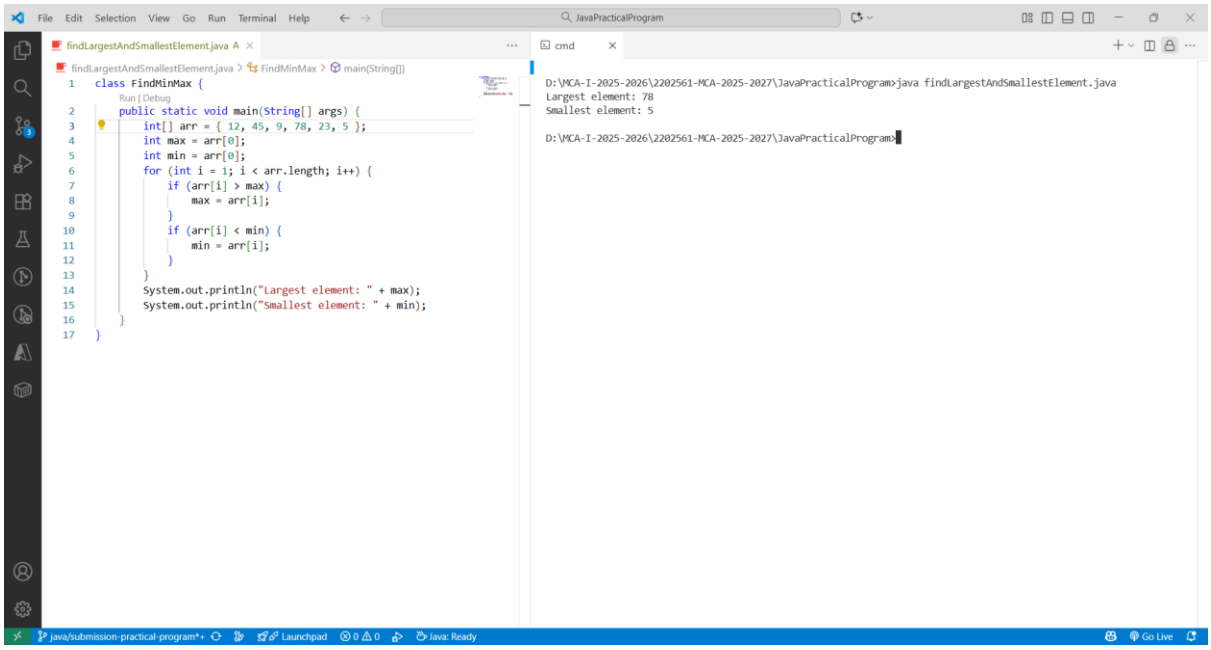
Name	Udhayarajan J
Class	MCA - I
Sub	Java Programming
Type	Experiment Practical Program
Date	2025-11-12
Roll No	2202561

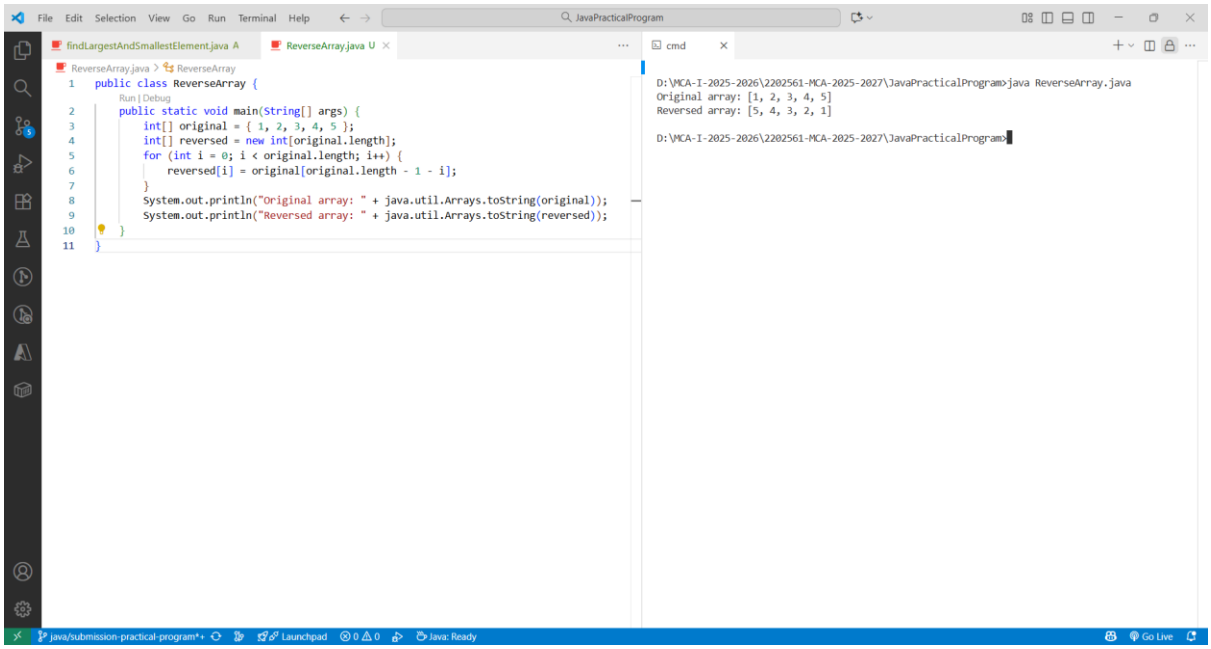
Experiment 2: Introduction of arrays and different operations on arrays

Design an application by using arrays

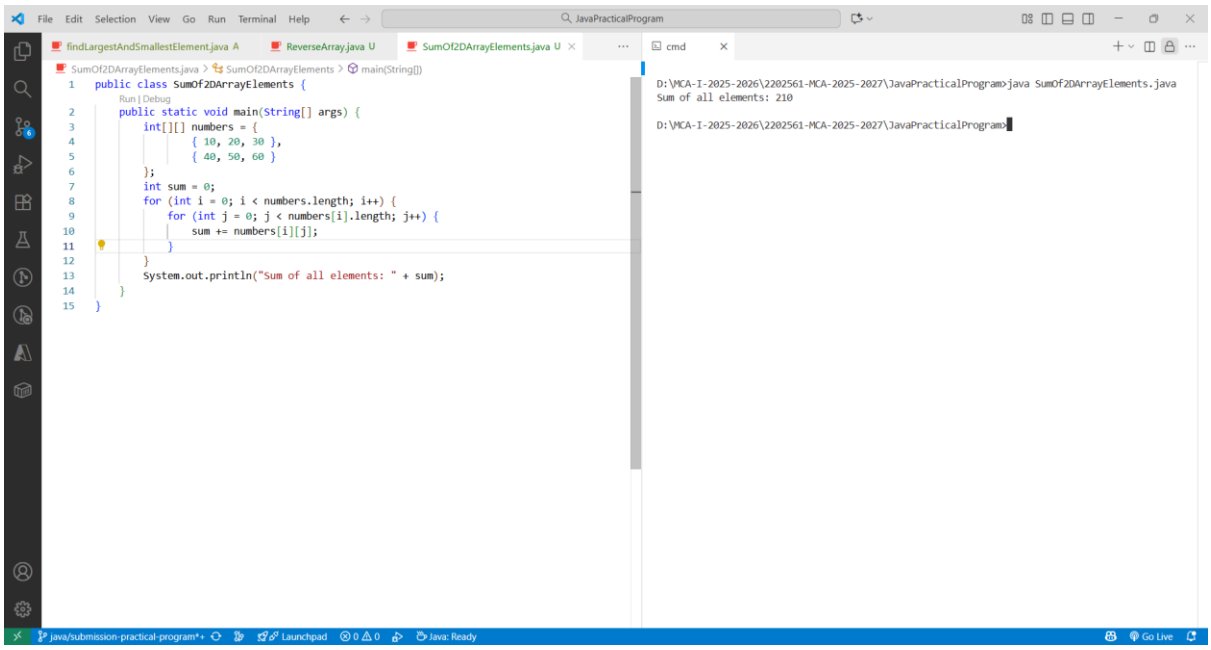
a. Finding the Largest/Smallest Element



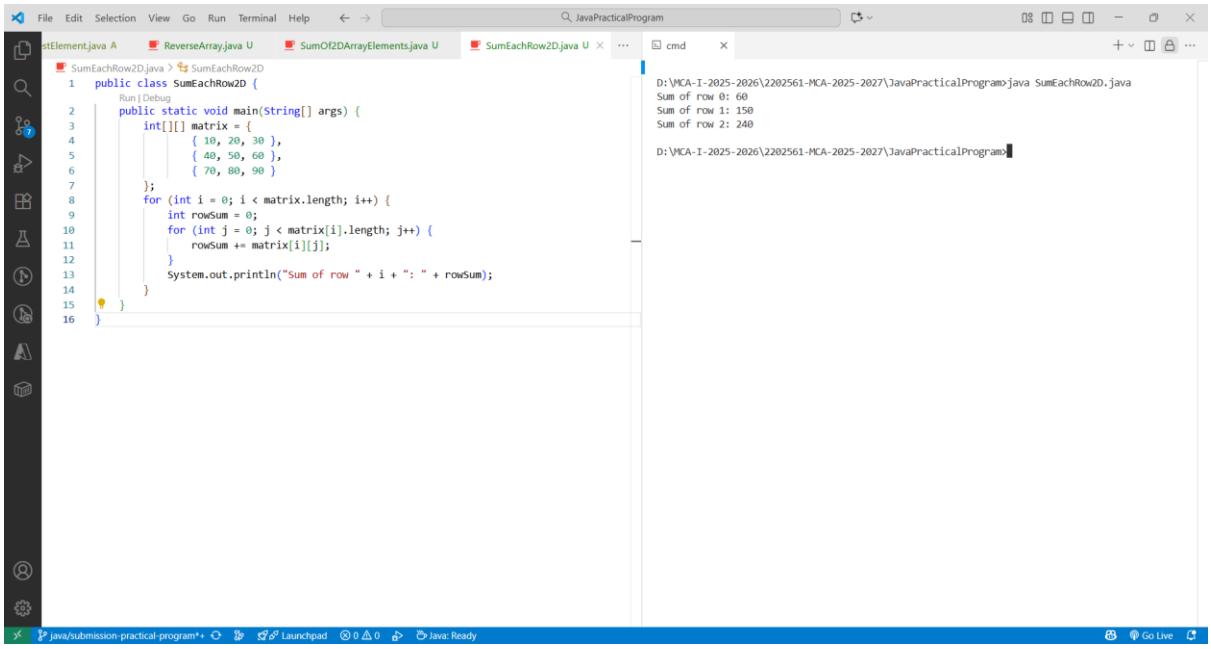
b. Reversing an Array



c. Sum of all elements in a 2D Array

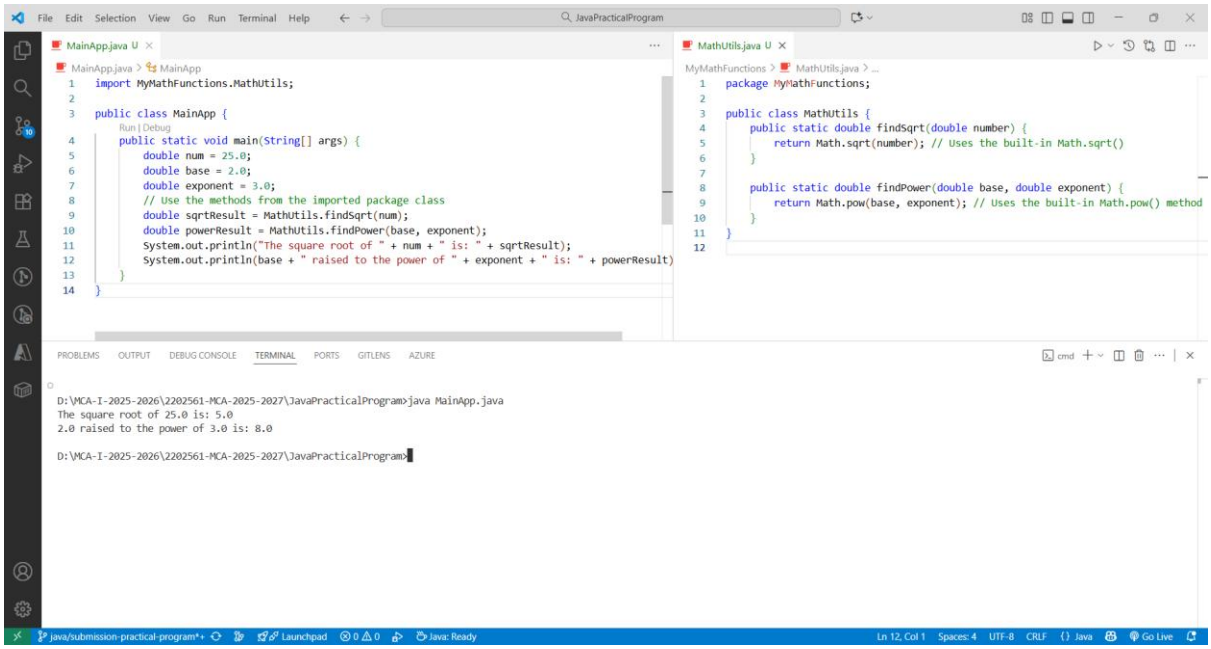


d. Sum of elements in each row of a 2D array

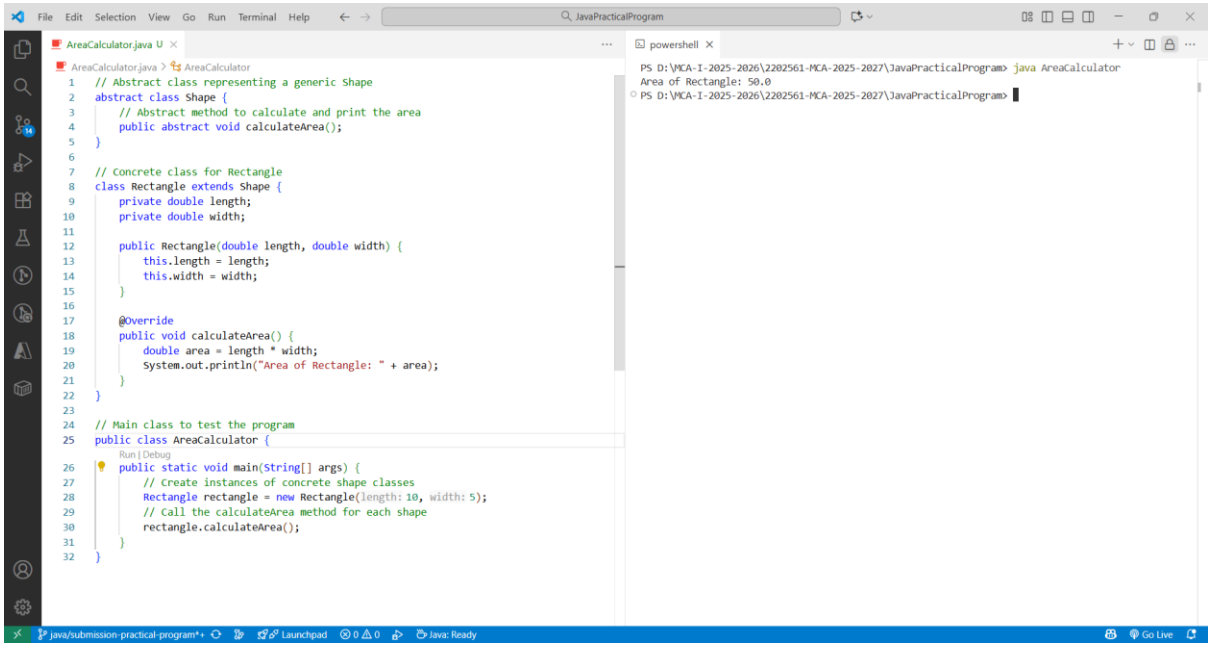


3: Package

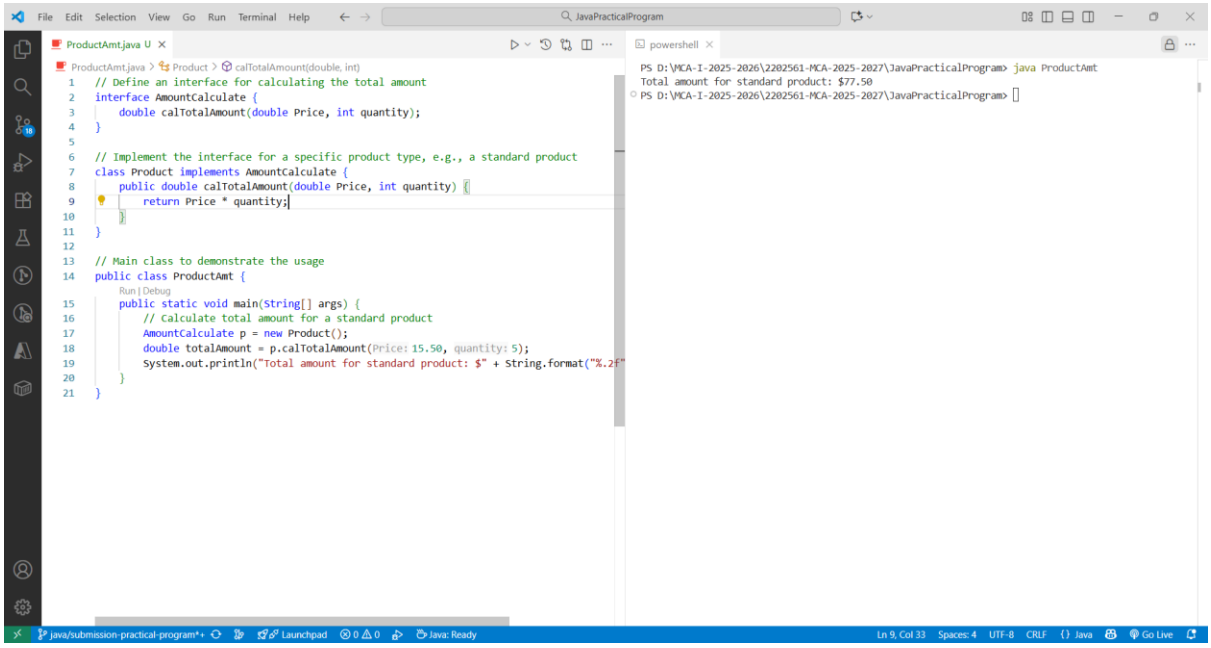
Write a program for Implementation of package to create class and methods in mynmathfunction package



b. abstract class Experiment for calculating area



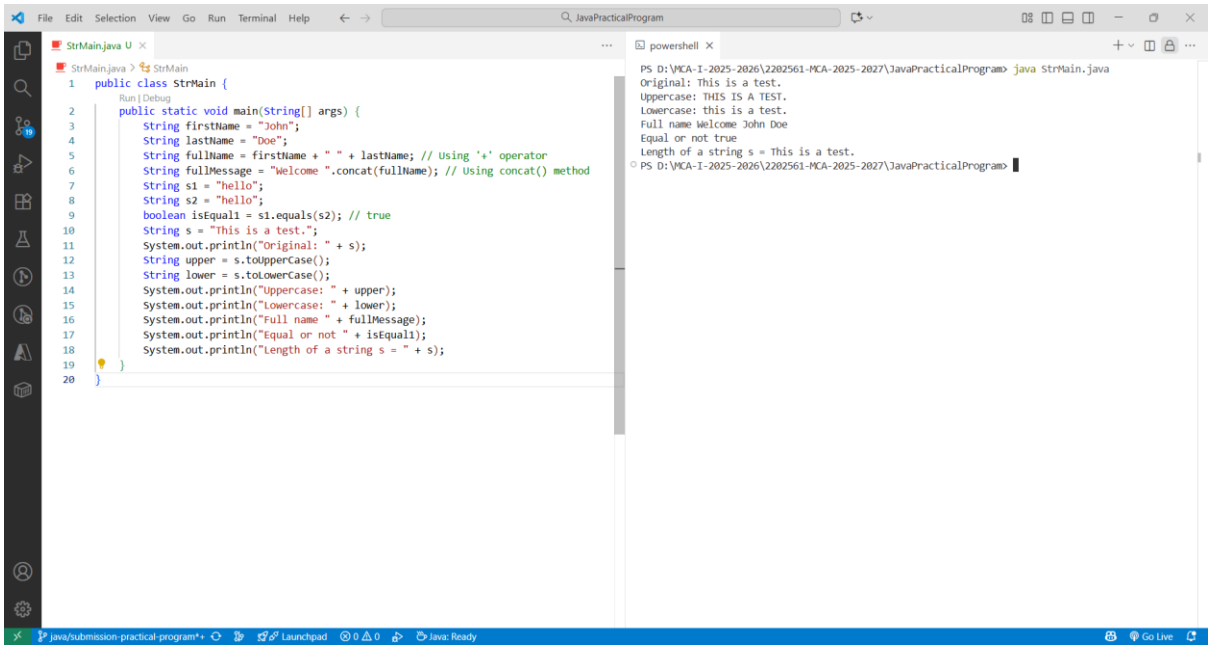
c. Interface program for calculating product amount



```java
// Define an interface for calculating the total amount
interface AmountCalculate {
    double calTotalAmount(double Price, int quantity);
}

// Implement the interface for a specific product type, e.g., a standard product
class Product implements AmountCalculate {
    public double calTotalAmount(double Price, int quantity) {
        return Price * quantity;
    }
}

// Main class to demonstrate the usage
public class ProductAmt {
    public static void main(String[] args) {
        // Calculate total amount for a standard product
        AmountCalculate p = new Product();
        double totalAmount = p.calTotalAmount(Price: 15.50, quantity: 5);
        System.out.println("Total amount for standard product: $" + String.format("%.2f"
    }
}
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java ProductAmt
Total amount for standard product: $77.50
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

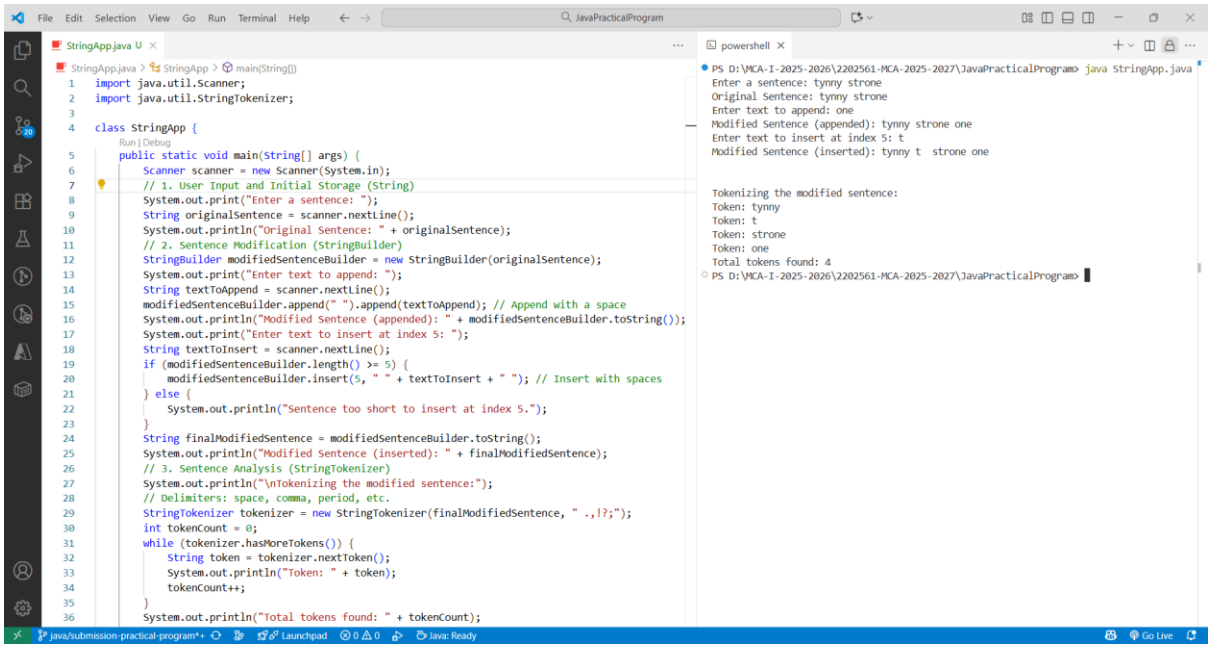
4: Introduction to string and its different operations
Use of Different string methods



```java
public class StrMain {
    public static void main(String[] args) {
        String firstName = "John";
        String lastName = "Doe";
        String fullName = firstName + " " + lastName; // Using '+' operator
        String fullMessage = "Welcome ".concat(fullName); // Using concat() method
        String s1 = "hello";
        String s2 = "hello";
        boolean isEqual1 = s1.equals(s2); // true
        String s = "This is a test.";
        System.out.println("Original: " + s);
        String upper = s.toUpperCase();
        String lower = s.toLowerCase();
        System.out.println("Uppercase: " + upper);
        System.out.println("Lowercase: " + lower);
        System.out.println("Full name " + fullMessage);
        System.out.println("Equal or not " + isEqual1);
        System.out.println("Length of a string s = " + s);
    }
}
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java StrMain.java
Original: This is a test.
Uppercase: THIS IS A TEST.
Lowercase: this is a test.
Full name Welcome John Doe
Equal or not true
Length of a string s = This is a test.
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

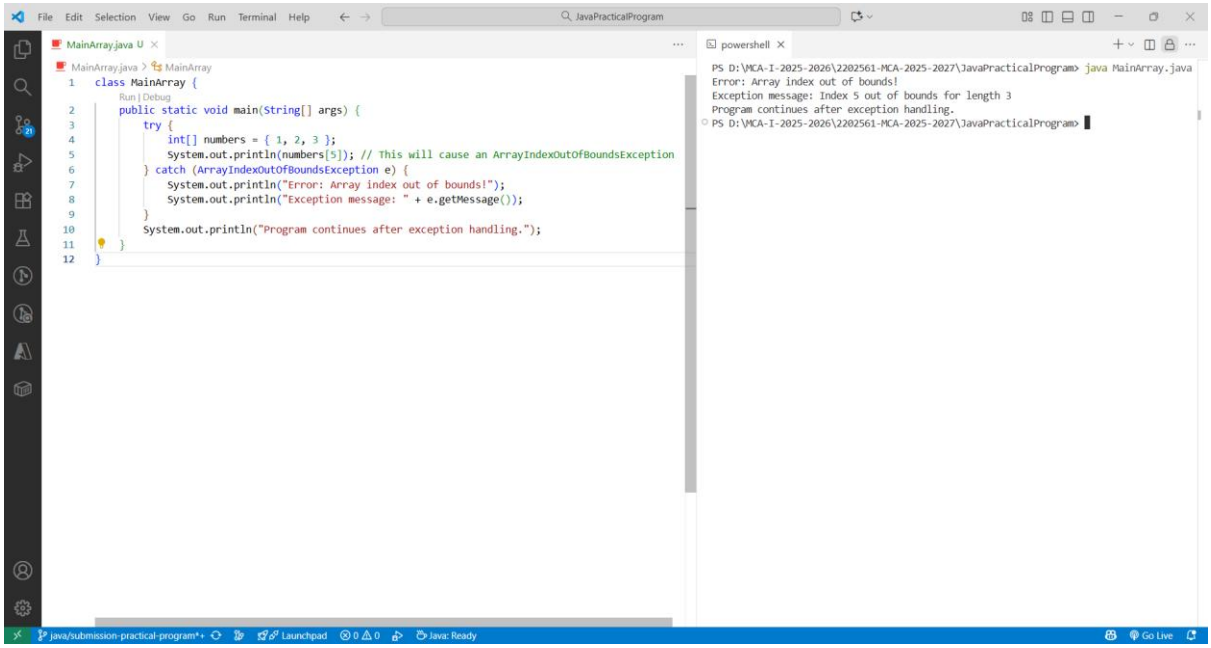
b. Design application using String, StringBuilder, StringTokenizer Experiment



```java
import java.util.Scanner;
import java.util.StringTokenizer;

class StringApp {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        // 1. User Input and Initial Storage (String)
        System.out.print("Enter a sentence: ");
        String originalSentence = scanner.nextLine();
        System.out.println("Original Sentence: " + originalSentence);
        // 2. Sentence Modification (StringBuilder)
        StringBuilder modifiedSentenceBuilder = new StringBuilder(originalSentence);
        System.out.print("Enter text to append: ");
        String textToAppend = scanner.nextLine();
        modifiedSentenceBuilder.append(" ").append(textToAppend); // Append with a space
        System.out.println("Modified Sentence (appended): " + modifiedSentenceBuilder.toString());
        System.out.print("Enter text to insert at index 5: ");
        String textToInsert = scanner.nextLine();
        if (modifiedSentenceBuilder.length() >= 5) {
            modifiedSentenceBuilder.insert(5, " " + textToInsert + " "); // Insert with spaces
        } else {
            System.out.println("Sentence too short to insert at index 5.");
        }
        String finalModifiedSentence = modifiedSentenceBuilder.toString();
        System.out.println("Modified Sentence (inserted): " + finalModifiedSentence);
        // 3. Sentence Analysis (StringTokenizer)
        System.out.println("\nTokenizing the modified sentence:");
        // Delimiters: space, comma, period, etc.
        StringTokenizer tokenizer = new StringTokenizer(finalModifiedSentence, " .,!?;");
        int tokenCount = 0;
        while (tokenizer.hasMoreTokens()) {
            String token = tokenizer.nextToken();
            System.out.println("Token: " + token);
            tokenCount++;
        }
        System.out.println("Total tokens found: " + tokenCount);
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java StringApp.java
Enter a sentence: tynny strone
Original Sentence: tynny strone
Enter text to append: one
Modified Sentence (appended): tynny strone one
Enter text to insert at index 5: t
Modified Sentence (inserted): tynny t  strone one


Tokenizing the modified sentence:
Token: tynny
Token: t
Token: strone
Token: one
Total tokens found: 4
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

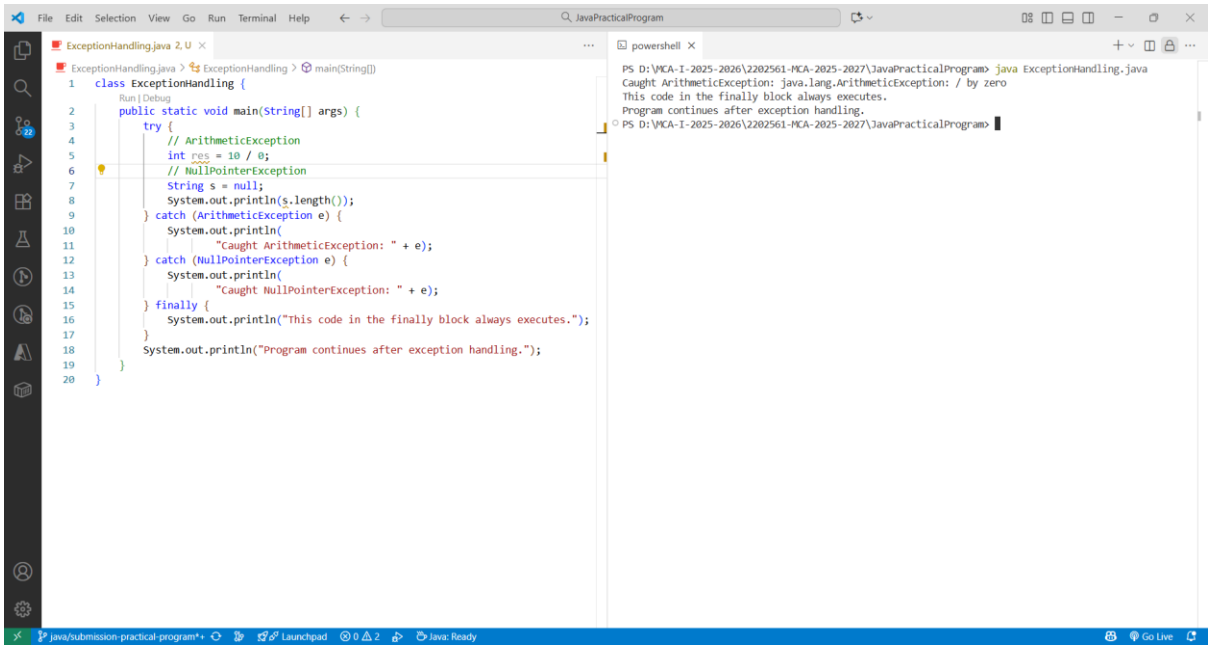
5: Exception Handling -Test any five of standard exception and user Defined Custom Exceptions in java



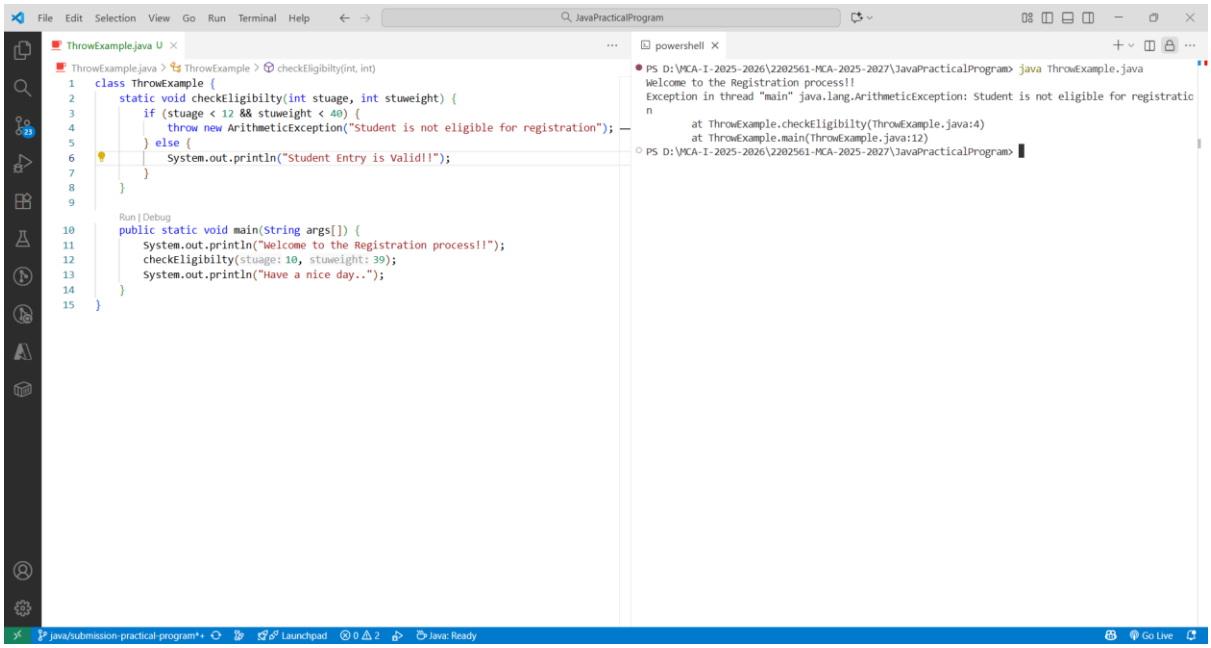
```java
class MainArray {
    public static void main(String[] args) {
        try {
            int[] numbers = { 1, 2, 3 };
            System.out.println(numbers[5]); // This will cause an ArrayIndexOutOfBoundsException
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Error: Array index out of bounds!");
            System.out.println("Exception message: " + e.getMessage());
        }
        System.out.println("Program continues after exception handling.");
    }
}
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java MainArray.java
Error: Array index out of bounds!
Exception message: Index 5 out of bounds for length 3
Program continues after exception handling.
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

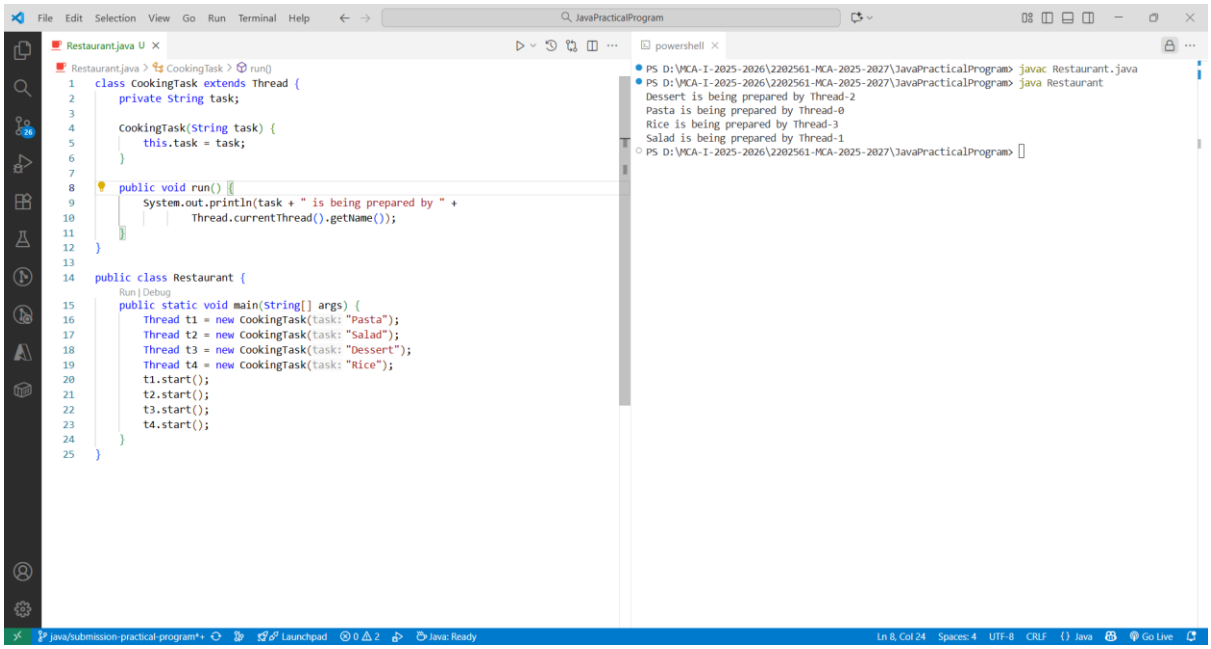
c. Try-catch-finally block with multiple catch statements.



d. User-defined exceptions



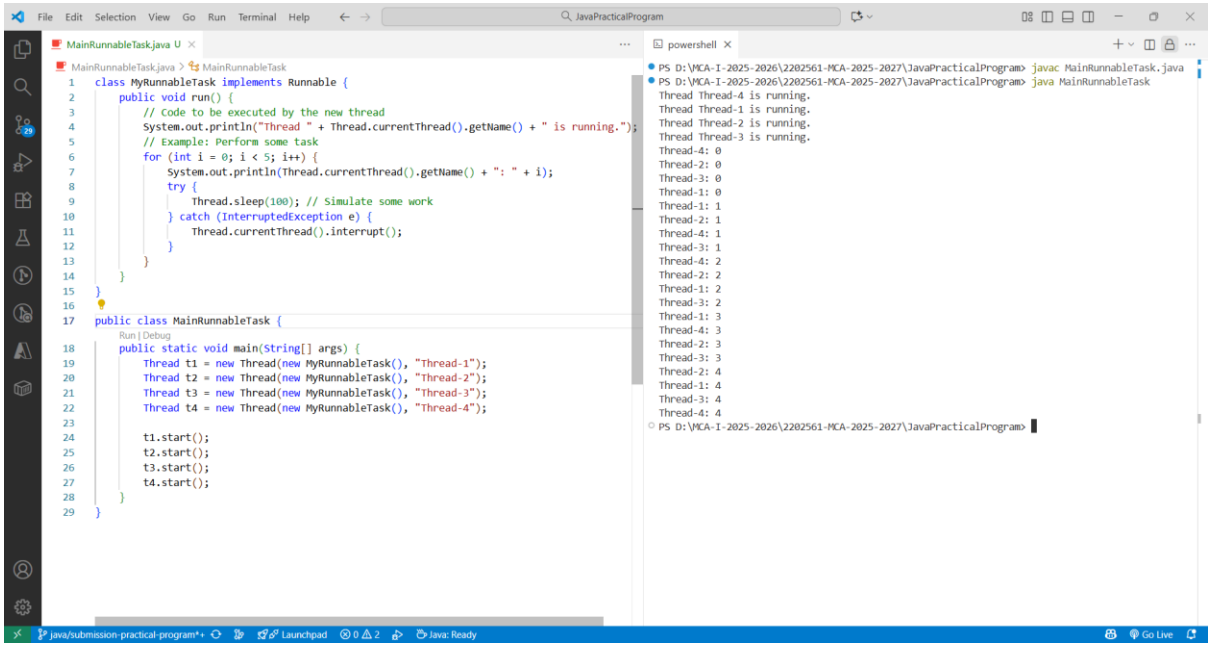
Experiment 6: Multithreading
Implement a Thread - 1. Extending the Thread class



```
class CookingTask extends Thread {
    private String task;

    CookingTask(String task) {
        this.task = task;
    }

    public void run() {
        System.out.println(task + " is being prepared by " +
                Thread.currentThread().getName());
    }
}

public class Restaurant {
    Run | Debug
    public static void main(String[] args) {
        Thread t1 = new CookingTask(task: "Pasta");
        Thread t2 = new CookingTask(task: "Salad");
        Thread t3 = new CookingTask(task: "Dessert");
        Thread t4 = new CookingTask(task: "Rice");
        t1.start();
        t2.start();
        t3.start();
        t4.start();
    }
}
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> javac Restaurant.java
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java Restaurant
Dessert is being prepared by Thread-2
Pasta is being prepared by Thread-0
Rice is being prepared by Thread-3
Salad is being prepared by Thread-1
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

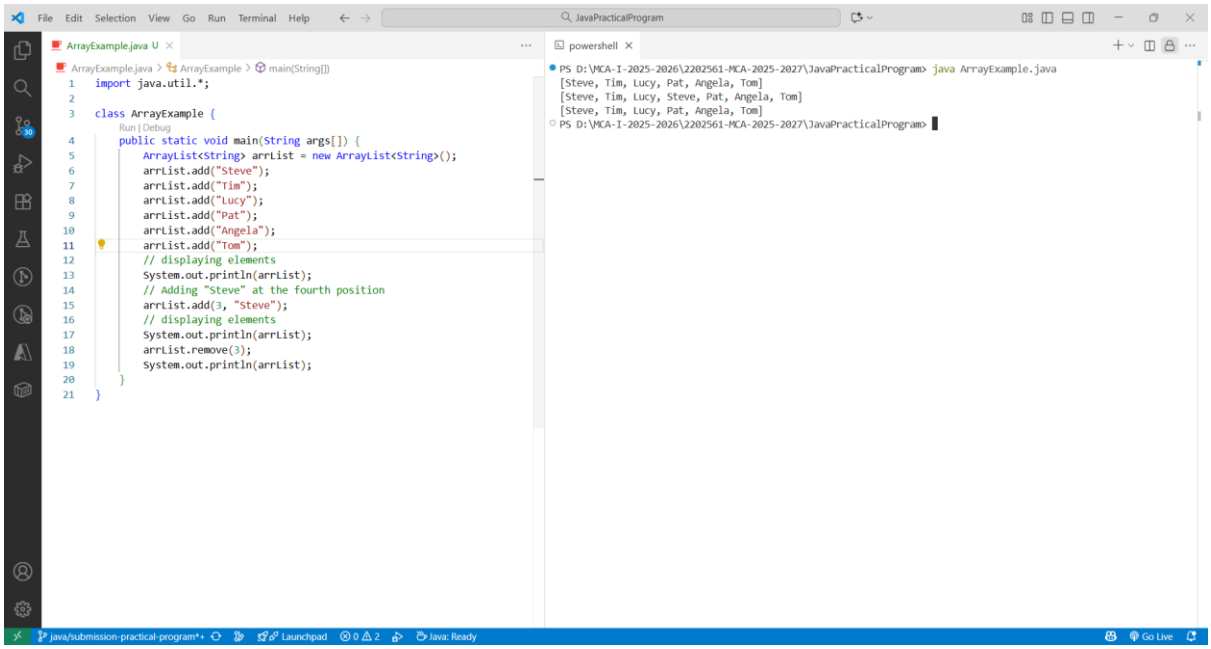
b. Implement Thread using the Runnable Interface



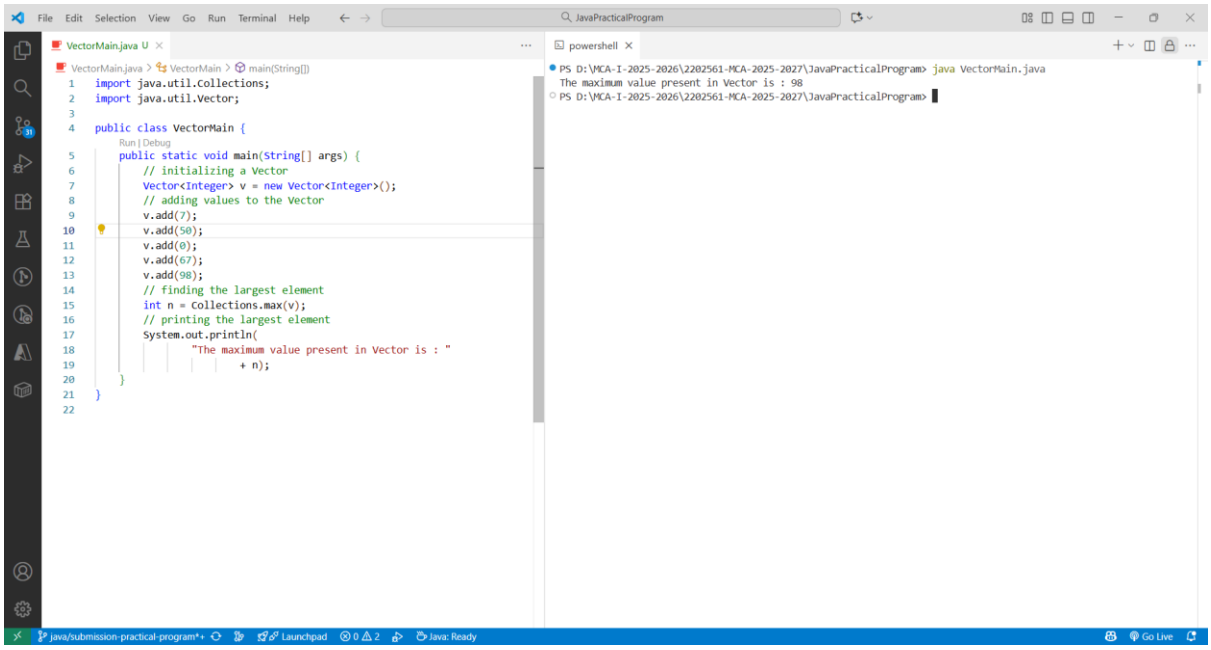
```
class MyRunnableTask implements Runnable {
    public void run() {
        // Code to be executed by the new thread
        System.out.println("Thread " + Thread.currentThread().getName() + " is running.");
        // Example: Perform some task
        for (int i = 0; i < 5; i++) {
            System.out.println(Thread.currentThread().getName() + ": " + i);
            try {
                Thread.sleep(100); // Simulate some work
            } catch (InterruptedException e) {
                Thread.currentThread().interrupt();
            }
        }
    }
}

public class MainRunnableTask {
    Run | Debug
    public static void main(String[] args) {
        Thread t1 = new Thread(new MyRunnableTask(), "Thread-1");
        Thread t2 = new Thread(new MyRunnableTask(), "Thread-2");
        Thread t3 = new Thread(new MyRunnableTask(), "Thread-3");
        Thread t4 = new Thread(new MyRunnableTask(), "Thread-4");

        t1.start();
        t2.start();
        t3.start();
        t4.start();
    }
}
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> javac MainRunnableTask.java
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java MainRunnableTask
Thread Thread-4 is running.
Thread Thread-1 is running.
Thread Thread-2 is running.
Thread Thread-3 is running.
Thread-4: 0
Thread-2: 0
Thread-3: 0
Thread-1: 0
Thread-1: 1
Thread-2: 1
Thread-4: 1
Thread-3: 1
Thread-4: 2
Thread-2: 2
Thread-1: 2
Thread-3: 2
Thread-1: 3
Thread-4: 3
Thread-2: 3
Thread-3: 3
Thread-2: 4
Thread-1: 4
Thread-3: 4
Thread-4: 4
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

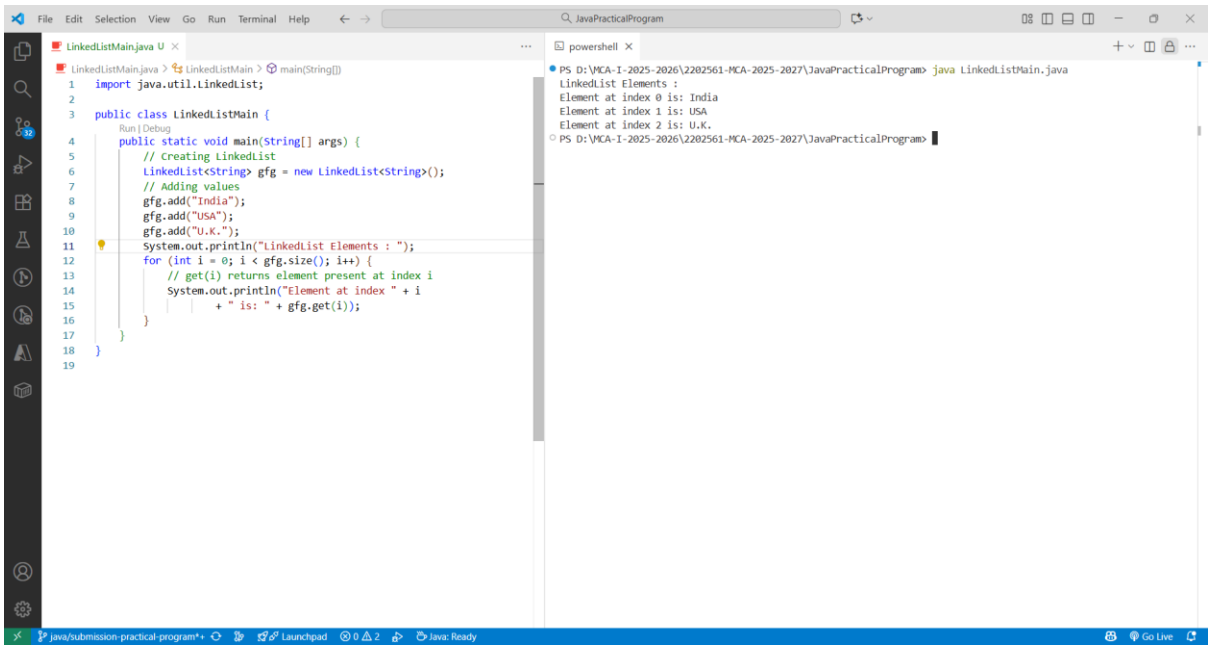
Experiment 7: Collections
Add and remove element using ArrayList



b. To find maximum element vector using predefined method



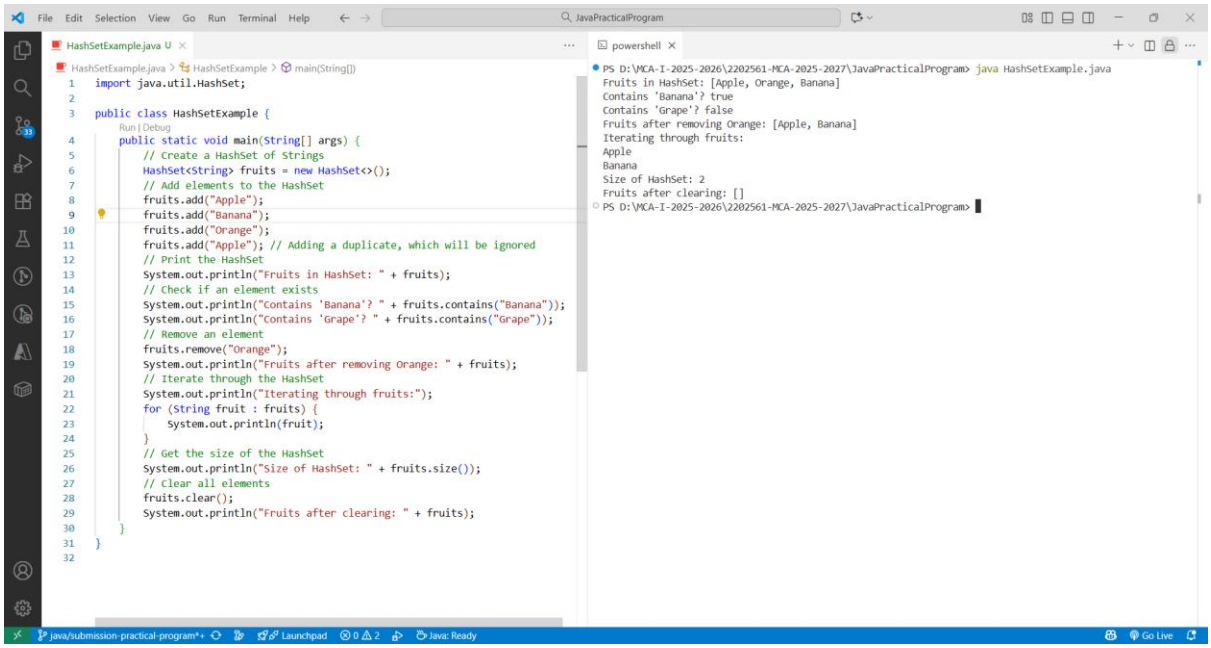
c. to get the elements of Linkedlist



```
import java.util.LinkedList;

public class LinkedListMain {
    Run | Debug
    public static void main(String[] args) {
        // Creating LinkedList
        LinkedList<String> gfg = new LinkedList<String>();
        // Adding values
        gfg.add("India");
        gfg.add("USA");
        gfg.add("U.K.");
        System.out.println("LinkedList Elements : ");
        for (int i = 0; i < gfg.size(); i++) {
            // get(i) returns element present at index i
            System.out.println("Element at index " + i
                    + " is: " + gfg.get(i));
        }
    }
}
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java LinkedListMain.java
LinkedList Elements :
Element at index 0 is: India
Element at index 1 is: USA
Element at index 2 is: U.K.
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

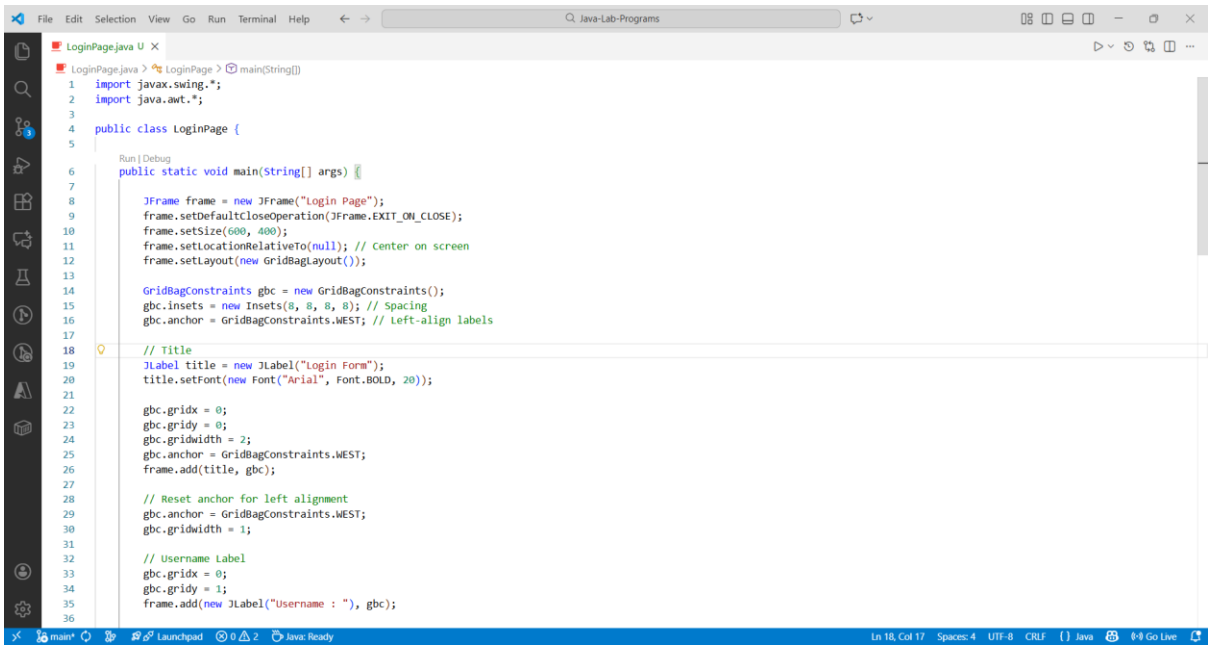
d. to get the elements of HashSet



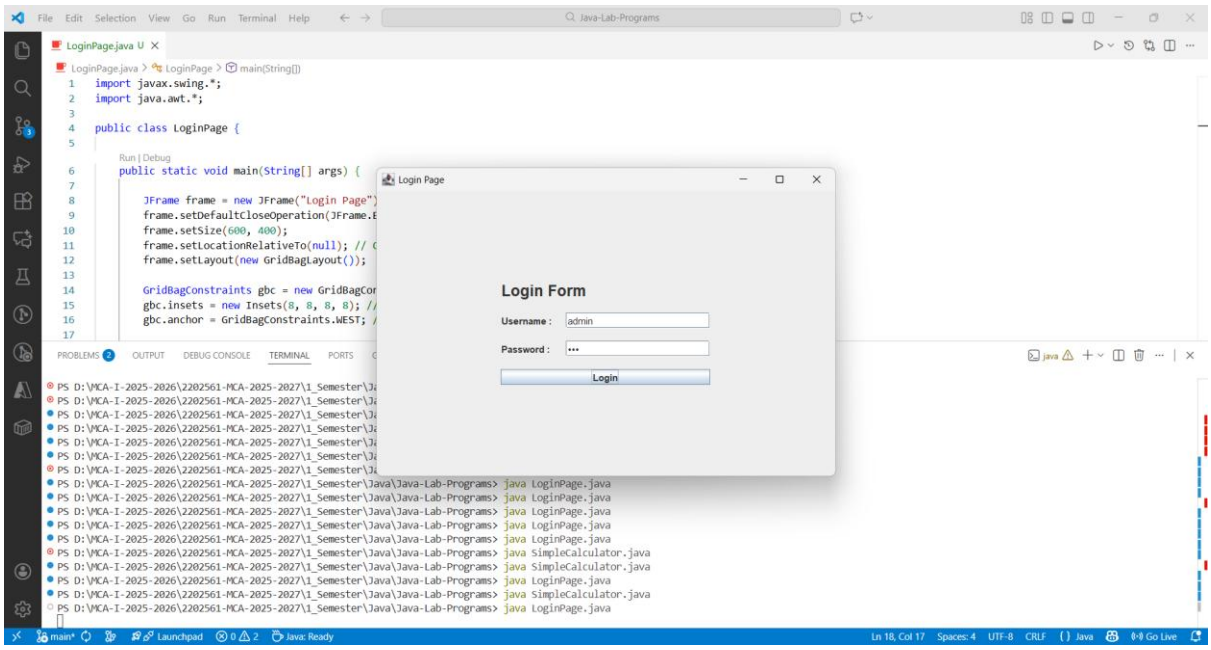
```
import java.util.HashSet;

public class HashSetExample {
    Run | Debug
    public static void main(String[] args) {
        // Create a HashSet of Strings
        HashSet<String> fruits = new HashSet<>();
        // Add elements to the HashSet
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Orange");
        fruits.add("Apple"); // Adding a duplicate, which will be ignored
        // Print the HashSet
        System.out.println("Fruits in HashSet: " + fruits);
        // Check if an element exists
        System.out.println("Contains 'Banana'? " + fruits.contains("Banana"));
        System.out.println("Contains 'Grape'? " + fruits.contains("Grape"));
        // Remove an element
        fruits.remove("Orange");
        System.out.println("Fruits after removing Orange: " + fruits);
        // Iterate through the HashSet
        System.out.println("Iterating through fruits:");
        for (String fruit : fruits) {
            System.out.println(fruit);
        }
        // Get the size of the HashSet
        System.out.println("Size of HashSet: " + fruits.size());
        // Clear all elements
        fruits.clear();
        System.out.println("Fruits after clearing: " + fruits);
    }
}
```

```
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram> java HashSetExample.java
Fruits in HashSet: [Apple, Orange, Banana]
Contains 'Banana'? true
Contains 'Grape'? false
Fruits after removing Orange: [Apple, Banana]
Iterating through fruits:
Apple
Banana
Size of HashSet: 2
Fruits after clearing: []
PS D:\MCA-I-2025-2026\2202561-MCA-2025-2027\JavaPracticalProgram>
```

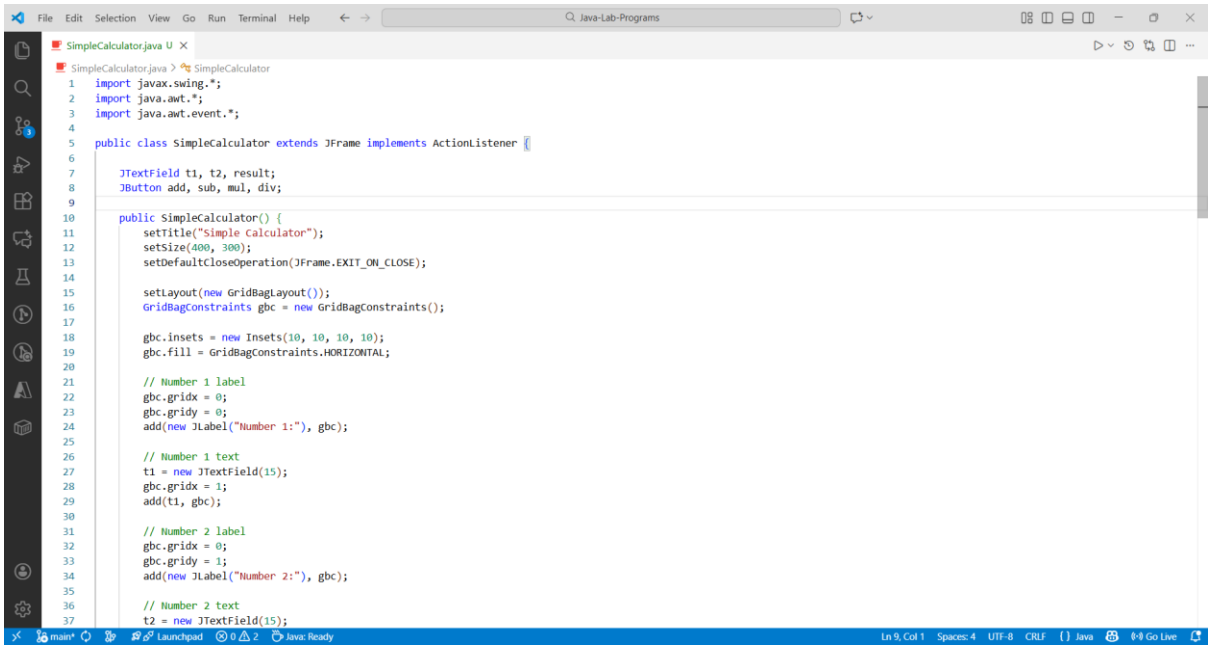
Login form using SWING



```java
import javax.swing.*;
import java.awt.*;

public class LoginPage {

    Run | Debug
    public static void main(String[] args) {

        JFrame frame = new JFrame("Login Page");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(600, 400);
        frame.setLocationRelativeTo(null); // Center on screen
        frame.setLayout(new GridBagLayout());

        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets(8, 8, 8, 8); // Spacing
        gbc.anchor = GridBagConstraints.WEST; // Left-align labels

        // Title
        JLabel title = new JLabel("Login Form");
        title.setFont(new Font("Arial", Font.BOLD, 20));

        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.gridwidth = 2;
        gbc.anchor = GridBagConstraints.WEST;
        frame.add(title, gbc);

        // Reset anchor for left alignment
        gbc.anchor = GridBagConstraints.WEST;
        gbc.gridwidth = 1;

        // Username Label
        gbc.gridx = 0;
        gbc.gridy = 1;
        frame.add(new JLabel("Username : "), gbc);
```



Simple calculator using SWING



```java
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class SimpleCalculator extends JFrame implements ActionListener {

    JTextField t1, t2, result;
    JButton add, sub, mul, div;

    public SimpleCalculator() {
        setTitle("Simple Calculator");
        setSize(400, 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();

        gbc.insets = new Insets(10, 10, 10, 10);
        gbc.fill = GridBagConstraints.HORIZONTAL;

        // Number 1 label
        gbc.gridx = 0;
        gbc.gridy = 0;
        add(new JLabel("Number 1:"), gbc);

        // Number 1 text
        t1 = new JTextField(15);
        gbc.gridx = 1;
        add(t1, gbc);

        // Number 2 label
        gbc.gridx = 0;
        gbc.gridy = 1;
        add(new JLabel("Number 2:"), gbc);

        // Number 2 text
        t2 = new JTextField(15);
```