



UNIVERSITY OF SCIENCE AND TECHNOLOGY CHITTAGONG

Faculty of Science Engineering & Technology (FSET)

Department of Computer Science & Engineering (CSE)

LAB TASK - 04

Submitted to

Debabarata Mallick
Lecturer
CSE, FSET, USTC

Submitted by

Name : S.M. ALI AKBAR KHALED

Roll: 24070103

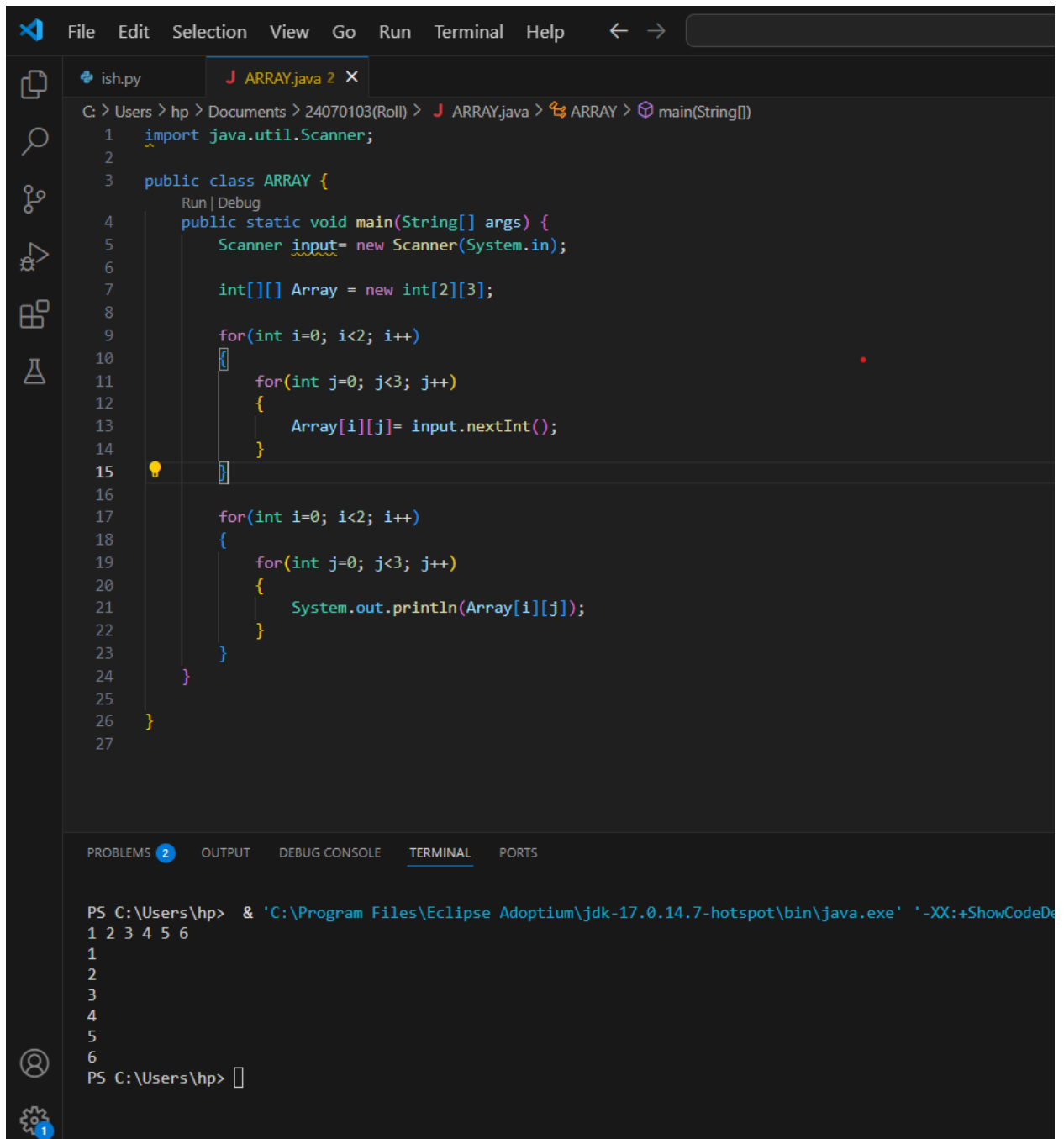
Section: "A"

Semester: 2nd

Date: 11.03.2025

Dept.: Computer Science & Engineering (CSE)

Question no-01: Write a Java program to print all elements of a given 2D array.

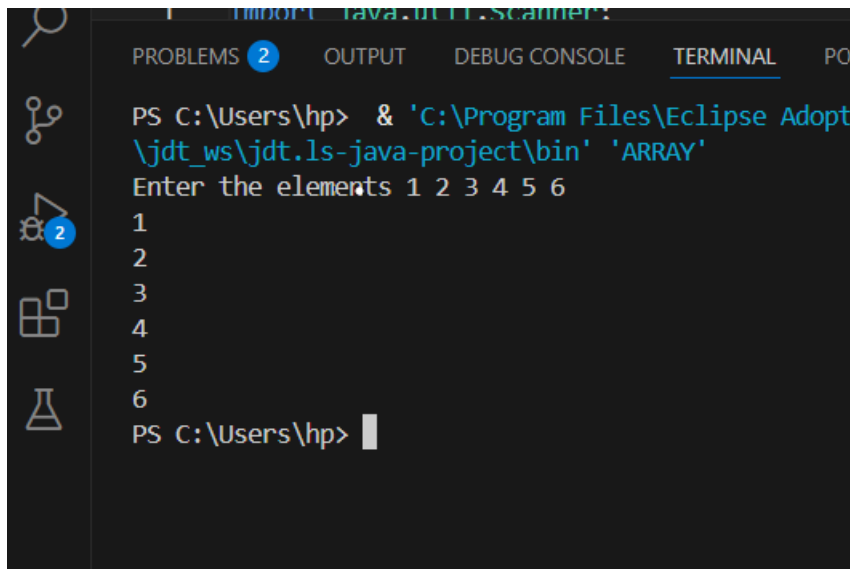


The screenshot shows the Eclipse IDE with a Java project named 'ARRAY'. The main class is 'ARRAY.java'. The code defines a 2D array of size 2x3 and prints its elements. The terminal output shows the execution of the program, which prints the numbers 1 through 6 in a 2x3 grid.

```
File Edit Selection View Go Run Terminal Help
ish.py ARRAY.java 2 X
C:\Users\hp\Documents\24070103(Roll) > J ARRAY.java > ARRAY > main(String[])
1 import java.util.Scanner;
2
3 public class ARRAY {
4     public static void main(String[] args) {
5         Scanner input = new Scanner(System.in);
6
7         int[][] Array = new int[2][3];
8
9         for(int i=0; i<2; i++)
10             for(int j=0; j<3; j++)
11             {
12                 Array[i][j] = input.nextInt();
13             }
14
15         for(int i=0; i<2; i++)
16         {
17             for(int j=0; j<3; j++)
18             {
19                 System.out.println(Array[i][j]);
20             }
21         }
22     }
23 }
24
25
26
27
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.14.7-hotspot\bin\java.exe' '-XX:+ShowCodeD
1 2 3 4 5 6
1
2
3
4
5
6
PS C:\Users\hp>
```



```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adopt
\jdt_ws\jdt.ls-java-project\bin' 'ARRAY'
Enter the elements 1 2 3 4 5 6
1
2
3
4
5
6
PS C:\Users\hp> |
```

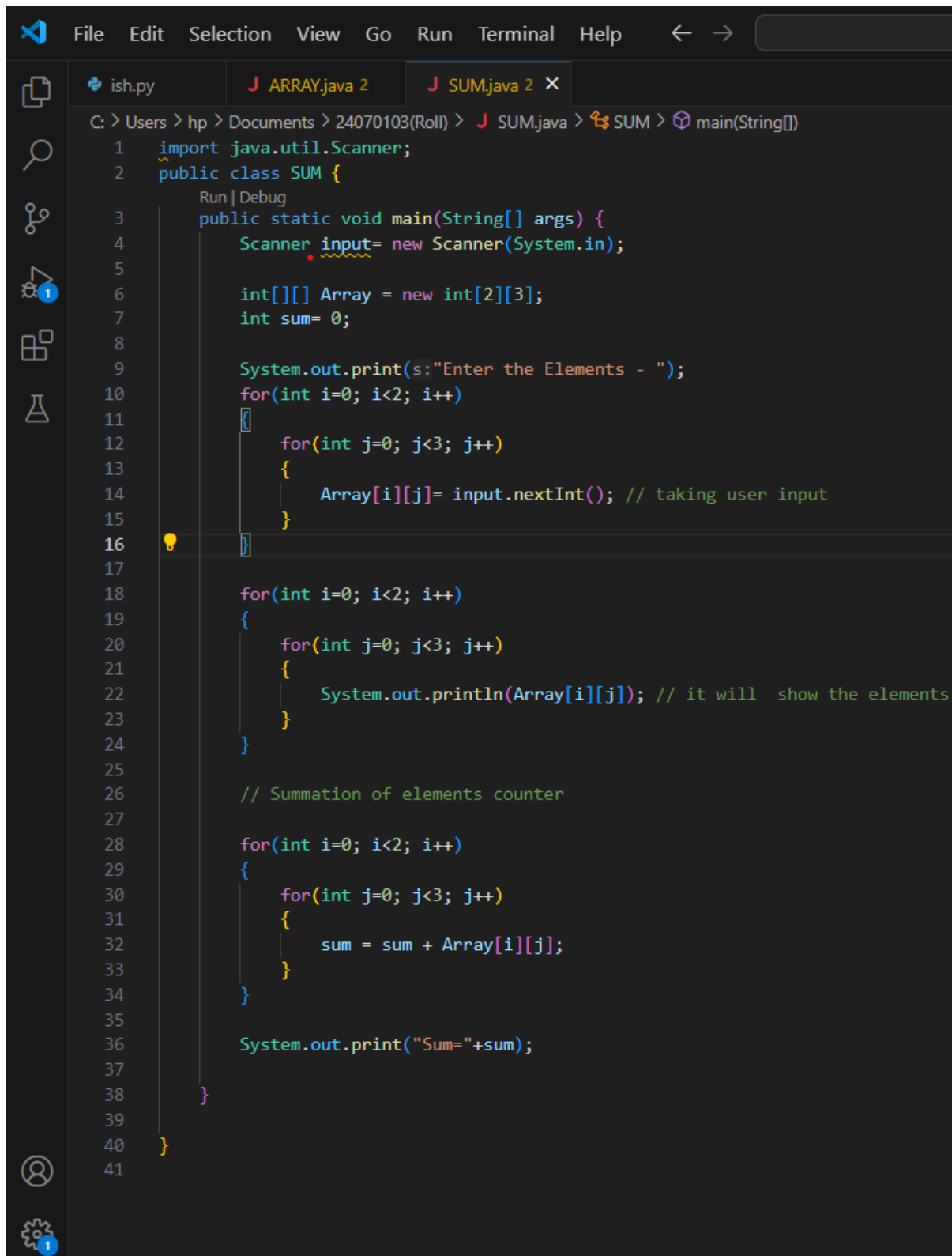
Explanation:

This Java program creates a 2D array of size 2x3, accepts user input to fill the array, and then prints its elements.

Key Steps:

1. **Array Initialization:** A 2D array of integers `Array[2][3]` is created.
2. **User Input:** Nested loops prompt the user to enter values to fill the array.
3. **Printing Array:** Another set of nested loops prints the elements of the array.

Question no.- 02: Write a Java program to calculate the sum of all elements in a 2D array.



The screenshot shows an IDE with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The tab bar shows three tabs: ish.py, ARRAY.java 2, and SUM.java 2. The main editor window displays the following Java code:

```
C: > Users > hp > Documents > 24070103(Roll) > J SUM.java > SUM > main(String[])
1  import java.util.Scanner;
2  public class SUM {
3      Run | Debug
4      public static void main(String[] args) {
5          Scanner input= new Scanner(System.in);
6
7          int[][] Array = new int[2][3];
8          int sum= 0;
9
10         System.out.print(s:"Enter the Elements - ");
11         for(int i=0; i<2; i++)
12         {
13             for(int j=0; j<3; j++)
14             {
15                 Array[i][j]= input.nextInt(); // taking user input
16             }
17
18             for(int i=0; i<2; i++)
19             {
20                 for(int j=0; j<3; j++)
21                 {
22                     System.out.println(Array[i][j]); // it will show the elements
23                 }
24             }
25
26             // Summation of elements counter
27
28             for(int i=0; i<2; i++)
29             {
30                 for(int j=0; j<3; j++)
31                 {
32                     sum = sum + Array[i][j];
33                 }
34             }
35
36             System.out.print("Sum="+sum);
37         }
38     }
39 }
40
41
```

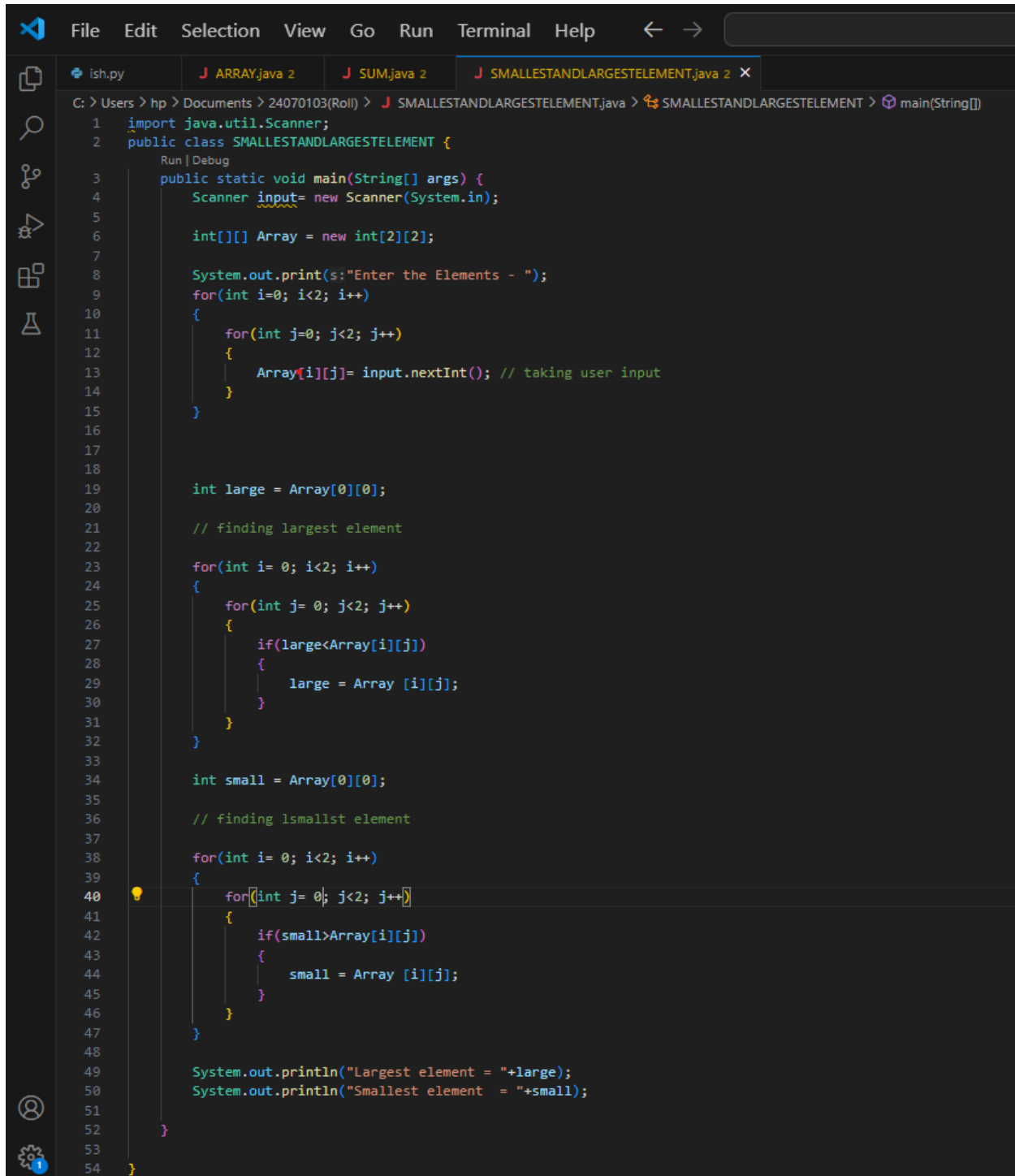
```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adoptium\j
\jdt_ws\jdt.ls-java-project\bin' 'SUM'
Enter the Elements - 1 2 3 4 5 6
1
2
3
4
5
6
Sum=21
PS C:\Users\hp> █
```

Explanation:

This Java program performs the following tasks:

1. **Creates a 2D Array:** A 2D array `Array` of size 2×3 is created to store integers.
2. **Takes User Input:** The program asks the user to input 6 elements (to fill the 2D array).
3. **Displays Array Elements:** The program then prints the elements of the array.
4. **Calculates Sum:** It computes the sum of all the elements in the 2D array.
5. **Displays Sum:** Finally, the program outputs the calculated sum.

Question 03: Write a Java program to find the largest and smallest elements in a 2D array.

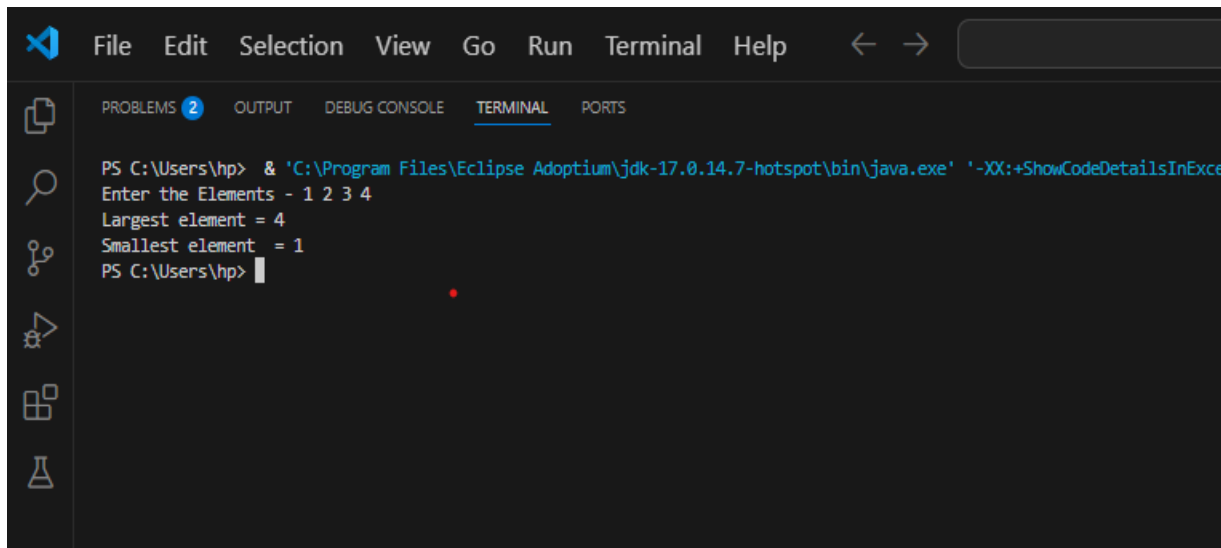


The screenshot shows an IDE with a Java file named `SMALLESTANDLARGESTELEMENT.java`. The code is as follows:

```
1  import java.util.Scanner;
2  public class SMALLESTANDLARGESTELEMENT {
3      public static void main(String[] args) {
4          Scanner input= new Scanner(System.in);
5
6          int[][] Array = new int[2][2];
7
8          System.out.print(s:"Enter the Elements - ");
9          for(int i=0; i<2; i++)
10             {
11                 for(int j=0; j<2; j++)
12                 {
13                     Array[i][j]= input.nextInt(); // taking user input
14                 }
15             }
16
17
18             int large = Array[0][0];
19
20             // finding largest element
21
22             for(int i= 0; i<2; i++)
23             {
24                 for(int j= 0; j<2; j++)
25                 {
26                     if(large<Array[i][j])
27                     {
28                         large = Array [i][j];
29                     }
30                 }
31             }
32
33             int small = Array[0][0];
34
35             // finding lsmallst element
36
37             for(int i= 0; i<2; i++)
38             {
39                 for(int j= 0; j<2; j++)
40                 {
41                     if(small>Array[i][j])
42                     {
43                         small = Array [i][j];
44                     }
45                 }
46             }
47
48             System.out.println("Largest element = "+large);
49             System.out.println("Smallest element = "+small);
50
51         }
52     }
53 }
54 }
```

The IDE interface includes a menu bar (File, Edit, Selection, View, Go, Run, Terminal, Help), a toolbar with icons for file operations, and a sidebar with icons for project structure, search, and other tools. The file explorer on the left shows the project structure with files `ish.py`, `ARRAY.java 2`, `SUM.java 2`, and `SMALLESTANDLARGESTELEMENT.java 2`. The terminal at the bottom is empty.

Output:



```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.14.7-hotspot\bin\java.exe' '-XX:+ShowCodeDetailsInExceptionMessages'
Enter the Elements - 1 2 3 4
Largest element = 4
Smallest element = 1
PS C:\Users\hp>
```

Explanation:

This Java program identifies the **largest** and **smallest** elements in a 2D array (matrix). It follows these steps:

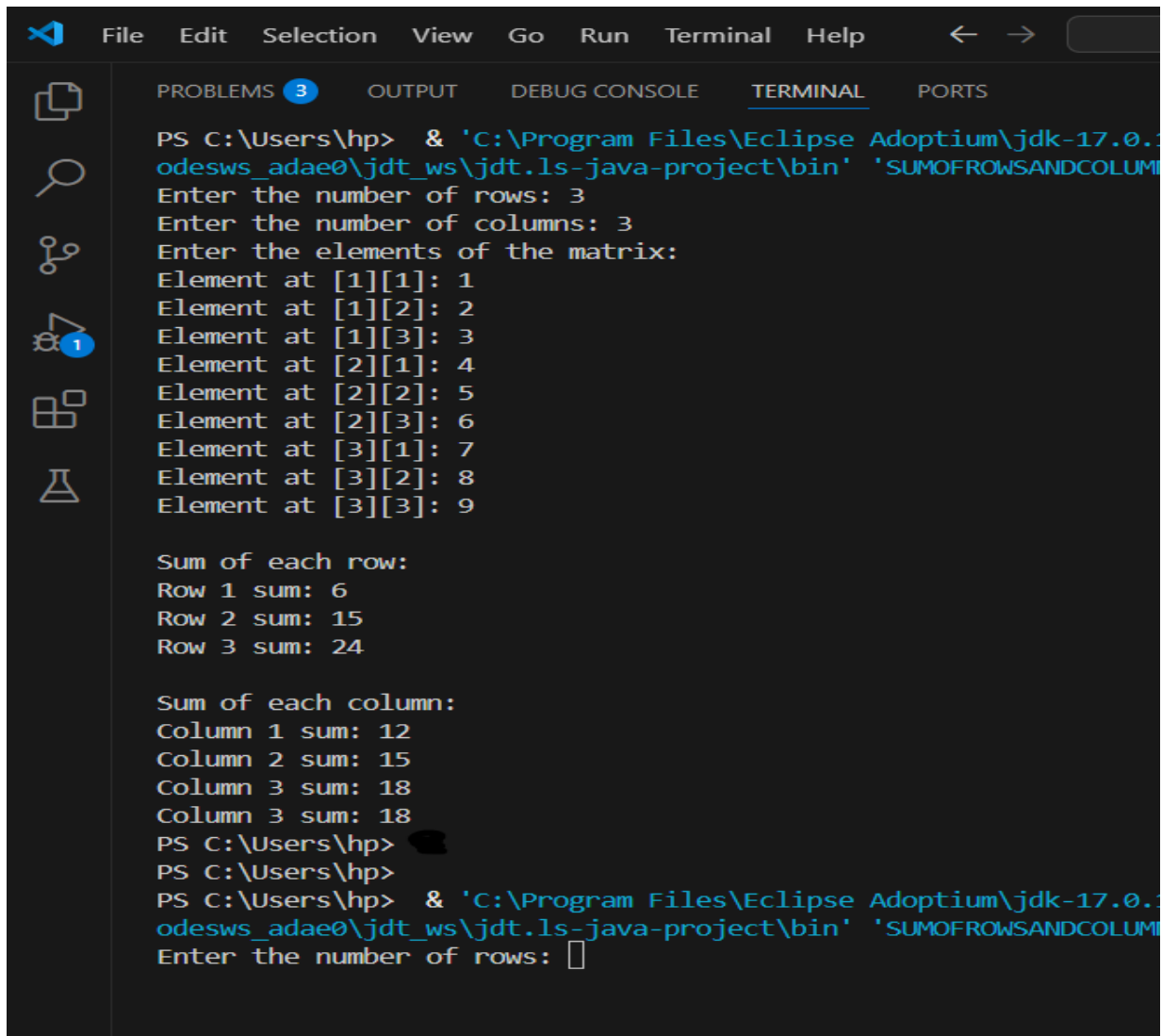
1. **Array Initialization:**
 - o A 2x2 array is created to store integer elements.
 - o User input is taken to populate the array.
2. **Finding the Largest Element:**
 - o The program iterates through the array and compares each element with the current largest value, updating the largest value if a larger element is found.
3. **Finding the Smallest Element:**
 - o Similarly, the program iterates through the array and compares each element with the current smallest value, updating the smallest value if a smaller element is found.
4. **Output:**
 - o The program prints the largest and smallest elements.

Question 4: Write a Java program to find the sum of each row and each column in a 2D array



```
File Edit Selection View Go Run Terminal Help
ish.py ARRAY.java SUM.java SMALLESTANDLARGESTELEMEN.java 2 SUMOFROWS
C: > Users > hp > Documents > 24070103(Roll) > SUMOFROWSANDCOLUMN.java > ...
1  import java.util.Scanner;
2
3  public class SUMOFROWSANDCOLUMN {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          // Prompt user for the number of rows and columns
8          System.out.print(s:"Enter the number of rows: ");
9          int rows = scanner.nextInt();
10         System.out.print(s:"Enter the number of columns: ");
11         int cols = scanner.nextInt();
12
13         // Initialize the matrix
14         int[][] matrix = new int[rows][cols];
15
16         // Input matrix elements
17         System.out.println(x:"Enter the elements of the matrix:");
18         for (int i = 0; i < rows; i++) {
19             for (int j = 0; j < cols; j++) {
20                 System.out.print("Element at [" + (i + 1) + "][" + (j + 1) + "]: ");
21                 matrix[i][j] = scanner.nextInt();
22             }
23         }
24
25         // Find the sum of each row
26         System.out.println(x:"\nSum of each row:");
27         for (int i = 0; i < rows; i++) {
28             int rowSum = 0;
29             for (int j = 0; j < cols; j++) {
30                 rowSum += matrix[i][j];
31             }
32             System.out.println("Row " + (i + 1) + " sum: " + rowSum);
33         }
34
35         // Find the sum of each column
36         System.out.println(x:"\nSum of each column:");
37         for (int j = 0; j < cols; j++) {
38             int colSum = 0;
39             for (int i = 0; i < rows; i++) {
40                 colSum += matrix[i][j];
41             }
42             System.out.println("Column " + (j + 1) + " sum: " + colSum);
43         }
44
45         // Close scanner
46         scanner.close();
47     }
48 }
49
```


Output:



```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.3\bin\java.exe' -Xms1g -Xmx1g -Djdt.ls-java-project\bin SUMOFROWSANDCOLUMNS.java
Enter the number of rows: 3
Enter the number of columns: 3
Enter the elements of the matrix:
Element at [1][1]: 1
Element at [1][2]: 2
Element at [1][3]: 3
Element at [2][1]: 4
Element at [2][2]: 5
Element at [2][3]: 6
Element at [3][1]: 7
Element at [3][2]: 8
Element at [3][3]: 9

Sum of each row:
Row 1 sum: 6
Row 2 sum: 15
Row 3 sum: 24

Sum of each column:
Column 1 sum: 12
Column 2 sum: 15
Column 3 sum: 18
Column 3 sum: 18
PS C:\Users\hp>
PS C:\Users\hp>
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.3\bin\java.exe' -Xms1g -Xmx1g -Djdt.ls-java-project\bin SUMOFROWSANDCOLUMNS.java
Enter the number of rows: 
```

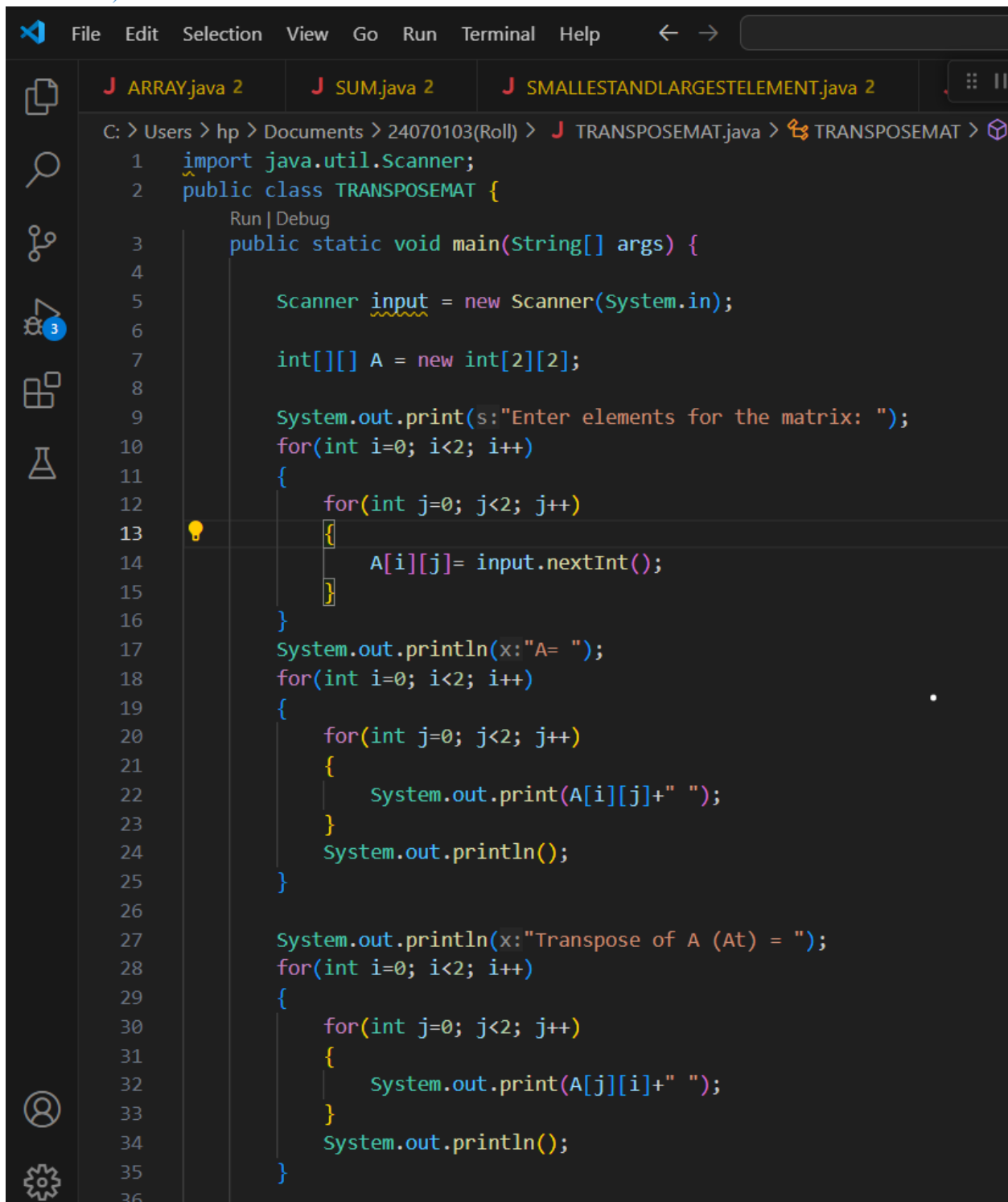
Explanation:

This Java program calculates the sum of each row and each column of a matrix.

Steps:

1. **Input Dimensions:** The program asks for the number of rows and columns.
2. **Input Matrix Elements:** It takes user input to fill the matrix.
3. **Row Sum:** It calculates and prints the sum of each row.
4. **Column Sum:** It calculates and prints the sum of each column.

Question 5: Write a Java program to find the transpose of a matrix (rows become columns and vice versa).



The screenshot shows an IDE with three tabs: ARRAY.java 2, SUM.java 2, and SMALLESTANDLARGESTELEMENT.java 2. The active tab is TRANSPOSEMAT.java, which contains the following Java code:

```
1  import java.util.Scanner;
2  public class TRANSPOSEMAT {
3      public static void main(String[] args) {
4
5          Scanner input = new Scanner(System.in);
6
7          int[][] A = new int[2][2];
8
9          System.out.print(s:"Enter elements for the matrix: ");
10         for(int i=0; i<2; i++)
11         {
12             for(int j=0; j<2; j++)
13             {
14                 A[i][j]= input.nextInt();
15             }
16         }
17         System.out.println(x:"A= ");
18         for(int i=0; i<2; i++)
19         {
20             for(int j=0; j<2; j++)
21             {
22                 System.out.print(A[i][j]+" ");
23             }
24             System.out.println();
25         }
26
27         System.out.println(x:"Transpose of A (At) = ");
28         for(int i=0; i<2; i++)
29         {
30             for(int j=0; j<2; j++)
31             {
32                 System.out.print(A[j][i]+" ");
33             }
34             System.out.println();
35         }
36     }
```

```
\jdk_ws\jdk-1.8-java-project\bin TRANSPOSEMAT
Enter elements for the matrix: 1 2 3 4
A=
1 2
3 4
Transpose of A (At) =
1 3
2 4
```

Explanation:

- **Matrix Initialization:**

- A 2×2 matrix `A` is created to store integer elements.

- **User Input:**

- The program prompts the user to input the elements of the matrix.

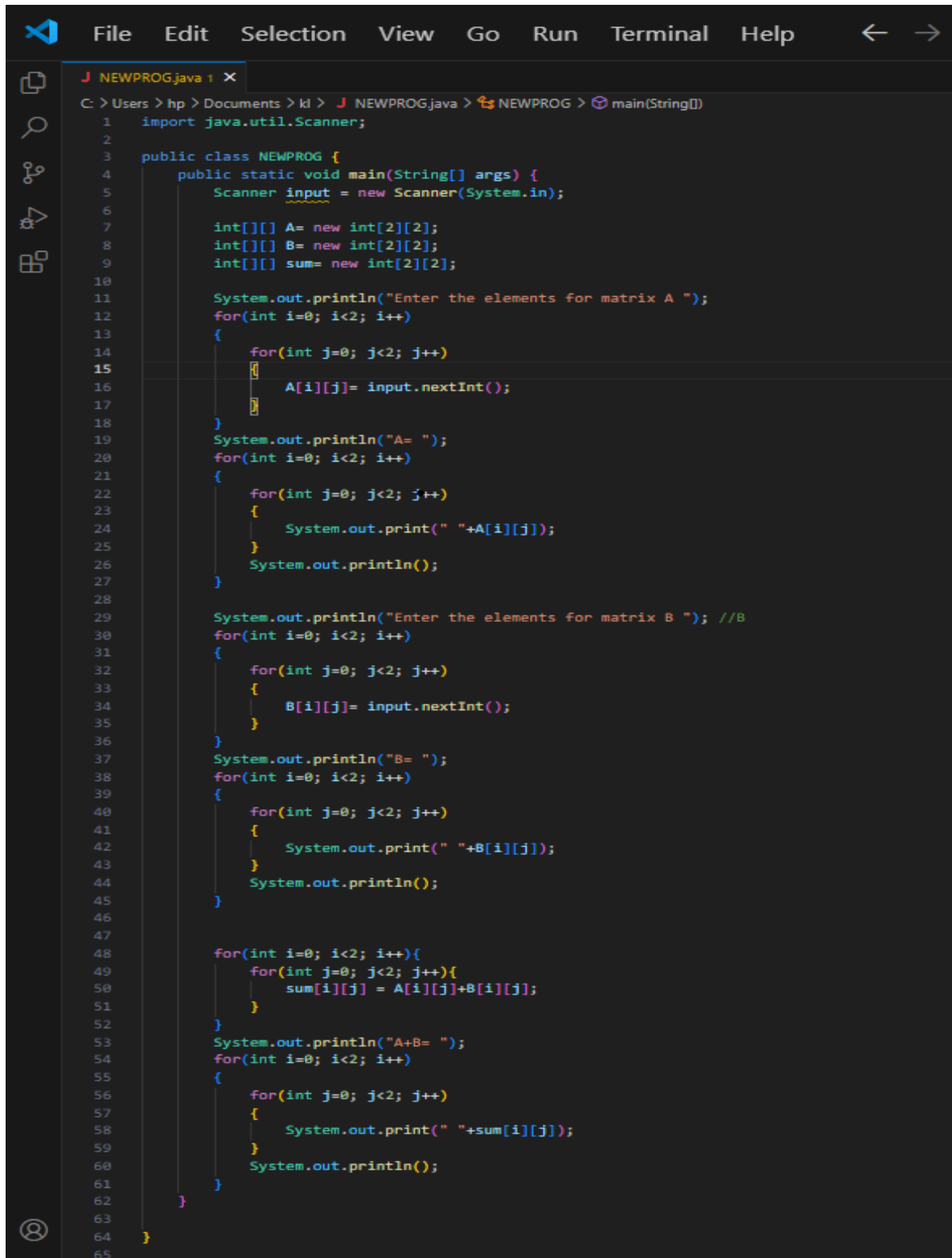
- **Display the Original Matrix:**

- The program prints the original matrix `A`.

- **Transpose Calculation:**

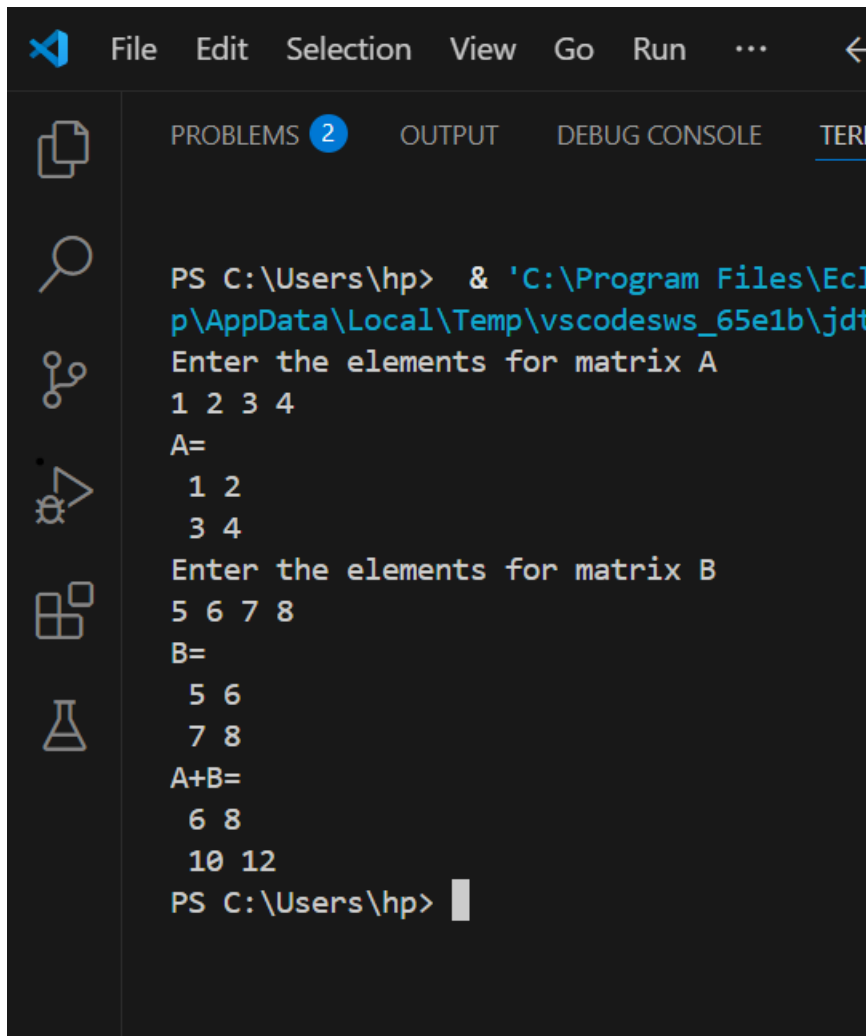
- The program calculates and prints the transpose of the matrix. In the transpose, the element at position `A[i][j]` in the original matrix is moved to `A[j][i]` in the transposed matrix.

Question 06: Write a Java program to add two matrices and store the result in another 2D array.



```
File Edit Selection View Go Run Terminal Help
J NEWPROG.java x
C:\Users\hp>Documents>kl>J NEWPROG.java>NEWPROG>main(String[])
1  import java.util.Scanner;
2
3  public class NEWPROG {
4      public static void main(String[] args) {
5          Scanner input = new Scanner(System.in);
6
7          int[][] A= new int[2][2];
8          int[][] B= new int[2][2];
9          int[][] sum= new int[2][2];
10
11         System.out.println("Enter the elements for matrix A ");
12         for(int i=0; i<2; i++)
13         {
14             for(int j=0; j<2; j++)
15             {
16                 A[i][j]= input.nextInt();
17             }
18         }
19         System.out.println("A= ");
20         for(int i=0; i<2; i++)
21         {
22             for(int j=0; j<2; j++)
23             {
24                 System.out.print(" "+A[i][j]);
25             }
26             System.out.println();
27         }
28
29         System.out.println("Enter the elements for matrix B "); //B
30         for(int i=0; i<2; i++)
31         {
32             for(int j=0; j<2; j++)
33             {
34                 B[i][j]= input.nextInt();
35             }
36         }
37         System.out.println("B= ");
38         for(int i=0; i<2; i++)
39         {
40             for(int j=0; j<2; j++)
41             {
42                 System.out.print(" "+B[i][j]);
43             }
44             System.out.println();
45         }
46
47         for(int i=0; i<2; i++){
48             for(int j=0; j<2; j++){
49                 sum[i][j] = A[i][j]+B[i][j];
50             }
51         }
52         System.out.println("A+B= ");
53         for(int i=0; i<2; i++)
54         {
55             for(int j=0; j<2; j++)
56             {
57                 System.out.print(" "+sum[i][j]);
58             }
59             System.out.println();
60         }
61     }
62 }
63
64 }
65
```

Output:



```
PS C:\Users\hp> & 'C:\Program Files\Ecl  
p\AppData\Local\Temp\vscodesws_65e1b\jdt  
Enter the elements for matrix A  
1 2 3 4  
A=  
1 2  
3 4  
Enter the elements for matrix B  
5 6 7 8  
B=  
5 6  
7 8  
A+B=  
6 8  
10 12  
PS C:\Users\hp> 
```

Explanation:

1. Matrix Declarations:

- The program creates three 2x2 matrices: A, B, and sum. A and B are input matrices, and sum will hold the result of adding A and B.

2. Input for Matrix A:

- The program prompts the user to input the elements of matrix A.
- Using nested loops, it reads 2 rows and 2 columns of data and stores them in the matrix A.

3. Display Matrix A:

- After taking input for A, the program displays the elements of matrix A to the console.

4. Input for Matrix B:

- The program then asks the user to input the elements of matrix B.
- Similarly, it reads 2 rows and 2 columns of data and stores them in the matrix B.

5. Display Matrix B:

- Once matrix B is input, the program displays the elements of matrix B to the console.

6. Matrix Addition:

- The program calculates the sum of matrices A and B element by element.
- For each corresponding element in A and B, it adds them and stores the result in the sum matrix.

7. Display Matrix A + B:

- Finally, the program prints the resulting matrix sum (which is the result of adding A and B), showing the user the output.

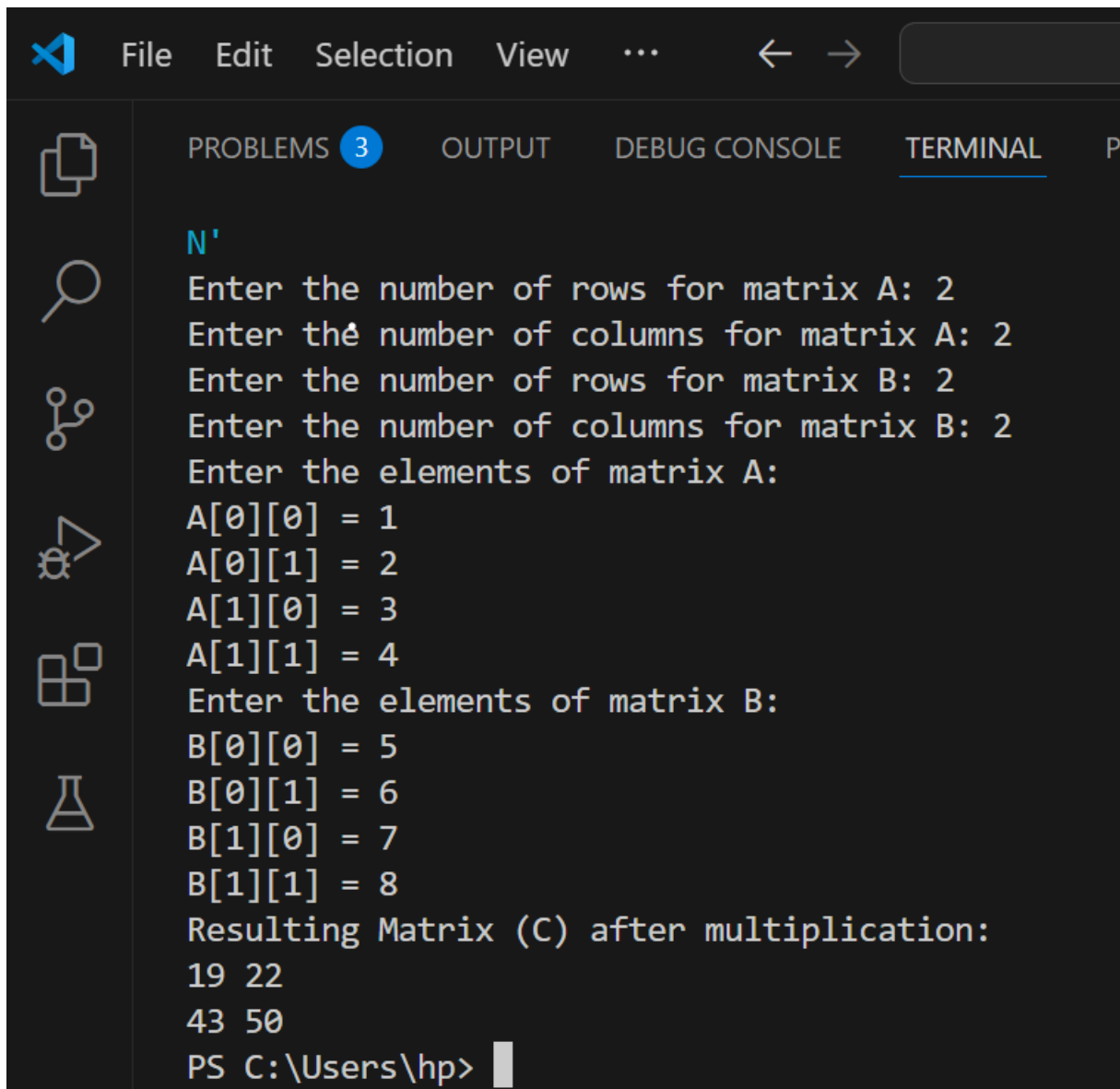
In essence, this program performs matrix addition and displays all three matrices: A, B, and the result of $A + B$.

Question 7: Write a Java program to multiply two matrices and store the result in another 2D array.



The screenshot shows an IDE with three tabs: TRANSPOSEMAT.java, SSYMMETRIC.java 1, and MATRIXMULTIPLICATION.java 2. The active tab is MATRIXMULTIPLICATION.java 2, which contains the following Java code:

```
1  import java.util.Scanner;
2
3  public class MATRIXMULTIPLICATION {
4      Run | Debug
5      public static void main(String[] args) {
6          Scanner scanner = new Scanner(System.in);
7
8          // Input the dimensions of matrix A
9          System.out.print(s:"Enter the number of rows for matrix A: ");
10         int rowsA = scanner.nextInt();
11         System.out.print(s:"Enter the number of columns for matrix A: ");
12         int colsA = scanner.nextInt();
13
14         // Input the dimensions of matrix B
15         System.out.print(s:"Enter the number of rows for matrix B: ");
16         int rowsB = scanner.nextInt();
17         System.out.print(s:"Enter the number of columns for matrix B: ");
18         int colsB = scanner.nextInt();
19
20         // Check if multiplication is possible
21         if (colsA != rowsB) {
22             System.out.println(x:"Matrix multiplication is not possible. Number of columns of A must be equal to number of rows of B.");
23             return;
24         }
25
26         // Initialize the matrices A, B and C (result matrix)
27         int[][] A = new int[rowsA][colsA];
28         int[][] B = new int[rowsB][colsB];
29         int[][] C = new int[rowsA][colsB];
30
31         // Input values for matrix A
32         System.out.println(x:"Enter the elements of matrix A:");
33         for (int i = 0; i < rowsA; i++) {
34             for (int j = 0; j < colsA; j++) {
35                 System.out.print("A[" + i + "][" + j + "] = ");
36                 A[i][j] = scanner.nextInt();
37             }
38         }
39
40         // Input values for matrix B
41         System.out.println(x:"Enter the elements of matrix B:");
42         for (int i = 0; i < rowsB; i++) {
43             for (int j = 0; j < colsB; j++) {
44                 System.out.print("B[" + i + "][" + j + "] = ");
45                 B[i][j] = scanner.nextInt();
46             }
47         }
48
49         // Perform matrix multiplication
50         for (int i = 0; i < rowsA; i++) {
51             for (int j = 0; j < colsB; j++) {
52                 C[i][j] = 0; // Initialize the result element
53                 for (int k = 0; k < colsA; k++) {
54                     C[i][j] += A[i][k] * B[k][j];
55                 }
56             }
57         }
58
59         // Output the result matrix C
60         System.out.println(x:"Resulting Matrix (C) after multiplication:");
61         for (int i = 0; i < rowsA; i++) {
62             for (int j = 0; j < colsB; j++) {
63                 System.out.print(C[i][j] + " ");
64             }
65             System.out.println();
66         }
67     }
68 }
```



The image shows a screenshot of the Visual Studio Code editor. The top menu bar includes 'File', 'Edit', 'Selection', 'View', and a search icon. Below the menu bar, the 'TERMINAL' tab is active, displaying the output of a C++ program. The program prompts the user to enter the number of rows and columns for two matrices, A and B, and then the elements of each matrix. The resulting matrix C is displayed after multiplication.

```
N'  
Enter the number of rows for matrix A: 2  
Enter the number of columns for matrix A: 2  
Enter the number of rows for matrix B: 2  
Enter the number of columns for matrix B: 2  
Enter the elements of matrix A:  
A[0][0] = 1  
A[0][1] = 2  
A[1][0] = 3  
A[1][1] = 4  
Enter the elements of matrix B:  
B[0][0] = 5  
B[0][1] = 6  
B[1][0] = 7  
B[1][1] = 8  
Resulting Matrix (C) after multiplication:  
19 22  
43 50  
PS C:\Users\hp>
```


Explanation:

Here's a brief explanation of the `MATRIXMULTIPLICATION` program:

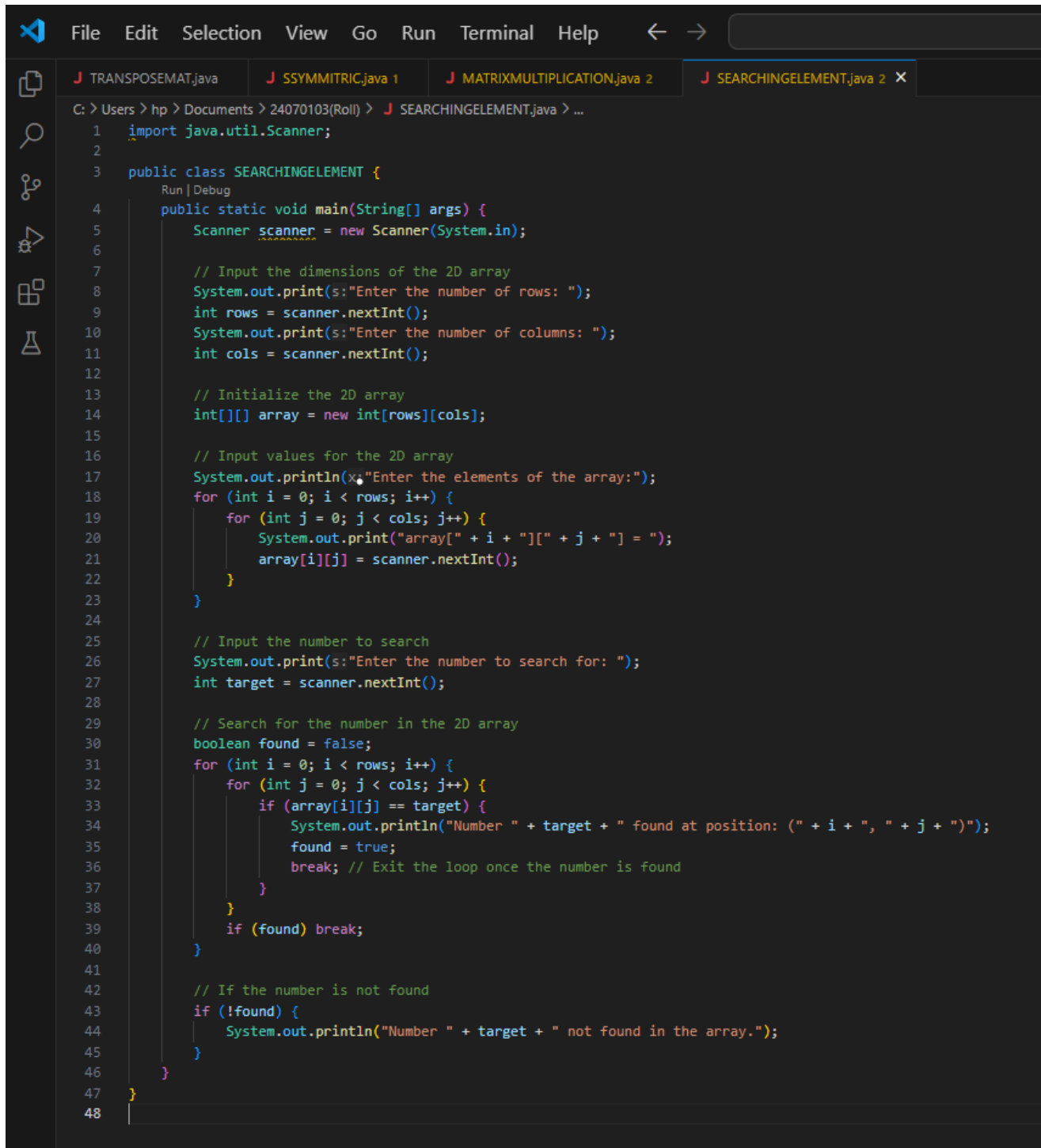
1. **User Input:**
 - The program asks the user to input the dimensions of two matrices A and B, then the elements for both matrices.
2. **Matrix Multiplication Check:**
 - It checks if matrix multiplication is possible (i.e., the number of columns of matrix A must be equal to the number of rows of matrix B).
3. **Matrix Multiplication:**
 - Using nested loops, the program performs matrix multiplication. Each element of the result matrix C is calculated by multiplying corresponding elements from A and B and summing them up.
4. **Result Output:**
 - After multiplication, the result matrix C is printed.

Example:

For matrices A (2x3) and B (3x2), the program calculates the resulting matrix C (2x2) and displays it.

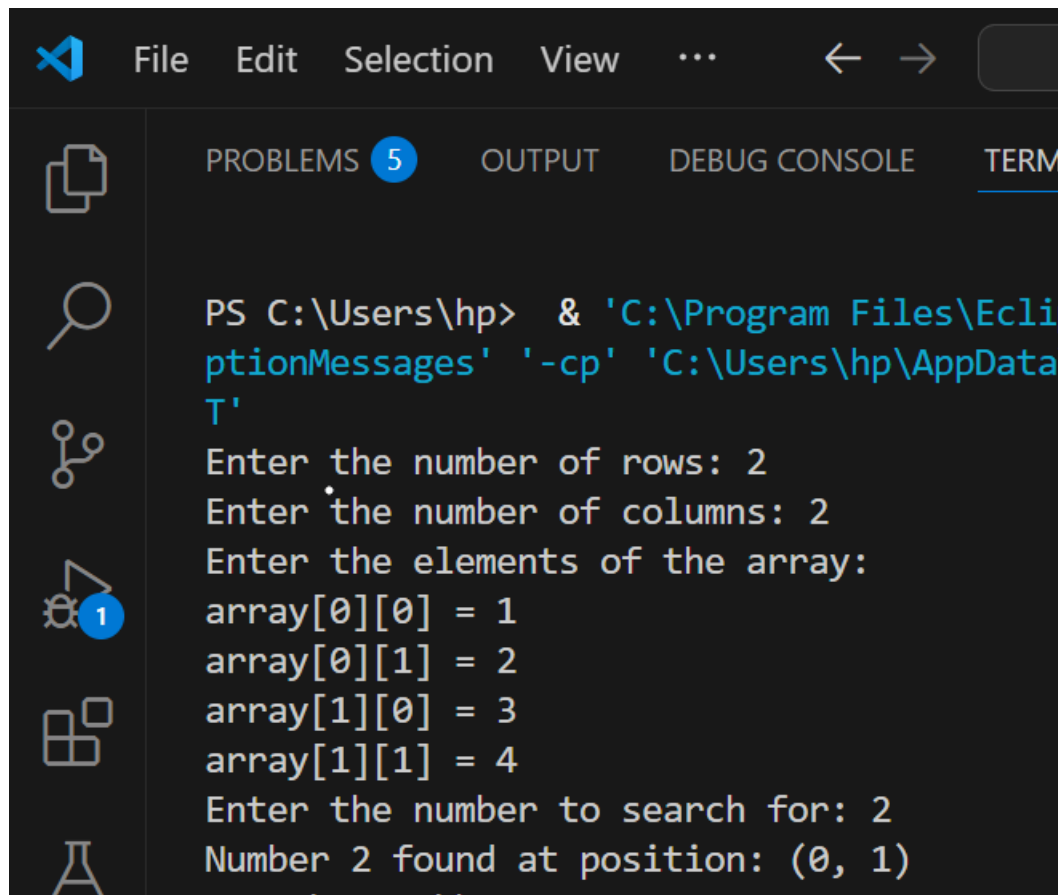
The class `MATRIXMULTIPLICATION` performs matrix multiplication and prints the result based on user inputs.

Question 8: Write a Java program to search for a given number in a 2D array and print its position.



The screenshot shows an IDE with the following tabs: TRANSPOSEMAT.java, SSYMMETRIC.java 1, MATRIXMULTIPLICATION.java 2, and SEARCHINGELEMENT.java 2. The active tab is SEARCHINGELEMENT.java 2. The code in the editor is as follows:

```
1  import java.util.Scanner;
2
3  public class SEARCHINGELEMENT {
4      public static void main(String[] args) {
5          Scanner scanner = new Scanner(System.in);
6
7          // Input the dimensions of the 2D array
8          System.out.print(s:"Enter the number of rows: ");
9          int rows = scanner.nextInt();
10         System.out.print(s:"Enter the number of columns: ");
11         int cols = scanner.nextInt();
12
13         // Initialize the 2D array
14         int[][] array = new int[rows][cols];
15
16         // Input values for the 2D array
17         System.out.println(x:"Enter the elements of the array:");
18         for (int i = 0; i < rows; i++) {
19             for (int j = 0; j < cols; j++) {
20                 System.out.print("array[" + i + "][" + j + "] = ");
21                 array[i][j] = scanner.nextInt();
22             }
23         }
24
25         // Input the number to search
26         System.out.print(s:"Enter the number to search for: ");
27         int target = scanner.nextInt();
28
29         // Search for the number in the 2D array
30         boolean found = false;
31         for (int i = 0; i < rows; i++) {
32             for (int j = 0; j < cols; j++) {
33                 if (array[i][j] == target) {
34                     System.out.println("Number " + target + " found at position: (" + i + ", " + j + ")");
35                     found = true;
36                     break; // Exit the loop once the number is found
37                 }
38             }
39             if (found) break;
40         }
41
42         // If the number is not found
43         if (!found) {
44             System.out.println("Number " + target + " not found in the array.");
45         }
46     }
47 }
48
```



```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Software\Eclipse.app\Contents\Win32\bin\java.exe' ^
    -cp 'C:\Users\hp\AppData\Local\Temp\Eclipse Software\Eclipse.app\Contents\Win32\bin\java.exe' ^
    -jar 'C:\Users\hp\AppData\Local\Temp\Eclipse Software\Eclipse.app\Contents\Win32\bin\java.exe'
Enter the number of rows: 2
Enter the number of columns: 2
Enter the elements of the array:
array[0][0] = 1
array[0][1] = 2
array[1][0] = 3
array[1][1] = 4
Enter the number to search for: 2
Number 2 found at position: (0, 1)
```

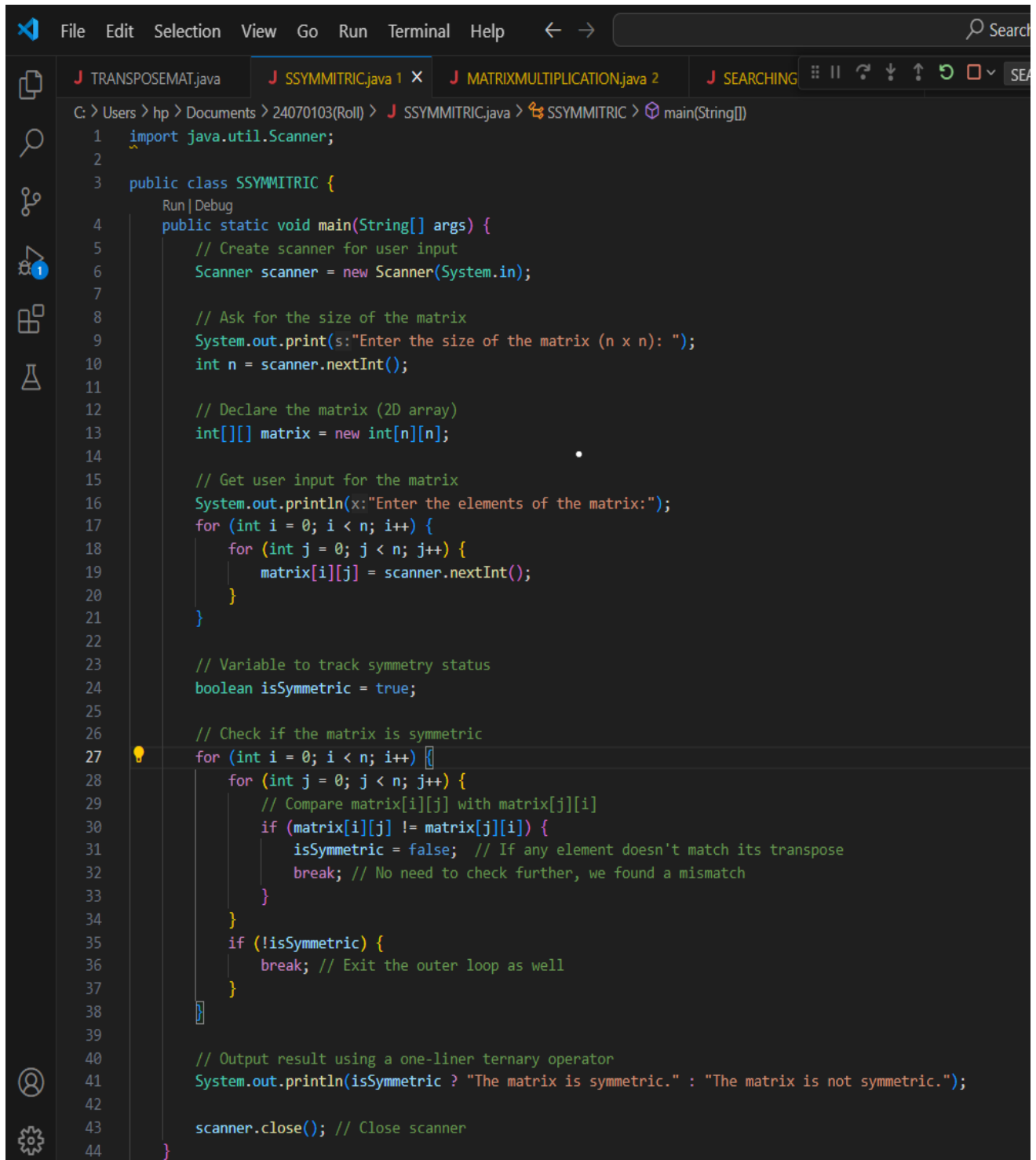
Explanation:

The SEARCHINGELEMENT program allows the user to input a 2D array (matrix) and search for a specific number within it. Here's a quick breakdown:

1. **User Input:**
 - o The user enters the dimensions of the matrix (rows and columns) and then fills the matrix with elements.
 - o The user also specifies the number they want to search for.
2. **Searching:**
 - o The program uses two nested loops to iterate through the 2D array and checks if the target number exists.
 - o If the number is found, it prints the position (row and column).
 - o If the number is not found, it displays a message saying the number is not in the array.
3. **Output:**
 - o If found: Number X found at position: (row, col)
 - o If not found: Number X not found in the array




Question 9:

Write a Java program to check if a matrix is symmetric (i.e., matrix is equal to its transpose).



```
File Edit Selection View Go Run Terminal Help
J TRANSPPOSEMAT.java J SSYMMITRIC.java 1 X J MATRIXMULTIPLICATION.java 2 J SEARCHING

C:\Users\hp\Documents\24070103(Roll)\SSYMMITRIC.java > SSYMMITRIC > main(String[])
1 import java.util.Scanner;
2
3 public class SSYMMITRIC {
4     public static void main(String[] args) {
5         // Create scanner for user input
6         Scanner scanner = new Scanner(System.in);
7
8         // Ask for the size of the matrix
9         System.out.print("Enter the size of the matrix (n x n): ");
10        int n = scanner.nextInt();
11
12        // Declare the matrix (2D array)
13        int[][] matrix = new int[n][n];
14
15        // Get user input for the matrix
16        System.out.println("Enter the elements of the matrix:");
17        for (int i = 0; i < n; i++) {
18            for (int j = 0; j < n; j++) {
19                matrix[i][j] = scanner.nextInt();
20            }
21        }
22
23        // Variable to track symmetry status
24        boolean isSymmetric = true;
25
26        // Check if the matrix is symmetric
27        for (int i = 0; i < n; i++) {
28            for (int j = 0; j < n; j++) {
29                // Compare matrix[i][j] with matrix[j][i]
30                if (matrix[i][j] != matrix[j][i]) {
31                    isSymmetric = false; // If any element doesn't match its transpose
32                    break; // No need to check further, we found a mismatch
33                }
34            }
35            if (!isSymmetric) {
36                break; // Exit the outer loop as well
37            }
38        }
39
40        // Output result using a one-liner ternary operator
41        System.out.println(isSymmetric ? "The matrix is symmetric." : "The matrix is not symmetric.");
42
43        scanner.close(); // Close scanner
44    }
}
```



```
bin' 'SSYMMITRIC'  
Enter the size of the matrix (n x n): 2 2  
Enter the elements of the matrix:  
1 2 2 1  
The matrix is not symmetric.  
PS C:\Users\hp>
```

Explanation:

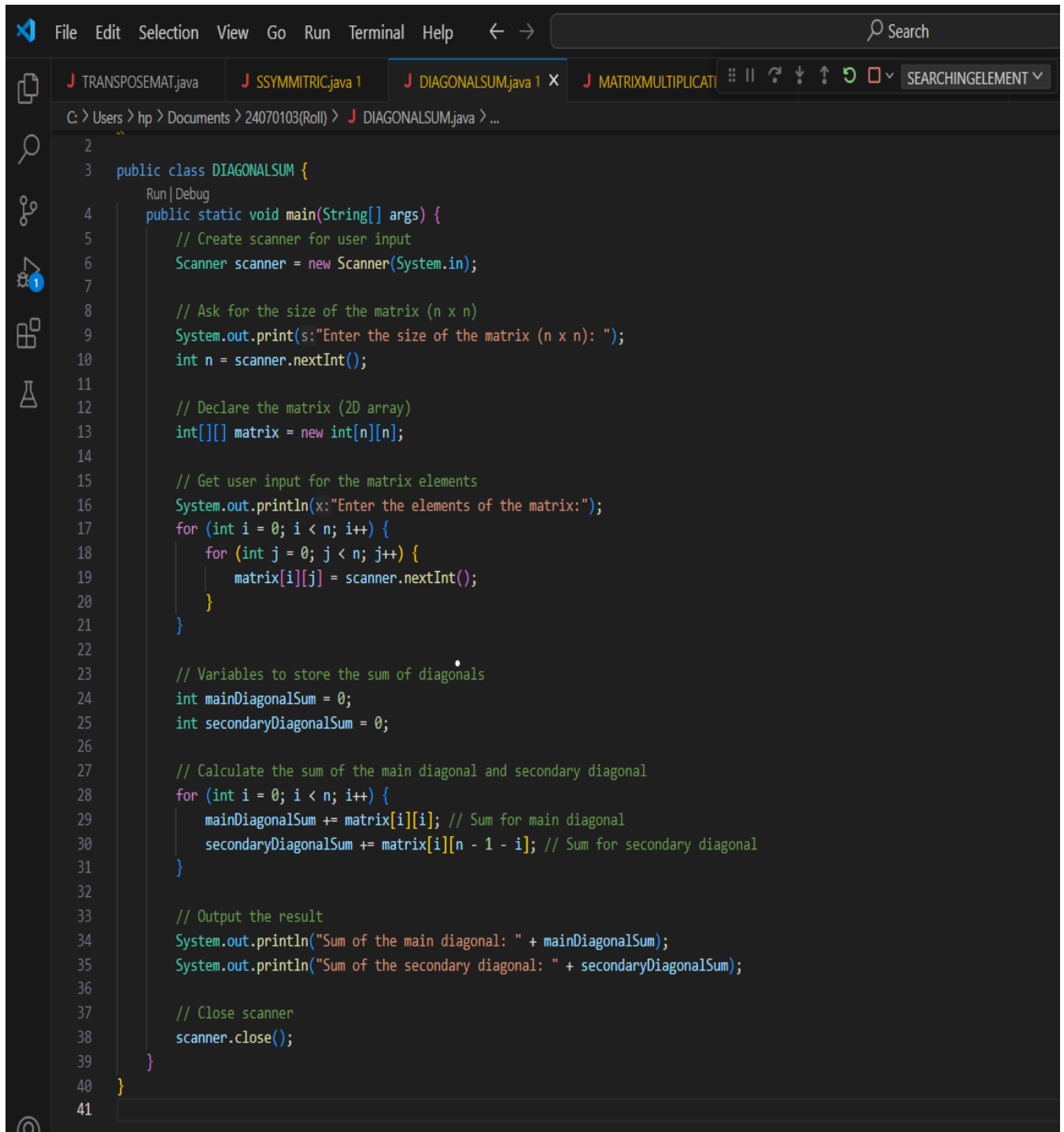
Explanation of the SSYMMITRIC Program:

This Java program checks if a square matrix ($n \times n$) is symmetric. A matrix is **symmetric** if its elements are equal to their corresponding elements in the transposed matrix. The element at position `matrix[i][j]` in the matrix should be equal to `matrix[j][i]` for all i and j .

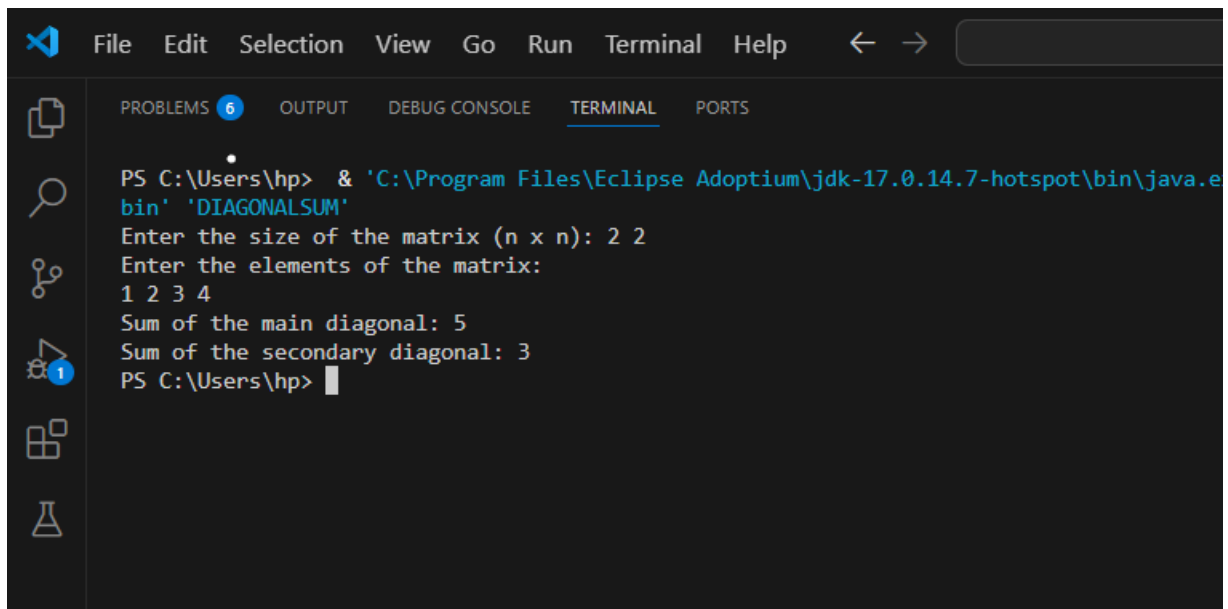
Detailed Breakdown:

1. **User Input:**
 - The program first prompts the user to input the size of the matrix ($n \times n$), where n is the number of rows and columns.
 - Then, the program asks the user to enter the elements of the matrix.
2. **Matrix Declaration:**
 - A 2D array `matrix` of size $n \times n$ is created to store the matrix elements.
3. **Matrix Input:**
 - The program uses nested `for` loops to populate the 2D array with user-inputted values.
4. **Symmetry Check:**
 - The program initializes a boolean variable `isSymmetric` as `true`. This will be used to track whether the matrix is symmetric or not.
 - Two nested loops are used to compare the elements of the matrix with their transposed counterparts:
 - `matrix[i][j]` is compared with `matrix[j][i]`.
 - If at any point `matrix[i][j]` is not equal to `matrix[j][i]`, `isSymmetric` is set to `false`, and the loops break immediately since we already know the matrix is not symmetric.
5. **Output:**
 - After checking all the elements, the program uses a ternary operator to output:
 - "The matrix is symmetric." if `isSymmetric` is `true`.
 - "The matrix is not symmetric." if `isSymmetric` is `false`.
6. **Closing the Scanner:**

Question 10 : Write a Java program to calculate the sum of the main diagonal and secondary diagonal of a square matrix.



```
2
3 public class DIAGONALSUM {
4     public static void main(String[] args) {
5         // Create scanner for user input
6         Scanner scanner = new Scanner(System.in);
7
8         // Ask for the size of the matrix (n x n)
9         System.out.print(s:"Enter the size of the matrix (n x n): ");
10        int n = scanner.nextInt();
11
12        // Declare the matrix (2D array)
13        int[][] matrix = new int[n][n];
14
15        // Get user input for the matrix elements
16        System.out.println(x:"Enter the elements of the matrix:");
17        for (int i = 0; i < n; i++) {
18            for (int j = 0; j < n; j++) {
19                matrix[i][j] = scanner.nextInt();
20            }
21        }
22
23        // Variables to store the sum of diagonals
24        int mainDiagonalSum = 0;
25        int secondaryDiagonalSum = 0;
26
27        // Calculate the sum of the main diagonal and secondary diagonal
28        for (int i = 0; i < n; i++) {
29            mainDiagonalSum += matrix[i][i]; // Sum for main diagonal
30            secondaryDiagonalSum += matrix[i][n - 1 - i]; // Sum for secondary diagonal
31        }
32
33        // Output the result
34        System.out.println("Sum of the main diagonal: " + mainDiagonalSum);
35        System.out.println("Sum of the secondary diagonal: " + secondaryDiagonalSum);
36
37        // Close scanner
38        scanner.close();
39    }
40 }
41
```



```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.14.7-hotspot\bin\java.exe' 'DIAGONALSUM'
Enter the size of the matrix (n x n): 2 2
Enter the elements of the matrix:
1 2 3 4
Sum of the main diagonal: 5
Sum of the secondary diagonal: 3
PS C:\Users\hp>
```

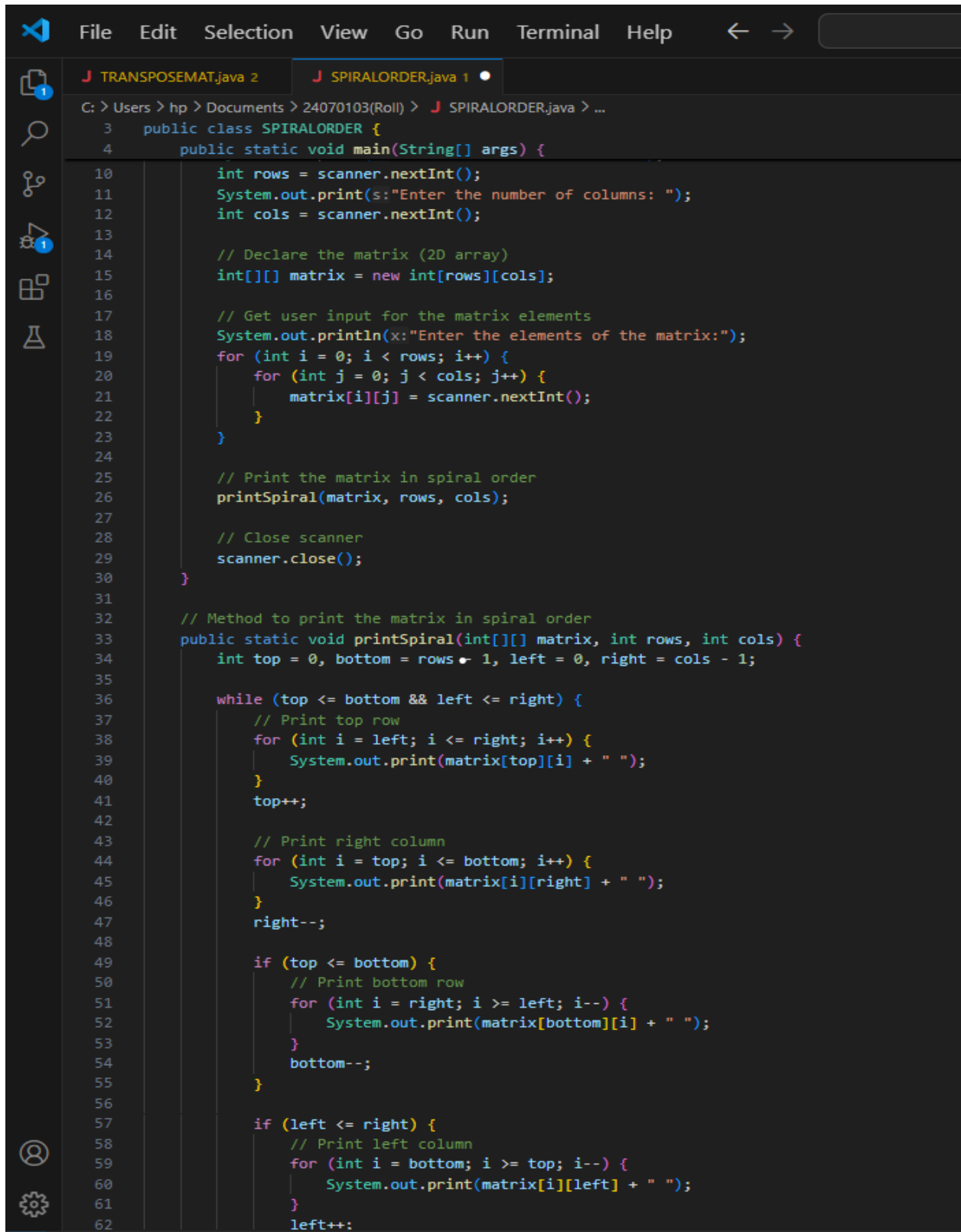
Explanation:

The `DIAGONALSUM` program calculates the sum of the main diagonal and secondary diagonal of a square matrix ($n \times n$).

Key Steps:

1. **Input:** The user provides the size of the matrix ($n \times n$) and the matrix elements.
2. **Main Diagonal:** The sum of elements where the row index equals the column index (`matrix[i][i]`).
3. **Secondary Diagonal:** The sum of elements where the row index and column index sum to $n - 1$ (`matrix[i][n-1-i]`).
4. **Output:** The program prints the sum of the main diagonal and the secondary diagonal.

Question 11 : Write a Java program to print a 2D array in spiral order (starting from the top-left corner and going inwards).

The image shows a screenshot of an IDE with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. Below the menu, there are two tabs: 'TRANPOSEMAT.java 2' and 'SPIRALORDER.java 1'. The active tab is 'SPIRALORDER.java 1', showing the following code:

```
3 public class SPIRALORDER {
4     public static void main(String[] args) {
5
6         // Scanner object
7         Scanner scanner = new Scanner(System.in);
8
9         // Get user input for the number of rows and columns
10        int rows = scanner.nextInt();
11        System.out.print(s:"Enter the number of columns: ");
12        int cols = scanner.nextInt();
13
14        // Declare the matrix (2D array)
15        int[][] matrix = new int[rows][cols];
16
17        // Get user input for the matrix elements
18        System.out.println(x:"Enter the elements of the matrix:");
19        for (int i = 0; i < rows; i++) {
20            for (int j = 0; j < cols; j++) {
21                matrix[i][j] = scanner.nextInt();
22            }
23        }
24
25        // Print the matrix in spiral order
26        printSpiral(matrix, rows, cols);
27
28        // Close scanner
29        scanner.close();
30    }
31
32    // Method to print the matrix in spiral order
33    public static void printSpiral(int[][] matrix, int rows, int cols) {
34        int top = 0, bottom = rows - 1, left = 0, right = cols - 1;
35
36        while (top <= bottom && left <= right) {
37            // Print top row
38            for (int i = left; i <= right; i++) {
39                System.out.print(matrix[top][i] + " ");
40            }
41            top++;
42
43            // Print right column
44            for (int i = top; i <= bottom; i++) {
45                System.out.print(matrix[i][right] + " ");
46            }
47            right--;
48
49            if (top <= bottom) {
50                // Print bottom row
51                for (int i = right; i >= left; i--) {
52                    System.out.print(matrix[bottom][i] + " ");
53                }
54                bottom--;
55            }
56
57            if (left <= right) {
58                // Print left column
59                for (int i = bottom; i >= top; i--) {
60                    System.out.print(matrix[i][left] + " ");
61                }
62                left++;
63            }
64        }
65    }
66 }
```



```
PS C:\Users\hp> & C:\Program Files\Easy Java Compiler\bin\jdt_ws\jdt.ls-java-project\bin' 'SPIRAL'
Enter the number of rows: 3
Enter the number of columns: 3
Enter the elements of the matrix:
1 2 3
4 5 6
7 8 9
1 2 3 6 9 8 7 4 5
PS C:\Users\hp>
```

Explanation:

- **User Input:**

- The program first asks the user to input the number of rows and columns of the matrix.
- It then asks the user to input the matrix elements.

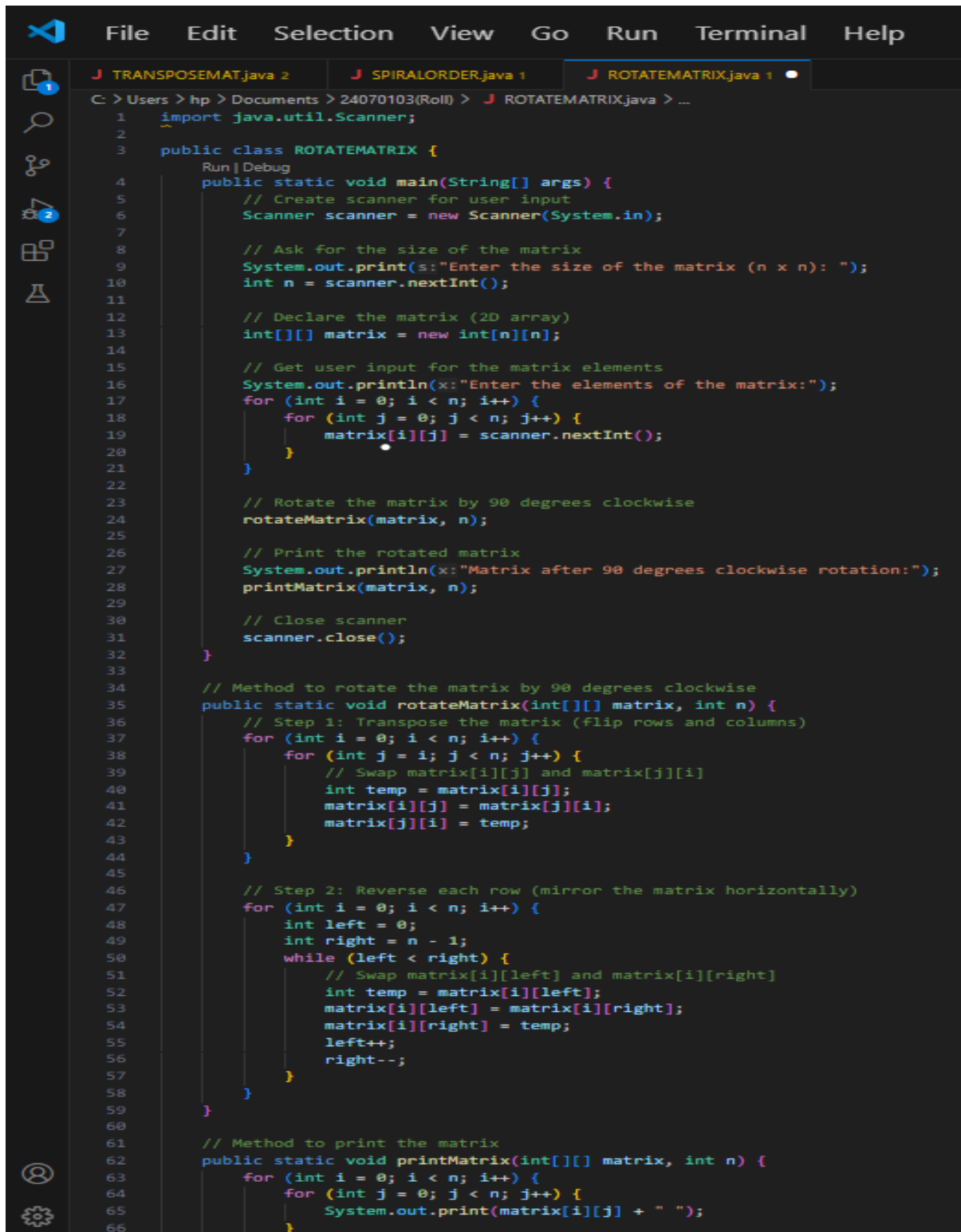
- **Spiral Order Logic:**

- **Boundaries:** Four boundaries are defined: `top`, `bottom`, `left`, and `right`. Initially, these boundaries cover the entire matrix.
- The program then iterates through the matrix in a spiral manner:
 - **Top row:** Traverse the topmost row from left to right.
 - **Right column:** Traverse the rightmost column from top to bottom.
 - **Bottom row:** If there are rows left, traverse the bottommost row from right to left.
 - **Left column:** If there are columns left, traverse the leftmost column from bottom to top.
- After each traversal, the corresponding boundary (`top`, `bottom`, `left`, or `right`) is adjusted inward.

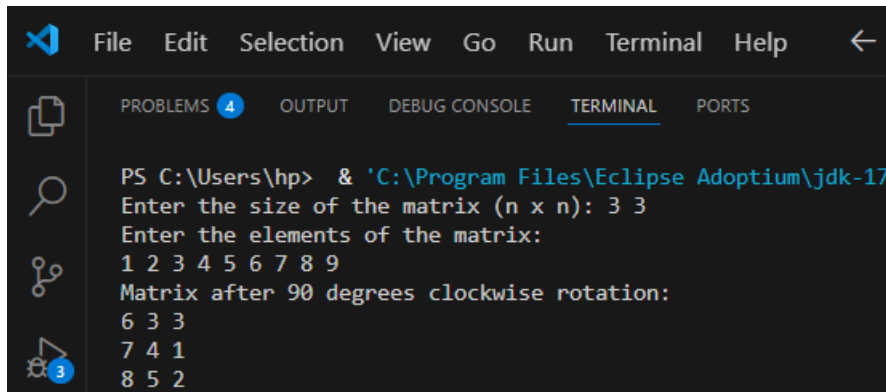
- **Output:**

- The matrix is printed in spiral order, starting from the top-left corner and going inwards.

Question 12: Write a Java program to rotate a square matrix by 90 degrees clockwise.



```
File Edit Selection View Go Run Terminal Help
J TRANSPPOSEMAT.java 2 J SPIRALORDER.java 1 J ROTATEMATRIX.java 1
C:\Users\hp\Documents\24070103(Roll)\> J ROTATEMATRIX.java > ...
1 import java.util.Scanner;
2
3 public class ROTATEMATRIX {
4     Run | Debug
5     public static void main(String[] args) {
6         // Create scanner for user input
7         Scanner scanner = new Scanner(System.in);
8
9         // Ask for the size of the matrix
10        System.out.print("Enter the size of the matrix (n x n): ");
11        int n = scanner.nextInt();
12
13        // Declare the matrix (2D array)
14        int[][] matrix = new int[n][n];
15
16        // Get user input for the matrix elements
17        System.out.println("Enter the elements of the matrix:");
18        for (int i = 0; i < n; i++) {
19            for (int j = 0; j < n; j++) {
20                matrix[i][j] = scanner.nextInt();
21            }
22        }
23
24        // Rotate the matrix by 90 degrees clockwise
25        rotateMatrix(matrix, n);
26
27        // Print the rotated matrix
28        System.out.println("Matrix after 90 degrees clockwise rotation:");
29        printMatrix(matrix, n);
30
31        // Close scanner
32        scanner.close();
33    }
34
35    // Method to rotate the matrix by 90 degrees clockwise
36    public static void rotateMatrix(int[][] matrix, int n) {
37        // Step 1: Transpose the matrix (flip rows and columns)
38        for (int i = 0; i < n; i++) {
39            for (int j = i; j < n; j++) {
40                // Swap matrix[i][j] and matrix[j][i]
41                int temp = matrix[i][j];
42                matrix[i][j] = matrix[j][i];
43                matrix[j][i] = temp;
44            }
45        }
46
47        // Step 2: Reverse each row (mirror the matrix horizontally)
48        for (int i = 0; i < n; i++) {
49            int left = 0;
50            int right = n - 1;
51            while (left < right) {
52                // Swap matrix[i][left] and matrix[i][right]
53                int temp = matrix[i][left];
54                matrix[i][left] = matrix[i][right];
55                matrix[i][right] = temp;
56                left++;
57                right--;
58            }
59        }
60
61        // Method to print the matrix
62        public static void printMatrix(int[][] matrix, int n) {
63            for (int i = 0; i < n; i++) {
64                for (int j = 0; j < n; j++) {
65                    System.out.print(matrix[i][j] + " ");
66                }
67            }
68        }
69    }
70 }
```

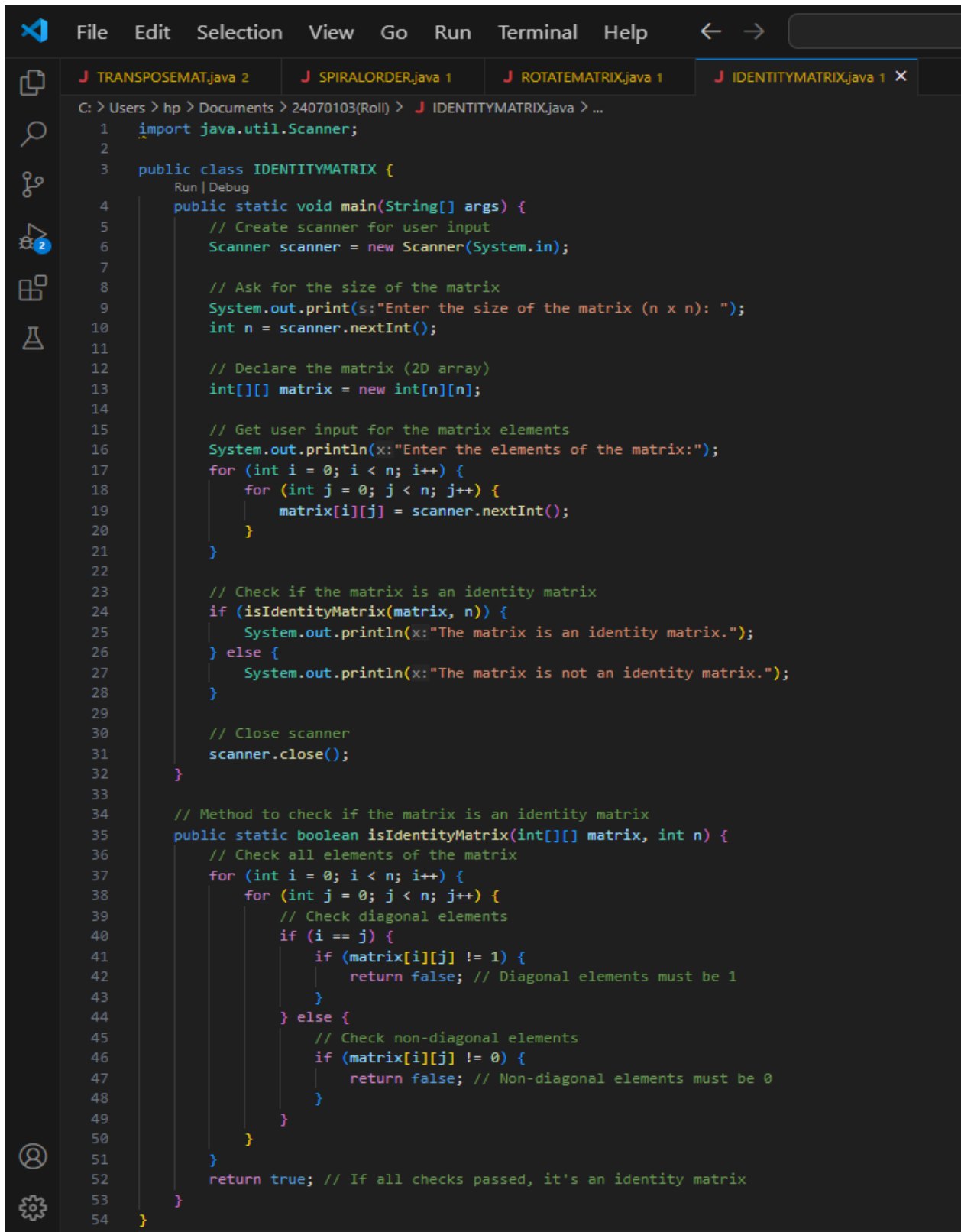


```
PS C:\Users\hp> & 'C:\Program Files\Eclipse Adoptium\jdk-17
Enter the size of the matrix (n x n): 3 3
Enter the elements of the matrix:
1 2 3 4 5 6 7 8 9
Matrix after 90 degrees clockwise rotation:
6 3 3
7 4 1
8 5 2
```

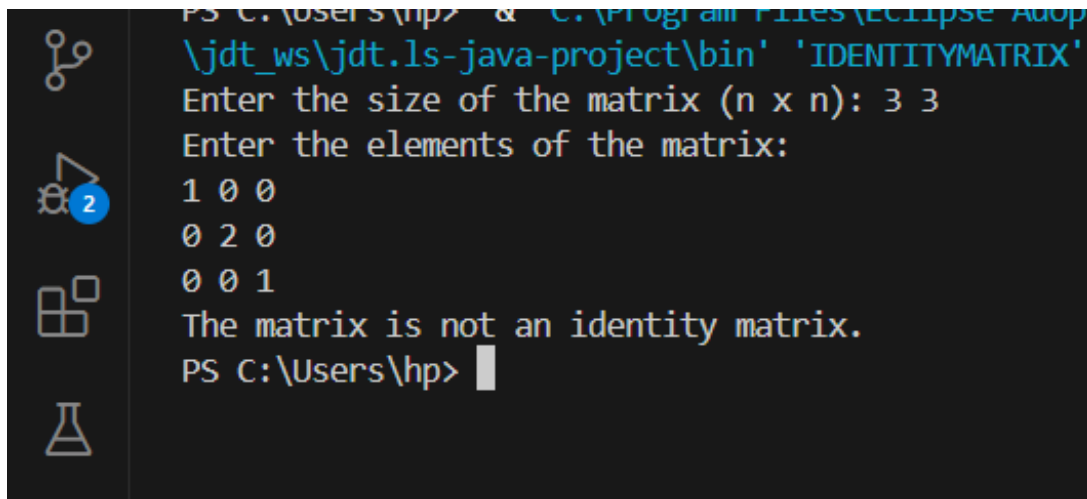
Explanation:

- **User Input:**
 - The program first prompts the user to input the size of the square matrix ($n \times n$).
 - Then, it collects the matrix elements, which are filled row by row.
- **Matrix Rotation:** The rotation happens in two steps:
 - **Step 1: Transpose the matrix:** The elements are swapped across the diagonal (`matrix[i][j]` with `matrix[j][i]`), which mirrors the matrix over the diagonal.
 - **Step 2: Reverse each row:** After the transpose, the rows are reversed (mirror the matrix horizontally) to achieve the 90-degree clockwise rotation.
- **Output:**
 - The program then prints the rotated matrix after applying the rotation logic.

Question 13: Write a Java program to check if a matrix is an identity matrix (diagonal elements are 1, and all others are 0).

The image shows a screenshot of an IDE with a dark theme. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. Below the menu bar, there are four tabs: TRANSPOSEMAT.java 2, SPIRALORDER.java 1, ROTATEMATRIX.java 1, and IDENTITYMATRIX.java 1 (which is the active tab). The main editor area shows the code for IDENTITYMATRIX.java. The code starts with an import statement for java.util.Scanner, followed by a public class IDENTITYMATRIX. Inside the class, there is a main method that prompts the user to enter the size of the matrix (n x n), reads the input, and then prompts the user to enter the elements of the matrix. It then checks if the matrix is an identity matrix by calling a static method isIdentityMatrix. The isIdentityMatrix method checks all elements of the matrix: if any diagonal element is not 1, it returns false; if any non-diagonal element is not 0, it returns false; otherwise, it returns true. The code is as follows:

```
1 import java.util.Scanner;
2
3 public class IDENTITYMATRIX {
4     public static void main(String[] args) {
5         // Create scanner for user input
6         Scanner scanner = new Scanner(System.in);
7
8         // Ask for the size of the matrix
9         System.out.print(s:"Enter the size of the matrix (n x n): ");
10        int n = scanner.nextInt();
11
12        // Declare the matrix (2D array)
13        int[][] matrix = new int[n][n];
14
15        // Get user input for the matrix elements
16        System.out.println(x:"Enter the elements of the matrix:");
17        for (int i = 0; i < n; i++) {
18            for (int j = 0; j < n; j++) {
19                matrix[i][j] = scanner.nextInt();
20            }
21        }
22
23        // Check if the matrix is an identity matrix
24        if (isIdentityMatrix(matrix, n)) {
25            System.out.println(x:"The matrix is an identity matrix.");
26        } else {
27            System.out.println(x:"The matrix is not an identity matrix.");
28        }
29
30        // Close scanner
31        scanner.close();
32    }
33
34    // Method to check if the matrix is an identity matrix
35    public static boolean isIdentityMatrix(int[][] matrix, int n) {
36        // Check all elements of the matrix
37        for (int i = 0; i < n; i++) {
38            for (int j = 0; j < n; j++) {
39                // Check diagonal elements
40                if (i == j) {
41                    if (matrix[i][j] != 1) {
42                        return false; // Diagonal elements must be 1
43                    }
44                } else {
45                    // Check non-diagonal elements
46                    if (matrix[i][j] != 0) {
47                        return false; // Non-diagonal elements must be 0
48                    }
49                }
50            }
51        }
52        return true; // If all checks passed, it's an identity matrix
53    }
54 }
```



```
PS C:\Users\hp> & "C:\Program Files\Eclipse Adoptium\jdk-17.0.10-windows\bin\java.exe" -cp "C:\Users\hp\Documents\jdt_ws\jdt.ls-java-project\bin" 'IDENTITYMATRIX'
Enter the size of the matrix (n x n): 3 3
Enter the elements of the matrix:
1 0 0
0 2 0
0 0 1
The matrix is not an identity matrix.
PS C:\Users\hp>
```

Explanation:

Explanation:

1. User Input:

- The program prompts the user to enter the size of the matrix ($n \times n$).
- The program then asks the user to input the elements of the matrix row by row.

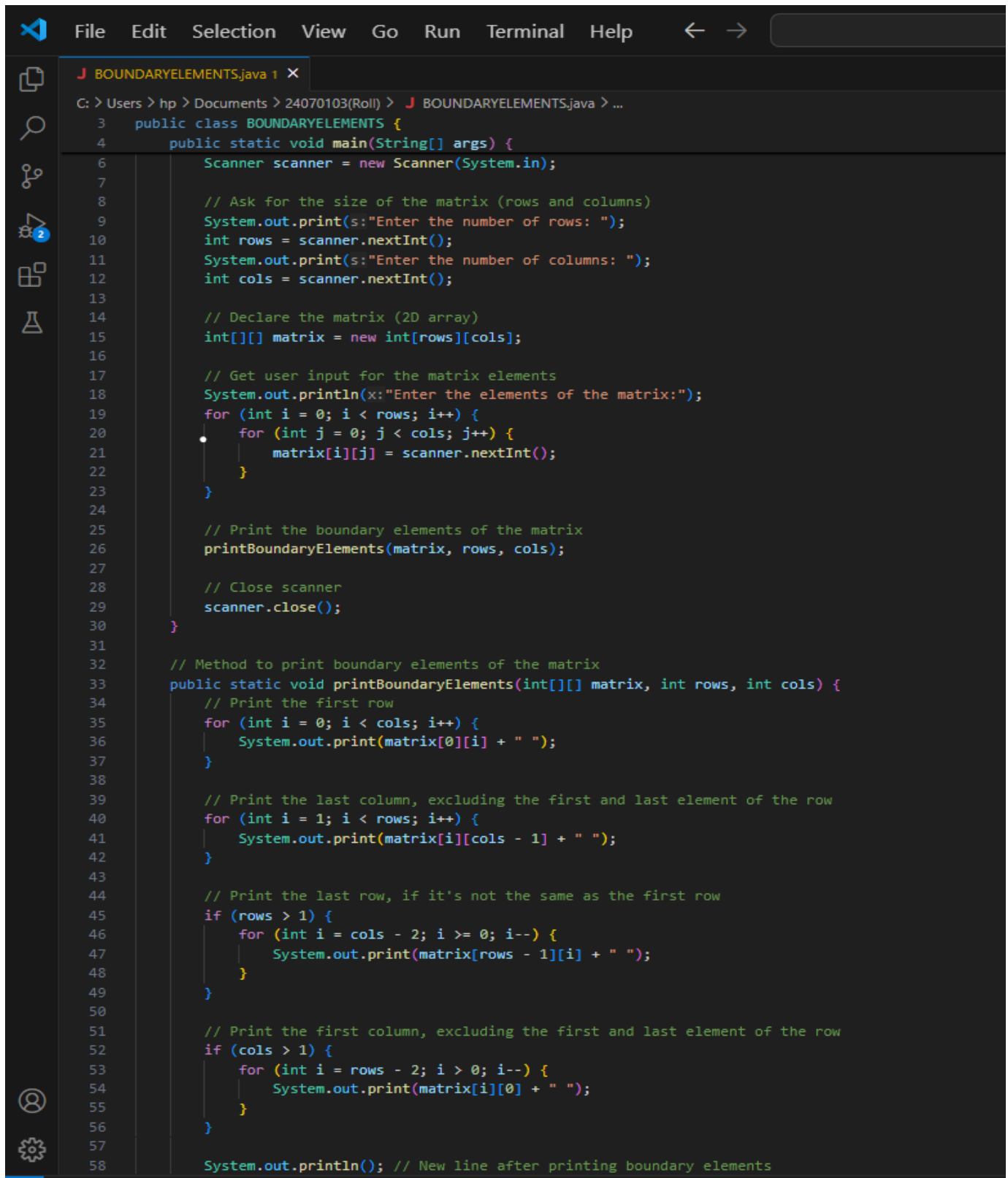
2. Identity Matrix Check:

- **Diagonal elements** (where $i == j$) should be 1.
- **Non-diagonal elements** (where $i \neq j$) should be 0.
- The program checks each element of the matrix to ensure these conditions are met.
- If any element violates these conditions, the matrix is not an identity matrix.

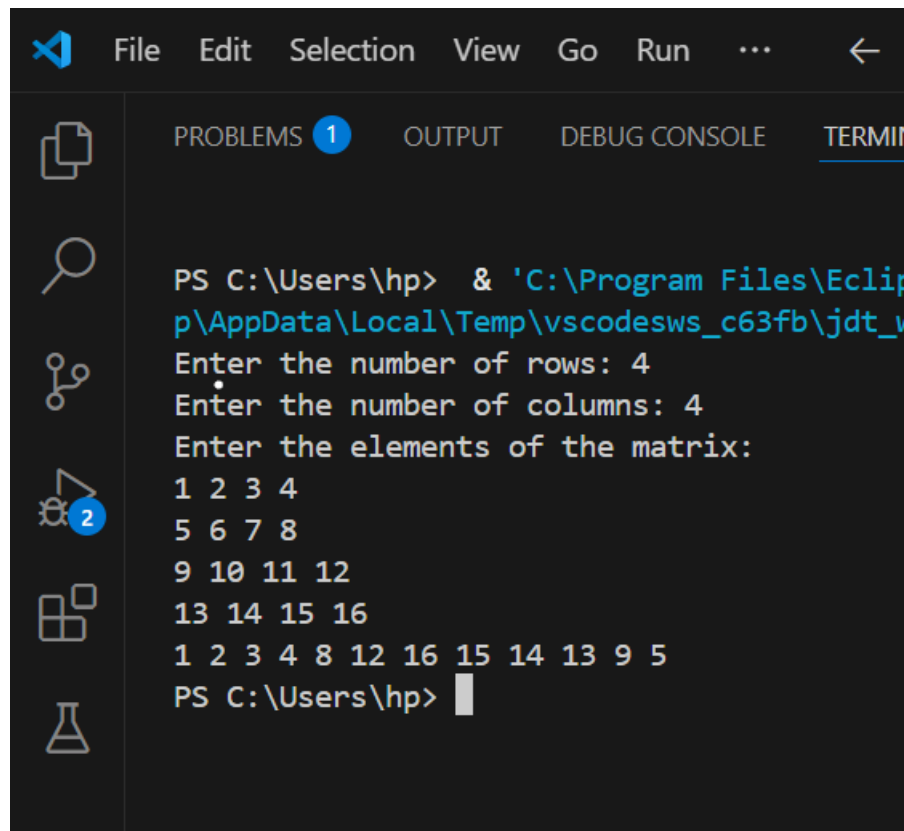
3. Output:

- If the matrix satisfies the identity matrix conditions, the program prints: "The matrix is an identity matrix."
- Otherwise, it prints: "The matrix is not an identity matrix."

Question 14: Write a Java program to print only the boundary elements of a 2D array.



```
File Edit Selection View Go Run Terminal Help
J BOUNDARYELEMENTS.java 1 X
C:\Users\hp\Documents\24070103(Roll)\> J BOUNDARYELEMENTS.java > ...
3 public class BOUNDARYELEMENTS {
4     public static void main(String[] args) {
5
6         Scanner scanner = new Scanner(System.in);
7
8         // Ask for the size of the matrix (rows and columns)
9         System.out.print(s:"Enter the number of rows: ");
10        int rows = scanner.nextInt();
11        System.out.print(s:"Enter the number of columns: ");
12        int cols = scanner.nextInt();
13
14        // Declare the matrix (2D array)
15        int[][] matrix = new int[rows][cols];
16
17        // Get user input for the matrix elements
18        System.out.println(x:"Enter the elements of the matrix:");
19        for (int i = 0; i < rows; i++) {
20            for (int j = 0; j < cols; j++) {
21                matrix[i][j] = scanner.nextInt();
22            }
23        }
24
25        // Print the boundary elements of the matrix
26        printBoundaryElements(matrix, rows, cols);
27
28        // Close scanner
29        scanner.close();
30    }
31
32    // Method to print boundary elements of the matrix
33    public static void printBoundaryElements(int[][] matrix, int rows, int cols) {
34        // Print the first row
35        for (int i = 0; i < cols; i++) {
36            System.out.print(matrix[0][i] + " ");
37        }
38
39        // Print the last column, excluding the first and last element of the row
40        for (int i = 1; i < rows; i++) {
41            System.out.print(matrix[i][cols - 1] + " ");
42        }
43
44        // Print the last row, if it's not the same as the first row
45        if (rows > 1) {
46            for (int i = cols - 2; i >= 0; i--) {
47                System.out.print(matrix[rows - 1][i] + " ");
48            }
49        }
50
51        // Print the first column, excluding the first and last element of the row
52        if (cols > 1) {
53            for (int i = rows - 2; i > 0; i--) {
54                System.out.print(matrix[i][0] + " ");
55            }
56        }
57
58        System.out.println(); // New line after printing boundary elements
```



```
File Edit Selection View Go Run ... <

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\hp> & 'C:\Program Files\Eclipse
p\AppData\Local\Temp\vscodesws_c63fb\jdt_v
Enter the number of rows: 4
Enter the number of columns: 4
Enter the elements of the matrix:
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
1 2 3 4 8 12 16 15 14 13 9 5
PS C:\Users\hp> 
```

Explanation:

- **User Input:**

- The program first asks the user for the number of rows and columns in the matrix.
- Then, the program prompts the user to input the matrix elements row by row.

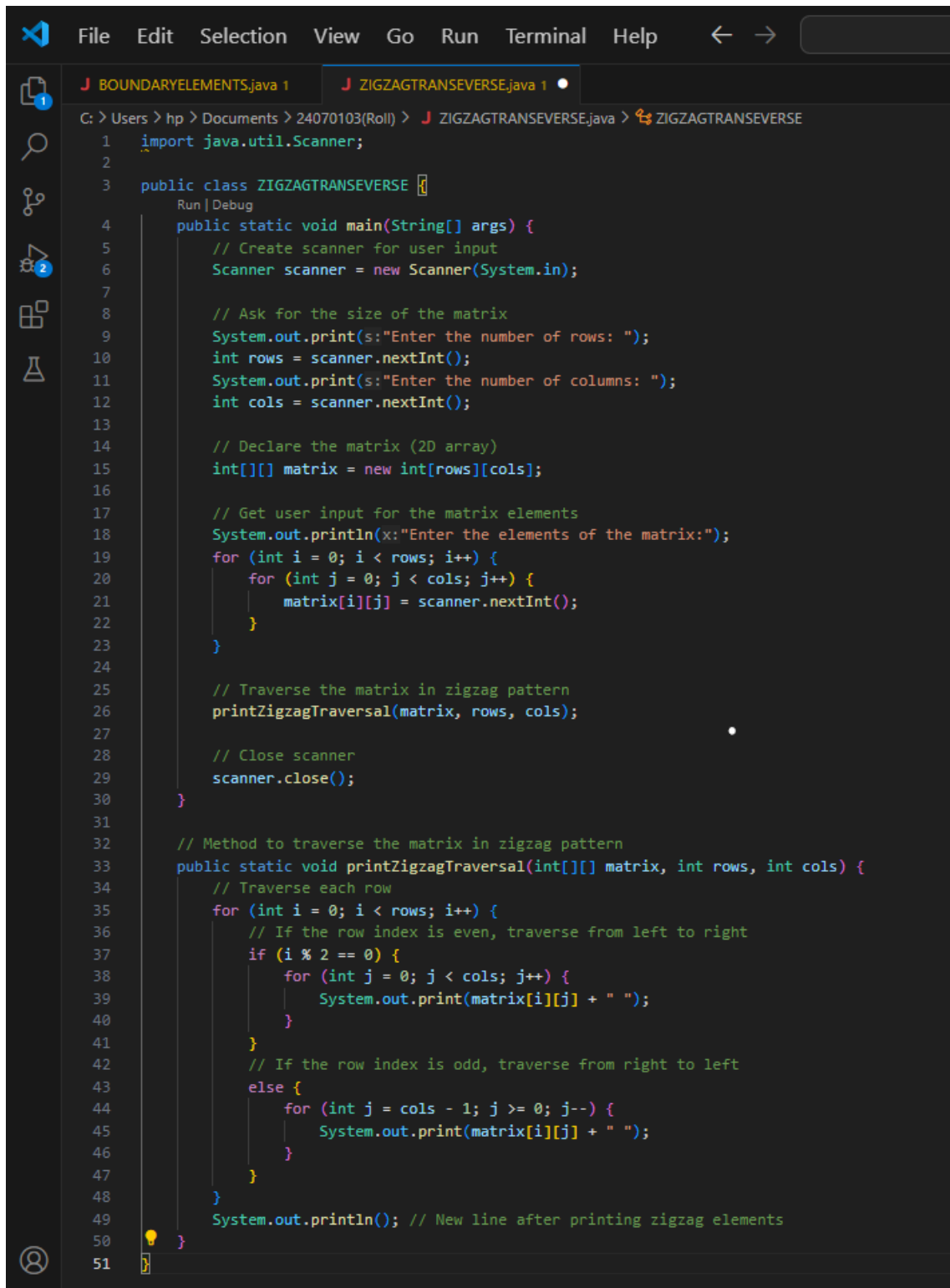
- **Boundary Element Extraction:**

- The boundary elements are the outermost elements in the matrix, which include:
 - **First row:** Traverse all elements in the first row.
 - **Last column:** Traverse all elements in the last column, excluding the already printed elements from the first row and the last row.
 - **Last row:** Traverse the last row in reverse order, excluding the elements from the first and last column.
 - **First column:** Traverse the first column in reverse order, excluding the already printed elements from the first and last row.

- **Output:**

- The boundary elements are printed in the order: first row, last column, last row (in reverse), and first column (in reverse).

Question 15: Write a Java program to traverse a 2D array in a zigzag pattern.



```
File Edit Selection View Go Run Terminal Help
J BOUNDARYELEMENTS.java 1 J ZIGZAGTRANSEVERSE.java 1
C:\Users\hp\Documents\24070103(Roll) > J ZIGZAGTRANSEVERSE.java > ZIGZAGTRANSEVERSE
1 import java.util.Scanner;
2
3 public class ZIGZAGTRANSEVERSE {
4     public static void main(String[] args) {
5         // Create scanner for user input
6         Scanner scanner = new Scanner(System.in);
7
8         // Ask for the size of the matrix
9         System.out.print(s:"Enter the number of rows: ");
10        int rows = scanner.nextInt();
11        System.out.print(s:"Enter the number of columns: ");
12        int cols = scanner.nextInt();
13
14        // Declare the matrix (2D array)
15        int[][] matrix = new int[rows][cols];
16
17        // Get user input for the matrix elements
18        System.out.println(x:"Enter the elements of the matrix:");
19        for (int i = 0; i < rows; i++) {
20            for (int j = 0; j < cols; j++) {
21                matrix[i][j] = scanner.nextInt();
22            }
23        }
24
25        // Traverse the matrix in zigzag pattern
26        printZigzagTraversal(matrix, rows, cols);
27
28        // Close scanner
29        scanner.close();
30    }
31
32    // Method to traverse the matrix in zigzag pattern
33    public static void printZigzagTraversal(int[][] matrix, int rows, int cols) {
34        // Traverse each row
35        for (int i = 0; i < rows; i++) {
36            // If the row index is even, traverse from left to right
37            if (i % 2 == 0) {
38                for (int j = 0; j < cols; j++) {
39                    System.out.print(matrix[i][j] + " ");
40                }
41            }
42            // If the row index is odd, traverse from right to left
43            else {
44                for (int j = cols - 1; j >= 0; j--) {
45                    System.out.print(matrix[i][j] + " ");
46                }
47            }
48        }
49        System.out.println(); // New line after printing zigzag elements
50    }
51 }
```

