UNIVARSITY 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

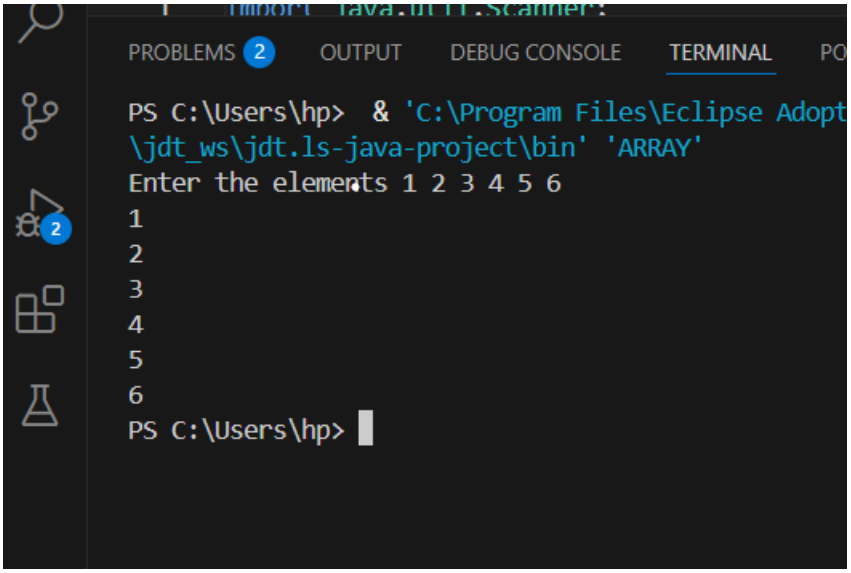
Date: 11. 03. 2025

Dept.: **Computer Science & Engineering (CSE)**

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

```java
import java.util.Scanner;

public class ARRAY {
    Run | Debug
    public static void main(String[] args) {
        Scanner input= new Scanner(System.in);

        int[][] Array = new int[2][3];

        for(int i=0; i<2; i++)
        {
            for(int j=0; j<3; j++)
            {
                Array[i][j]= input.nextInt();
            }
        }

        for(int i=0; i<2; i++)
        {
            for(int j=0; j<3; j++)
            {
                System.out.println(Array[i][j]);
            }
        }
    }

}
```

```
PS C:\Users\hp>  & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.14.7-hotspot\bin\java.exe' '-XX:+ShowCodeDe
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.

File Edit Selection View Go Run Terminal Help

ish.py | ARRAY.java 2 | SUM.java 2

C: > Users > hp > Documents > 24070103(Roll) > SUM.java > SUM > main(String[])

```java
import java.util.Scanner;
public class SUM {
    Run | Debug
    public static void main(String[] args) {
        Scanner input= new Scanner(System.in);

        int[][] Array = new int[2][3];
        int sum= 0;

        System.out.print(s:"Enter the Elements - ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<3; j++)
            {
                Array[i][j]= input.nextInt(); // taking user input
            }
        }

        for(int i=0; i<2; i++)
        {
            for(int j=0; j<3; j++)
            {
                System.out.println(Array[i][j]); // it will  show the elements
            }
        }

        // Summation of elements counter

        for(int i=0; i<2; i++)
        {
            for(int j=0; j<3; j++)
            {
                sum = sum + Array[i][j];
            }
        }

        System.out.print("Sum="+sum);

    }

}
```

```
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 `2x3` 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.

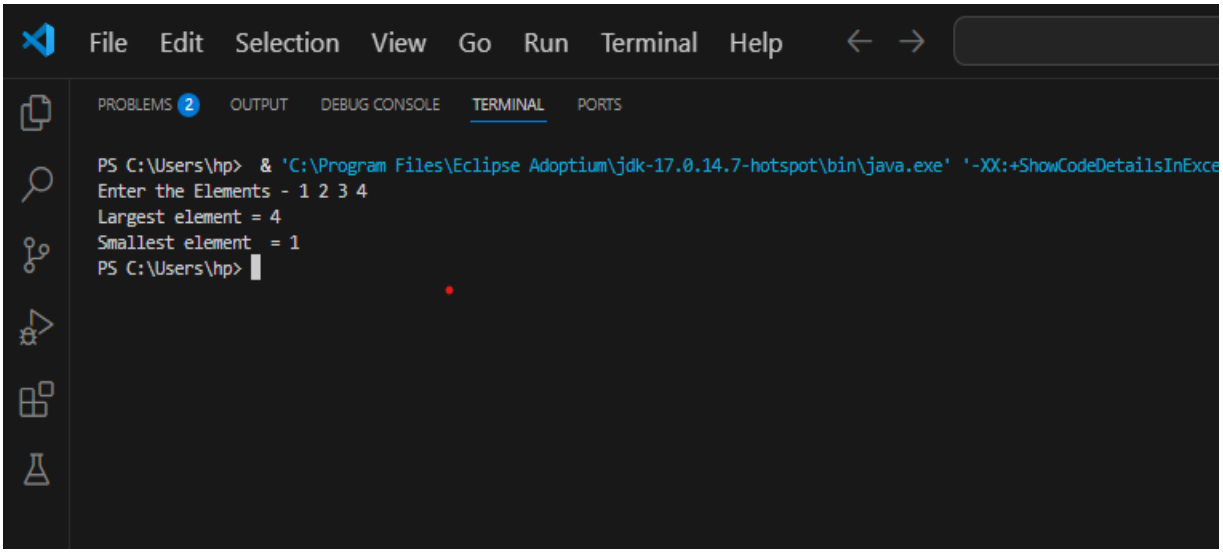
File Edit Selection View Go Run Terminal Help

ish.py | ARRAY.java 2 | SUM.java 2 | SMALLESTANDLARGESTELEMENT.java 2

C: > Users > hp > Documents > 24070103(Roll) > SMALLESTANDLARGESTELEMENT.java > SMALLESTANDLARGESTELEMENT > main(String[])

```java
import java.util.Scanner;
public class SMALLESTANDLARGESTELEMENT {
    Run | Debug
    public static void main(String[] args) {
        Scanner input= new Scanner(System.in);

        int[][] Array = new int[2][2];

        System.out.print(s:"Enter the Elements - ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                Array[i][j]= input.nextInt(); // taking user input
            }
        }



        int large = Array[0][0];

        // finding largest element

        for(int i= 0; i<2; i++)
        {
            for(int j= 0; j<2; j++)
            {
                if(large<Array[i][j])
                {
                    large = Array [i][j];
                }
            }
        }

        int small = Array[0][0];

        // finding lsmallst element

        for(int i= 0; i<2; i++)
        {
            for(int j= 0; j<2; j++)
            {
                if(small>Array[i][j])
                {
                    small = Array [i][j];
                }
            }
        }

        System.out.println("Largest element = "+large);
        System.out.println("Smallest element  = "+small);

    }

}
```

Output:



Explanation:

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

1. **Array Initialization**:
   - A `2x2` array is created to store integer elements.
   - User input is taken to populate the array.
2. **Finding the Largest Element**:
   - 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**:
   - 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**:
   - 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 | SMALLESTANDLARGESTELEMENT.java 2 | SUMOFROWS

C: > Users > hp > Documents > 24070103(Roll) > SUMOFROWSANDCOLUMN.java > ...

```java
import java.util.Scanner;

public class SUMOFROWSANDCOLUMN {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Prompt user for the number of rows and columns
        System.out.print(s:"Enter the number of rows: ");
        int rows = scanner.nextInt();
        System.out.print(s:"Enter the number of columns: ");
        int cols = scanner.nextInt();

        // Initialize the matrix
        int[][] matrix = new int[rows][cols];

        // Input matrix elements
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print("Element at [" + (i + 1) + "][" + (j + 1) + "]: ");
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Find the sum of each row
        System.out.println(x:"\nSum of each row:");
        for (int i = 0; i < rows; i++) {
            int rowSum = 0;
            for (int j = 0; j < cols; j++) {
                rowSum += matrix[i][j];
            }
            System.out.println("Row " + (i + 1) + " sum: " + rowSum);
        }

        // Find the sum of each column
        System.out.println(x:"\nSum of each column:");
        for (int j = 0; j < cols; j++) {
            int colSum = 0;
            for (int i = 0; i < rows; i++) {
                colSum += matrix[i][j];
            }
            System.out.println("Column " + (j + 1) + " sum: " + colSum);
        }

        // Close scanner
        scanner.close();
    }
}
```

Output:

```
File  Edit  Selection  View  Go  Run  Terminal  Help

PROBLEMS 3   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

PS C:\Users\hp>  & 'C:\Program Files\Eclipse Adoptium\jdk-17.0.
odesws_adae0\jdt_ws\jdt.ls-java-project\bin' 'SUMOFROWSANDCOLUM
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.
odesws_adae0\jdt_ws\jdt.ls-java-project\bin' 'SUMOFROWSANDCOLUM
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).

File Edit Selection View Go Run Terminal Help

ARRAY.java 2 | SUM.java 2 | SMALLESTANDLARGESTELEMENT.java 2

C: > Users > hp > Documents > 24070103(Roll) > TRANSPOSEMAT.java > TRANSPOSEMAT >

```java
import java.util.Scanner;
public class TRANSPOSEMAT {
    Run | Debug
    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);

        int[][] A = new int[2][2];

        System.out.print(s:"Enter elements for the matrix: ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                A[i][j]= input.nextInt();
            }
        }
        System.out.println(x:"A= ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                System.out.print(A[i][j]+" ");
            }
            System.out.println();
        }

        System.out.println(x:"Transpose of A (At) = ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                System.out.print(A[j][i]+" ");
            }
            System.out.println();
        }
```

```
\jdt_ws\jdt.ls-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 `2x2` 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.

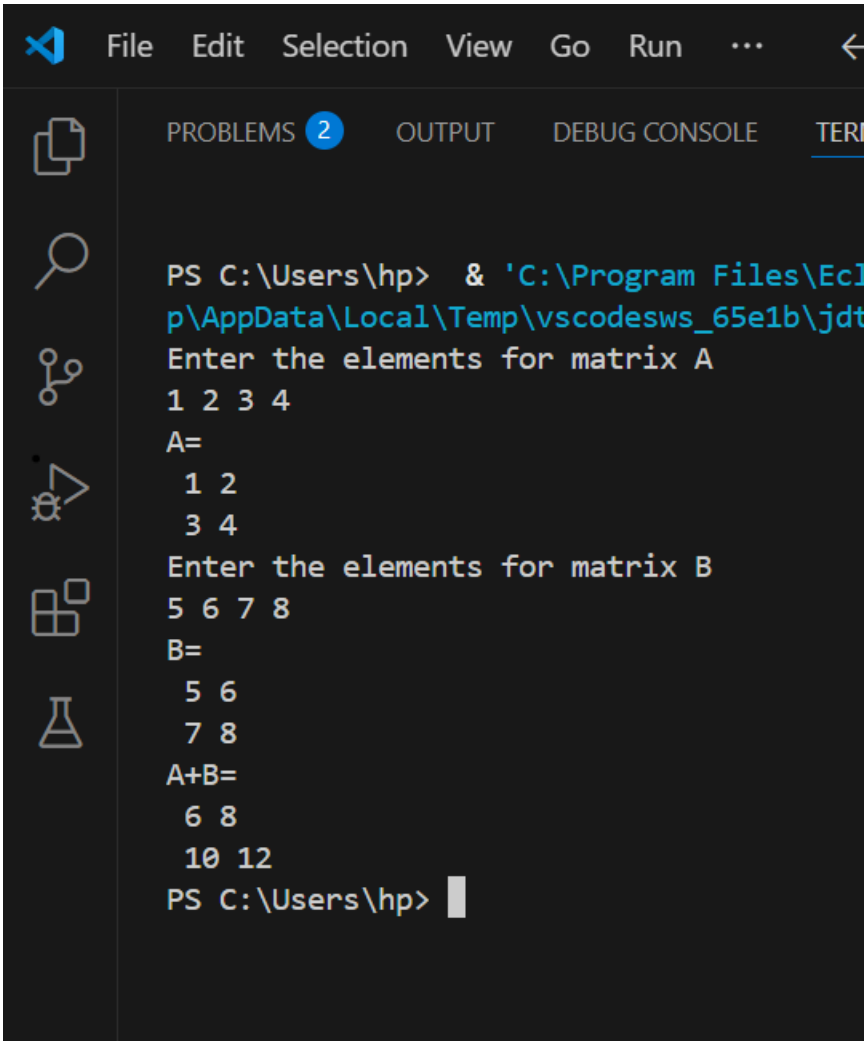
File Edit Selection View Go Run Terminal Help

NEWPROG.java 1

C: > Users > hp > Documents > kl > NEWPROG.java > NEWPROG > main(String[])

```java
import java.util.Scanner;

public class NEWPROG {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        int[][] A= new int[2][2];
        int[][] B= new int[2][2];
        int[][] sum= new int[2][2];

        System.out.println("Enter the elements for matrix A ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                A[i][j]= input.nextInt();
            }
        }
        System.out.println("A= ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                System.out.print(" "+A[i][j]);
            }
            System.out.println();
        }

        System.out.println("Enter the elements for matrix B "); //B
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                B[i][j]= input.nextInt();
            }
        }
        System.out.println("B= ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                System.out.print(" "+B[i][j]);
            }
            System.out.println();
        }


        for(int i=0; i<2; i++){
            for(int j=0; j<2; j++){
                sum[i][j] = A[i][j]+B[i][j];
            }
        }
        System.out.println("A+B= ");
        for(int i=0; i<2; i++)
        {
            for(int j=0; j<2; j++)
            {
                System.out.print(" "+sum[i][j]);
            }
            System.out.println();
        }
    }

}
```

Output:


File Edit Selection View Go Run ···
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERI
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.

File Edit Selection View Go Run Terminal Help

TRANSPOSEMAT.java | SSYMMITRIC.java 1 | MATRIXMULTIPLICATION.java 2

C: > Users > hp > Documents > 24070103(Roll) > MATRIXMULTIPLICATION.java > ...

```java
import java.util.Scanner;

public class MATRIXMULTIPLICATION {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the dimensions of matrix A
        System.out.print(s:"Enter the number of rows for matrix A: ");
        int rowsA = scanner.nextInt();
        System.out.print(s:"Enter the number of columns for matrix A: ");
        int colsA = scanner.nextInt();

        // Input the dimensions of matrix B
        System.out.print(s:"Enter the number of rows for matrix B: ");
        int rowsB = scanner.nextInt();
        System.out.print(s:"Enter the number of columns for matrix B: ");
        int colsB = scanner.nextInt();

        // Check if multiplication is possible
        if (colsA != rowsB) {
            System.out.println(x:"Matrix multiplication is not possible. Number of columns of A must be equal to number of rows of B.");
            return;
        }

        // Initialize the matrices A, B and C (result matrix)
        int[][] A = new int[rowsA][colsA];
        int[][] B = new int[rowsB][colsB];
        int[][] C = new int[rowsA][colsB];

        // Input values for matrix A
        System.out.println(x:"Enter the elements of matrix A:");
        for (int i = 0; i < rowsA; i++) {
            for (int j = 0; j < colsA; j++) {
                System.out.print("A[" + i + "][" + j + "] = ");
                A[i][j] = scanner.nextInt();
            }
        }

        // Input values for matrix B
        System.out.println(x:"Enter the elements of matrix B:");
        for (int i = 0; i < rowsB; i++) {
            for (int j = 0; j < colsB; j++) {
                System.out.print("B[" + i + "][" + j + "] = ");
                B[i][j] = scanner.nextInt();
            }
        }

        // Perform matrix multiplication
        for (int i = 0; i < rowsA; i++) {
            for (int j = 0; j < colsB; j++) {
                C[i][j] = 0; // Initialize the result element
                for (int k = 0; k < colsA; k++) {
                    C[i][j] += A[i][k] * B[k][j];
                }
            }
        }

        // Output the result matrix C
        System.out.println(x:"Resulting Matrix (C) after multiplication:");
        for (int i = 0; i < rowsA; i++) {
            for (int j = 0; j < colsB; j++) {
                System.out.print(C[i][j] + " ");
            }
            System.out.println();
        }
    }
```

File Edit Selection View ···

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL P

```
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.

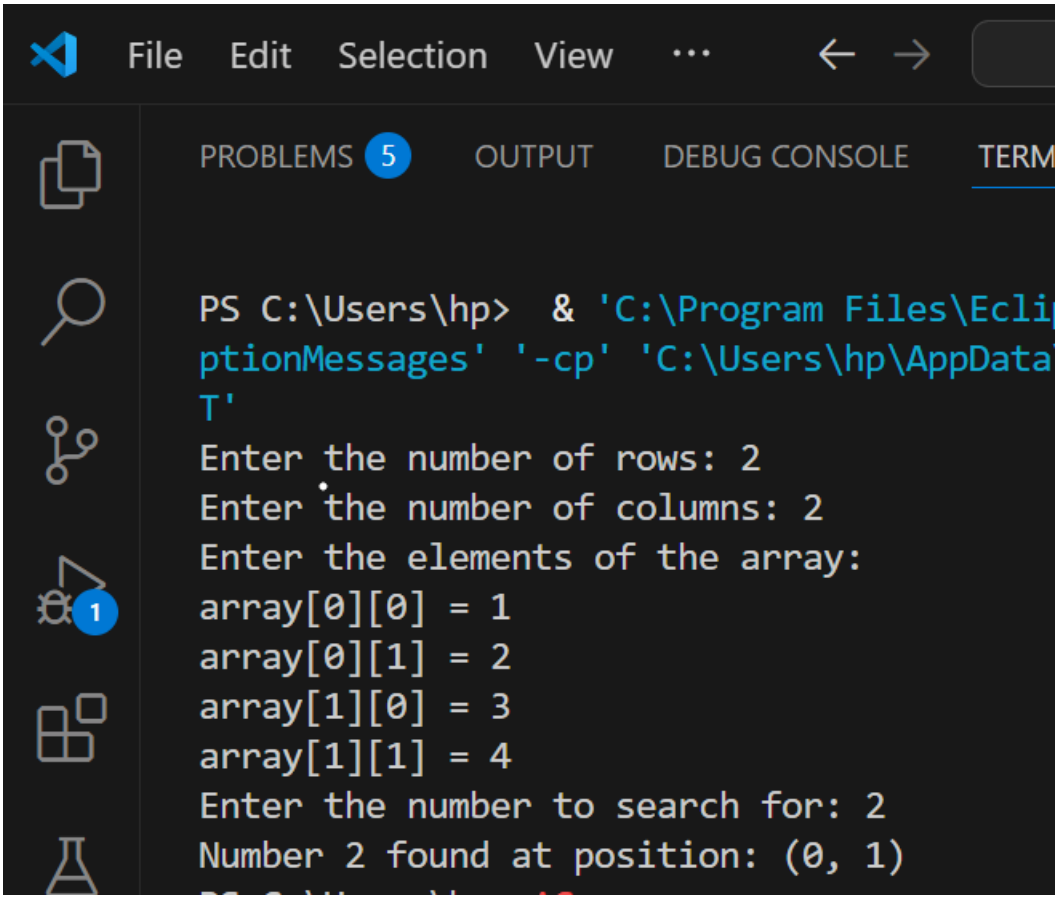
Example:

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.

```java
import java.util.Scanner;

public class SEARCHINGELEMENT {
    Run | Debug
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Input the dimensions of the 2D array
        System.out.print(s:"Enter the number of rows: ");
        int rows = scanner.nextInt();
        System.out.print(s:"Enter the number of columns: ");
        int cols = scanner.nextInt();

        // Initialize the 2D array
        int[][] array = new int[rows][cols];

        // Input values for the 2D array
        System.out.println(x:"Enter the elements of the array:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                System.out.print("array[" + i + "][" + j + "] = ");
                array[i][j] = scanner.nextInt();
            }
        }

        // Input the number to search
        System.out.print(s:"Enter the number to search for: ");
        int target = scanner.nextInt();

        // Search for the number in the 2D array
        boolean found = false;
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                if (array[i][j] == target) {
                    System.out.println("Number " + target + " found at position: (" + i + ", " + j + ")");
                    found = true;
                    break; // Exit the loop once the number is found
                }
            }
            if (found) break;
        }

        // If the number is not found
        if (!found) {
            System.out.println("Number " + target + " not found in the array.");
        }
    }
}
```



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

```java
import java.util.Scanner;

public class SSYMMITRIC {
    Run | Debug
    public static void main(String[] args) {
        // Create scanner for user input
        Scanner scanner = new Scanner(System.in);

        // Ask for the size of the matrix
        System.out.print(s:"Enter the size of the matrix (n x n): ");
        int n = scanner.nextInt();

        // Declare the matrix (2D array)
        int[][] matrix = new int[n][n];

        // Get user input for the matrix
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Variable to track symmetry status
        boolean isSymmetric = true;

        // Check if the matrix is symmetric
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                // Compare matrix[i][j] with matrix[j][i]
                if (matrix[i][j] != matrix[j][i]) {
                    isSymmetric = false;  // If any element doesn't match its transpose
                    break; // No need to check further, we found a mismatch
                }
            }
            if (!isSymmetric) {
                break; // Exit the outer loop as well
            }
        }

        // Output result using a one-liner ternary operator
        System.out.println(isSymmetric ? "The matrix is symmetric." : "The matrix is not symmetric.");

        scanner.close(); // Close scanner
    }
```

```
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:

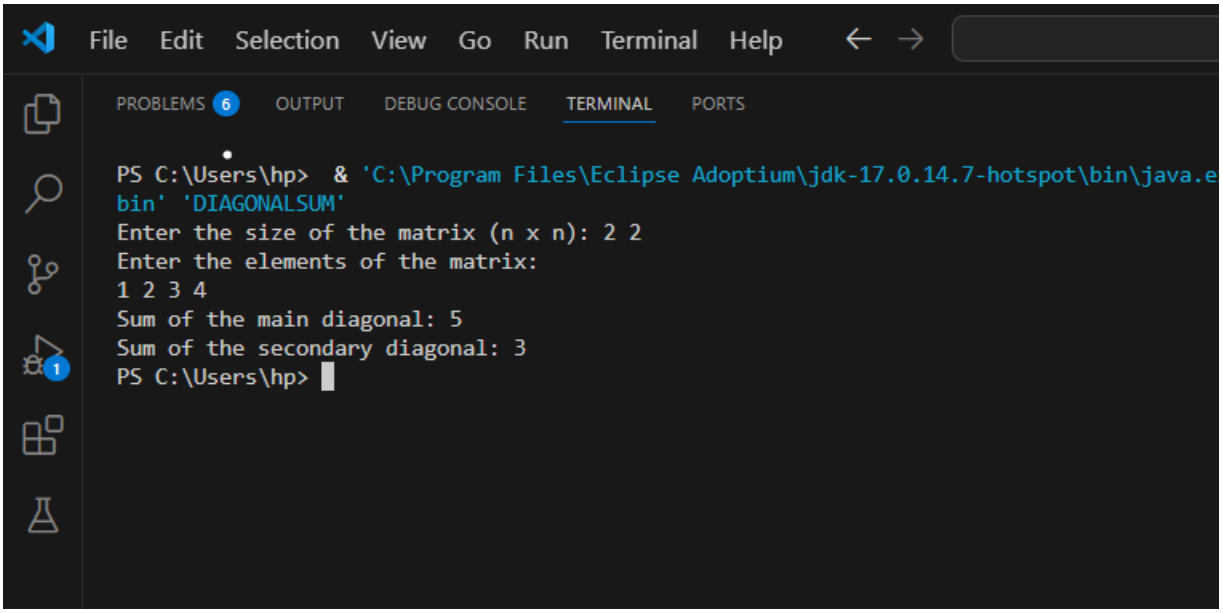
This Java program checks if a square matrix (n x n) is symmetric. A matrix is **symmetric** if its elements are equal to their corresponding elements in the transposed matrix. The element at position `matrix[i][j]` in the matrix should be equal to `matrix[j][i]` for all `i` and `j`.

Detailed Breakdown:

1. **User Input**:
   - The program first prompts the user to input the size of the matrix (`n x 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 x 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.

```java

public class DIAGONALSUM {
    Run | Debug
    public static void main(String[] args) {
        // Create scanner for user input
        Scanner scanner = new Scanner(System.in);

        // Ask for the size of the matrix (n x n)
        System.out.print(s:"Enter the size of the matrix (n x n): ");
        int n = scanner.nextInt();

        // Declare the matrix (2D array)
        int[][] matrix = new int[n][n];

        // Get user input for the matrix elements
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Variables to store the sum of diagonals
        int mainDiagonalSum = 0;
        int secondaryDiagonalSum = 0;

        // Calculate the sum of the main diagonal and secondary diagonal
        for (int i = 0; i < n; i++) {
            mainDiagonalSum += matrix[i][i]; // Sum for main diagonal
            secondaryDiagonalSum += matrix[i][n - 1 - i]; // Sum for secondary diagonal
        }

        // Output the result
        System.out.println("Sum of the main diagonal: " + mainDiagonalSum);
        System.out.println("Sum of the secondary diagonal: " + secondaryDiagonalSum);

        // Close scanner
        scanner.close();
    }
}
```



Explanation:

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

Key Steps:

1. **Input**: The user provides the size of the matrix (`n x 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).

File Edit Selection View Go Run Terminal Help

TRANSPOSEMAT.java 2 | SPIRALORDER.java 1

C: > Users > hp > Documents > 24070103(Roll) > SPIRALORDER.java > ...

```java
public class SPIRALORDER {
    public static void main(String[] args) {
        int rows = scanner.nextInt();
        System.out.print(s:"Enter the number of columns: ");
        int cols = scanner.nextInt();

        // Declare the matrix (2D array)
        int[][] matrix = new int[rows][cols];

        // Get user input for the matrix elements
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Print the matrix in spiral order
        printSpiral(matrix, rows, cols);

        // Close scanner
        scanner.close();
    }

    // Method to print the matrix in spiral order
    public static void printSpiral(int[][] matrix, int rows, int cols) {
        int top = 0, bottom = rows - 1, left = 0, right = cols - 1;

        while (top <= bottom && left <= right) {
            // Print top row
            for (int i = left; i <= right; i++) {
                System.out.print(matrix[top][i] + " ");
            }
            top++;

            // Print right column
            for (int i = top; i <= bottom; i++) {
                System.out.print(matrix[i][right] + " ");
            }
            right--;

            if (top <= bottom) {
                // Print bottom row
                for (int i = right; i >= left; i--) {
                    System.out.print(matrix[bottom][i] + " ");
                }
                bottom--;
            }

            if (left <= right) {
                // Print left column
                for (int i = bottom; i >= top; i--) {
                    System.out.print(matrix[i][left] + " ");
                }
                left++;
```

```
PS C:\Users\hp>  & 'C:\Program Files\Ec
\jdt_ws\jdt.ls-java-project\bin' 'SPIRA
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.

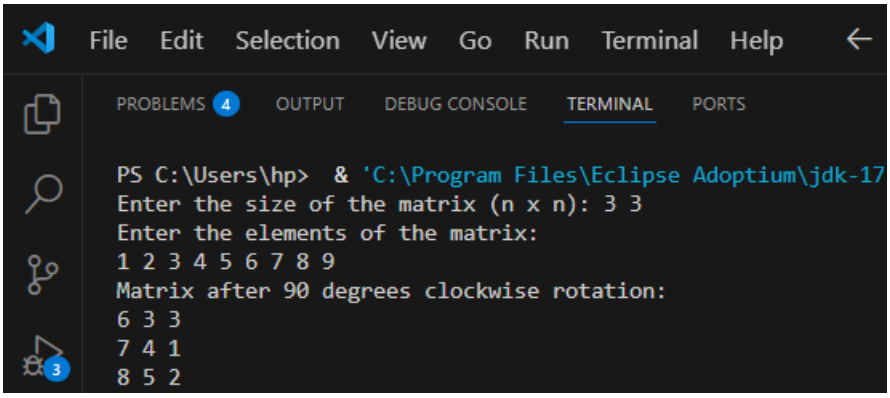
Question 12: Write a Java program to rotate a square matrix by 90 degrees clockwise.

File Edit Selection View Go Run Terminal Help

TRANSPOSEMAT.java 2 | SPIRALORDER.java 1 | ROTATEMATRIX.java 1

C: > Users > hp > Documents > 24070103(Roll) > ROTATEMATRIX.java > ...

```java
import java.util.Scanner;

public class ROTATEMATRIX {
    Run | Debug
    public static void main(String[] args) {
        // Create scanner for user input
        Scanner scanner = new Scanner(System.in);

        // Ask for the size of the matrix
        System.out.print(s:"Enter the size of the matrix (n x n): ");
        int n = scanner.nextInt();

        // Declare the matrix (2D array)
        int[][] matrix = new int[n][n];

        // Get user input for the matrix elements
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Rotate the matrix by 90 degrees clockwise
        rotateMatrix(matrix, n);

        // Print the rotated matrix
        System.out.println(x:"Matrix after 90 degrees clockwise rotation:");
        printMatrix(matrix, n);

        // Close scanner
        scanner.close();
    }

    // Method to rotate the matrix by 90 degrees clockwise
    public static void rotateMatrix(int[][] matrix, int n) {
        // Step 1: Transpose the matrix (flip rows and columns)
        for (int i = 0; i < n; i++) {
            for (int j = i; j < n; j++) {
                // Swap matrix[i][j] and matrix[j][i]
                int temp = matrix[i][j];
                matrix[i][j] = matrix[j][i];
                matrix[j][i] = temp;
            }
        }

        // Step 2: Reverse each row (mirror the matrix horizontally)
        for (int i = 0; i < n; i++) {
            int left = 0;
            int right = n - 1;
            while (left < right) {
                // Swap matrix[i][left] and matrix[i][right]
                int temp = matrix[i][left];
                matrix[i][left] = matrix[i][right];
                matrix[i][right] = temp;
                left++;
                right--;
            }
        }
    }

    // Method to print the matrix
    public static void printMatrix(int[][] matrix, int n) {
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                System.out.print(matrix[i][j] + " ");
            }
```



Explanation:

- **User Input**:
  - The program first prompts the user to input the size of the square matrix (`n x 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).

File Edit Selection View Go Run Terminal Help

TRANSPOSEMAT.java 2 | SPIRALORDER.java 1 | ROTATEMATRIX.java 1 | IDENTITYMATRIX.java 1

C: > Users > hp > Documents > 24070103(Roll) > IDENTITYMATRIX.java > ...

```java
import java.util.Scanner;

public class IDENTITYMATRIX {
    Run | Debug
    public static void main(String[] args) {
        // Create scanner for user input
        Scanner scanner = new Scanner(System.in);

        // Ask for the size of the matrix
        System.out.print(s:"Enter the size of the matrix (n x n): ");
        int n = scanner.nextInt();

        // Declare the matrix (2D array)
        int[][] matrix = new int[n][n];

        // Get user input for the matrix elements
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Check if the matrix is an identity matrix
        if (isIdentityMatrix(matrix, n)) {
            System.out.println(x:"The matrix is an identity matrix.");
        } else {
            System.out.println(x:"The matrix is not an identity matrix.");
        }

        // Close scanner
        scanner.close();
    }

    // Method to check if the matrix is an identity matrix
    public static boolean isIdentityMatrix(int[][] matrix, int n) {
        // Check all elements of the matrix
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < n; j++) {
                // Check diagonal elements
                if (i == j) {
                    if (matrix[i][j] != 1) {
                        return false; // Diagonal elements must be 1
                    }
                } else {
                    // Check non-diagonal elements
                    if (matrix[i][j] != 0) {
                        return false; // Non-diagonal elements must be 0
                    }
                }
            }
        }
        return true; // If all checks passed, it's an identity matrix
    }
}
```

```
PS C:\Users\hp> & "C:\Program Files\Eclipse Adopt
\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>
```

<u>Explanation:</u>

Explanation:

1. **User Input**:
   - The program prompts the user to enter the size of the matrix (`n x 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 != 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

BOUNDARYELEMENTS.java 1

C: > Users > hp > Documents > 24070103(Roll) > BOUNDARYELEMENTS.java > ...

```java
public class BOUNDARYELEMENTS {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Ask for the size of the matrix (rows and columns)
        System.out.print(s:"Enter the number of rows: ");
        int rows = scanner.nextInt();
        System.out.print(s:"Enter the number of columns: ");
        int cols = scanner.nextInt();

        // Declare the matrix (2D array)
        int[][] matrix = new int[rows][cols];

        // Get user input for the matrix elements
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Print the boundary elements of the matrix
        printBoundaryElements(matrix, rows, cols);

        // Close scanner
        scanner.close();
    }

    // Method to print boundary elements of the matrix
    public static void printBoundaryElements(int[][] matrix, int rows, int cols) {
        // Print the first row
        for (int i = 0; i < cols; i++) {
            System.out.print(matrix[0][i] + " ");
        }

        // Print the last column, excluding the first and last element of the row
        for (int i = 1; i < rows; i++) {
            System.out.print(matrix[i][cols - 1] + " ");
        }

        // Print the last row, if it's not the same as the first row
        if (rows > 1) {
            for (int i = cols - 2; i >= 0; i--) {
                System.out.print(matrix[rows - 1][i] + " ");
            }
        }

        // Print the first column, excluding the first and last element of the row
        if (cols > 1) {
            for (int i = rows - 2; i > 0; i--) {
                System.out.print(matrix[i][0] + " ");
            }
        }

        System.out.println(); // New line after printing boundary elements
```

File Edit Selection View Go Run ···

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMI

```
PS C:\Users\hp>  & 'C:\Program Files\Eclip
p\AppData\Local\Temp\vscodesws_c63fb\jdt_w
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).

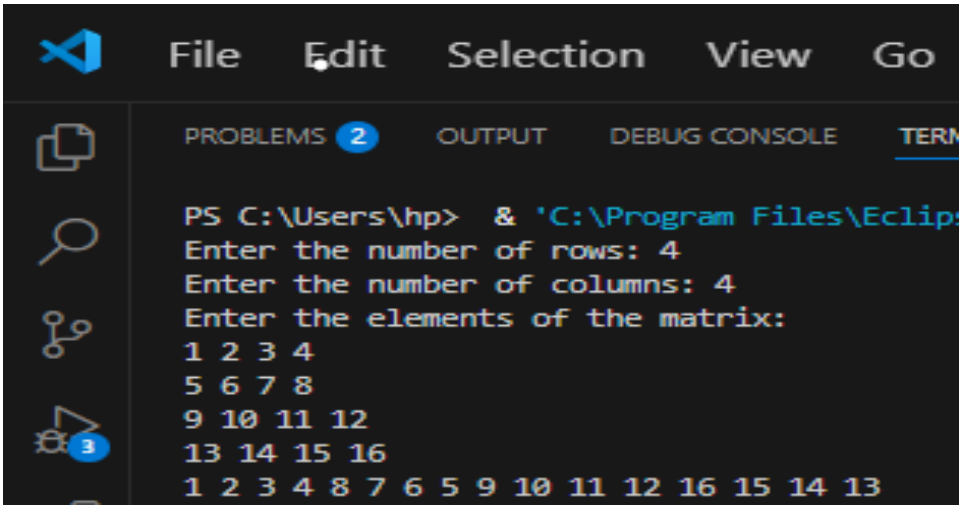
Question 15: Write a Java program to traverse a 2D array in a zigzag pattern.

File Edit Selection View Go Run Terminal Help

BOUNDARYELEMENTS.java 1 | ZIGZAGTRANSEVERSE.java 1

C: > Users > hp > Documents > 24070103(Roll) > ZIGZAGTRANSEVERSE.java > ZIGZAGTRANSEVERSE

```java
import java.util.Scanner;

public class ZIGZAGTRANSEVERSE {
    Run | Debug
    public static void main(String[] args) {
        // Create scanner for user input
        Scanner scanner = new Scanner(System.in);

        // Ask for the size of the matrix
        System.out.print(s:"Enter the number of rows: ");
        int rows = scanner.nextInt();
        System.out.print(s:"Enter the number of columns: ");
        int cols = scanner.nextInt();

        // Declare the matrix (2D array)
        int[][] matrix = new int[rows][cols];

        // Get user input for the matrix elements
        System.out.println(x:"Enter the elements of the matrix:");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < cols; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Traverse the matrix in zigzag pattern
        printZigzagTraversal(matrix, rows, cols);

        // Close scanner
        scanner.close();
    }

    // Method to traverse the matrix in zigzag pattern
    public static void printZigzagTraversal(int[][] matrix, int rows, int cols) {
        // Traverse each row
        for (int i = 0; i < rows; i++) {
            // If the row index is even, traverse from left to right
            if (i % 2 == 0) {
                for (int j = 0; j < cols; j++) {
                    System.out.print(matrix[i][j] + " ");
                }
            }
            // If the row index is odd, traverse from right to left
            else {
                for (int j = cols - 1; j >= 0; j--) {
                    System.out.print(matrix[i][j] + " ");
                }
            }
        }
        System.out.println(); // New line after printing zigzag elements
    }
}
```



Explanation:

1. **User Input**:
   - The program first asks the user for the number of **rows** and **columns** of the matrix.
   - Then, the program takes input for the elements of the matrix row by row.
2. **Zigzag Traversal Logic**:
   - **Even rows** (0, 2, 4, etc.): The elements are traversed from **left to right**.
   - **Odd rows** (1, 3, 5, etc.): The elements are traversed from **right to left**.
3. **Output**:
   - The program prints the elements of the matrix in a zigzag pattern as it traverses row by row.