Korn Rojrattanapanya (64010009)

```java
package exams.exam1;

import java.util.Scanner;

public class S2Q1_64010009 {

    public static final int ROW = 1;
    public static final int COL = 0;

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        int matrix_size;

        while (true) {

            System.out.print("Input size of matrix: ");
            matrix_size = scanner.nextInt();

            if (matrix_size ≥ 3) break;
            System.out.println("ERROR: matrix size must not be less than 3");
        }
        scanner.close();

        int[][] matrix = makeMatrix(matrix_size);
        int[] largest_position = findLargest(matrix);

        System.out.print("The first found largest member is: ");
        System.out.print(matrix[largest_position[ROW]][largest_position[COL]]);
        System.out.printf(" at (%d, %d)\n", largest_position[ROW], largest_position[COL]);

        System.out.print("(a) Members on the Top: ");
        printMembers(getMembersTop(matrix, largest_position));

        System.out.print("(b) Members on the Right: ");
        printMembers(getMembersRight(matrix, largest_position));

        System.out.print("(c) Members on the Bottom: ");
        printMembers(getMembersBottom(matrix, largest_position));

        System.out.print("(d) Members on the Left: ");
        printMembers(getMembersLeft(matrix, largest_position));

        System.out.println("End of program.");
    }

    public static int[] findLargest(int[][] matrix) {

        int[] largest_position = new int[2];

        int largest = -1;

        for (int row = 0; row < matrix.length; row++) {
            for (int col = 0; col < matrix.length; col++) {
                if (matrix[row][col] > largest) {

                    largest = matrix[row][col];
                    largest_position[ROW] = row;
                    largest_position[COL] = col;
                }
            }
        }

        return largest_position;
    }

    public static int[][] makeMatrix(int size) {

        int[][] matrix = new int[size][size];

        System.out.println("Random matrix:");
        for (int row = 0; row < size; row++) {
            for (int col = 0; col < size; col++) {

                matrix[row][col] = (int) (Math.random() * 10);
                System.out.print(matrix[row][col] + " ");
            }
            System.out.print("\n");
        }
        return matrix;
    }

    public static int[] getMembersTop(int[][] matrix, int[] largest) {

        int member_size = largest[ROW];
        int[] members = new int[member_size];
        if (member_size == 0) return members;

        for (int i = 0; i < member_size; i++)
            members[i] = matrix[largest[ROW] - i - 1][largest[COL]];

        return members;
    }

    public static int[] getMembersRight(int[][] matrix, int[] largest) {

        int member_size = matrix.length - 1 - largest[COL];
        int[] members = new int[member_size];
        if (member_size == 0) return members;

        for (int i = 0; i < member_size; i++)
            members[i] = matrix[largest[ROW]][largest[COL] + i + 1];

        return members;
    }

    public static int[] getMembersBottom(int[][] matrix, int[] largest) {
        int member_size = matrix.length - 1 - largest[ROW];
        int[] members = new int[member_size];
        if (member_size == 0) return members;

        for (int i = 0; i < member_size; i++)
            members[i] = matrix[largest[ROW] + i + 1][largest[COL]];

        return members;
    }

    public static int[] getMembersLeft(int[][] matrix, int[] largest) {

        int member_size = largest[COL];
        int[] members = new int[member_size];
        if (member_size == 0) return members;

        for (int i = 0; i < member_size; i++)
            members[i] = matrix[largest[ROW]][largest[COL] - i - 1];

        return members;
    }

    public static void printMembers(int[] members) {

        if (members.length == 0) {

            System.out.println("NO");
            return;
        }

        for (int i = 0; i < members.length; i++) {
            if (i != members.length - 1) System.out.print(members[i] + ", ");
            else System.out.print(members[i]);
        }
        System.out.print("\n");
    }
}
```