Exercises with Arrays and Matrices

Objectives

- Array Exercises
- Matrix exercises
- Homework exercises
- Guidelines

Array exercises

1. Given an array with n elements, create a JAVA program which will determine the sum of the elements between the first even element and the last even element, including both of them.

- Sample Input:
  - n = 5
  - 7 6 1 2 8
- Sample Output: 17
- Solution:

```
    import java.util.Scanner;

    public class Application {

        public static void main(String[] args) {
            Scanner keyboard = new Scanner(System.in);
            System.out.print("Enter n: ");
            int n = keyboard.nextInt();
            int[] arr= new int[n];
            for (int i = 0; i < n; i++) {
                System.out.print("Enter a number: ");
                arr[i] = keyboard.nextInt();
            }
            int firstEvenNumberPos=-1, lastEvenNumberPos = -1, sum = 0;
            boolean foundFirst = false;

            for(int i =0; i < n; i++) {
                if(arr[i] % 2 == 0 && !foundFirst) {
                    firstEvenNumberPos = i;
                    lastEvenNumberPos = i;
                    foundFirst = true;
                } else if (arr[i] %2 == 0) {
                    lastEvenNumberPos = i;
                }
            }

            for(int i = firstEvenNumberPos; i <= lastEvenNumberPos; i++) {
                sum += arr[i];
            }

            if(firstEvenNumberPos == -1) {
                System.out.println("There are no even numbers");
            } else {
                System.out.println(sum);
            }
        }

    }
```

2. Given an array with n elements, create a JAVA program which will compute how many elements are strictly greater than the average of the elements from the array.

   - Sample Input:
     - n = 5;
     - 5 0 1 2 4
   - Sample Output: 2
   - Solution:

```
    import java.util.Scanner;

    public class Application {

        public static void main(String[] args) {
            Scanner keyboard = new Scanner(System.in);
            System.out.print("Enter n: ");
            int n = keyboard.nextInt();
            int[] arr= new int[n];
            int sum = 0, count = 0;

            for (int i = 0; i < n; i++) {
                System.out.print("Enter a number: ");
                arr[i] = keyboard.nextInt();
            }

            for(int i = 0; i < n; i++){
            sum += arr[i];
            }

            double average = sum / n;

            for(int i = 0; i < n; i++) {
            if(arr[i] > average) {
                count++;
            }
            }
            System.out.println(count);
        }
    }
```

3. Given an array with `n` elements, create a JAVA program which will determin how many elements are outside the closed interval determined by the first and last element

- Sample Input:
  - n = 6
  - 2 0.5 4 -1 -8 -3
- Sample Output: 2
- Solution:

```
import java.util.Scanner;

public class Application {

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Enter n: ");
        int n = keyboard.nextInt();
        double[] arr= new double[n];
        int count = 0;

        for (int i = 0; i < n; i++) {
            System.out.print("Enter a number: ");
            arr[i] = keyboard.nextDouble();
        }

        double firstElement = arr[0];
        double lastElement = arr[n-1];

        double startInterval = Math.min(firstElement, lastElement);
        double endInterval = Math.max(firstElement, lastElement);

        for(int i = 0; i < n;i++) {
            if( arr[i] > endInterval || arr[i] < startInterval) {
                count++;
            }
        }
        System.out.println(count);
    }

}
```

Matrix exercises

1. Let's consider a square matrix with N rows and N columns. In this matrix we have 4 areas:
    - 1, the zone which contains the elements which are strictly above the main diagonale and strictly above the second diagonale
    - 2, the zone which contains the elements strictly above the main diagonale and strictly below the second diagonale.
    - 3, the zone which contains the elements strictly below the main diagonale and strictly below the second diagonale.
    - 4, the zone which contains the elements strictly below the main diagonale and strictly above the second diagonale.

- Given a squared matrix and an integer number Z, which represent san area from the matrix, create a program which computes the sum of the elements in the Z area.
    - Sample Input:

      ```
          n = 5
          z = 2

          7 4 8 5 10
          7 7 10 2 2
          1 2 8 8 4
          9 9 5 3 2
          3 6 7 1 7
      ```

    - Sample Output: 16
    - Solution:

      ```
      import java.util.Scanner;

      public class Application {

          public static void main(String[] args) {
              Scanner keyboard = new Scanner(System.in);
              int[][] matrix = new int[][]{
                      {7,4,8,5,10},
                      {7,7,10,2,2},
                      {1,2,8,8,4},
                      {9,9,5,3,2},
                      {3,6,7,1,7}
              };

              System.out.print("Enter the zone for which you want to compute
      the sum of the elements: (1-4) ");
              int z = keyboard.nextInt();
              int sum =0;

              for(int i = 0; i < matrix.length; i++){
                  for(int j = 0; j < matrix.length; j++) {
      ```

```
                if(z == 1 && (j > i && j < matrix[i].length-i-1)) {
                    sum+= matrix[i][j];
                } else if (z == 2 &&  ( j > i && j > (matrix[i].length
-i-1) )) {
                    sum+= matrix[i][j];
                } else if (z == 3 && (i > j && j > (matrix[i].length-i-
1))) {
                    sum+= matrix[i][j];
                } else if (z == 4 && (i > j &&  j < (matrix[i].length-
i-1) )) {
                    sum+= matrix[i][j];
                }
            }
        }
        System.out.println("The sum of the elements in the zone #" + z
+ " is: " + sum);
    }

}
```

2. Given a 2D array with n rows and n columns which contains natural number, create a JAVA program which will compute the absolute difference between the sum of the elements on the main diagonale and the elements on the second diagonale.
    - Sample Input:

```
    n = 4
    1  2  3  4
    5  6  7  8
    9  10 81 12
    13 14 15 16
```

    - Sample Output: 70
    - Solution:

```
    import java.util.Scanner;

    public class Application {

        public static void main(String[] args) {
            int[][] matrix = new int[][]{
                {1,2,3,4},
                {5,6,7,8},
                {9,10,81,12},
                {13,14,15,16}
            };

            int sumMainDiagonale = 0;
            int sumSecondDiagonale = 0;
            for (int i = 0; i < 4; i++ ) {
                for (int j = 0; j < 4; j++) {
                    if (i == j) {
                        sumMainDiagonale += matrix[i][j];
                    } else if (j == (4-1-i)) {
                        sumSecondDiagonale += matrix[i][j];
                    }
                }
            }

            System.out.println("The difference between the main
diagonal and second diagonal is: " + Math.abs(sumMainDiagonale -
sumSecondDiagonale));
        }

    }
```

3. Given a 2D array with n rows and n columns, create a JAVA program which will build another matrix which will be symetric with respect to the main diagonale of the given matrix.

   - Sample Input:

```
    n = 4
    3 1 8 5
    7 8 5 1
    2 2 6 7
    9 8 1 3
```

   - Sample Output:

```
    3 7 2 9
    1 8 2 8
    8 5 6 1
    5 1 7 3
```

   - Solution:

```
    import java.util.Scanner;

    public class Application {

        public static void main(String[] args) {
            Scanner keyboard = new Scanner(System.in);
            System.out.println("Enter n: ");
            int n = keyboard.nextInt();
            int[][] matrix = new int[n][n];
            for(int i = 0; i < n; i++) {
                for(int j = 0; j < n; j++) {
                    System.out.println("Enter an element for
["+i+","+j+"]: ");
                    matrix[i][j] = keyboard.nextInt();
                }
            }

            int[][] result = new int[n][n];

            for (int i = 0; i < n; i++ ) {
                for (int j = 0; j < n; j++) {
                result[i][j] = matrix[j][i];
                }
            }

            for (int i = 0; i < result.length; i++ ) {
                for (int j = 0; j < result[i].length; j++) {
                    System.out.print(result[i][j] + " ");
                }
                System.out.println();
            }

        }

    }
```

4. Given a square matrix, create a JAVA program that will iterate clockwise the outside of the matrix.

   - Sample Input:

   ```
   n = 5
   1 2 3 4 5
   6 7 8 9 1
   2 3 4 5 6
   7 8 9 1 2
   3 4 5 6 7
   ```

   - Sample Output:

   ```
   1 2 3 4 5 1 6 2 7 6 5 4 3 7 2 6
   ```

   - Solution:

```java
import java.util.Scanner;

public class Application {

    public static void main(String[] args) {
        Scanner keyboard = new Scanner(System.in);
        System.out.println("Enter n: ");
        int n = keyboard.nextInt();

        int[][] matrix = new int[n][n];
        for(int i = 0; i < n; i++) {
            for(int j = 0; j < n; j++) {
                System.out.println("Enter an element for
["+i+","+j+"]: ");
                matrix[i][j] = keyboard.nextInt();
            }
        }

        for(int i = 0; i < n; i++) {
            System.out.print(matrix[0][i] + " ");
        }

        for (int i = 1; i < n; i++) {
            System.out.print(matrix[i][n-1] + " ");
        }

        for (int i = n-2; i >=0;i-- ){
        System.out.print(matrix[n-1][i] + " ");
        }

        for(int i = n-2; i > 0; i--) {
            System.out.print(matrix[i][0] + " ");
        }

    }

}
```

5. Given a matrix with n rows and n columns, create a JAVA program which will compute the sum of the elements which are on the two diagonals that are neighbor with the main diagonal.

- Sample Input:

```
n = 5
3 1 8 5 4
7 8 5 1 2
2 2 6 7 3
9 8 1 3 6
7 5 3 1 7
```

- Sample Output: 30
- Solution:

```java
    import java.util.Scanner;

    public class Application {

        public static void main(String[] args) {
            Scanner keyboard = new Scanner(System.in);
            System.out.println("Enter n: ");
            int n = keyboard.nextInt();

            int[][] matrix = new int[n][n];
            for(int i = 0; i < n; i++) {
                for(int j = 0; j < n; j++) {
                    System.out.println("Enter an element for ["+i+","+j+"]:
");

                    matrix[i][j] = keyboard.nextInt();
                }
            }

            int sum = 0;
            for(int i = 0; i < n; i++) {
                if(i == 0) {
                    sum += matrix[i][i+1];
                } else if (i == (n-1)) {
                    sum += matrix[n-1][n-2];
                } else {
                    sum += matrix[i][i-1];
                    sum += matrix[i][i+1];
                }
            }

            System.out.println(sum);

        }

    }
```

Homework exercises

1. Given an array with `n` elements of the form `a1, a2, ..., an`, natural numbers, Create a JAVA program which will:

   a. Display the elements of the array from the right to the left b. Compute the sum of the even elements c. Compute the sum of the elements on even indices d. Cmpute the number of the elements which are divisible with 10 e. Compute the sum of the numberd which are divisible with 3 and on odd indices.

Note: the program should display the output in the following way:

```
- On the first line, it will display the elements from the right to left

- On the second line, it will display the sum of the even elements - On the third
line, it will display the sum of the elements on even indices

- On the fourth line it will display how many elements are divisible with 10

- On the fifth line, it will display the sum of the numbers which are both
divisible with 3 and on odd indices
```

- Sample Input:
  - n = 10
  - 1 2 3 4 5 6 7 8 9 10
- Sample Output:

  ```
  10 9 8 7 6 5 4 3 2 1
  30
  30
  1
  12
  ```

2. Given an array with `n` rows and `n` columns and natural number elements, create a JAVA program which will Display, in ascending order, the sums of the elements in the four areas delimited by diagonals..
   - Sample Input:

     ```
     3 1 8 5 4
     7 8 5 1 2
     2 2 6 7 3
     9 8 1 3 6
     7 5 3 1 7
     ```

   - Sample Output:
     - `10 18 19 20`

Guidelines

- When solving the exercises, try to review what we have done because most of them are based on what we have already performed in the class.