Exercises with Arrays and Matrices

Objectives

- Recap previous session
- Exercises with arrays
- Exercises with matrices
- Homework exercises
- Guidelines

Recap previous session

- Give an example of a real life situation where sorting helps
- How can we swap the value of two variables?
- What is a multidimensional array?
- What is the difference between a simple array and a 2D array?
- How do you access an element from an array?
- How do you access an element from a 2D array?

Exercises with arrays

1. Given an array with n elements, create a JAVA program which will compute the absolute difference between the number of even numbers and number of odd numbers.

   - Sample Input:
     - n = 5
     - 15 245 28 33 11
   - Sample Output:
     - 3
   - 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[] numbers = new int[n];
        for(int i = 0; i < numbers.length; i++) {
            System.out.println("Enter a number: ");
            numbers[i] = keyboard.nextInt();
        }

        int numberOfEvenElements = 0;
        int numberOfOddElements = 0;
        for(int i = 0; i < numbers.length; i++) {
```

```
            if(numbers[i] % 2 == 0) {
                numberOfEvenElements++;
            } else {
                numberOfOddElements++;
            }
        }

        int difference = Math.abs(numberOfEvenElements -
    numberOfOddElements);
            System.out.println(difference);
        }
    }
```

2. Given an array with n elements, create a JAVA program which will display the elements which are multiples of the last element of the array.

   ○ Sample Input:

     ▪ n = 5
     ▪ 7 4 9 6 2

   ○ Sample Output: 4 6 2

   ○ 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[] numbers = new int[n];
        for(int i = 0; i < numbers.length; i++) {
            System.out.println("Enter a number: ");
            numbers[i] = keyboard.nextInt();
        }

        for(int i = 0; i < numbers.length; i++) {
            if(numbers[i] % numbers[numbers.length-1] == 0) {
                System.out.print(numbers[i] + " ");
            }
        }
    }
}
```

3. Given an array with n elements, create a JAVA program which will display the elements with even indices in the ascending order of the indices, and the elements with odd indices, in the descending order of the indices.

- Sample Input:

  - n = 5
  - 7 9 2 6 8

- Sample Output:

  - 7 2 8
  - 6 9

- 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[] numbers = new int[n];
        for(int i = 0; i < numbers.length; i++) {
            System.out.println("Enter a number: ");
            numbers[i] = keyboard.nextInt();
        }

        for(int i = 0; i < numbers.length; i++) {
            if(i % 2 == 0) {
                System.out.print(numbers[i] + " ");
            }
        }

        System.out.println();

        for(int i = numbers.length-1; i  > 0; i--) {
            if(i % 2 != 0) {
                System.out.print(numbers[i] + " ");
            }
        }
    }
}
```

4. Given an array with n elements, create a JAVA program which will display the elements of the array in the following order: first, last, second, second to last, etc

   - Sample Input:

     - n = 4
     - 2 9 1 5 8

   - Sample Output:

- 2 8 9 5 1

- 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[] numbers = new int[n];
        for(int i = 0; i < numbers.length; i++) {
            System.out.println("Enter a number: ");
            numbers[i] = keyboard.nextInt();
        }

        int lastIndex = numbers.length-1;

        for(int startIndex = 0; startIndex < lastIndex;
startIndex++, lastIndex--) {
            System.out.print(numbers[startIndex] + " " +
numbers[lastIndex] + " ");
        }

        if(numbers.length % 2 != 0)  {
            System.out.print(numbers[numbers.length/2]);
        }
    }
}
```

5. Given an array with n elements, create a JAVA program which will display the numbers between the element with the minimum value and the element with the maximum value, including both of them (closed interval)

   - Sample Input:

     - n = 5
     - 7 9 6 2 8

   - Sample Output:

     - 9 6 2

   - Solution:

```
import java.util.Scanner;

public class Application {
```

```java
        public static void main(String[] args) {
            Scanner keyboard = new Scanner(System.in);
            System.out.println("Enter n: ");
            int n = keyboard.nextInt();
            int[] numbers = new int[n];
            for(int i = 0; i < numbers.length; i++) {
                System.out.println("Enter a number: ");
                numbers[i] = keyboard.nextInt();
            }

            int minPos = findMinPos(numbers);
            int maxPos = findMaxPos(numbers);
            displaySection(numbers, minPos, maxPos);
        }

        private static void displaySection(int[] numbers, int minPos,
    int maxPos) {
            int startIndex = Math.min(minPos, maxPos);
            int endIndex = Math.max(minPos, maxPos);

            for(int i = startIndex; i <= endIndex; i++) {
                System.out.print(numbers[i] + " ");
            }

        }

        private static int findMinPos(int[] numbers) {
            int minPos = 0;
            for(int i =0; i < numbers.length; i++) {
                if(numbers[i] < numbers[minPos]) {
                    minPos = i;
                }
            }
            return minPos;
        }

        private static int findMaxPos(int[] numbers) {
            int maxPos = 0;
            for(int i =0; i < numbers.length; i++) {
                if(numbers[i] > numbers[maxPos]) {
                    maxPos = i;
                }
            }
            return maxPos;
        }
    }
```

Exercises with matrices

1. Write a JAVA program which will replace each element from the main diagonale with the average of its neighbors

○ Sample Input:

```
12 13 21 17
8  9  15 4
2  3  7  9
21 24 29 18
```

○ Sample Output:

```
10 13 21 17
8 9 15 4
2 3 14 9
21 24 29 19
```

○ Solution:

```java
public class Application {

    public static void main(String[] args) {
        int [][] matrixA = new int[][]{
                {12, 13,  21, 17},
                {8,   9,  15,  4},
                {2,   3,  7,  9},
                {21,  24, 29, 18}
        };

        for(int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
                if(i == j) {
                    if(i == 0 && j == 0) {
                        matrixA[i][j] = computeAverage(matrixA[0]
[1], matrixA[1][0]);
                    } else if (i==3 && j == 3) {
                        matrixA[i][j] = computeAverage(matrixA[3]
[2], matrixA[2][3]);
                    } else {
                        matrixA[i][j] = computeAverage(matrixA[i-1]
[j], matrixA[i][j-1], matrixA[i+1][j], matrixA[i][j+1]);
                    }
                }
            }
        }

        for(int i = 0; i < 4; i++) {
            for(int j = 0; j < 4; j++) {
                System.out.print(matrixA[i][j] +  " ");
            }
            System.out.println();
```

```java
            }
        }

        public static int computeAverage(int a, int b) {
            return (a+b) /2;
        }

        public static int computeAverage(int a, int b, int c, int d) {
            return ( a + b + c + d ) / 4;
        }
    }
```

2. Write a JAVA program which will replace the main diagonale with the second diagonale.

- Sample Input:

```
12 13 21 17
8  9  15 4
2  3  7  9
21 24 29 18
```

- Sample Output:

```
17 13 21 12
8  15 9  4
2  7  3  9
18 24 29 21
```

- Solution:

```java
    public class Application {

        public static void main(String[] args) {
            int [][] matrixA = new int[][]{
                    {12, 13,  21, 17},
                    {8,   9,  15,  4},
                    {2,   3,  7,  9},
                    {21,  24, 29, 18}
            };

            for(int i = 0; i < matrixA.length; i++) {
                int temp = matrixA[i][i];
                matrixA[i][i] =  matrixA[i][matrixA.length - i-1];
                matrixA[i][matrixA.length-i-1] = temp;
            }

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

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

3. Write a JAVA program which will multiply a scalar with a two dimensional matrix

  ○ The theory says that the result will be a matrix where each element is the element from the first matrix, multiplied with the scalar.

  ○ Sample Input:

```
Matrix = 2 9 0
         1 3 5
         2 4 7
         8 1 5
Scalar = 4
```

  ○ Sample Output:

```
8 36 0
4 12 20
8 16 28
32 4 20
```

  ○ Solution:

```java
public class Application {

    public static void main(String[] args) {
        int [][] matrixA = new int[][]{
            {2, 9, 0},
            {1, 3, 5},
            {2, 4, 7},
            {8, 1, 5}
        };

        int scalar = 4;

        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 3; j++) {
                matrixA[i][j] *= scalar;
            }
        }
```

```java
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 3; j++) {
                System.out.print(matrixA[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

4. Write a JAVA program to multiply 2-dimensional arrays one by the other. This is also called matrix multiplication.

   o Theory:

      ▪ Make sure that the number of columns in the 1st matrix, is equal to the number of rows in the 2nd matrix
      ▪ Multiply the elements of each row of the first matrix by the elements of each column in the second matrix
      ▪ Add the products as follows, considering the matrix from the example:
         ▪ (3*2 + 2*1 + 1*2 + 5*8) => This will be result[0][0];
         ▪ (3*9 + 2*3 + 1*4 + 5*1) => This will be result[0][1];
         ▪ (3*0 + 2*5 + 1*7 + 5*5) => This will be result[0][2];
         ▪ (9*2 + 1*1 + 3*2 + 0*8) => This will be result[1][0];
         ▪ (9*9 + 1*3 + 3*4 + 0*1) => This will be result[1][1];
         ▪ (9*0 + 1*5 + 3*7 + 0*5) => This will be result[1][2];
      ▪ The resulting matrix has M rows X N columns where M is the number of rows of the first matrix and N is the number of columns of the second matrix

   o Sample Input:

```
3 2 1 5    2 9 0
9 1 3 0    1 3 5
           2 4 7
           8 1 5
```

   o Sample Output:

```
50 42 42
25 96 26
```

   o Solution:

```java
public class Application {
```

```java
        public static void main(String[] args) {
            int [][] matrixA = new int[][]{
                {3,2,1,5},
                {9,1,3,0}
            };

            int [][] matrixB = new int[][]{
                {2,9,0},
                {1,3,5},
                {2,4,7},
                {8,1,5}
            };
```

```java
        int [][] resultMatrix = new int[2][3];

        for(int i = 0; i < 2; i++) {
            for (int j = 0; j < 3; j++) {
                resultMatrix[i][j] = 0;
                for (int k = 0; k < 4; k++) {
                    resultMatrix[i][j] += matrixA[i][k]* matrixB[k][j];
                }
            }
        }

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

5. Write a JAVA program which adds two matrices of same dimensions (Same number of rows and columns)

   o Theory: We should add the elements which are on the same position and put the result back in the resulting matrix

   o Sample Input:

```
3 2 1      2 9 0
9 1 3      1 3 5
2 6 11     2 4 7
```

- Sample Output:

```
5   11 1
10 4   8
4   10 18
```

- Solution:

```java
public class Application {

    public static void main(String[] args) {
        int [][] matrixA = new int[][]{
            {3, 2, 1},
            {9, 1, 3},
            {2,6,11}
        };

        int [][] matrixB = new int[][]{
            {2,9,0},
            {1,3,5},
            {2,4,7}
        };

        int [][] resultMatrix = new int[3][3];

        for(int i = 0; i < 3; i++) {
            for(int j = 0; j < 3; j++) {
                resultMatrix[i][j] = matrixA[i][j] + matrixB[i][j];
            }
        }

        for(int i = 0; i < 3; i++) {
            for(int j = 0; j < 3; j++) {
                System.out.print(resultMatrix[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Homework exercises

1. Create a JAVA program which computes the sum of all even numbers in a matrix.

   - Sample Input:

```
2 9 0
1 3 5
2 4 7
```

- Sample Output:
    - 8

2. Create a JAVA program which traverses a matrix and in each cell it places the maximum between the index of the row and the index of the column

    - Sample Input:

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

    - Sample Output:

```
0 1 2 3
1 1 2 3
2 2 2 3
3 3 3 3
4 4 4 4
```

3. Write a JAVA program which will replace each element from the second diagonale with the average of its neighbors

    - Sample Input:

```
12 13 21 17
8  9  15 4
2  3  7  9
21 24 29 18
```

    - Sample Output:

```
12 13 21 12
8  9  10 4
2  10 7  9
13 24 29 18
```

4. Write a JAVA program which will subtract two matrices of same dimensions (Same number of rows and columns)

- Sample Input:

```
3 2 1    2 9 0
9 1 3    1 3 5
2 6 11   2 4 7
```

- Sample Output:

```
1 -7  1
8 -2 -2
0  2  4
```

5. Write a JAVA program which will determine if two matrices are equal. Theory sais that two matrices are equal if and only if they have the same dimensions and same elements.

- Sample Input:

```
3 2 1    2 9 0
9 1 3    1 3 5
2 6 11   2 4 7
```

- Sample Output: `false`
- Sample Input2:

```
0 0    0 0 0
0 0    0 0 0
0 0    0 0 0
```

- Sample Output2: `false`
- Sample Input2:

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

- Sample Output2: `true`

6. Write a C++ program which will divide a matrix by a scalar. Note that you can only divide a matrix by a scalar! You cannot divide two matrices!

- Sample Input:

```
        2.0 9.0 7.0
4.0     1.0 3.0 5.0
        2.0 4.0 7.0
```

- Sample Output:

```
0.5     2.25    1.75
0.25    0.75    1.25
0.5     1       1.75
```

Guidelines

- Each recap section contains topics that we discussed in previous meetings
- It is best that you answer them and write them down, not to learn it by hard but it will speed the process of learning due to visualization.
- Everything that is new, should be noted and maybe discussed in the following session(s) in case if it is not well understood
- Of course, at each session, you can choose to speak about a certain topic that you are interested in.