In a one-dimensional array, the array elements are specified using one index.

Example:
```java
int[] primes = {2, 3, 5, 7, 11, 13};
```

The following code prints all the components of the array:
```java
for(int i = 0; i < primes.length; i++)
    System.out.print(primes[i] + " ");
System.out.println();
```

A two-dimensional array is an array where its elements are specified using two indices.

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

The following code prints all the components of the array:
```java
for(int i=0; i < x.length; i++)
    for(int j=0; j < x[i].length; j++){
        System.out.print(x[i][j] + " ");
    }
System.out.println();
```

In Java, multi-dimensional arrays are really arrays of arrays. Thus the following is permissible.

```java
int[][] n = {{1,2,3},{4,5},{6,7,8,9}};
```

The number of columns vary; `n[row].length` stores the number of columns. The value in `n[0][1]` is 2, and the value in `n[2][3]` is 9.
Array Declaration:

Declare an array object reference variable referring to a 2-dimensional array.

```java
double[][] a;
```

Create the array and store the location of the first element of the array in the array object reference variable.

```java
a = new double[3][4]; // defines an array with 3 rows and 4 columns
```

The following declares and initializes a 2-dimensional array.

```java
int[][] a = {{0,1,2,3}, {4,5,6,7}, {8,9,0,1}};
```

The following nested loop traverses a 2-dimensional array.

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

The following declares and initializes a 3-dimensional array.

```java
int[][][] a = { { {0,1},{2,3} }, { {4,5},{6,7} } };
```

`a` stores a reference to an array of 2 items, where each item is a reference to an array of 2 items.

The following nested loop traverses a 3-dimensional array.

```java
for(int i=0; i < a.length; i++){
    for(int j=0; j < a[i].length; j++){
        for(int k=0; k < a[i][j].length; k++){
            System.out.print(a[i][j][k] + " ");
        }
        System.out.println();
    }
    System.out.println();
}
```
The purpose of this lab is to learn how to create and use two-dimensional arrays in Java. In this exercise, you will create a matrix class, as specified below, and create a driver class to test the methods created in the matrix class.

A matrix is a collection of numbers arranged in a rectangular array with a fixed number of rows and columns – an $m \times n$ array. Arithmetic operations such as addition and multiplication are defined for matrices. For example, two matrices may be added or multiplied together to yield a new matrix.

**Matrix Addition and Subtraction:**

Given $A = (a_{ij})$ and $B = (b_{ij})$ to be $m \times n$ matrices. We define their *sum*, denoted by $A+B$, and their *difference*, denoted by $A-B$, to be the respective matrices $(a_{ij}+b_{ij})$ and $(a_{ij}-b_{ij})$. When two matrices have the same dimensions, we just add or subtract their corresponding entries.

Example:

Given $A = \begin{bmatrix} 1 & 2 \\ 5 & 3 \end{bmatrix}$ and $B = \begin{bmatrix} 3 & 2 \\ 4 & 5 \end{bmatrix}$

$A + B = \begin{bmatrix} 1 & 2 \\ 5 & 3 \end{bmatrix} + \begin{bmatrix} 3 & 2 \\ 4 & 5 \end{bmatrix} = \begin{bmatrix} 1+3 & 2+2 \\ 5+4 & 3+5 \end{bmatrix} = \begin{bmatrix} 4 & 4 \\ 9 & 8 \end{bmatrix}$

$A - B = \begin{bmatrix} 1 & 2 \\ 5 & 3 \end{bmatrix} - \begin{bmatrix} 3 & 2 \\ 4 & 5 \end{bmatrix} = \begin{bmatrix} 1-3 & 2-2 \\ 5-4 & 3-5 \end{bmatrix} = \begin{bmatrix} -2 & 0 \\ 1 & -2 \end{bmatrix}$

**Matrix Multiplication:**

To multiply two matrices, Matrix $A$ must have the same number of columns as the rows in Matrix $B$. If $A$ is an $m \times p$ matrix and $B$ is a $p \times n$ matrix. We define their product, denoted by $AB$, to be the $m \times n$ matrix whose $ij$-entry, $1 \leq i \leq m$ and $1 \leq j \leq n$, is the product of the $i$-th row of $A$ and the $j$-th column of $B$.

Example:

Given $A = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$ and $B = \begin{bmatrix} 3 & 2 \\ 4 & 5 \end{bmatrix}$

$AB = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} \times \begin{bmatrix} 3 & 2 \\ 4 & 5 \end{bmatrix} = \begin{bmatrix} 1(3) + 2(4) & 1(2) + 2(5) \\ 4(3) + 3(4) & 4(2) + 3(5) \end{bmatrix} = \begin{bmatrix} 11 & 12 \\ 24 & 23 \end{bmatrix}$
Scalar Multiplication:
Scalar multiplication is defined by, for any \( r \in \mathbb{R} \), \( rA \) is the matrix \( (ra_{ij}) \). The scalar multiplication \( rA \) of a matrix \( A \) and a number \( r \) is given by multiplying every entry of \( A \) by \( r \).

Example:
Given \( A = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} \)

\[
2A = 2 \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} = \begin{bmatrix} 2 \times 1 & 2 \times 2 \\ 2 \times 4 & 2 \times 3 \end{bmatrix} = \begin{bmatrix} 2 & 4 \\ 8 & 6 \end{bmatrix}
\]

Transpose a Matrix:
Given \( A = (a_{ij}) \) to be an \( m \times n \) matrix. The transpose of \( A \), denoted by \( A^T \), is the matrix whose \( i \)-th column is the \( i \)-th row of \( A \), or equivalently, whose \( j \)-th row is the \( j \)-th column of \( A \). Notice that \( A^T \) is an \( n \times m \) matrix. We write \( A^T = (a_{ji}) \) where \( a_{ji} = a_{ij} \).

Examples:
Suppose \( A = \begin{bmatrix} 1 & 2 \\ 5 & 3 \end{bmatrix} \) then \( A^T \) is: \( \begin{bmatrix} 1 & 5 \\ 2 & 3 \end{bmatrix} \)

Suppose \( A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \) then \( A^T \) is: \( \begin{bmatrix} 1 & 2 \\ 4 & 5 \\ 3 & 6 \end{bmatrix} \)

Equality.
Two matrices \( A = (a_{ij}) \) and \( B = (b_{ij}) \) are equal, written \( A = B \), provided they have the same size and their corresponding entries are equal, that is, their sizes are both \( m \times n \) and for each \( 1 \leq i \leq m \) and \( 1 \leq j \leq n \), \( a_{ij} = b_{ij} \).

Examples:
Given \( A = \begin{bmatrix} 1 & 2 \\ 5 & 3 \end{bmatrix} \) and \( B = \begin{bmatrix} 1 & 2 \\ 5 & 3 \end{bmatrix} \) then \( A = B \) is true

Given \( A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \) and \( B = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \) then \( A = B \) is false
Instructions

Part 1

1. Create a Java Project named Lab1.
2. Create a package named lab1_YourLastName_FirstInitial.
3. Import the Matrix class that implements addition, subtraction, multiplication, scalar multiplication, equal, and transpose methods for m x n matrices. Copy, paste, and modify the template from Lab 0 to the Matrix class. The class should also include a constructor Matrix(int[][] m), and a toString method. Add another class (driver class with main method) to test those methods.

Part 2

Write Java code for the following and create a word document with the answers. Test whether or not the following are true by writing the necessary statements in the driver class:

1. $(A^T)^T = A$
2. $(A + B)^T = A^T + B^T$
3. $(rA)^T = rA^T$
4. $(AB)^T = B^T A^T$
5. $AB \neq BA$
6. $A(BC) = (AB)C$
7. $A(B+C) = AB + AC$
8. $(A+B)C = AC + BC$
9. $(rA)B = r(AB) = A(rB)$

// Java code to verify (A+B)C = AC + BC
Matrix a = new Matrix(new int[][]{{1,2},{2,0}});
Matrix b = new Matrix(new int[][]{{1,2},{2,0}});
Matrix c = new Matrix(new int[][]{{1,2},{2,0}});
System.out.println(a.add(b).mult(c));
System.out.println(a.mult(c).add(b.mult(c)));

For the following, write the java code in the main method.

Given
\[ A= \begin{bmatrix} 1 & -2 & 3 \\ 1 & -1 & 0 \end{bmatrix} \quad B= \begin{bmatrix} 3 & 4 \\ 5 & -1 \\ 1 & -1 \end{bmatrix} \quad C= \begin{bmatrix} 4 & -1 & 2 \\ 5 & 1 \end{bmatrix} \quad D= \begin{bmatrix} -1 & 0 & 1 \\ 0 & 2 & 1 \end{bmatrix} \]

\[ E= \begin{bmatrix} 3 & 4 \\ -2 & 3 \\ 0 & 1 \end{bmatrix} \quad F= \begin{bmatrix} 2 \\ -3 \end{bmatrix} \quad G= \begin{bmatrix} 2 & -1 \end{bmatrix} \]

Compute each of the following and simplify, whenever possible. If a computation is not possible, state why.

(a) $3C - 4D$  (b) $A - (D + 2C)$  (c) $A - E$  (d) $AE$
(e) $3BC - 4BD$  
(f) $CB + D$  
(g) $GC$  
(h) $FG$

(i) $C^2$  
(j) $C + D$