Lab #5: 2-D Arrays

The main aim of the lab is to continue dealing with some selected problems using **2-D arrays** and their applications for **Tic Tac Toe game**.

(Deadline: 23h59 23/10/2023)

===

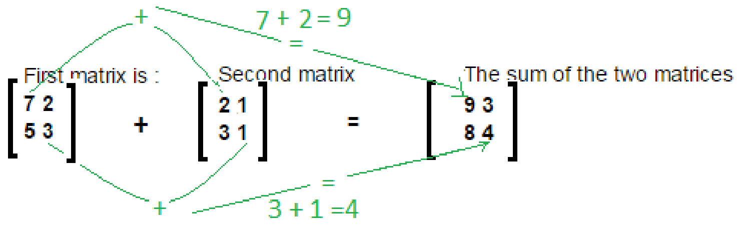
Task 1: Basic Problems

===

Task 1.1: Implement the following method for adding 2 matrices.

```
// add 2 matrices
	public static int[][] add(int[][] a, int[][] b) {
		// TODO
		return null;

	}
```

Example:

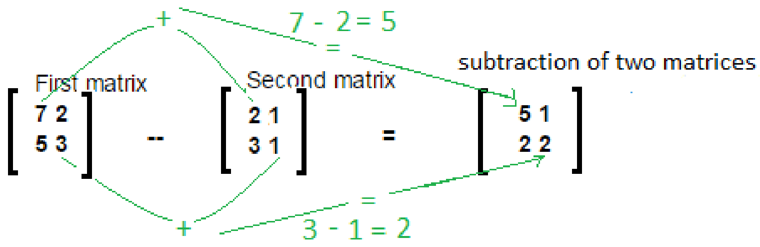


===

Task 1.2: Implement the following method for subtracting 2 matrices.

```
// subtract 2 matrices
	public static int[][]subtract(int[][] a, int[][] b) {
		// TODO
		return null;

	}
```

Example:



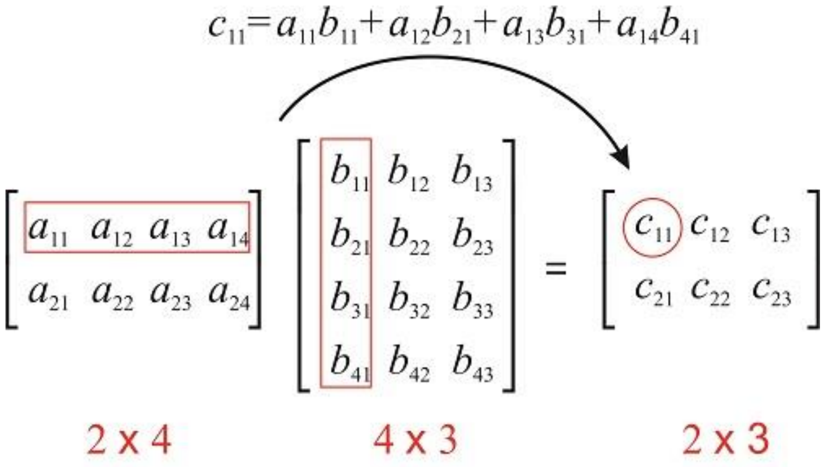
$$\begin{bmatrix}7&2\\5&3\end{bmatrix} - \begin{bmatrix}2&1\\3&1\end{bmatrix} = \begin{bmatrix}5&1\\2&2\end{bmatrix}$$

===

Task 1.3: Implement the following method for multiplying 2 matrices.

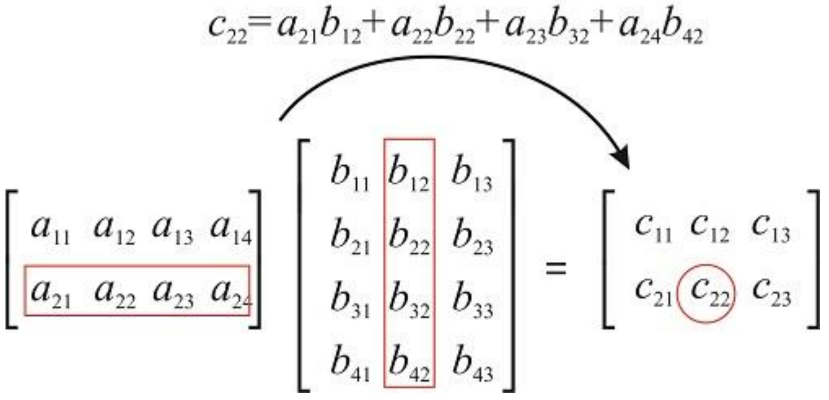
```
// multiply 2  matrices
public static int[][] multiply(int[][] a, int[][] b) {
     // TODO
     return null;

}
```

Example:



$$c_{11}=a_{11}b_{11}+a_{12}b_{21}+a_{13}b_{31}+a_{14}b_{41}$$

$$\begin{bmatrix}a_{11}&a_{12}&a_{13}&a_{14}\\a_{21}&a_{22}&a_{23}&a_{24}\end{bmatrix}\begin{bmatrix}b_{11}&b_{12}&b_{13}\\b_{21}&b_{22}&b_{23}\\b_{31}&b_{32}&b_{33}\\b_{41}&b_{42}&b_{43}\end{bmatrix}=\begin{bmatrix}c_{11}&c_{12}&c_{13}\\c_{21}&c_{22}&c_{23}\end{bmatrix}$$



$$c_{22}=a_{21}b_{12}+a_{22}b_{22}+a_{23}b_{32}+a_{24}b_{42}$$

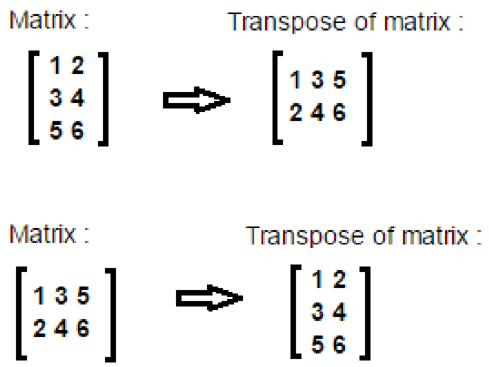
$$\begin{bmatrix}a_{11}&a_{12}&a_{13}&a_{14}\\a_{21}&a_{22}&a_{23}&a_{24}\end{bmatrix}\begin{bmatrix}b_{11}&b_{12}&b_{13}\\b_{21}&b_{22}&b_{23}\\b_{31}&b_{32}&b_{33}\\b_{41}&b_{42}&b_{43}\end{bmatrix}=\begin{bmatrix}c_{11}&c_{12}&c_{13}\\c_{21}&c_{22}&c_{23}\end{bmatrix}$$

===

Task 1.4: Implement the following method for transposing a given matrix:

```
// tranpose a matrix
public static int[][] transpose(int[][] a) {
        // TODO
        return null;

        }
```

Example:



Matrix :

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix} \Rightarrow$$

Transpose of matrix :

$$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix}$$

Matrix :

$$\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 6 \end{bmatrix} \Rightarrow$$

Transpose of matrix :

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

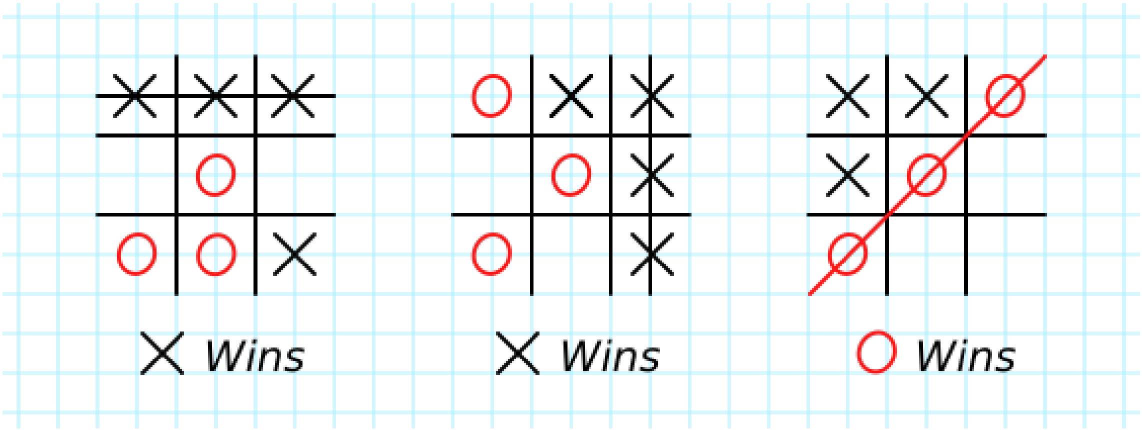
==

Task 2: Application of 2D Arrays

==

TIC TAC TOE Game

The board is an ***3 x 3*** matrix containing symbols **'X'**, **'O'**, or an empty char (' ').



For a given TicTacToe class as follows:

```
public class TicTacToe {
     private static final char EMPTY = ' ';

     private char[][] board;

//…
}
```

Task 2.1. Implement the following method to check whether a player wins or not based on checking **rows**?

```
/*
      * This method checks all rows and returns true if any of
them are marked with
      * all of a single player's markers.
      * Otherwise, returns false.
      */
     public boolean checkRows() {
          //TODO
          return false;
     }
```

Task 2.2. Implement the following method to check whether a player wins or not based on checking **columns**?

```
/*
     * This method checks all columns and returns true if any
of them are marked
      * with all of a single player's.
      * Otherwise, returns false.
      */
     public boolean checkColumns() {
          //TODO
          return false;
     }
```

Task 2.3. Implement the following method to check whether a player wins or not based on checking **diagonals**?

```
/*
      * This method checks both diagonals and returns true if
any of them are marked
      * with all of a single player's markers. Otherwise,
returns false.
      */
     public boolean checkDiagonals() {
          // Check top-left to bottom-right
```

```
        //TODO

        // Check bottom-left to top-right
        //TODO
        return false;

    }
```

Task 2.4 (advanced): Expand the implemented methods for handling a board with ***n x n*** matrix.