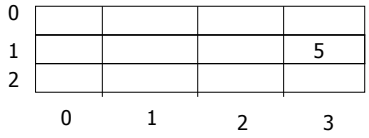
Two dimensional arrays. Applications with matrices

- Multidimensional arrays can be described as arrays of arrays.
- For example, a bidimensional array can be imagined as a bidimensional table of a uniform concrete data *type*

- Declaring matrices

```
int m1[][] = new int[3][3];
int m2[][] = {{1,2,3}, {4,5,6}};
```

- int jimmy [3][4];



jimmy[1][3] = 5

Print content of a Matrix

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


Method that prints content of a Matrix

```
public class Matrix {
    public static void main(String args[]) {
        int m[][] = { {7, 2},
                      {1, 5},
                      {8, 6} };
        printMatrix(m);
    }
    static void printMatrix(int m[][]) {
        for(int i=0; i<m.length; i++)
            for(int j=0; j<m[0].length; j++)
                System.out.print(m[i][j] + " " );
    }
}
```


Matrix multiplication

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ a_{21} & a_{22} & & a_{2m} \\ \vdots & & & \vdots \\ a_{n1} & a_{n2} & & a_{nm} \end{pmatrix} \qquad B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1k} \\ b_{21} & b_{22} & & b_{2k} \\ \vdots & & & \vdots \\ b_{j1} & b_{j2} & & b_{jk} \end{pmatrix}$$

m=j

$$A\ B = \begin{pmatrix} a_{11}xb_{11}+a_{12}xb_{21}+..+a_{1m}xb_{j1} & a_{11}xb_{12}+a_{12}xb_{22}+..+a_{1m}xb_{j2} & a_{11}xb_{1k}+a_{12}xb_{2k}+..+a_{1m}xb_{jk} \\ a_{21}xb_{12}+a_{22}xb_{22}+..+a_{2m}xb_{j2} & a_{21}xb_{12}+a_{22}xb_{22}+..+a_{2m}xb_{j2} & a_{21}xb_{12}+a_{22}xb_{22}+..+a_{2m}xb_{j2} \\ \vdots & \vdots & \\ a_{n1}xb_{1k}+a_{n2}xb_{2k}+..+a_{nm}xb_{jk} & a_{n1}xb_{1k}+a_{n2}xb_{2k}+..+a_{nm}xb_{jk} & a_{n1}xb_{1k}+a_{n2}xb_{2k}+..+a_{nm}xb_{jk} \end{pmatrix}$$

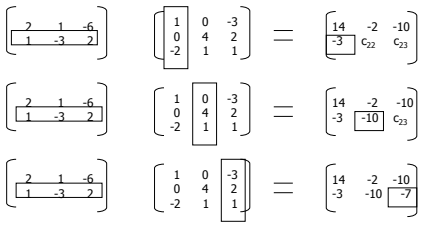
Example

$$\begin{pmatrix} \boxed{2} & \boxed{1} & \boxed{-6} \\ 1 & -3 & 2 \end{pmatrix} \begin{pmatrix} \boxed{1} & 0 & -3 \\ \boxed{0} & 4 & 2 \\ \boxed{-2} & 1 & 1 \end{pmatrix} = \begin{pmatrix} \boxed{14} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \end{pmatrix}$$

$$\begin{pmatrix} \boxed{2} & \boxed{1} & \boxed{-6} \\ 1 & -3 & 2 \end{pmatrix} \begin{pmatrix} 1 & \boxed{0} & -3 \\ 0 & \boxed{4} & 2 \\ -2 & \boxed{1} & 1 \end{pmatrix} = \begin{pmatrix} 14 & \boxed{-2} & c_{13} \\ c_{21} & c_{22} & c_{23} \end{pmatrix}$$

$$\begin{pmatrix} \boxed{2} & \boxed{1} & \boxed{-6} \\ 1 & -3 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 & \boxed{-3} \\ 0 & 4 & \boxed{2} \\ -2 & 1 & \boxed{1} \end{pmatrix} = \begin{pmatrix} 14 & -2 & \boxed{-10} \\ c_{21} & c_{22} & c_{23} \end{pmatrix}$$

Example



- Code that multiplies matrices

```
for(int i=0; i<a.length; i++) // matrix rows (vertical)
        for(int j=0; j<b[0].length; j++) // matrix columns (horizontal)
            for(int k=0; k<b.length; k++)
                r[i][j] = r[i][j] + a[i][k]* b[k][j];
```

- Activity:
  - Make a program that populates a matrix with the following values and prints them out. Each value in a row is printed in a row separated by comas:

```
1  2  3  4  5
3  4  6  8  10
3  6  9  12 15
```


- Activity:
  - Make a program that prints out the greatest number in the matrix

```
int m[3][5] = {{1,2,3,4,5},
               {2,4,6,8,10},
               {3,6,9,12,15}};
```

- Activity:
  - Make a program that sums up all the odd numbers in the matrix

```
int m[3][5] = {{1,2,3,4,5},
               {2,4,6,8,10},
               {3,6,9,12,15}};
```

s11

- Activity:
  - Make a program that multiplies a matrix by a vector

```
int r[] = new int[3];
      int v[] = {1, 2};
      int m[][] = { {7, 2},
                    {1, 5},
                    {8, 6} };

static void printVector(int v[]) {}
static void printMatrix(int m[][]) {}
static void matrixXvector(int m[][], int v[], int r[]) {}
```

s11

- Activity:
  - Make a program that multiplies the following matrices. Each value in a row is printed in a row separated by comas :

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

int b[2][3] = { {2, 5, 8},
                {3, 4, 9} };

static void printMatrix(int m[][]) {}
static void matrixXmatrix(int a[][], int b[], int r[][]) {}
```

s12