

Grid Lock

2D array creation and access



Three workers, three arrays

PAUSE AND TRY

The shop has 30 workers now. What goes wrong with this plan?

HOURS WORKED, MON TO WED

```
int[] ana = {6, 8, 5};  
int[] ben = {7, 4, 9};  
int[] cy = {3, 6, 6};
```

Each worker gets an array of hours.
What would you need for 30 workers?
For the whole shop in one variable?

By the end of class you can...

- 1 Explain how a 2D array is an array of arrays, and picture it as rows and columns.
- 2 Create a 2D array with an initializer list or with new, and name the default values.
- 3 Read and change one element with `grid[row][col]`.
- 4 Use `grid[r]` as a whole row, a 1D array that is shared with the grid.
- 5 Find the rows and columns with `length`, and predict an `ArrayIndexOutOfBoundsException`.

A large white square with rounded corners, tilted slightly to the right, containing a large, bold, dark teal number '1'.

PART 1

An array of arrays

KEY IDEA

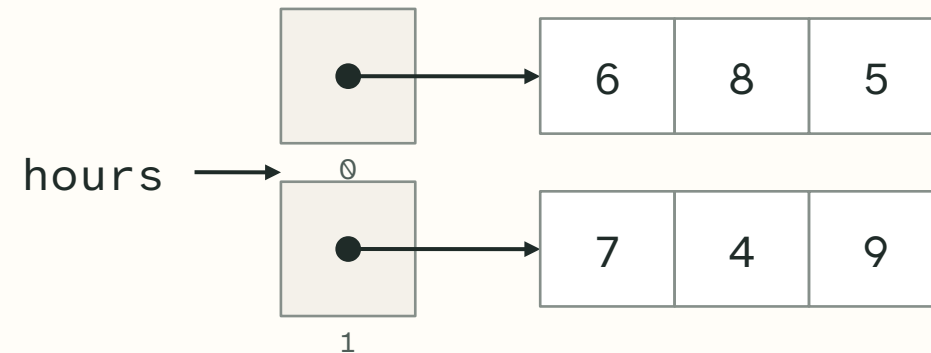
Two pictures of one 2D array

```
int[][] hours = { {6, 8, 5},  
                  {7, 4, 9} };
```

AS A GRID

	col 0	col 1	col 2
row 0	6	8	5
row 1	7	4	9

AS JAVA STORES IT



A **2D array** is an outer array whose elements are 1D arrays. Each inner array is a **row**.

Reading an initializer list

BUS.JAVA

```
String[][] bus = { {"Ana", "Bo"},  
                  {"Cam", "Dot"},  
                  {"Eve", "Fin"} };  
  
System.out.println(bus[2][0]);  
System.out.println(bus[0][1]);
```

OUTPUT

Eve
Bo

Piece	Means
String[][]	array of String[] rows
inner lists: 3	3 rows
values per list: 2	2 columns



Size is always said **rows first**: this is a 3-by-2 array.

Two levels of braces need [][]



The type `int[]` holds ints. A 2D initializer list holds rows, so the type must be `int[][]`.

```
int[] hours = { {6, 8, 5}, {7, 4, 9} };
```

COMPILER SAYS

```
WrongDims.java:3: error: illegal initializer for int
```

```
int[][] hours = { {6, 8, 5}, {7, 4, 9} };
```

FIX



2

PART 2

One box at a time

grid[row][col] reads and changes one box

HOURS.JAVA

```
int[][] hours = { {6, 8, 5},  
                  {7, 4, 9} };  
System.out.println(hours[1][2]);  
System.out.println(hours[0][1]);  
hours[1][1] += 3;  
hours[0][0] = hours[1][1] * 2;  
System.out.println(hours[0][0]);
```

OUTPUT

```
9  
8  
14
```

AFTER THE CHANGES

	col 0	col 1	col 2
row 0	14	8	5
row 1	7	7	9



The **first** index picks the row.
The **second** picks the column.
Both count from 0.

Swapped indices: wrong box, no error



On a square grid, **grid[col][row]** still runs. It just reads the wrong box.

LOCKERS.JAVA

```
int[][] lockers = { {11, 12, 13},  
                    {21, 22, 23},  
                    {31, 32, 33} };  
  
// Want: row 2, column 0  
System.out.println(lockers[0][2]);
```

OUTPUT

13

Fix: lockers[2][0]
prints 31

	col 0	col 1	col 2
row 0	11	12	13
row 1	21	22	23
row 2	31	32	33

Update the grid, then print

PAUSE AND TRY

Draw the grid, update it line by line, and write the output.

```
int[][] p = { {2, 5},  
              {8, 1},  
              {4, 7} };  
p[2][0] = p[0][1] + p[1][1];  
p[1][0] -= 3;  
System.out.println(p[2][0]);  
System.out.println(p[1][0] + p[0][0]);
```

Draw it here: 3 rows, 2 columns.

	col 0	col 1
row 0		
row 1		
row 2		

A large, white, rounded square containing a bold, teal number '3'.

3

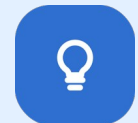
PART 3

new and default values

new type[rows][cols] fills in defaults

DEFAULTS.JAVA

```
int[][] a = new int[2][4];
double[][] b = new double[3][3];
boolean[][] c = new boolean[2][2];
String[][] d = new String[4][2];
System.out.println(a[1][3]);
System.out.println(b[0][0]);
System.out.println(c[1][0]);
System.out.println(d[3][1]);
```



The first size is the rows. The size is set at creation and **cannot change**.

Element type	Default
int	0
double	0.0
boolean	false
String (any reference)	null

OUTPUT

```
0
0.0
false
null
```

An empty String box holds null

CREW.JAVA

```
String[][] crew = new String[2][3];  
crew[0][0] = "Max";  
System.out.println(crew[0][0].length());  
System.out.println(crew[1][2].length());
```

OUTPUT

```
3  
Exception in thread "main"  
java.lang.NullPointerException
```

THE FIX

```
String name = crew[1][2];  
if (name != null) {  
    System.out.println(name.length());  
} else {  
    System.out.println("Seat is empty");  
}
```

OUTPUT

```
Seat is empty
```



Students expect "", but **new** makes no String objects. Check **!= null** before calling a method.

A large, white, rounded square containing a bold, teal number '4'.

4

PART 4

Rows and lengths

grid[r] is a whole row, and it is shared

MONDAY.JAVA

```
int[][] hours = { {6, 8, 5},  
                  {7, 4, 9} };  
  
int[] mon = hours[0];  
mon[2] = 10;  
System.out.println(hours[0][2]);  
System.out.println(mon.length);
```

OUTPUT

```
10  
3
```

Expression	Type
hours	int[][]
hours[0]	int[]
hours[0][2]	int



Each pair of brackets peels off **one level**: grid, then row, then one value.

`int x = hours[0];` does not compile: a row is an `int[]`, not an `int`.

length gives rows and columns

PLOT.JAVA

```
int[][] plot = new int[4][6];  
int lastRow = plot.length - 1;  
int lastCol = plot[0].length - 1;  
plot[lastRow][lastCol] = 8;  
System.out.println(plot.length);  
System.out.println(plot[0].length);  
System.out.println(plot[3][5]);
```

OUTPUT

```
4  
6  
8
```

Ask for	Write	Here
rows	<code>plot.length</code>	4
columns	<code>plot[0].length</code>	6
last row	<code>plot.length - 1</code>	3
last col	<code>plot[0].length - 1</code>	5

The first size in **new int[4][6]** is the rows, so **plot.length** is 4.

A length is never a valid index



Last index is **length - 1**. Rows from **grid.length**, columns from **grid[0].length**.

PLOT.JAVA

```
int[][] plot = new int[4][6];  
System.out.println(plot[0][plot[0].length]);
```

OUTPUT

```
Exception in thread "main"  
java.lang.ArrayIndexOutOfBoundsException: Index 6 out of bounds for length 6
```

Fix: `plot[0][plot[0].length - 1]`, the last column.

Valid, or off the edge?

PAUSE AND TRY

Which expressions are valid? What is the type of rain[2]?

Expression	Valid, or crash?
rain[3][5]	
rain[4][0]	
rain[0][6]	
rain[5][3]	
rain[2] (its type)	

```
double[][] rain =  
    new double[4][6];
```

Rows first: 4 weeks of 6 days.

Words to know

2D array

An array whose elements are 1D arrays, pictured as a grid.

Row

One inner array: a line across the grid. `grid[r]`

Column

The elements at the same index in every row.

Row index

The first index in `grid[r][c]`.

Column index

The second index in `grid[r][c]`.

Default value

What new puts in each box: 0, 0.0, false, or null.

One bracket or two?



Explain to your partner what `hours`, `hours[1]`, and `hours[1][2]` each give you.

SENTENCE STARTER

No brackets gives the _____, one pair gives a _____, and two pairs give one _____.

Next: the Grid Lock puzzle worksheet.