

2D ARRAYS

Do Now

I want to track student progress on exams over time by creating a series of Arrays that track each students score:

```
int[] exam1 = {85, 90, 78, 97, 94};
```

```
int[] exam2 = {94, 75, 58, 78, 93};
```


```
int[] exam3 = {96, 91, 88, 70, 89};
```

```
int[] exam4 = {97, 71, 90, 93, 97};
```

```
int[] exam5 = {66, 99, 78, 79, 80};
```

How do I print the 5 scores for Student1 (Student1 scores are at index 0 in the arrays)? ⇒
`System.out.println(your_code_here)`

What about if we have more scores (more arrays) would your previous printing statement is a good solution to print all the scores for Student1?



Storing Arrays in Arrays

Arrays can store Arrays

```
int[][] gradeBook = {exam1, exam2, exam3, exam4, exam5}
```



We add an additional set of square brackets



Storing Arrays in Arrays

```
Int[][] gradeBook = new int[2][5];
```

The number of arrays
in the array

The number of
values in the array

```
gradeBook[0] = exam1;
```

```
gradeBook[1] = exam2;
```



2D Arrays

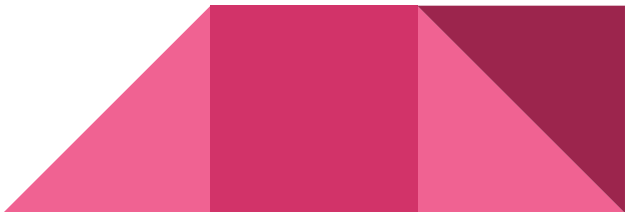
Arrays that store arrays are called **2D Arrays**.

We can see and access them as a grid:

	Students
Exams	{ 85, 90, 78, 97, 94},
	{ 94, 75, 58, 78, 93},
	{ 96, 91, 88, 70, 89},
	{ 97, 71, 90, 93, 97},
	{ 66, 99, 78, 79, 80} } ;

The rows represent the different exams.

The columns represent the students.



2D Arrays

Students

```
int[][] gradeBook = { {85, 90, 78, 97, 94},  
                      {94, 75, 58, 78, 93},  
                      {96, 91, 88, 70, 89},  
                      {97, 71, 90, 93, 97},  
                      {66, 99, 78, 79, 80} };
```

Exams

Find Student2's grade for exam 3

```
int grade = gradeBook[2][1];
```



2D Arrays

Students

```
int[][] gradeBook = { {85, 90, 78, 97, 94},  
                      {94, 75, 58, 78, 93},  
                      {96, 91, 88, 70, 89},  
                      {97, 71, 90, 93, 97},  
                      {66, 99, 78, 79, 80} };
```

Exams

Find Student2's grade for exam 3

```
int grade = gradeBook[2][1];
```

Access the 3rd
exam in the array



2D Arrays

Students

```
int[][] gradeBook = { {85, 90, 78, 97, 94},  
                      {94, 75, 58, 78, 93},  
                      {96, 91, 88, 70, 89},  
                      {97, 71, 90, 93, 97},  
                      {66, 99, 78, 79, 80} };
```

Exams

Find Student2's grade for exam 3

```
int grade = gradeBook[2][1];
```

Access the 2nd student in
the exams array



2D Arrays - Important to remember

When accessing elements in a 2D array, we are accessing the row and the column

gradeBook[row][column];

Find array



Find value in array

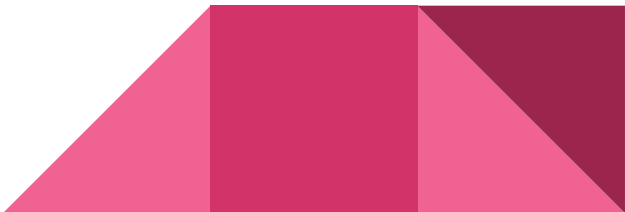
Modify elements

Same notation to modify element values:

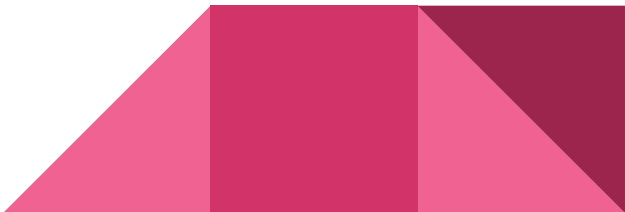
gradeBook[row][column] = newValue;

gradeBook[3][0] = 82;

```
int[][] gradeBook = { {85, 90, 78, 97, 94},  
                      {94, 75, 58, 78, 93},  
                      {96, 91, 88, 70, 89},  
                      {82, 71, 90, 93, 97},  
                      {66, 99, 78, 79, 80} } ;
```



2D Arrays - Important to remember

- Declaring a 2D array: `int[][] array = new int[3][2];`
 - Initialize with values: `int[][] array = {{7, 1}, {4, 9}, {1, 0}};` // Same row size
 - Initialize with values: `int[][] array = {{7, 1, 5}, {4, 9}, {1}};` // Different row size (**ragged array**)
 - Length: `array.length` (number of rows); `array[0].length` (number columns specific row)
 - Getting a value: `int value = array[2][1];`.
 - Setting a value: `array[2][1] = 10;`
- 

2D Arrays - Important to remember

- Assign an entire sub array: `array[1] = new int[5];`

Remember: `array[0] = array[1]` does not make a copy of the sub-array. It only changes the reference to point the same location in memory.

- Create a 2D array row by row: When we do not know the initial length of the sub-arrays:

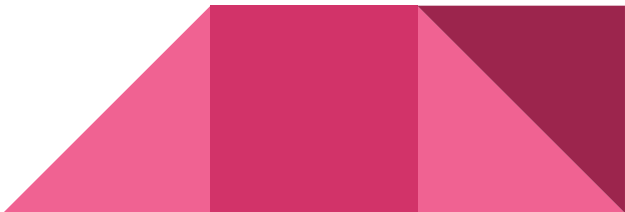
```
int[][] array = new int[3][];
```

It will create a 2D array with a null value for each row: { null, null, null }

- When you know the sub-arrays length:

```
array[0]=new int[4];  
array[1]=new int[2];  
array[2]=new int[3];
```

} {0, 0, 0, 0}, {0, 0}, {0, 0, 0}



2D Arrays - Important to remember

- Changing a 2D array after declaration:

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


- Pass 2D array into a method:

```
myMethod(new int[][]{{1, 2, 3}, {4, 5}, {6}});
```



Traverse 2D Arrays

```
int[][] array = { {1,2,3},{4,5,6}};  
for (int row = 0; row < array.length; row++)  
{  
    for (int col = 0; col < array[0].length; col++)  
        System.out.println( array[row][col] );  
}
```



Traverse 2D Arrays - Enhanced For Loop

```
String[][] array;  
  
// Nested For-each loops that traverse a 2D String array  
for (String[] innerArray : array)  
{  
    for (String val : innerArray)  
    {  
        System.out.println(val);  
    }  
}
```

