Lecture 11

Multidimensional arrays
Two-dimensional Arrays

• Just an array of arrays

• useful for storing data in a table, or pixel information, for example

• syntax is very similar to one-dimensional array
Two-dimensional Arrays

• Syntax:

        elementType[][][] arrayRefVar;

• Example:

        int[][][] matrix;
Two-dimensional Arrays

- Creating a two-dimensional array:

```java
int[][] matrix = new int[5][5];
```
Two-dimensional Arrays

matrix = new int[5][5];
matrix[2][1] = 7;

int[][] array = {
    {1, 2, 3},
    {4, 5, 6},
    {7, 8, 9},
    {10, 11, 12}
};
Two-dimensional Arrays

- Initializing a two-dimensional array:

```java
int[][] matrix = {
    {1,2,3,4,5},
    {6,7,8,9,10},
    {11,12,13,14,15},
    {16,17,18,19,20},
    {21,22,23,24,25}
};
```
Two-dimensional Arrays

- Accessing item in a two-dimensional array:
Two-dimensional Arrays

- Accessing item in a two-dimensional array:

```java
int num = matrix[2][3];
```

row  column
Two-dimensional Arrays

• Not all rows need to be the same length.

• Called a Ragged Array

• You do need to know how many rows there will be (length of outermost array)
Two-dimensional Arrays

- Processing two-dimensional arrays:
Two-dimensional Arrays

• Processing two-dimensional arrays: For loops!
Two-dimensional Arrays

• Processing two-dimensional arrays: For loops!

```java
for (int row = 0; row < matrix.length; row++){
    for (int column = 0; column < matrix[row].length; column++){
        System.out.print(matrix[row][column] + " ");
    }
    System.out.println(" ");
}
```
**Two-dimensional Arrays - Practice**

- Let’s find which row has the largest sum from the following table:

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>12</td>
<td>6</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>43</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>23</td>
<td>1</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>32</td>
<td>34</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>3</td>
<td>45</td>
<td>56</td>
<td></td>
</tr>
</tbody>
</table>
Two-dimensional Arrays - Practice

• Let’s find which row has the largest sum from the table

```java
int[][] matrix = {
    {7,12,6,23},
    {45,43,3,5},
    {56,23,1,4},
    {67,32,34,29},
    {78,3,45,56}
};

int largestRowIndex = 0;
int largestSum = 0;

for (int row = 0; row < matrix.length; row++){
    int currentRowSum = 0;
    for (int column = 0; column < matrix[row].length; column++){
        currentRowSum += matrix[row][column];
    }
    if (currentRowSum > largestSum){
        largestSum = currentRowSum;
        largestRowIndex = row;
    }
}

System.out.println("The largest sum was " + largestSum + " found in row " + largestRowIndex);
```
Multi-dimensional Arrays

- Yo dawg, I heard you like arrays, so I put an array in your array of arrays...
Multi-dimensional Arrays

- Useful for a text file of data, multiple items of data associated with a timestamp, etc.
## Multi-dimensional Arrays

<table>
<thead>
<tr>
<th>Day</th>
<th>Hour</th>
<th>Temperature</th>
<th>Humidity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>76.4</td>
<td>0.92</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>77.7</td>
<td>0.93</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>97.7</td>
<td>0.71</td>
</tr>
<tr>
<td>10</td>
<td>23</td>
<td>98.7</td>
<td>0.71</td>
</tr>
<tr>
<td>10</td>
<td>24</td>
<td>98.7</td>
<td>0.74</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Day</th>
<th>Hour</th>
<th>Temperature</th>
<th>Humidity</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>24</td>
<td>98.7</td>
<td>0.74</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>77.7</td>
<td>0.93</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>97.7</td>
<td>0.71</td>
</tr>
<tr>
<td>10</td>
<td>23</td>
<td>97.7</td>
<td>0.71</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>76.4</td>
<td>0.92</td>
</tr>
</tbody>
</table>

Which day: \[ \text{data} [i] \]
Which hour: \[ \text{data} [i][j] \]
Temperature or humidity: \[ \text{data} [i][j][k] \]