Introduction to Programming

with Java, for Beginners

2D Arrays

- Array can have 2, 3, or more dimensions
- When declaring a variable of such an array, use a pair of square brackets for each dimension
- For 2D arrays, the elements are indexed [row][column]
- Remember “RC” [row][column]

Example 1: Table

- int[ ][ ] table = new int[3][2];
- int[ ][ ] table = { {1, 2}, {3, 6}, {7, 8} };
- For example, table[1][1] contains 6
- table[2][1] contains 8, and
- table[1][2] is “array out of bounds”

Processing 2D Arrays

- How to zero out this table on the previous slide?
  ```java
  for (int i = 0; i < 3; i++){
      for (int j = 0; j < 2; j++){
          table[i][j] = 0;
      }
  }
  ```
- Use a doubly-nested for-loop to process a 2D array
- In this example we know the number of rows (3) and columns (2)
- In general, it’s better not to use “magic numbers” (here the 3 and 2) in the loop.
- How could this code be improved?
Size of 2D Array

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1 2</td>
</tr>
<tr>
<td>1</td>
<td>3 6</td>
</tr>
<tr>
<td>2</td>
<td>7 8</td>
</tr>
</tbody>
</table>

- int[ ][ ] table = new int[3][2];
- The length of this array is the number of rows: `table.length` is 3
- Each row contains an array
- To get the number of columns, pick a row and ask for its length: e.g. `table[0].length` is 2

Printing a 2D array

```java
int[][] data = new int[3][2];

//Printing 2D array example
for (int row = 0; row < 3; row++){
    for (int col = 0; col < 2; col++){
        System.out.print(data[row][col] + 't');
    }
    System.out.println();
}
```

Generalized Printing

- This illustrates a general purpose way to print a 2D array
- It works even for "ragged" arrays, whose row lengths vary

```java
public static void printArray(int[][] data){
    for (int row = 0; row < data.length; row++){
        for (int col = 0; col < data[row].length; col++){
            System.out.print(data[row][col] + "t");
        }
        System.out.println();
    }
}
```

Ragged Array

- Row lengths vary.
- Motivation: save space

```java
> int[] one = {1,2,3}
> int[] two = {1,2,3,4,5,6}
> int[] three = {1,2};
> int[][] data = {one, two, three}
> data[0].length 3
> data[1].length 6
> data[2].length 2
> data[0] = three;
> data[1] = two;
> data[2] = one;
> data[0].length 2
> data[1].length 6
> data[2].length 3
```
Error Checking with arrays

```java
public static void printArray(int[][] data) {
    if (data == null || data.length == 0) {
        System.out.println("Array is empty");
        return;
    }
    for (int row = 0; row < data.length; row++) {
        for (int col = 0; col < data[row].length; col++) {
            System.out.print(data[row][col] + "\t");
        }
        System.out.println();
    }
}
```