Multidimensional Arrays

Outline

- Arrays Methods
- Multidimensional Data
- Rectangular Two-Dimensional Arrays
- Generalization of Multidimensional Arrays
- Jagged Arrays
- Exercise 27
Arrays Methods

- Arrays class has useful array manipulation methods
- Import java.util.*;

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>binarySearch(array, value)</td>
<td>Returns the index of the given value in a sorted array (returns neg. number of value doesn’t exist)</td>
</tr>
<tr>
<td>equals(array1, array2)</td>
<td>Returns true if arrays contain the same values in the same order</td>
</tr>
<tr>
<td>fill(array, value)</td>
<td>Sets every element to the value</td>
</tr>
<tr>
<td>sort(array)</td>
<td>Sorts the array into ascending order</td>
</tr>
<tr>
<td>toString(array)</td>
<td>Returns a String containing the array elements, like: [3, 8, 7, 9, 2]</td>
</tr>
<tr>
<td>deepToString(array)</td>
<td>Returns a String containing the array elements of multidimensional arrays</td>
</tr>
</tbody>
</table>

Multidimensional Data

- What are some examples of multidimensional data?
  - int: one int
  - int[]: one-dimensional array of ints
  - int[][]: a two-dimensional grid of ints
  - int[][][]: a three-dimensional collection of ints
  - ...
Rectangular Two-Dimensional Array

• Temperature data (five readings per day) for three consecutive days
  – Convention: [<rows>][<columns>]

```java
//Rows = days, columns = reading
double[][][] temps = new double[3][5];
```

![Diagram of 2D Array]

Accessing a Cell in a 2D Array

• Steps for accessing a cell
  1. temps the entire grid
  2. temps[2] the entire third row
  3. temps[2][0] the first element of the third row

```java
temps[0][3] = 98.3;
temps[2][0] = 99.4;
```

![Diagram of Accessing Cells]
Printing a Multidimensional Array

• What control structure should we use?
• What are the bounds for the outer loop?
• What are the bounds for the inner loop?

```java
public static void print(double[][] grid) {
    for (int i = 0; i < grid.length; i++) {
        for (int j = 0; j < grid[i].length; j++) {
            System.out.println(grid[i][j] + " ");
        }
        System.out.println();
    }
}
```

Generalizing Multidimensional Arrays

• Three-dimensional array
  - int[][][] numbers = new int[4][4][4];
  - Plane by row by column
  - Plane is array of two dimensional arrays
  - Row is array of arrays
  - Column is array of ints.

• Multidimensional arrays
  - Consistency on what you consider each array of arrays to be
  - Comments to remind you (and others) what each dimension is – program context!
Jagged Arrays

- An array of arrays of varying lengths
  - First construct rows array
  - Construct array for each row

```java
int[][] jagged = new int[3][];
jagged[0] = new int[3];
jagged[1] = new int[5];
jagged[2] = new int[4];
```

Pascal’s Triangle Example

- Row $n$ of Pascal’s Triangle are the coefficients of the expanded $(x + y)^n$
- Row $n$ calculated from row $n - 1$
- Row 4 calculates middle part of Row 5
  
  $$(1 + 4)(4 + 6)(6 + 4)(4 + 1)$$

  $$5 10 10 5$$
Pascal’s Triangle Example (2)

```java
public static void fillIn(int[][] triangle) {
    for (int i = 0; i < triangle.length; i++) {
        // Create new row

        // Put in leading and trailing 1s

        // Fill middle of triangle
    }
}
```

References

- Pascal’s Triangle Picture from http://mathforum.org/workshops/usd/pascal/patterns_pascal.html