Announcements

- Lab 8 solution posted
- Program 4 due in 1 week
- Final exam
  - 6/13, 8–11 AM
  - SN014
Questions?
Today in COMP 110

- More about arrays
- 2D arrays
public class Weather
{
    private double[] temperature;
    private double[] pressure;

    public void initializeTemperature(int len)
    {
        temperature = new double[len];
    }
}
Arrays of objects

- When you create an array of objects like this:
  ```java
  Student[] students = new Student[35];
  ```
- Each of the elements of students is not yet an object
- You have to instantiate each individual one
  ```java
  students[0] = new Student();
  students[1] = new Student();
  ```
- …or do this in a loop
Arrays of objects

Smiley[] smilies = new Smiley[3];
for (int i = 0; i < smilies.length; i++)
{
    smilies[i] = new Smiley();
}
Arrays of objects

```java
Student[] students = new Student[5];
for (int i = 0; i < students.length; i++)
{
    students[i] = new Student(keyboard.nextInt());
    students[i].printAge();
}
```
Arrays as parameters

```java
public void changeArray(int[] arr) {
    int len = arr.length;
    arr[len - 1] = 25;
}
```

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>47</td>
<td>52</td>
<td>14</td>
<td>25</td>
</tr>
</tbody>
</table>
Arrays as return types

```java
public double[] buildArray(int len) {
    double[] retArray = new double[len];
    for (int i = 0; i < retArray.length; i++) {
        retArray[i] = i * 1.5;
    }
    return retArray;
}
```
Indexed variables as method arguments

- No different from using a regular variable

```java
public void printNum(int num) {
    System.out.println(num);
}

public void doStuff() {
    int[] scores = { 15, 37, 95 };

    for (int index = 0; index < scores.length; index++) {
        printNum(scores[index]);
    }
}
```
2D arrays

- Arrays having more than one index are often useful
  - Tables
  - Grids

<table>
<thead>
<tr>
<th></th>
<th>0: Open</th>
<th>1: High</th>
<th>2: Low</th>
<th>3: Close</th>
</tr>
</thead>
<tbody>
<tr>
<td>0: Apple Inc.</td>
<td>99.24</td>
<td>99.85</td>
<td>95.72</td>
<td>98.24</td>
</tr>
<tr>
<td>1: Walt Disney Co.</td>
<td>21.55</td>
<td>24.20</td>
<td>21.41</td>
<td>23.36</td>
</tr>
<tr>
<td>2: Google Inc.</td>
<td>333.12</td>
<td>341.15</td>
<td>325.33</td>
<td>331.14</td>
</tr>
<tr>
<td>3: Microsoft Corp.</td>
<td>21.32</td>
<td>21.54</td>
<td>21.00</td>
<td>21.50</td>
</tr>
</tbody>
</table>
Declaring and creating 2D arrays

```java
int[][] table = new int[4][3];

or

int[][] table;
table = new int[4][3];
```
Declaring and creating 2D arrays

```java
int[][] table = new int[4][3];

gives you the ability to use

table[0][0]
table[0][1]
table[0][2]
table[1][0]
table[1][1]
table[1][2]
table[2][0]
table[2][1]
table[2][2]
table[3][0]
table[3][1]
table[3][2]
```
We used a loop to iterate over a 1D array

```java
int[] scores = { 13, 57, 93, 60, 102 };
for (int i = 0; i < scores.length; i++)
{
    System.out.println(scores[i]);
}
```
How about a 2D array?

```java
int[][] table = new int[4][3];
```

Use a nested loop

```java
for (int row = 0; row < 4; row++)
{
    for (int column = 0; column < 3; column++)
    {
        table[row][column] = 25;
    }
}
```
Multidimensional arrays

- You can have more than two dimensions

```java
int[][][] table = new int[4][3][5];
```

- Use more nested loops to access all elements
public void print2DArray(int[][] arr)
{
    for (int row = 0; row < arr.length; row++)
    {
        for (int column = 0; column < arr[row].length; column++)
        {
            System.out.print(arr[row][column] + " ");
        }
        System.out.println();
    }
}
```java
public int[][] giveMeAnArray()
{
    int[][] table = new int[4][3];
    // put values in the table
    return table;
}
```
length for a 2D array

```java
int[][] table = new int[4][3];
```

- `table.length` is the number of rows, or the integer in the first pair of brackets (4)
- `table[i].length` is the number of columns, or the integer in the second pair of brackets (3)
int[] scores = new int[5];

- scores is a one-dimensional array
  - base type is int

int[][] table = new int[4][3];

- table is also in fact a one-dimensional array
  - base type is int[]
- We still refer to table as a two-dimensional array
Program 4: Memory

- START EARLY.