Announcements:

```
python.reset()
```

Also moving to a more reasonable room (CSE 403)

Building Java Programs

Chapter 7

Lecture 7-1: Arrays

reading: 7.1

self-checks: #1-9

videos: Ch. 7 #4

Can we solve this problem?

- Consider the following program (input underlined):

```
How many days' temperatures? 7
Day 1's high temp: 45
Day 2's high temp: 44
Day 3's high temp: 39
Day 4's high temp: 48
Day 5's high temp: 37
Day 6's high temp: 46
Day 7's high temp: 53
Average temp = 44.6
4 days were above average.
```

Why the problem is hard

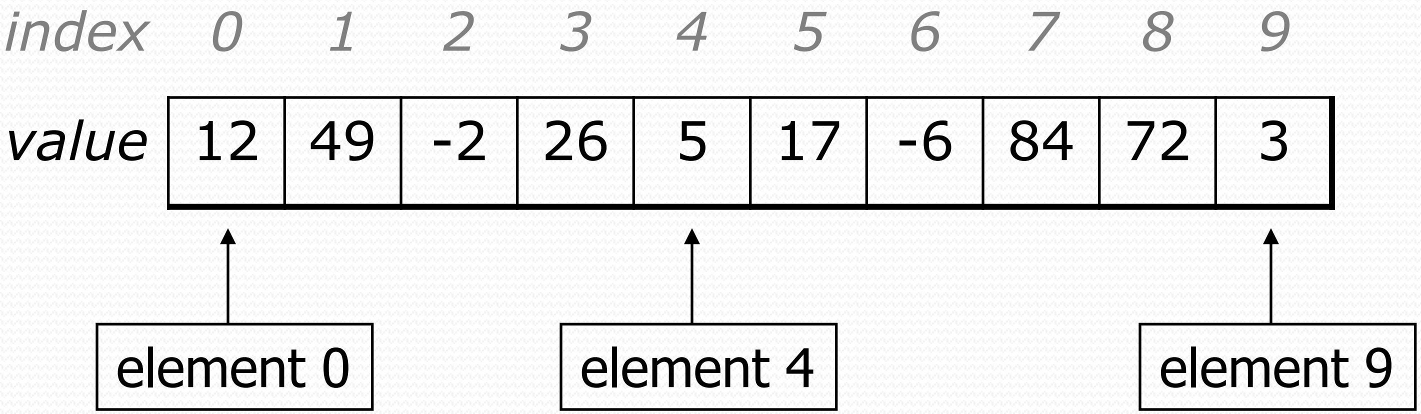
- We need each input value twice:
  - to compute the average (a cumulative sum)
  - to count how many were above average

- We could read each value into a variable... but we:
  - don't know how many days are needed until the program runs
  - don't know how many variables to declare

- We need a way to hold *a bunch of data*, not just one thing per variable
  - very important need we carefully/awkwardly avoided until now
  - every programming language needs ways to handle bunches of data; arrays are just the first way you’re seeing

Arrays

- **array**: object that stores many values of the same type.
  - **element**: One value in an array.
  - **index**: A 0-based integer to access an element from an array.



index	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
value	12	49	-2	26	5	17	-6	84	72	3

element 0

element 4

element 9

Array declaration

type`[]` **name** `= new` **type**`[`**length**`];`

- Example:

```
int[] numbers = new int[10];
```

index	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
value	0	0	0	0	0	0	0	0	0	0

Array declaration, cont.

- The length can be any integer expression.

```
int x = 2 * 3 + 1;
int[] data = new int[x % 5 + 2];
```

- Each element initially gets a "zero-equivalent" value.

Type	Default value
`int`	`0`
`double`	`0.0`
`boolean`	`false`
`String` or other object	`null` (means, "no object")

Accessing elements

```
name[index]                 // access
name[index] = value;        // modify
```

- Example:

```
int[] numbers = new int[10];
numbers[0] = 27;
numbers[3] = -6;

System.out.println(numbers[0]);
if (numbers[3] < 0) {
    System.out.println("Element 3 is negative.");
}
```

index	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
value	**27**	0	0	**-6**	0	0	0	0	0	0

Mini-exercise

- What does this print?

```
int[] a = new int[5];
a[2] = 8;
a[3] = 10;
System.out.println(a[3]);
System.out.println(a[2]*a[3]);
System.out.println(a[0]*a[2]);
```

Mini-exercise - answer

- What does this print?

```
int[] a = new int[5];
a[2] = 8;
a[3] = 10;
System.out.println(a[3]);
System.out.println(a[2]*a[3]);
System.out.println(a[0]*a[2]);
```

Result:

```
10
80
0
```

Arrays of other types

```
double[] results = new double[5];
results[2] = 3.4;
results[4] = -0.5;
```

index	*0*	*1*	*2*	*3*	*4*
value	0.0	0.0	**3.4**	0.0	**-0.5**

```
boolean[] tests = new boolean[6];
tests[3] = true;
```

index	*0*	*1*	*2*	*3*	*4*	*5*
value	false	false	false	**true**	false	false

Out-of-bounds

- Legal indexes: between **0** and the **array's length - 1**.
  - Reading or writing any index outside this range will throw an `ArrayIndexOutOfBoundsException`.

- Example:

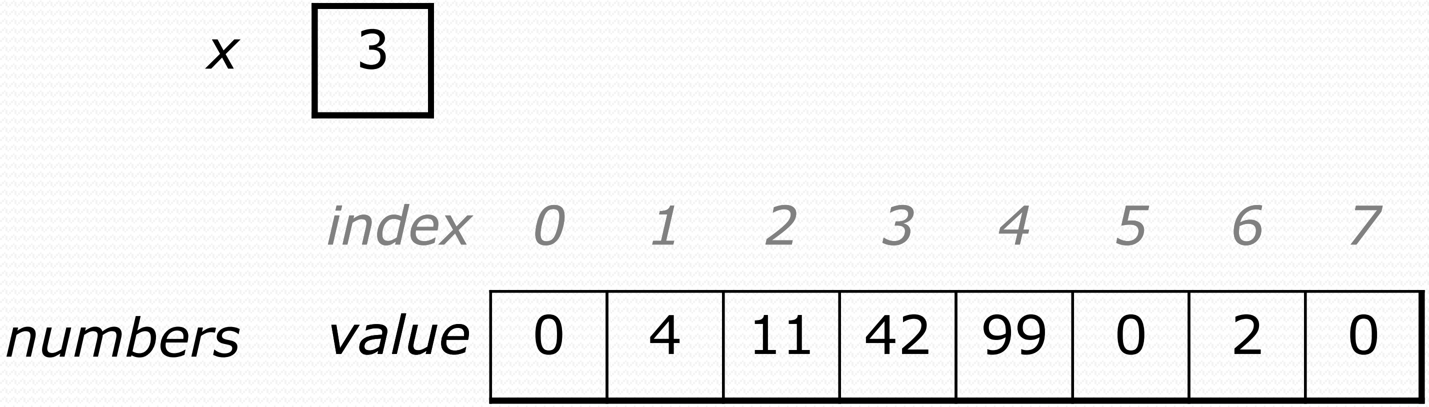
```
int[] data = new int[10];
System.out.println(data[0]);        // okay
System.out.println(data[9]);        // okay
System.out.println(data[-1]);       // exception
System.out.println(data[10]);       // exception
```

index	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
value	0	0	0	0	0	0	0	0	0	0

Accessing array elements

```
int[] numbers = new int[8];
numbers[1] = 3;
numbers[4] = 99;
numbers[6] = 2;

int x = numbers[1];
numbers[x] = 42;
numbers[numbers[6]] = 11; // use numbers[6] as index
```



x 3

	index	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
numbers	*value*	0	4	11	42	99	0	2	0

Arrays and `for` loops

- It is common to use `for` loops to access array elements.

```
for (int i = 0; i < 8; i++) {
    System.out.print(numbers[i] + " ");
}
System.out.println();  // output: 0 4 11 0 44 0 0 2
```

- Sometimes we assign each element a value in a loop.

```
for (int i = 0; i < 8; i++) {
    numbers[i] = 2 * i;
}
```

index	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*
value	0	2	4	6	8	10	12	14

The `length` field

- An array's `length` field stores its number of elements.

```
name.length

for (int i = 0; i < numbers.length; i++) {
    System.out.print(numbers[i] + " ");
}
// output: 0 2 4 6 8 10 12 14
```

  - It does not use parentheses like a String's `.length()`.

- What expressions refer to:
  - The last element of any array?
  - The middle element?

Mini-exercise

- What does this print?

```
int[] a = new int[5];
for (int i=0; i<a.length; i++) {
   a[i] = 10*i +1;
}
for (int i=0; i<a.length; i++) {
   System.out.println(a[i]);
}
```

Mini-exercise - answer

- What does this print?

```
int[] a = new int[5];
for (int i=0; i<a.length; i++) {
   a[i] = 10*i+1;
   }
for (int i=0; i<a.length; i++) {
   System.out.println(a[i]);
}
```

Result:

```
1
11
21
31
41
```

Summary of features

- In each of the following, **`e`** can be any expression that produces an **`int`**
  - Variable referring to an array: **`type[] name;`**
  - Array creation: **`new type[e]`**
  - Access array element: **`a[e]`**
  - Update array element: **`a[e] = …;`**
  - Get array's length: **`a.length`**
- That's it, but there are many important *idioms*
  - Like with many other things we've learned:
    a few rules and many idioms

Weather question

- Use an array to solve the weather problem:

```
How many days' temperatures? 7
Day 1's high temp: 45
Day 2's high temp: 44
Day 3's high temp: 39
Day 4's high temp: 48
Day 5's high temp: 37
Day 6's high temp: 46
Day 7's high temp: 53
Average temp = 44.6
4 days were above average.
```

Weather answer

```
// Reads temperatures from the user, computes average and # days above average.
import java.util.*;

public class Weather {
    public static void main(String[] args) {
        Scanner console = new Scanner(System.in);
        System.out.print("How many days' temperatures? ");
        int days = console.nextInt();

        int[] temperatures = new int[days];  // array to store days' temperatures
        int sum = 0;

        for (int i = 0; i < days; i++) {     // read/store each day's temperature
            System.out.print("Day " + (i + 1) + "'s high temp: ");
            temperatures[i] = console.nextInt();
            sum += temperatures[i];
        }
        double average = (double) sum / days;

        int count = 0;                       // see if each day is above average
        for (int i = 0; i < days; i++) {
            if (temperatures[i] > average) {
                count++;
            }
        }

        // report results
        System.out.printf("Average temp = %.1f\n", average);
        System.out.println(count + " days above average");
    }
}
```