CSCI 112
Friday, July 2, 2010
Instructor: James Church

Sometimes we need to find different properties of an array.

Basic Algorithm: Finding the max value in a list of integers

```
int[] numbers = new int [100];
.
.       //populate the array with values
.
int maxIndex = 0;
for( int index = 1; index < number.length; index++ ){
        if( number[maxIndex] < numbers[index] ){
                maxIndex = index;
        }
}
int largestValue = numbers[maxIndex];
```

Bounds Checking

```
     { 10, 20, 30, 40, 50 }
      First          Last
     Bound          Bound
```

index inside bounds: legal
index outside bounds: illegal

If you try to access data outside of the bounds, an exception is thrown.

- *ArrayOutOfBoundsException*

Java Methods

Method – set of procedures that can be called as needed.

- so, if you create a method and never call it, that code is never used, you must explicitly call methods to have them execute.

Three Parts of a Method

- **Return Type** – What data type is produced by this method ('void' if no type).
- **Signature** – A combination of names and input parameters.
- **Body** – Code which is executed when the method is called.

Example Method: Area of a Circle

```
static double areaOfCircle( double r ){
//declaring this as a static method means we can call it from other methods
	return 2.14159*r*r;
}
```

Example Method: How to Find the Average of a List of Numbers (Version 1)

```
static double average( int[] numbers ){
	double sum = 0.0;
	for( int i=0; i<numbers.length; i++ ){
		sum = sum + numbers[i];
	}
	return sum/(double)number.length;
}
```

Example Method: How to Find the Average of a List of Numbers (Version 2)

```
static double average( int … numbers ){
// "..." operator allows us to take as many arguments as we want and pack them into an array called
// numbers
	double sum = 0.0;
	for( int I: numbers ){
		sum += I;
	}
	return sum / (double)numbers.length;
}
```

How to Use Method: Average (works for v1 and v2)

```
int[] a = { 10, 20, 30, 40 };
System.out.println( average(a) );
```

How to Use Method: Average (works only for v2)

```
System.out.println( average(10, 20, 30, 40) );
```

Parameter Passing

- **Pass-by-value:** Transfer the value of a variable directly to the method.
- **Pass-by-reference:** Transfer the address of a variable to the method.

For Primitives, Java passes by value.
For Arrays and Objects, Java passes by reference.

The key advantage to pass-by-value is that there is no ambiguity about what is being sent.
Pass-by-reference has the advantage of knowing how many elements there are.

If you are passing by value, and there are 1,000,000 objects being sent, they must all be sent.
If you are passing by reference, and there are 1,000,000 objects being sent, only the one needed is sent.

Java takes this behavior out of your hands, and there is no way to bypass it. Thus, there is no extra syntax to learn.