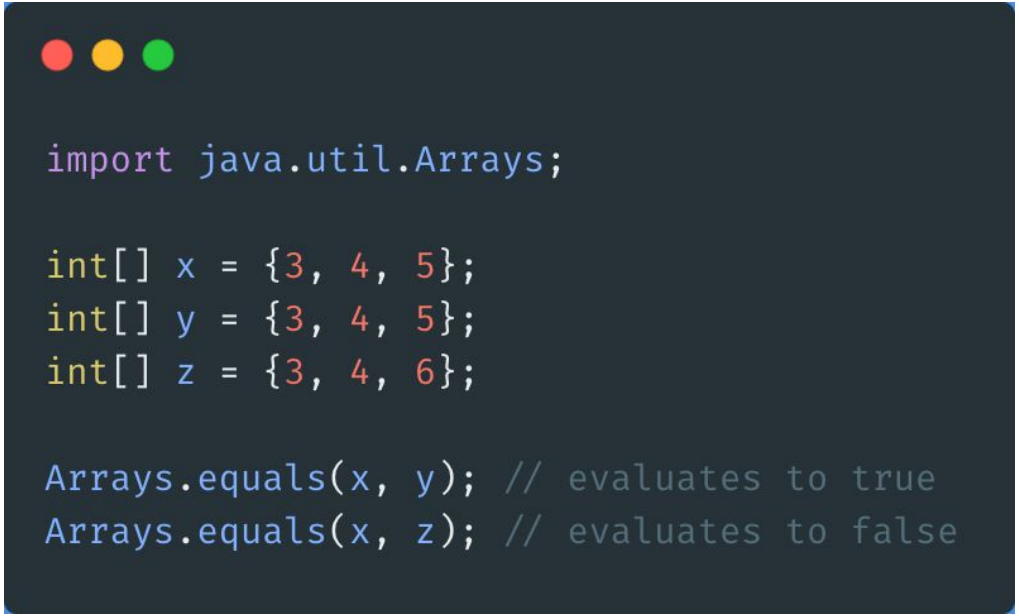
Odds and Ends of Java

Or: If every lecture had been 3 minutes longer

java.util.Arrays

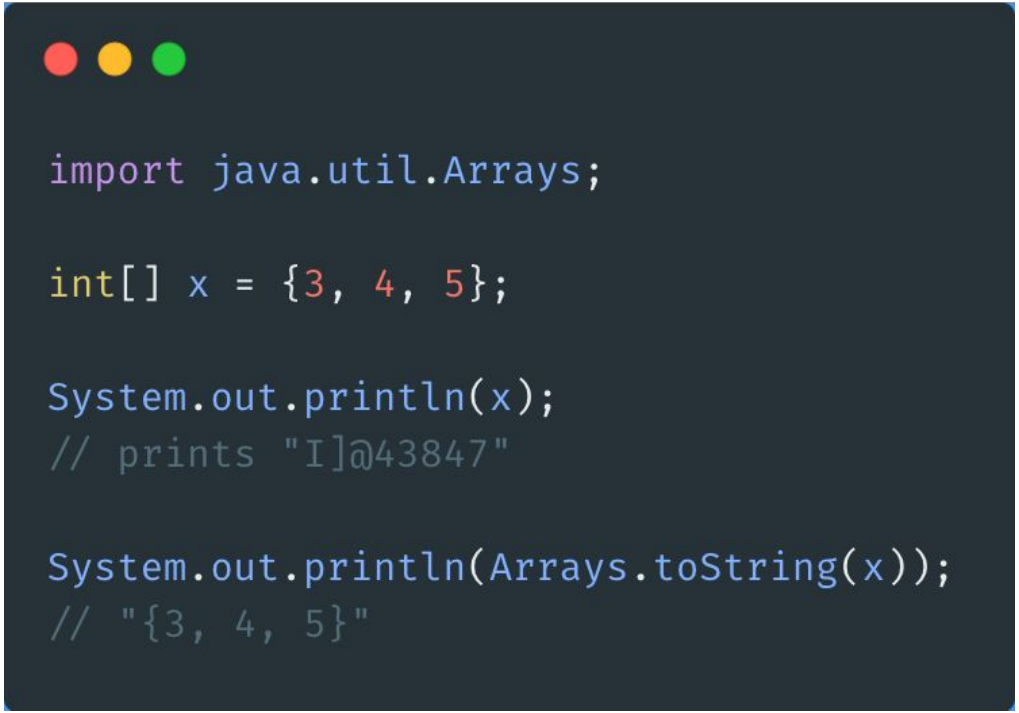
equals(a, b)

Returns true when arrays a and b contain the same elements



toString(a)

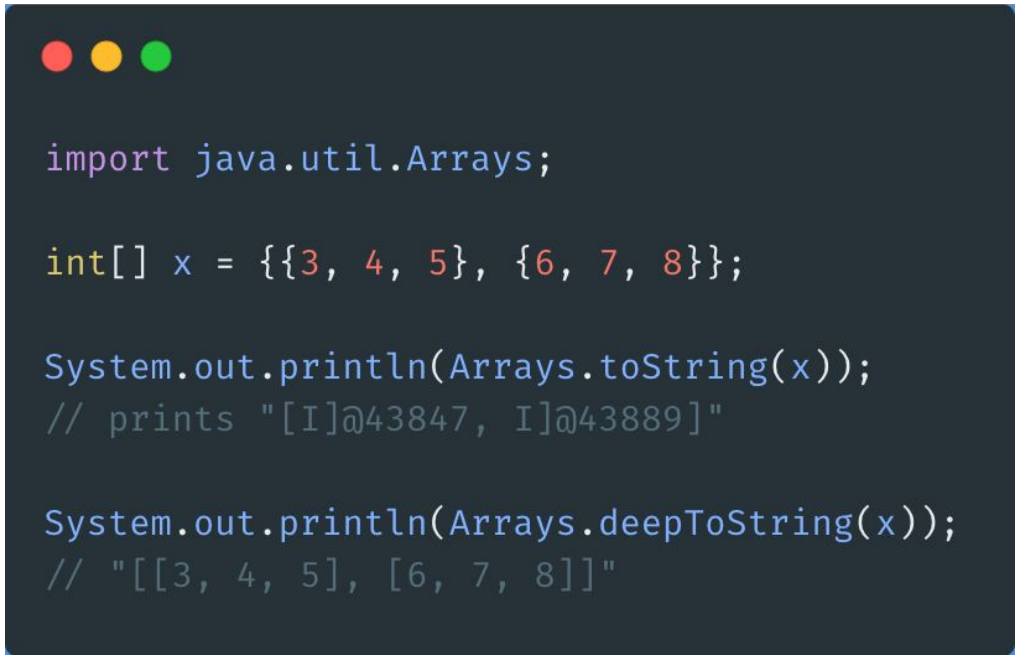
Returns a String representation of the array that's not just garbage. Useful on 1D arrays.



```
import java.util.Arrays;

int[] x = {3, 4, 5};

System.out.println(x);
// prints "I]@43847"

System.out.println(Arrays.toString(x));
// "{3, 4, 5}"
```

deepToString(a)

Returns a String representation of the array that's not just garbage. Useful on 2D arrays. (works on 1D arrays, too)



sort(a)

Modifies a in place and sorts its values in increasing order. Requires that a is an array containing Comparable types, or just primitive types.

```
import java.util.Arrays;

int[] x = {9, 0, 29, 3};

System.out.println(Arrays.toString(x));
// prints "[9, 0, 29, 3]"

Arrays.sort(x);

System.out.println(Arrays.toString(x));
// "[0, 3, 9, 29]"
```

binarySearch,
copyOf,
mismatch,
fill,
& more

All the details here.

Java's built-in LinkedList & ArrayList

Don’t have to write your own! Java has done it for you.

```java
import java.util.List;
import java.util.ArrayList;

List<Integer> inputArgs = new ArrayList<Integer>();
inputArgs.add(3);
inputArgs.add(4);
inputArgs.add(9);
inputArgs.add(-10);

System.out.println(inputArgs.isEmpty()); // false
System.out.println(inputArgs.contains(4)); // true
```

Pay attention to generics

You have to specify the type of List/ArrayList/LinkedList you want using the <> notation.

Remember that List should be the type of the variable, but you use the LinkedList or ArrayList constructor

```
import java.util.List;
import java.util.LinkedList;

List<Integer> ints = new LinkedList<Integer>();
List<Double> doubles = new LinkedList<Double>();
List<Book> books = new LinkedList<Book>();
List<Spider> spiders = new LinkedList<Spider>();
```

For-each loop

Complicated, error-prone, old fashioned, retro, etc

```java
import java.util.List;
import java.util.LinkedList;

List<Integer> ints = new LinkedList<Integer>();

for (int i = 0; i < ints.size(); i++) {
  int current = ints.get(i);
  doSomething(current);
}
```

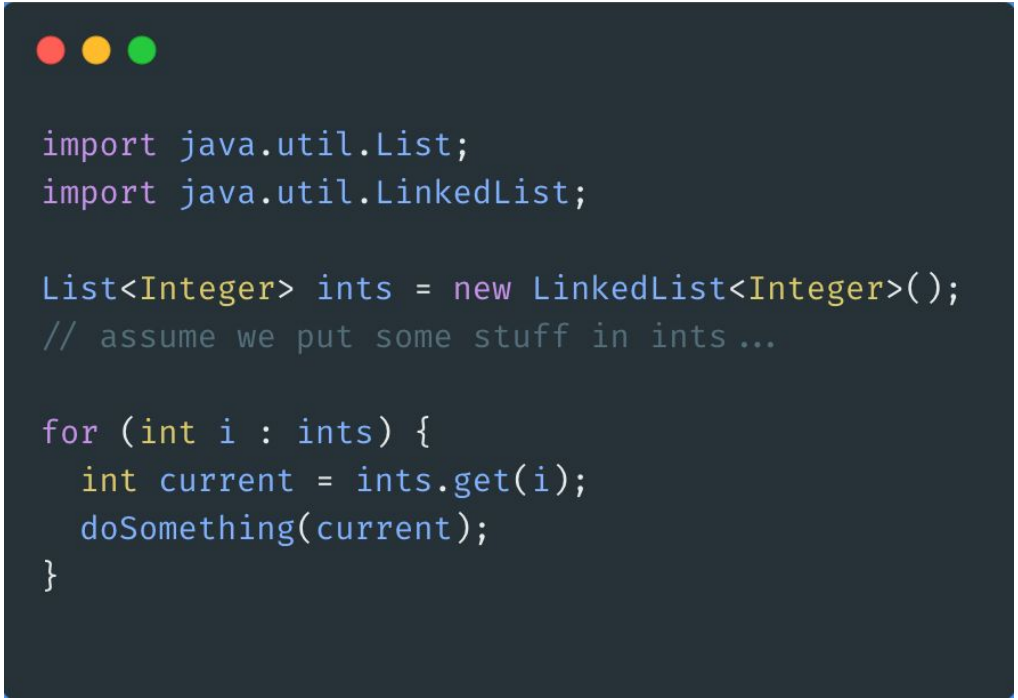
For-each loops are short and simple

for (type var : collection) {

// variable var in scope here

// var is not an index!!

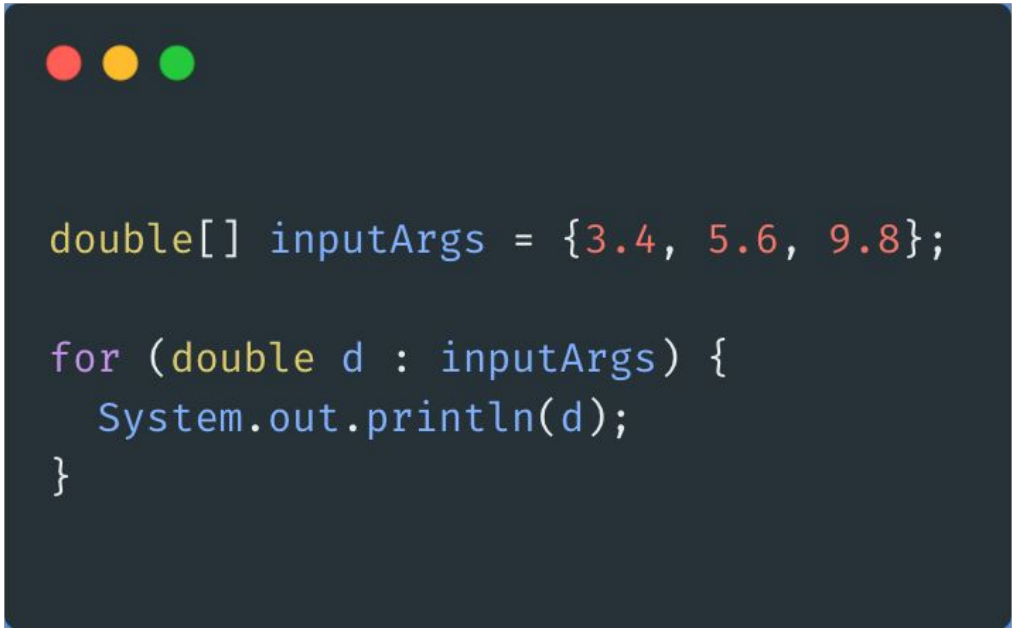
}



```
import java.util.List;
import java.util.LinkedList;

List<Integer> ints = new LinkedList<Integer>();
// assume we put some stuff in ints ...

for (int i : ints) {
  int current = ints.get(i);
  doSomething(current);
}
```

Works over arrays, too

Again, *d* is not an index: it's the array element itself!



Switch Statements

More concise way of matching a value than ifs

- The “switch” must be performed on a value of an enumerable type (commonly int, String, or char)
- Provide a series of cases to be tested for matches, along with a default
- Replaces if/else if/else if/.../else

```
switch (name) {
  case "Harry": case "Eric":
    title = "Professor";
    break;
  case "Jules": case "Michael": case "Gian": case "Ben":
    title = "Head TA";
    break;
  default:
    System.out.println("Name not matched, default case used");
    title = "TA";
    break;
}
```

JShell

A Java REPL (Read-Evaluate-Print-Loop)

- Codio's terminal is already like a REPL, but not for Java
  - Enter some command, and the terminal evaluates it and prints the result for you
  - Works in bash
- Wouldn't it be nice to be able to test some small code behavior without writing it into your classes?
  - The answer is yes.
- Requires you to install Java on your computer—not always trivial

Demo!

Next Steps

“Now that I’ve gotten through CIS 110, what’s next?”

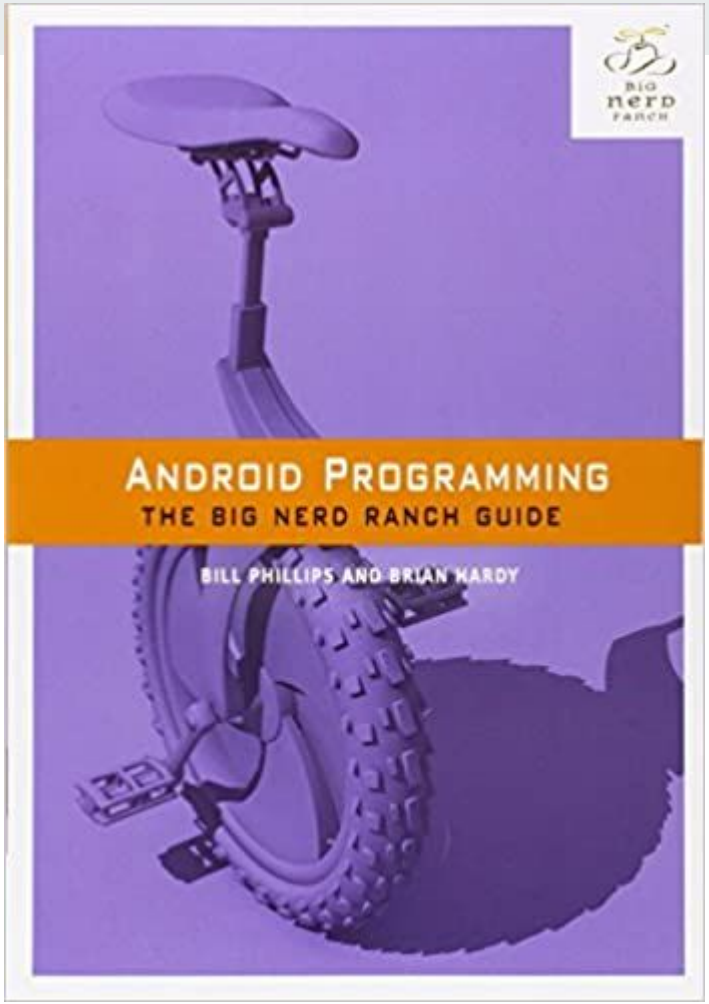
- For many folks, the answer is simple: take CIS 160 and CIS 121 and go from there
- Students often ask in either case: “How can I apply what I’ve learned?”

“I want to make a small project to show off what I can do.”

- Use Processing and get creative!
  - It’s like PennDraw, but much better
- Build another game
- Create a small facial recognition app
- Make some art
- Write a Twitter bot

“I want to build an app.”

- Lucky for you, Android apps can be easily developed using Java.
  - Technically, Kotlin
- There are plenty of video/written guides on the internet
- Penn Libraries also lets you access several books on the topic
  - (see right)



“I want to use coding to make my life easier in the other work that I’ll do.”

Java may not exactly be right for you, although it will always be there when you need it!

Learning your first programming language is hard. Learning your second is fun and easy.