Comp 215: Intro To Program Design

Prof. Chris Jermaine
cmj4@cs.rice.edu
“Chris”
“Prof. Chris”
“Dr. Chris”
This Class

• 50% of content: modern programming and program design

  — The Java programming language will be used (no prior Java assumed)
  — But this is not a “Java class” per se
  — Goal is proficiency in modern OO program development
  — Will fixate on Java a lot (can’t get around that), but will try to stay general, as well
This Class

• 50% of content: modern programming and program design
  — The Java programming language will be used (no prior Java assumed)
  — But this is not a “Java class” per se
  — Goal is proficiency in modern OO program development
  — Will fixate on Java a lot (can’t get around that), but will try to stay general, as well

• 50% of content: modern algorithms
  — Modern computer science IS NOT programming
  — Modern computer science IS algorithmic thinking
  — We only program because we have to... it’s a necessary evil :-(
  — In keeping with this view, will try to avoid studying programming in a vacuum
  — Means we’ll have to cover a lot of algorithms to motivate our programming
What Is “Modern OO Program Development”?

• Simply a paradigm that forces abstraction when applied correctly
  — Unfortunately, very few people apply it correctly!
  — Despite what you might have heard, OO is no silver bullet
What Is “Modern OO Program Development”?  

• Simply a paradigm that forces **abstraction** when applied correctly
  — Unfortunately, very few people apply it correctly!
  — Despite what you might have heard, OO is no silver bullet

• **What is abstraction?**
  — Abstraction means taking complicated machinery and wrapping it up in such a way that people can use the machinery w/o understanding the details
  — Abstraction is all about defining and enforcing **interfaces**
What Is “Modern OO Program Development”? 

- Simply a paradigm that forces **abstraction** when applied correctly
  - Unfortunately, very few people apply it correctly!
  - Despite what you might have heard, OO is no silver bullet

- What is **abstraction**?
  - Abstraction means taking complicated machinery and wrapping it up in such a way that people can use the machinery w/o understanding the details
  - Abstraction is all about defining and enforcing **interfaces**

- Why is **abstraction** good?
  - In theory, effort to build/maintain system is linear in system functionality, SLOC
  - Why? With proper abstraction, everything is local
  - W/o abstraction, effort is quadratic: $1 + 2 + 3 + ... + n$ is prop. to $n^2$
On To Java!

• “Java”... what is it? What’s the big idea?
  — OO programming language first released in 1995 (wow, pretty old already!)
  — Unique in that it was designed to run on a “virtual machine” (JVM)
  — To run a Java program, someone must have JVM installed on their machine
  — You write your program, software called compiler translates into Java bytecode
  — You give your bytecode to someone who wants to run your program
  — JVM on their machine interprets that bytecode
  — Interprets = uses underlying hardware to take actions spec’d by bytecode
On To Java! (cont’d)

• The JVM is a really big deal!
  — Same bytecode runs on a Mac, PC, Unix server, smartphone, anything with a JVM
  — JVM ensures a set of key algs, data structures, and I/O capabilities are there to use
  — All bundled in the “Java Class Library”... comes with the JVM
  — Back in ‘95, a typical PL might have a target-specific compiler + some low-level libraries (such as libc for C/C++ I/O)
  — Needed hash table? You wrote your own, asked your friends, or searched the web
  — Using Java? Just use the hash table that comes with the JVM
  — Hard to overstate significance of this change!
Compilation vs. Interpretation

• Java is a “compiled” language
  — Since entire program is translated into low-level bytecode, all at once
  — Then the bytecode is run

• Python is an “interpreted” language
  — Python interpreter actually executes Python

• Why might we prefer a compiled language?
Compilation vs. Interpretation

• Java is a “compiled” language
  — Since entire program is translated into low-level bytecode, all at once
  — Then the bytecode is run

• Python is an “interpreted” language
  — Python interpreter actually executes Python

• Why might we prefer a compiled language?
  — Often far faster to execute (though use of JVM does hurt a bit in the case of Java)
  — Can catch many errors at compile time
So Is Java A Good Language?

• The bad stuff:
  — It’s going on 20 years old, and showing its age a bit
  — Some stuff (such as “generics”) are arguably not done correctly
  — It’s a big and complicated language (so our class is quite back-loaded!)
So Is Java A Good Language?

• The bad stuff:
  — It’s going on 20 years old, and showing its age a bit
  — Some stuff (such as “generics”) are arguably not done correctly
  — It’s a big and complicated language (so our class is quite back-loaded!)

• The good stuff:
  — It’s the most modern of the widely-used languages
  — The JVM
  — It’s somewhat difficult to really screw up in Java (compared to C, for example)
So Is Java A Good Language?

• The bad stuff:
  — It’s going on 20 years old, and showing its age a bit
  — Some stuff (such as “generics”) are arguably not done correctly
  — It’s a big and complicated language (so our class is quite back-loaded!)

• The good stuff:
  — It’s the most modern of the widely-used languages
  — The JVM
  — It’s somewhat difficult to really screw up in Java (compared to C, for example)

• Why are we using it?
  — For me, comes down to pragmatics
  — Meanwhile, we can sit and hope for a better alternative! C# anyone?
Writing Your First Java Program

• You’ll have to do this to “pass” the second class meeting
  — More explicit instructions will be given at that time!
  — But most should be able to do this at home, with little difficulty
Writing Your First Java Program

• You’ll have to do this to “pass” the second class meeting
  — More explicitly instructions will be given at that time!
  — But most should be able to do this at home, with little difficulty

• First, you’ll need to download a “JDK”
  — Contains compiler, JRE (JVM, object library), and a bunch of other stuff
  — The one you want comes from Sun/Oracle (just Google “JDK”)
  — You’ll want version 7
Writing Your First Java Program

• You’ll have to do this to “pass” the second class meeting
  — More explicitly instructions will be given at that time!
  — But most should be able to do this at home, with little difficulty

• First, you’ll need to download a “JDK”
  — Contains compiler, JRE (JVM, object library), and a bunch of other stuff
  — The one you want comes from Sun/Oracle (just Google “JDK”)
  — You’ll want version 7

• Then, you’ll want an “IDE”
  — Allows you to manage the source files associated with a program
  — Typically has support for debugging, testing, editing source code
  — Standard open source IDE is called “Eclipse”
  — We’ll use one targeted towards “Java 101” users called DrJava
Once You’ve Installed the JDK and DrJava

• Fire up DrJava

  — Check “Full Java” under “Language Level”

• Then type in your first program:

```
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
```

• Press “compile” and then “run” and watch it go...
What Does This Program Do?

// A “class” is a bundle of data plus functionality (methods)
// A “program” consists of a set of classes

class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}


public class HelloWorld {

    // This is a “static” method called “main”
    // “static” means that it is shared by all objects of
    //   type HelloWorld
    // “public” means it can be called from outside the
    //   class
    // In every program, a class must have a public, static
    //   method called “main” that’s invoked at startup
    // “args” is the list of parameters passed via the
    //   command line when the program is run
    public static void main(String[] args) {
        System.out.println("Hello World");
    }
}
public class HelloWorld {
    public static void main(String[] args) {

        // “System” is a special class in the SCL
        // It has a data item (aka, a “member”) called “out”
        // that corresponds to your console
        // “out” has a method called “println” that accepts a
        // string and outputs it
        System.out.println("Hello World");
    }
}