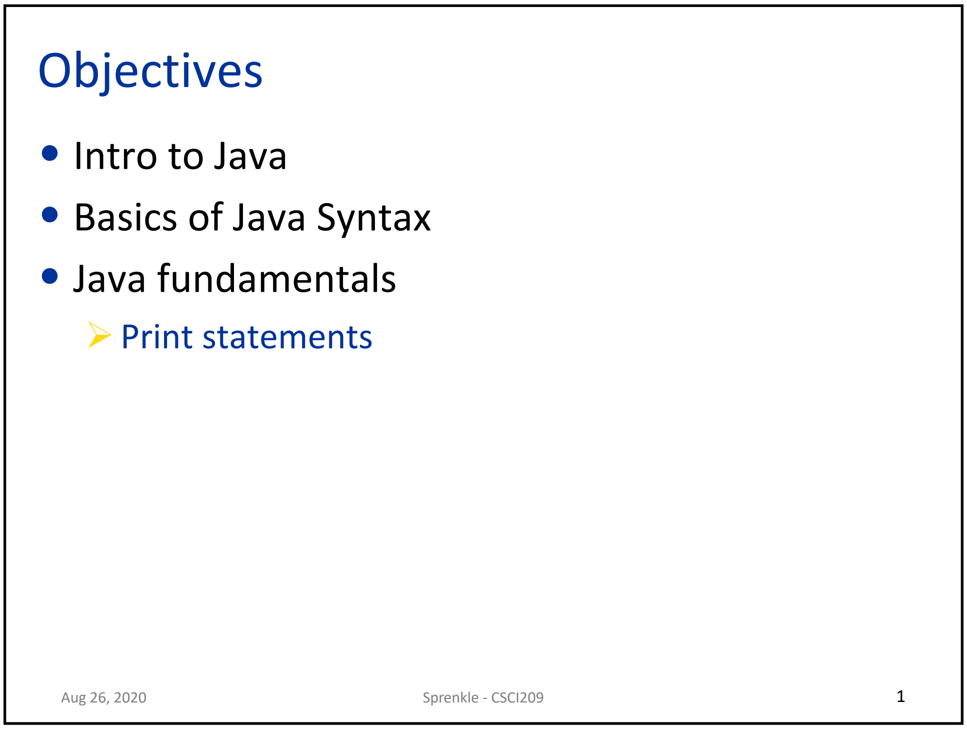
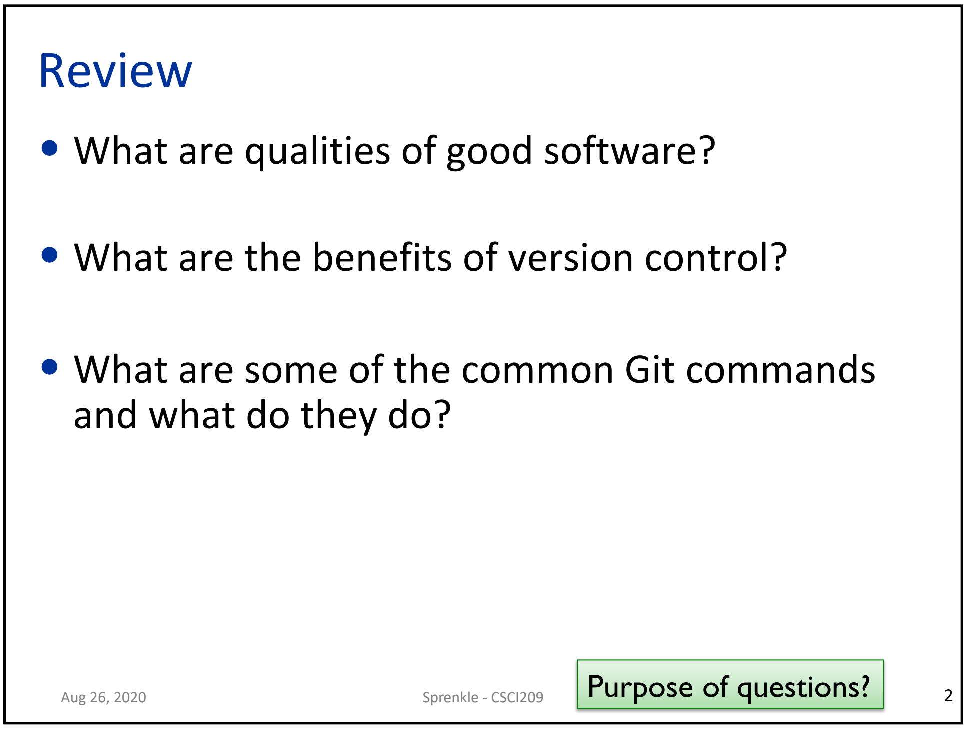


Objectives

- Intro to Java
- Basics of Java Syntax
- Java fundamentals
  - Print statements

1

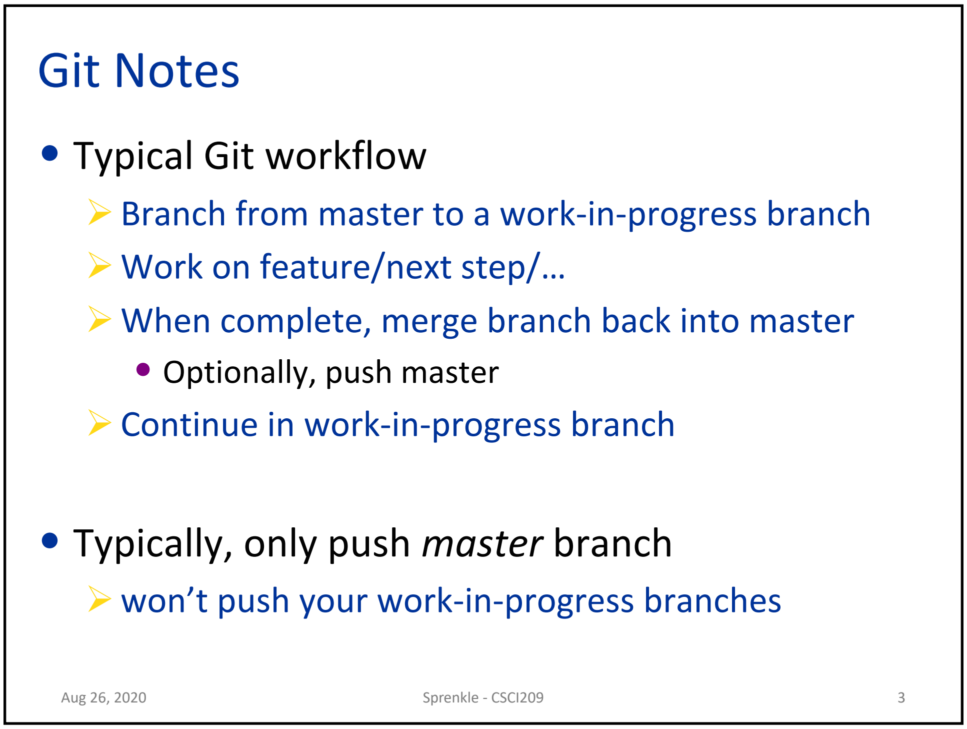


Review

- What are qualities of good software?
- What are the benefits of version control?
- What are some of the common Git commands and what do they do?

Purpose of questions?

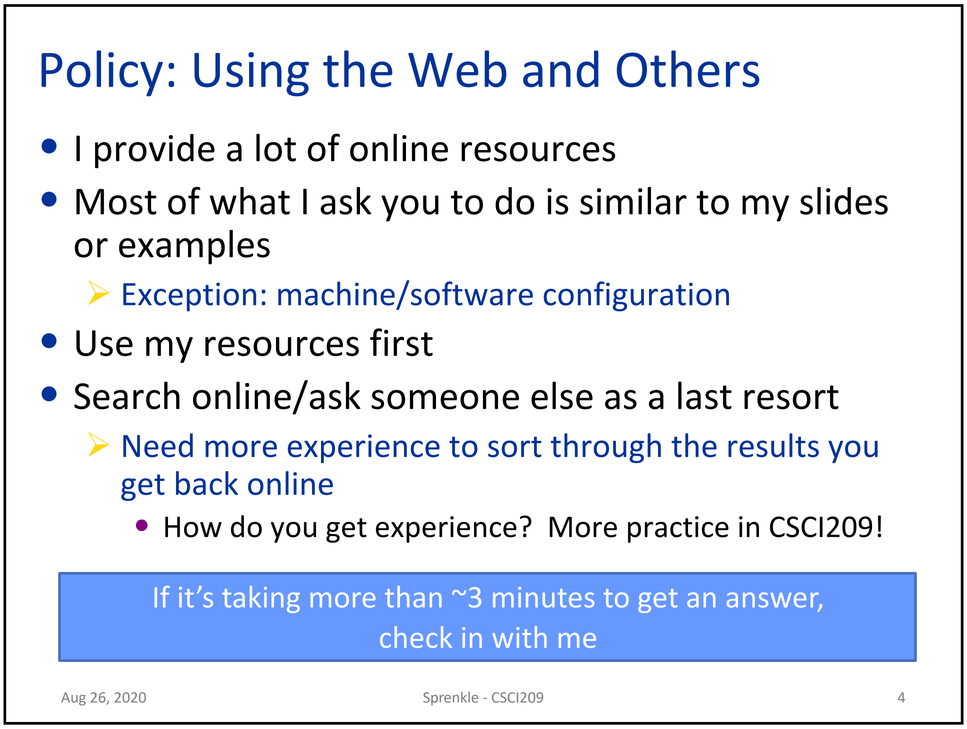
2



Git Notes

- Typical Git workflow
  - Branch from master to a work-in-progress branch
  - Work on feature/next step/…
  - When complete, merge branch back into master
    - Optionally, push master
  - Continue in work-in-progress branch
- Typically, only push *master* branch
  - won't push your work-in-progress branches

3

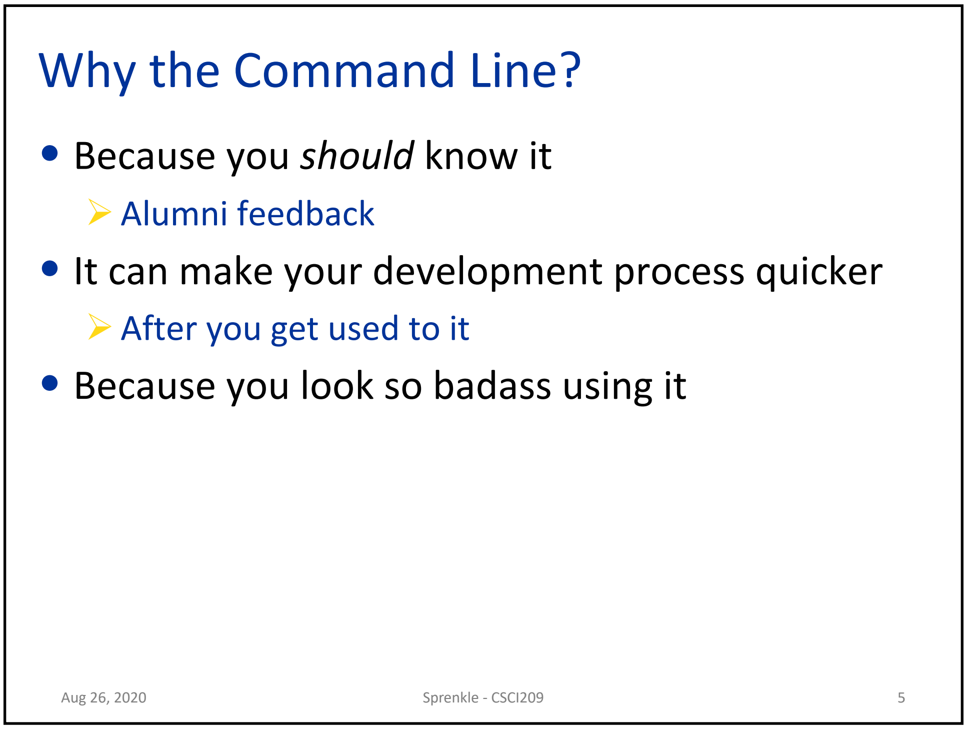


Policy: Using the Web and Others

- I provide a lot of online resources
- Most of what I ask you to do is similar to my slides or examples
  - Exception: machine/software configuration
- Use my resources first
- Search online/ask someone else as a last resort
  - Need more experience to sort through the results you get back online
    - How do you get experience? More practice in CSCI209!

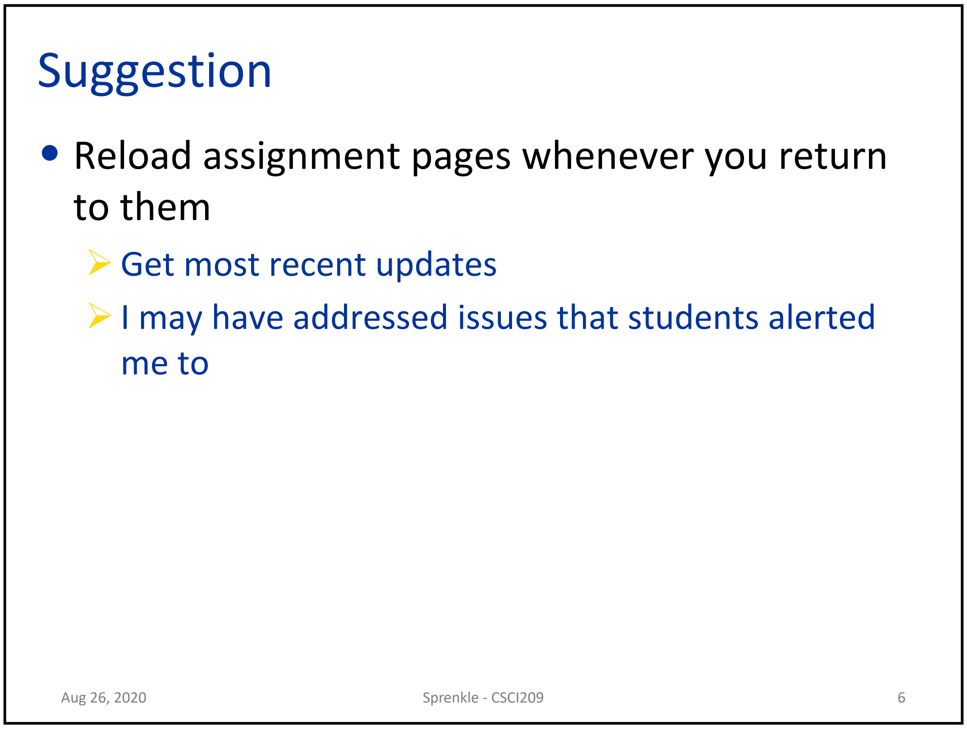
If it's taking more than ~3 minutes to get an answer, check in with me

4



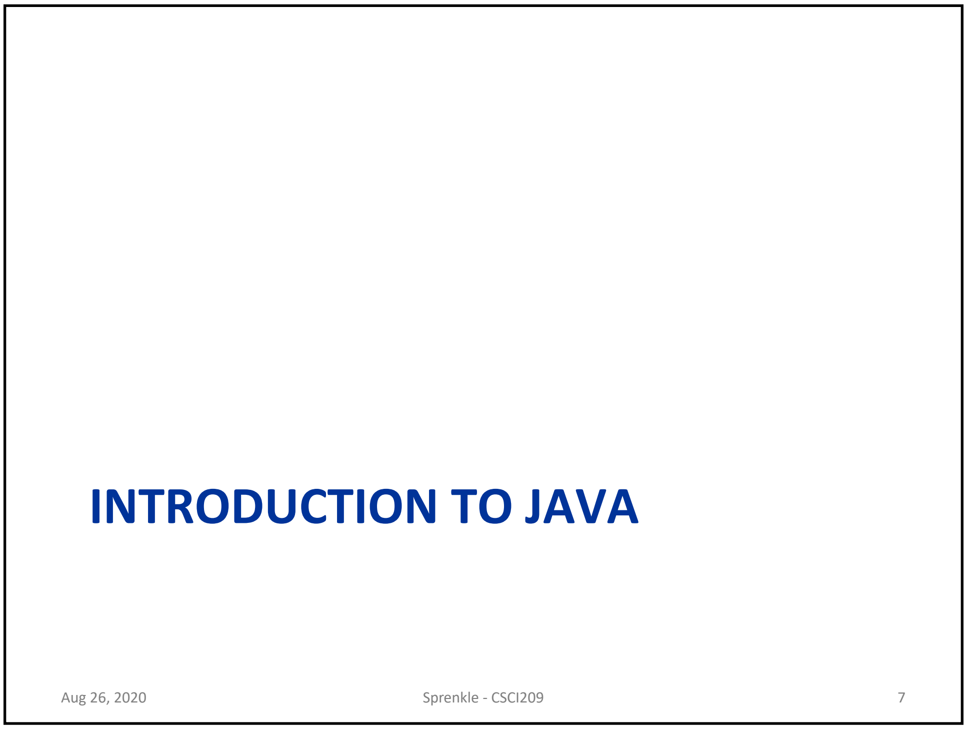
Why the Command Line?

- Because you *should* know it
  - Alumni feedback
- It can make your development process quicker
  - After you get used to it
- Because you look so badass using it



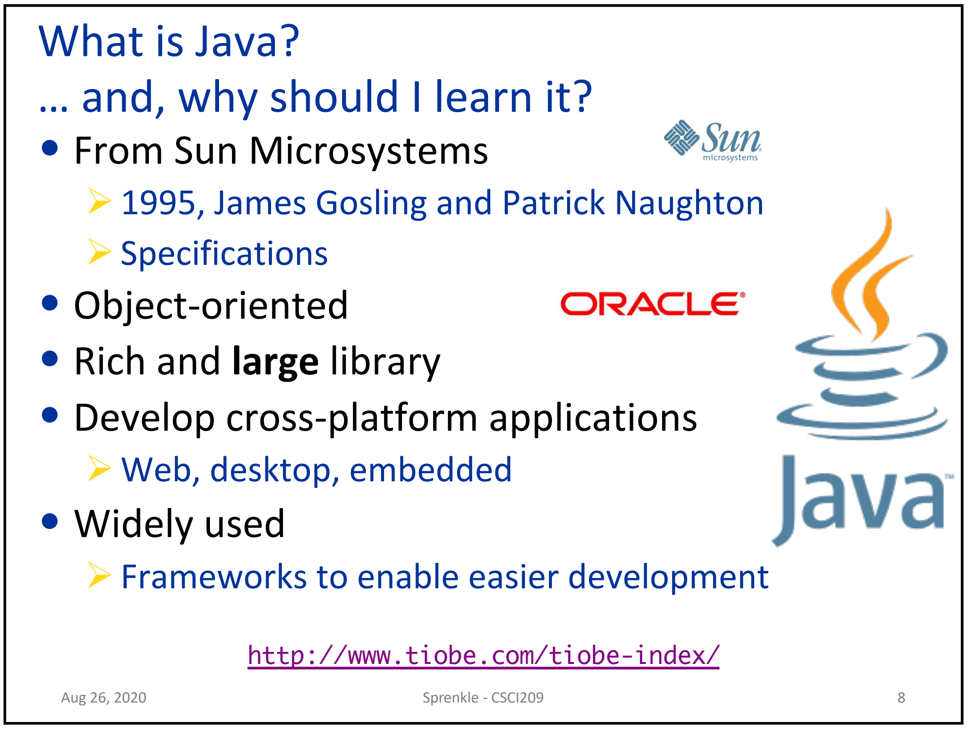
Suggestion

- Reload assignment pages whenever you return to them
  - Get most recent updates
  - I may have addressed issues that students alerted me to



INTRODUCTION TO JAVA

7



What is Java?
… and, why should I learn it?

- From Sun Microsystems
  - 1995, James Gosling and Patrick Naughton
  - Specifications
- Object-oriented
- Rich and **large** library
- Develop cross-platform applications
  - Web, desktop, embedded
- Widely used
  - Frameworks to enable easier development

Sun microsystems
ORACLE
Java

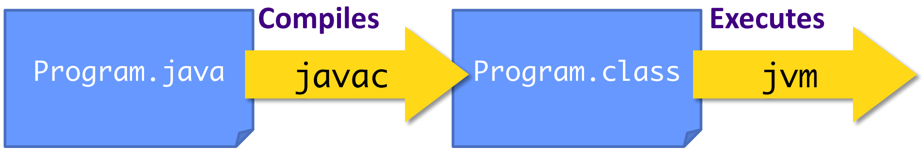
http://www.tiobe.com/tiobe-index/

8

What is Java?

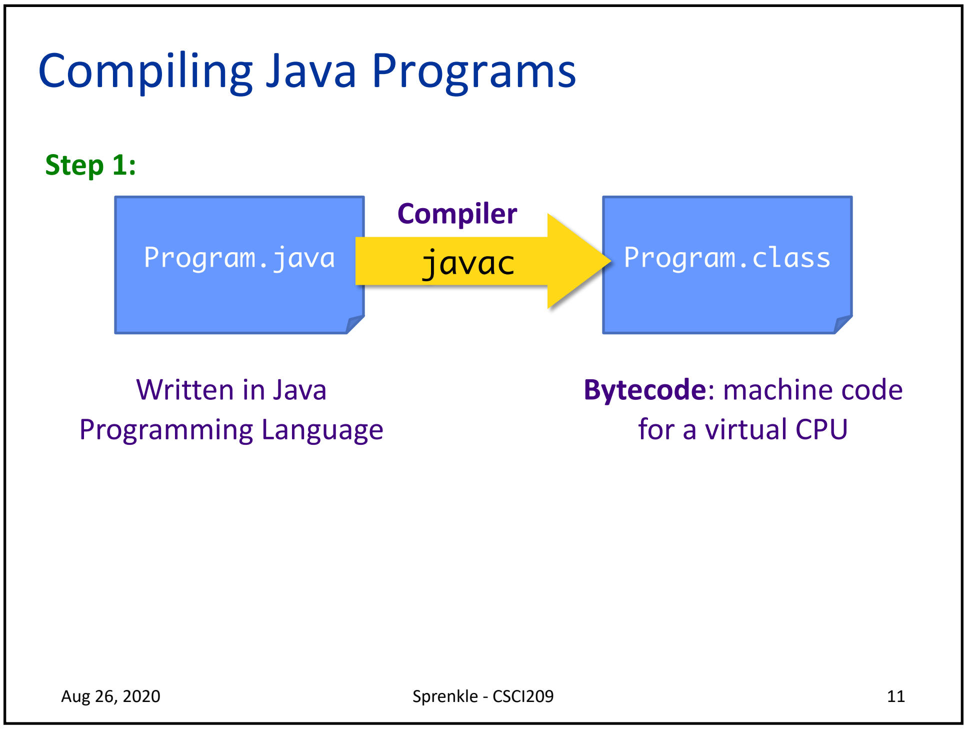
- Java Programming Language
- Java Virtual Machine
- Java Class Libraries

Overview: Compiling, Executing Java Programs



Written in Java Programming Language

Bytecode: machine code for a virtual CPU



Compiling Java Programs

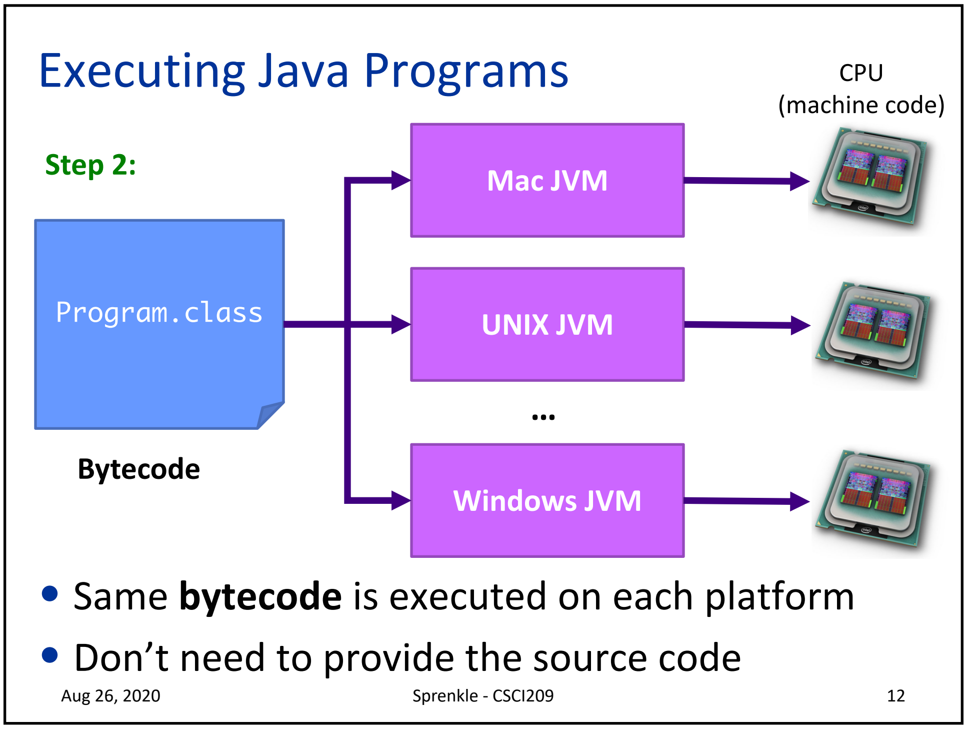
Step 1:

Program.java → **Compiler** javac → Program.class

Written in Java Programming Language

Bytecode: machine code for a virtual CPU

11



Executing Java Programs

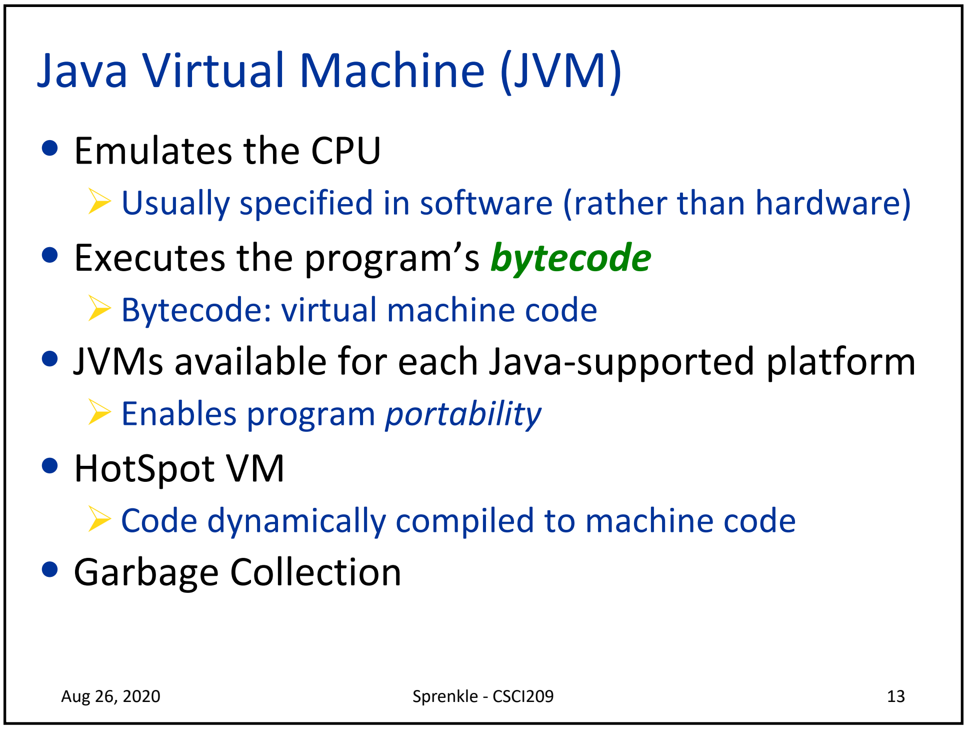
Step 2:

Program.class

Bytecode

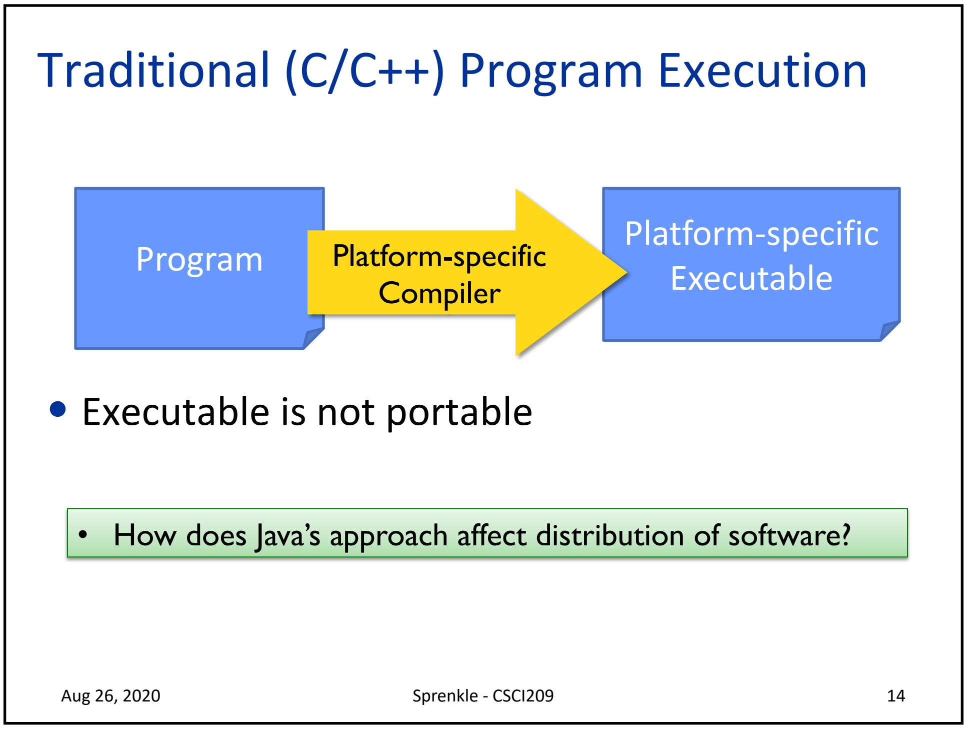
- Same **bytecode** is executed on each platform
- Don’t need to provide the source code

12



Java Virtual Machine (JVM)

- Emulates the CPU
  - Usually specified in software (rather than hardware)
- Executes the program’s ***bytecode***
  - Bytecode: virtual machine code
- JVMs available for each Java-supported platform
  - Enables program *portability*
- HotSpot VM
  - Code dynamically compiled to machine code
- Garbage Collection



Traditional (C/C++) Program Execution

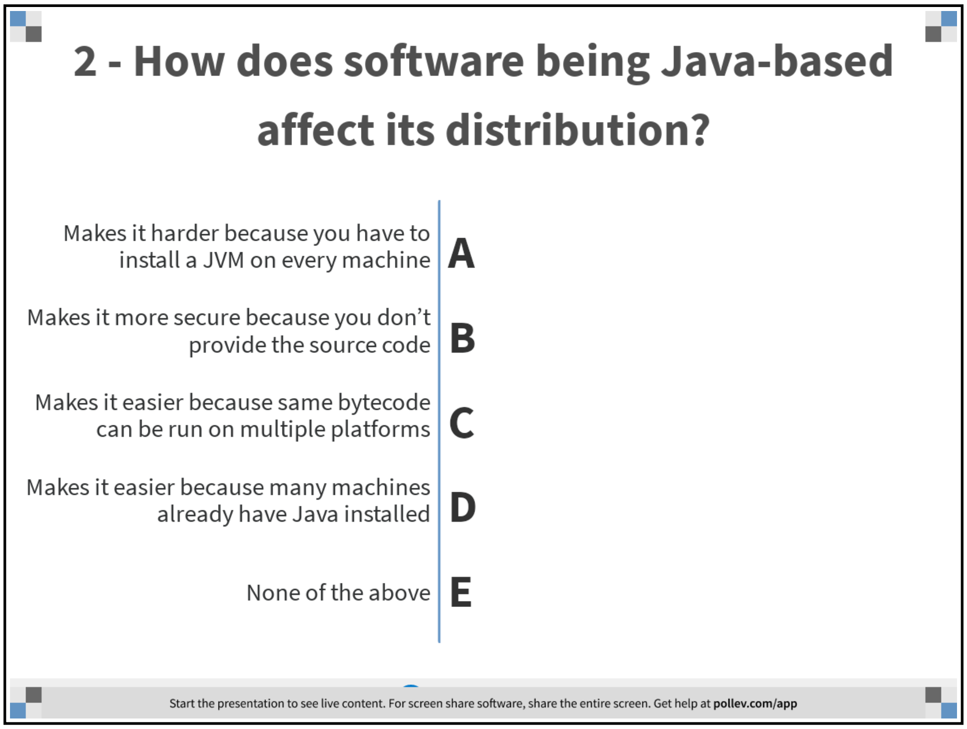
Program

Platform-specific Compiler

Platform-specific Executable

- Executable is not portable

- How does Java’s approach affect distribution of software?



2 - How does software being Java-based affect its distribution?

Makes it harder because you have to install a JVM on every machine **A**

Makes it more secure because you don't provide the source code **B**

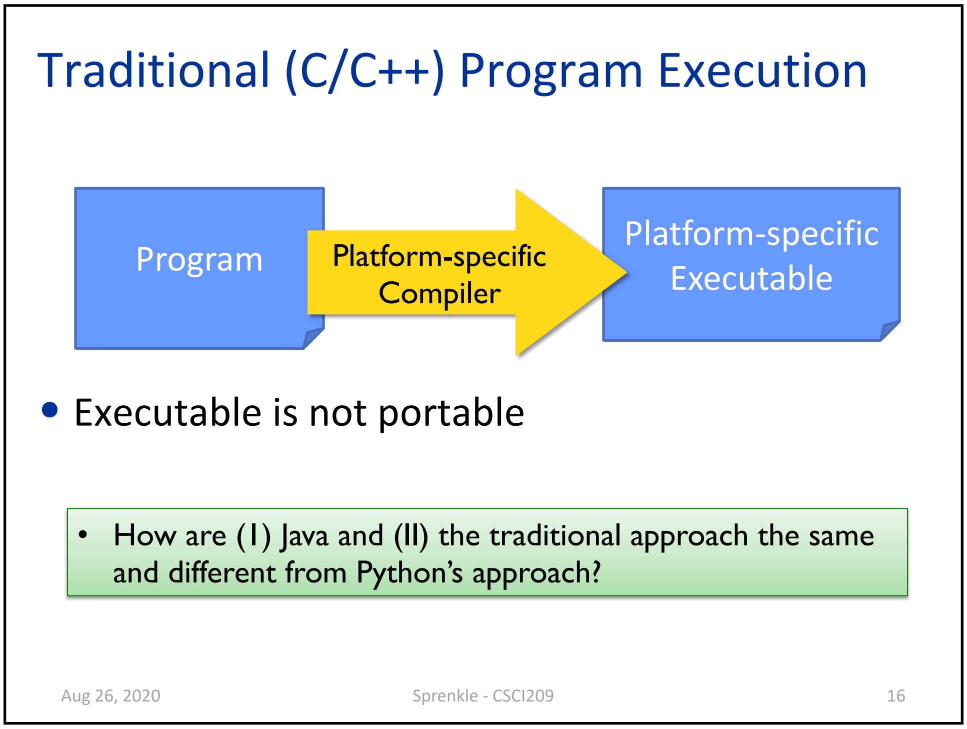
Makes it easier because same bytecode can be run on multiple platforms **C**

Makes it easier because many machines already have Java installed **D**

None of the above **E**

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

15



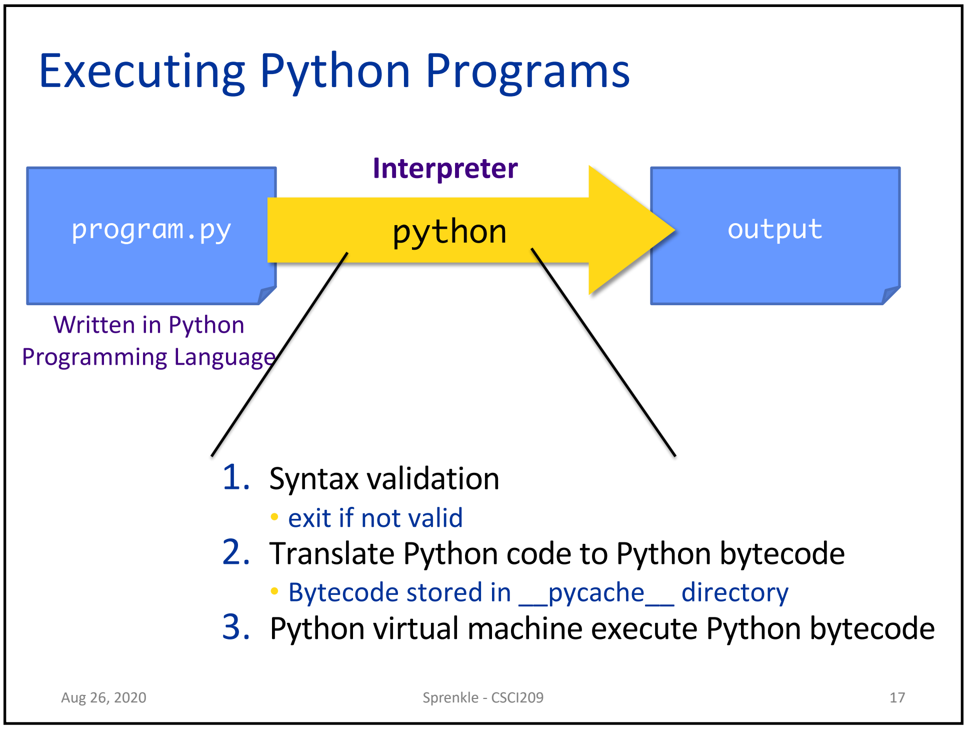
Traditional (C/C++) Program Execution

Program
Platform-specific Compiler
Platform-specific Executable

- Executable is not portable

- How are (I) Java and (II) the traditional approach the same and different from Python's approach?

16



Executing Python Programs

program.py

Written in Python Programming Language

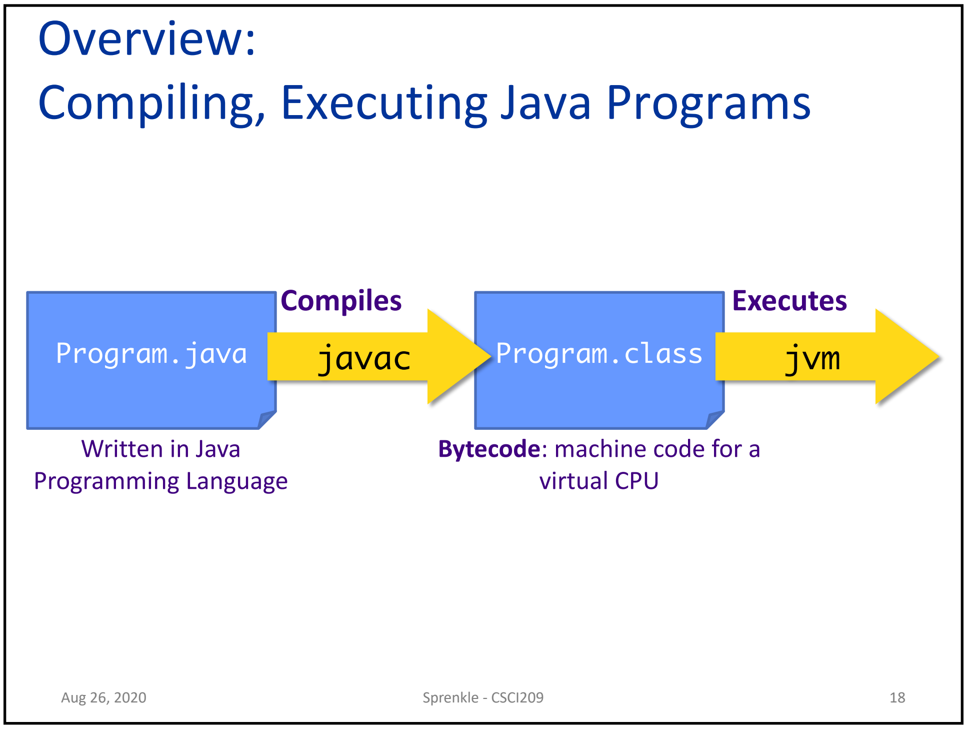
Interpreter

python

output

1. Syntax validation
   - exit if not valid
2. Translate Python code to Python bytecode
   - Bytecode stored in __pycache__ directory
3. Python virtual machine execute Python bytecode

17



Overview: Compiling, Executing Java Programs

Program.java

Written in Java Programming Language

Compiles

javac

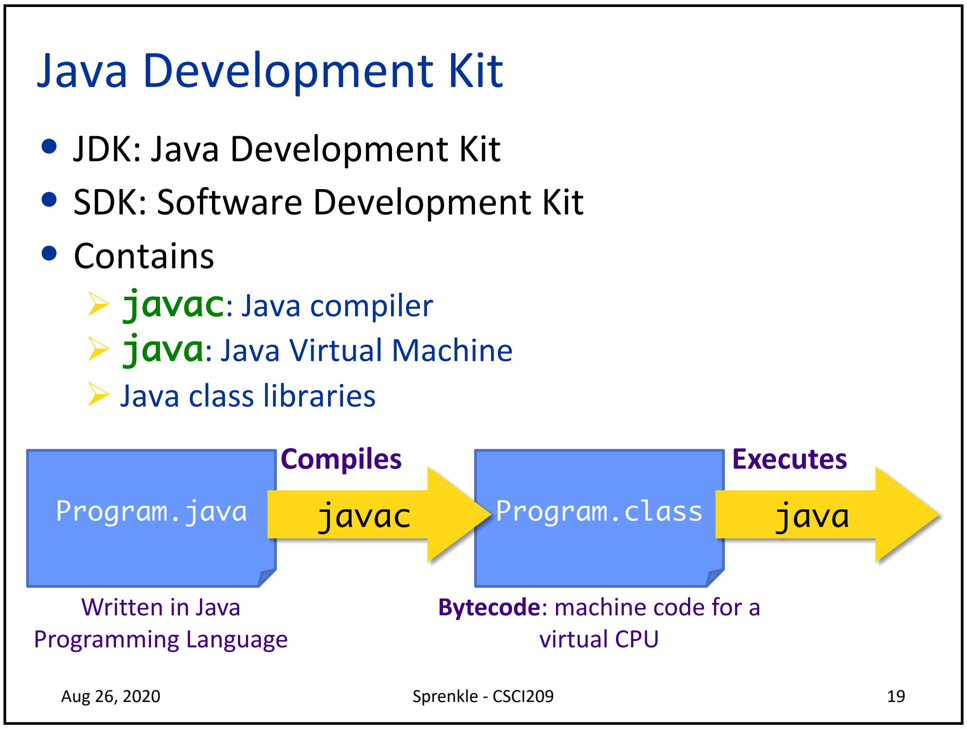
Program.class

Bytecode: machine code for a virtual CPU

Executes

jvm

18



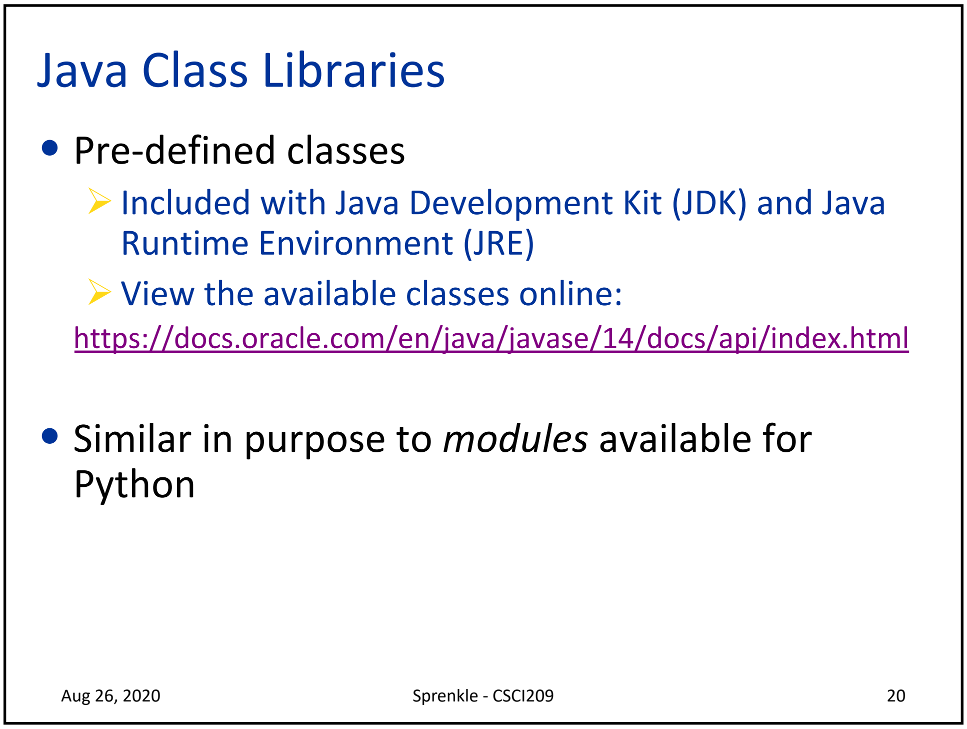
Java Development Kit

- JDK: Java Development Kit
- SDK: Software Development Kit
- Contains
  - `javac`: Java compiler
  - `java`: Java Virtual Machine
  - Java class libraries

Program.java
Compiles
javac
Program.class
Executes
java

Written in Java Programming Language

Bytecode: machine code for a virtual CPU

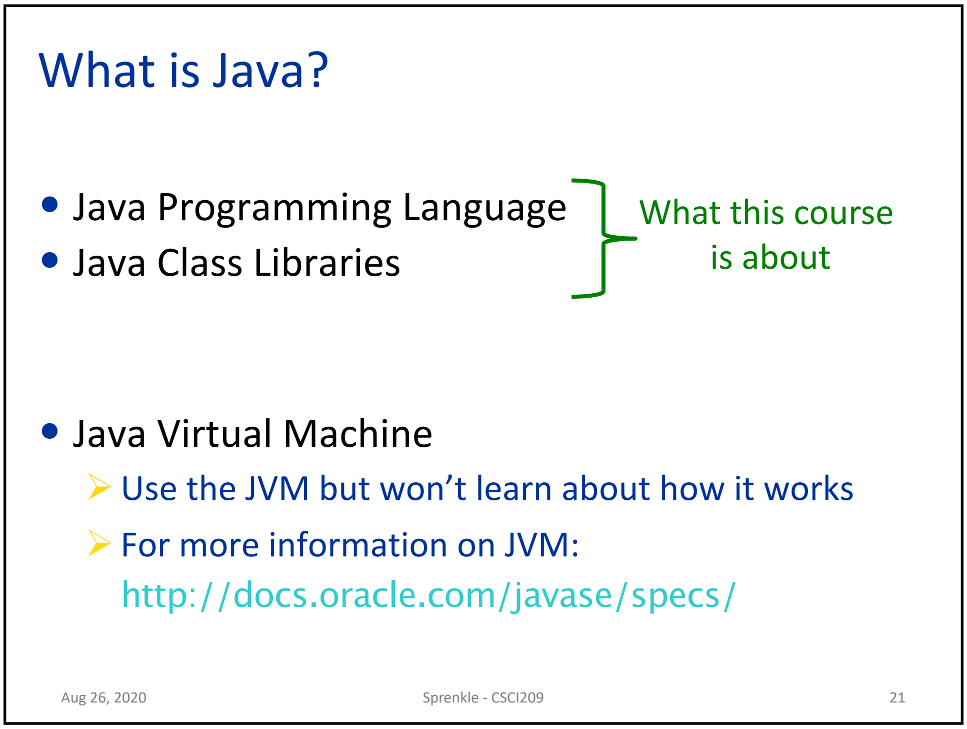


Java Class Libraries

- Pre-defined classes
  - Included with Java Development Kit (JDK) and Java Runtime Environment (JRE)
  - View the available classes online:

  https://docs.oracle.com/en/java/javase/14/docs/api/index.html

- Similar in purpose to *modules* available for Python



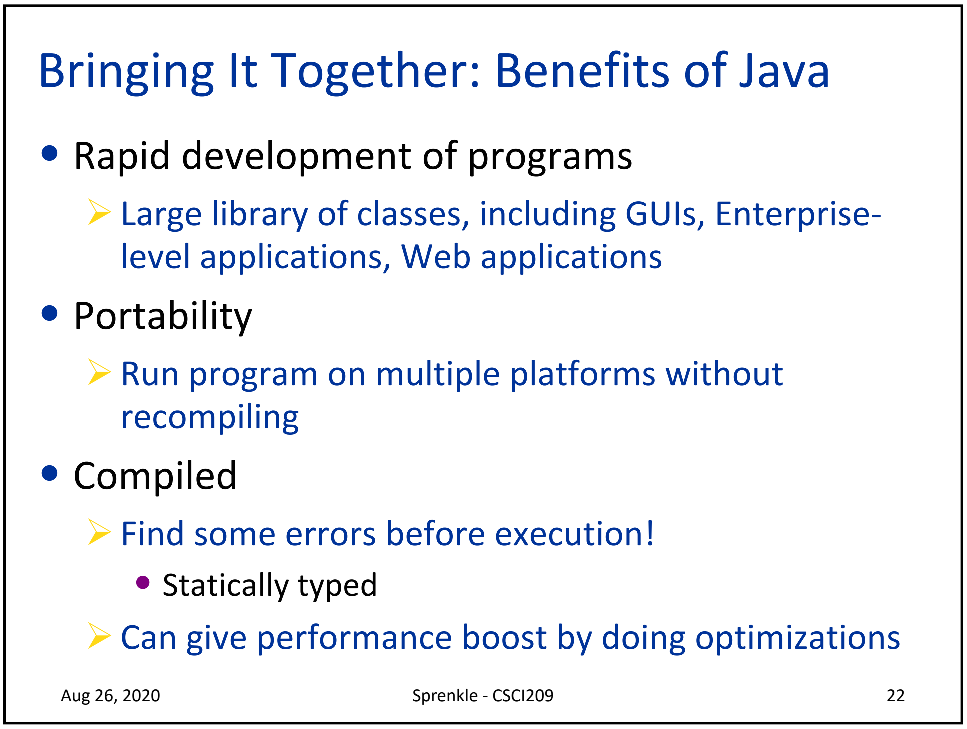
What is Java?

- Java Programming Language
- Java Class Libraries

What this course is about

- Java Virtual Machine
  - Use the JVM but won’t learn about how it works
  - For more information on JVM: http://docs.oracle.com/javase/specs/

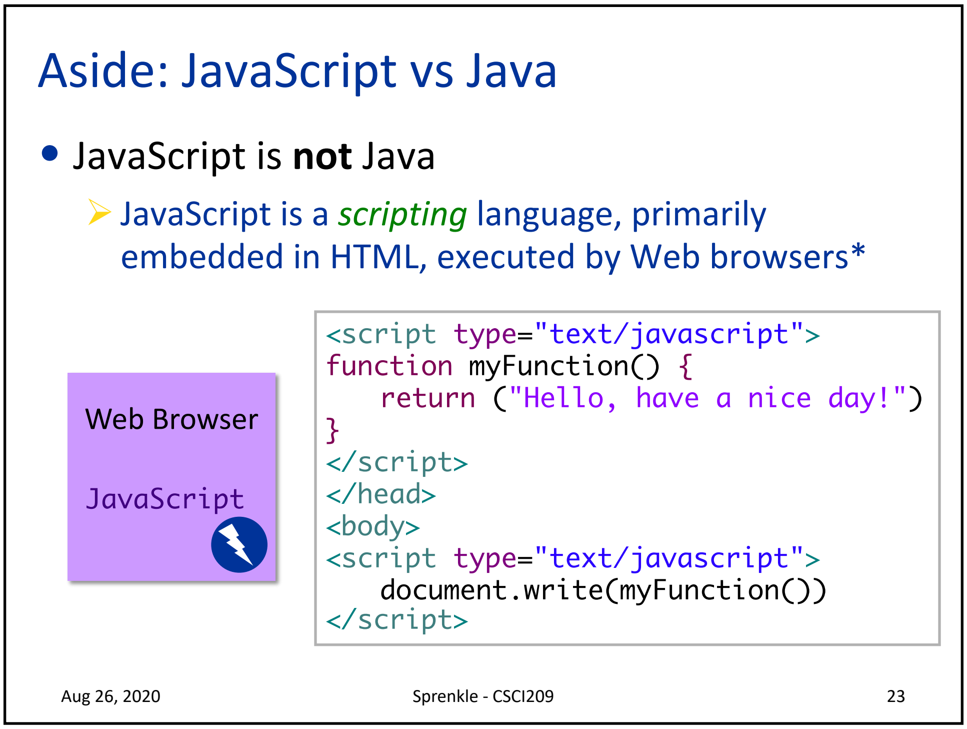
21



Bringing It Together: Benefits of Java

- Rapid development of programs
  - Large library of classes, including GUIs, Enterprise-level applications, Web applications
- Portability
  - Run program on multiple platforms without recompiling
- Compiled
  - Find some errors before execution!
    - Statically typed
  - Can give performance boost by doing optimizations

22



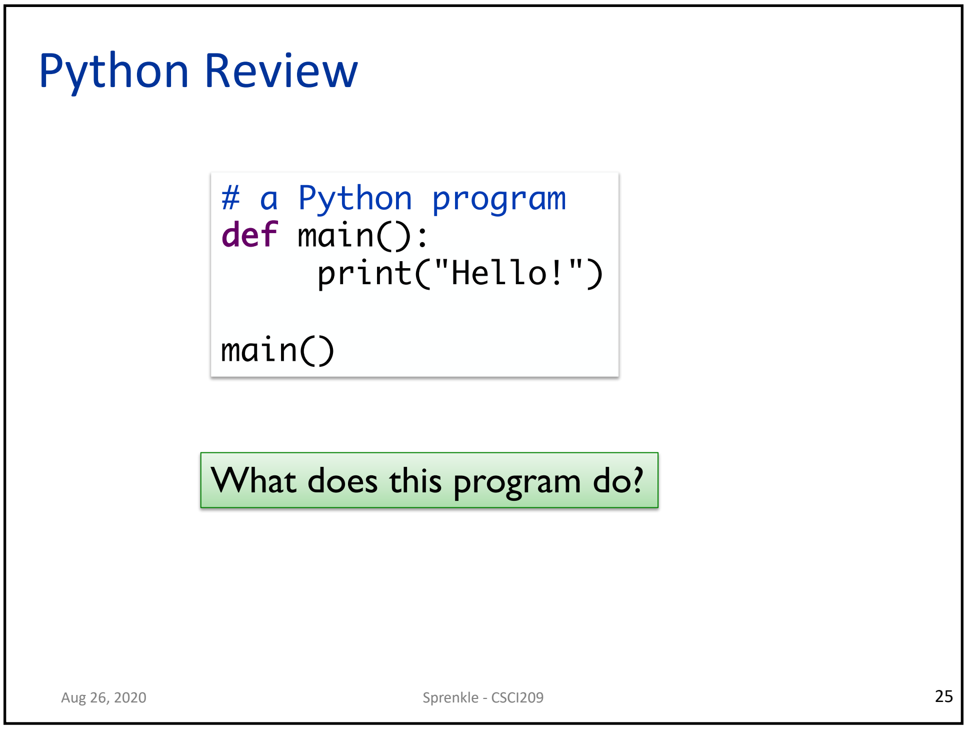
Aside: JavaScript vs Java

- JavaScript is **not** Java
  - JavaScript is a *scripting* language, primarily embedded in HTML, executed by Web browsers*

Web Browser
JavaScript

```
<script type="text/javascript">
function myFunction() {
    return ("Hello, have a nice day!")
}
</script>
</head>
<body>
<script type="text/javascript">
    document.write(myFunction())
</script>
```

LET’S PROGRAM!

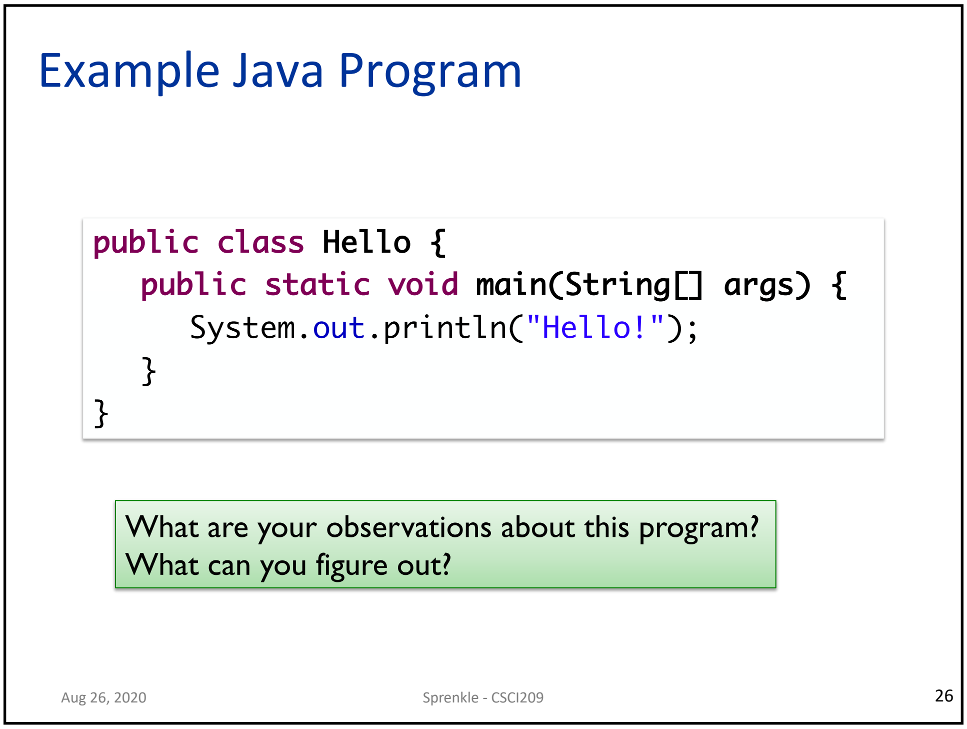


Python Review

```
# a Python program
def main():
     print("Hello!")

main()
```

What does this program do?



Example Java Program

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

What are your observations about this program?
What can you figure out?

Example Java Program

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

- Everything in Java is inside a `class`
  - Java is *entirely* object-oriented*
  - This class is named **`Hello`**

Example Java Program

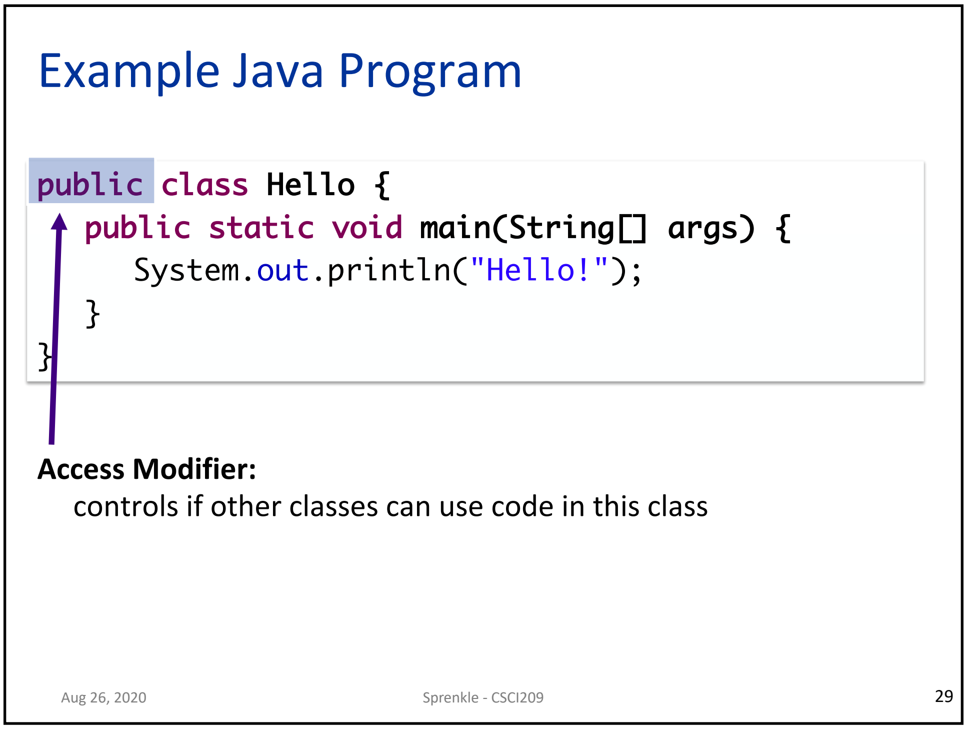
Blocks of code marked with { }

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

Defines the class "`Hello`"

- In general, each Java program file contains **one** class definition*

- Name of the class is name of file
  - E.g., `Hello.java`



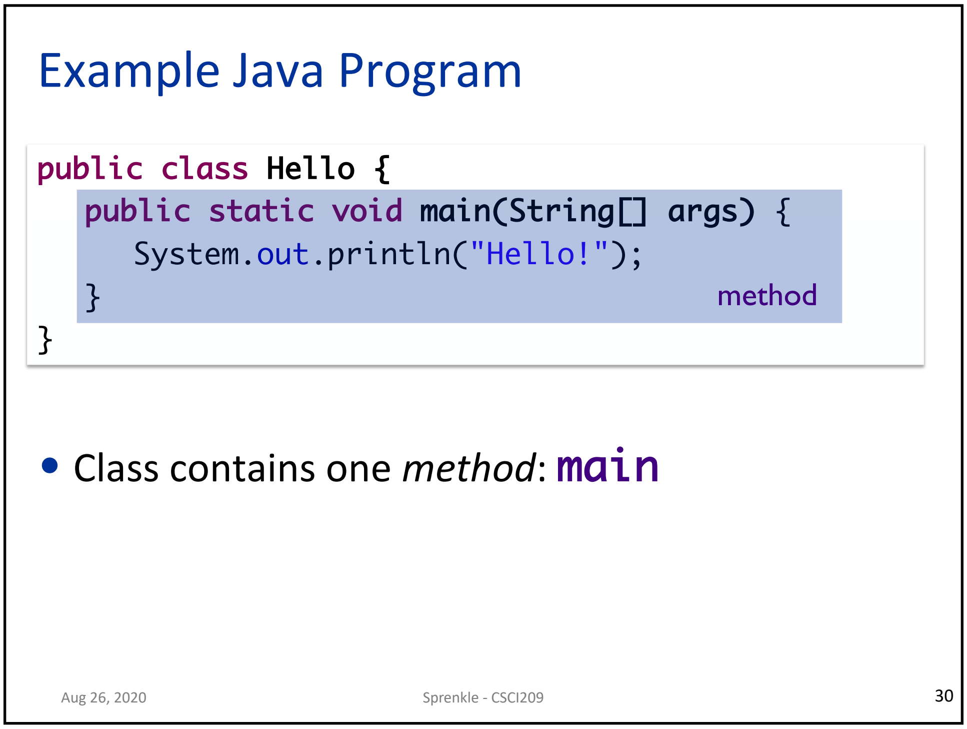
Example Java Program

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

Access Modifier:

controls if other classes can use code in this class

29

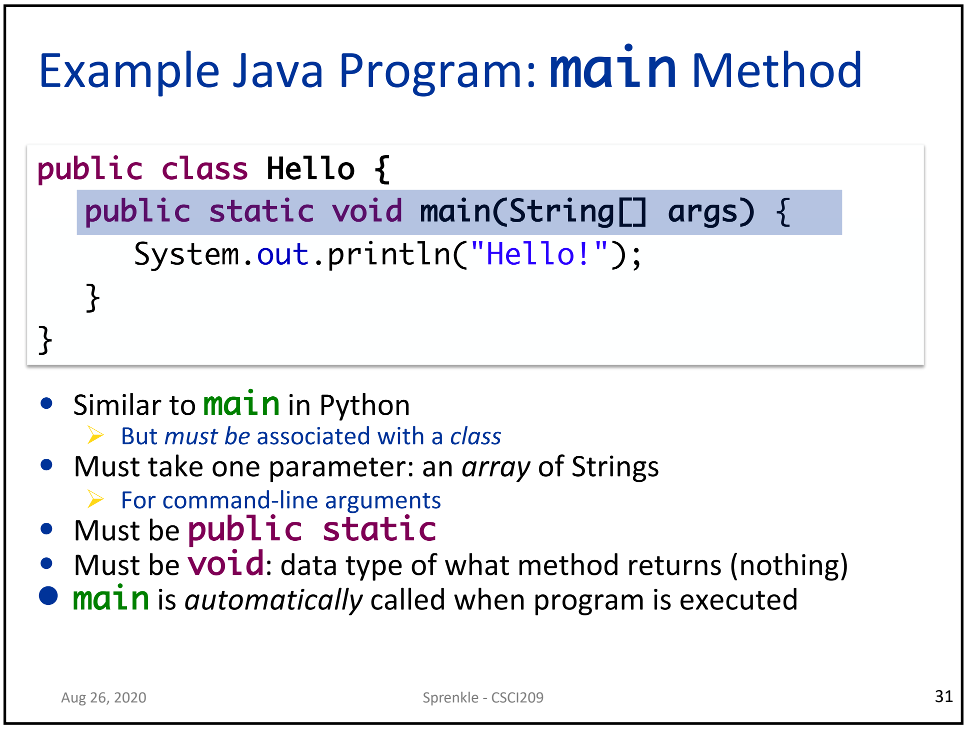


Example Java Program

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

- Class contains one *method*: **`main`**

30

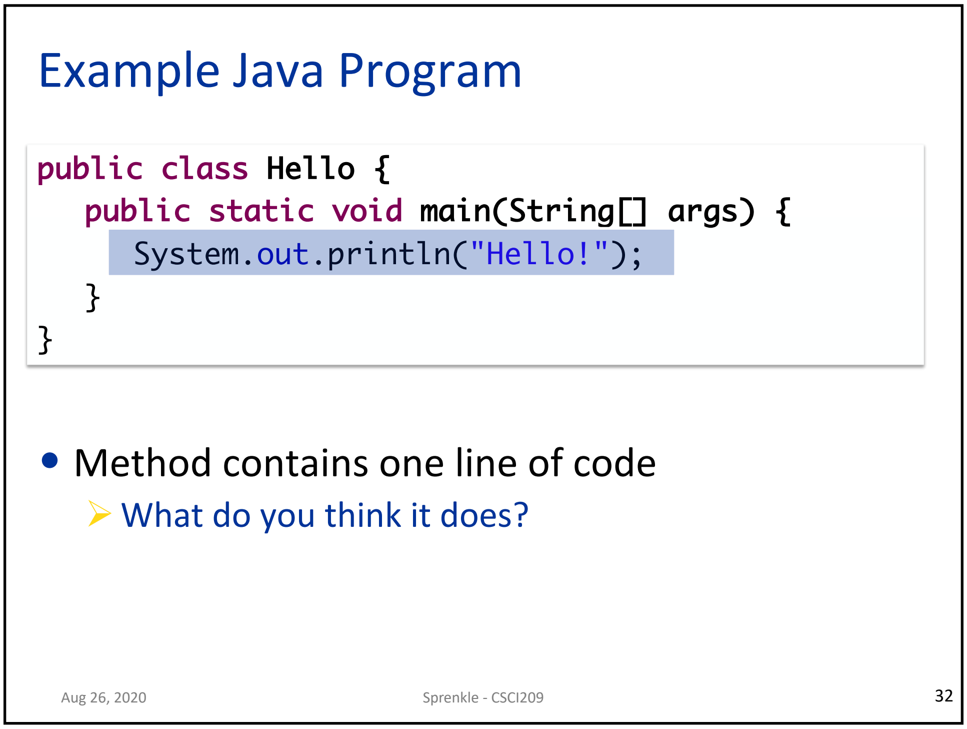


Example Java Program: **main** Method

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

- Similar to **main** in Python
  - But *must be* associated with a *class*
- Must take one parameter: an *array* of Strings
  - For command-line arguments
- Must be **public static**
- Must be **void**: data type of what method returns (nothing)
- **main** is *automatically* called when program is executed

31

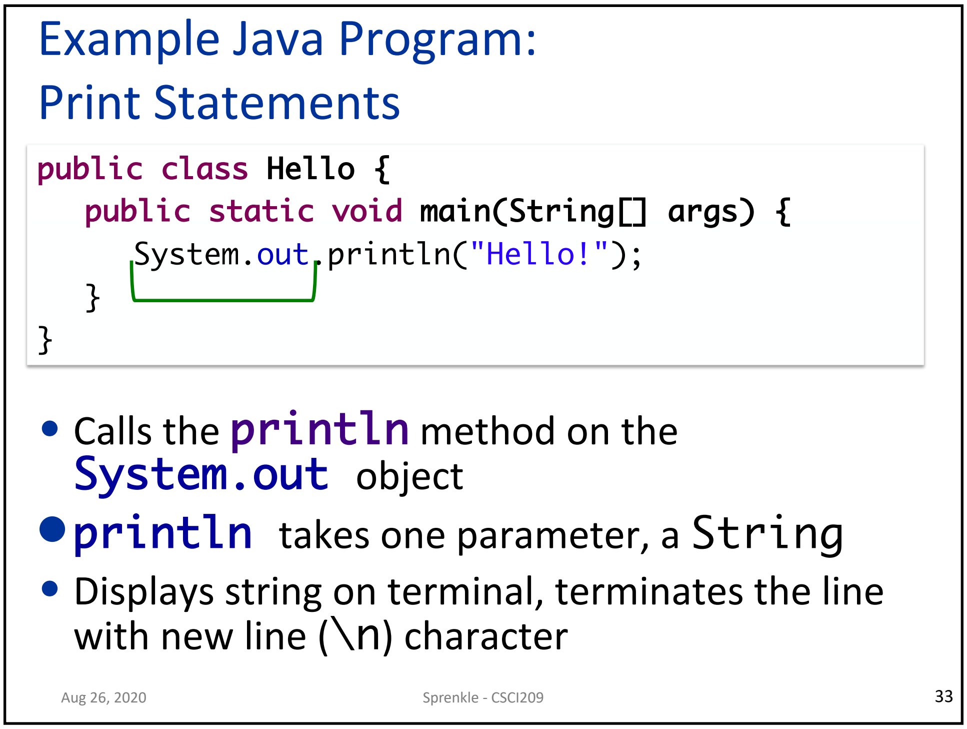


Example Java Program

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

- Method contains one line of code
  - What do you think it does?

32

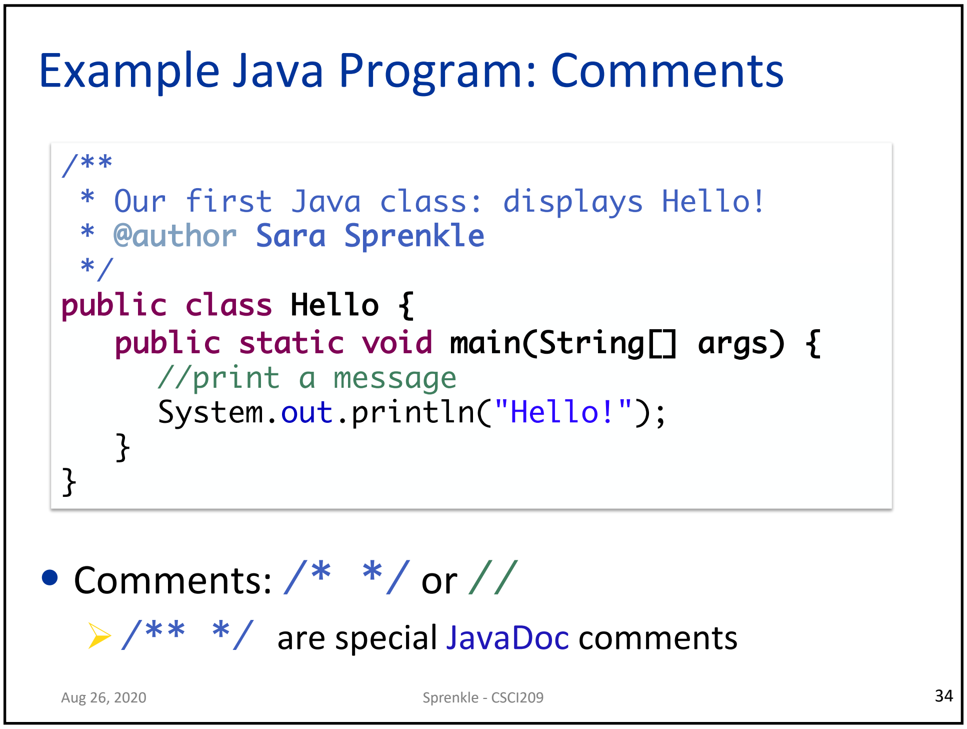


Example Java Program: Print Statements

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

- Calls the **println** method on the **System.out** object
- **println** takes one parameter, a String
- Displays string on terminal, terminates the line with new line (\n) character

33

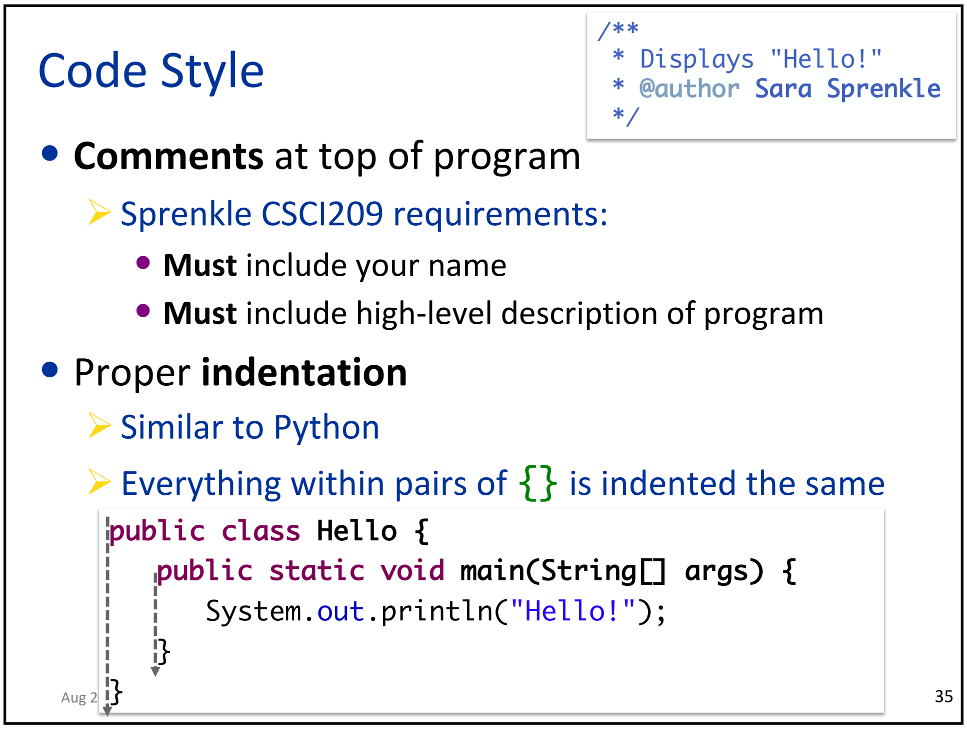


Example Java Program: Comments

```
/**
 * Our first Java class: displays Hello!
 * @author Sara Sprenkle
 */
public class Hello {
   public static void main(String[] args) {
      //print a message
      System.out.println("Hello!");
   }
}
```

- Comments: /* */ or //
    - /** */ are special JavaDoc comments

34



Code Style

```
/**
 * Displays "Hello!"
 * @author Sara Sprenkle
 */
```

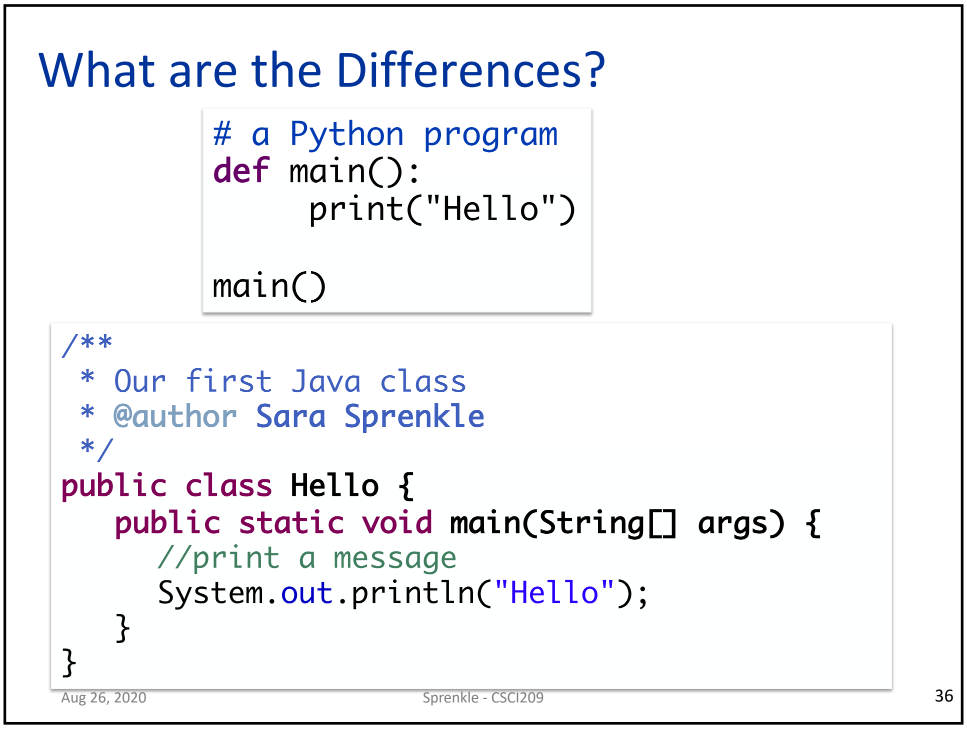
- **Comments** at top of program
  - Sprenkle CSCI209 requirements:
    - **Must** include your name
    - **Must** include high-level description of program
- Proper **indentation**
  - Similar to Python
  - Everything within pairs of {} is indented the same

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

Aug 2

35

35



What are the Differences?

```
# a Python program
def main():
     print("Hello")

main()
```

```
/**
 * Our first Java class
 * @author Sara Sprenkle
 */
public class Hello {
   public static void main(String[] args) {
      //print a message
      System.out.println("Hello");
   }
}
```

Aug 26, 2020 Sprenkle - CSCI209 36

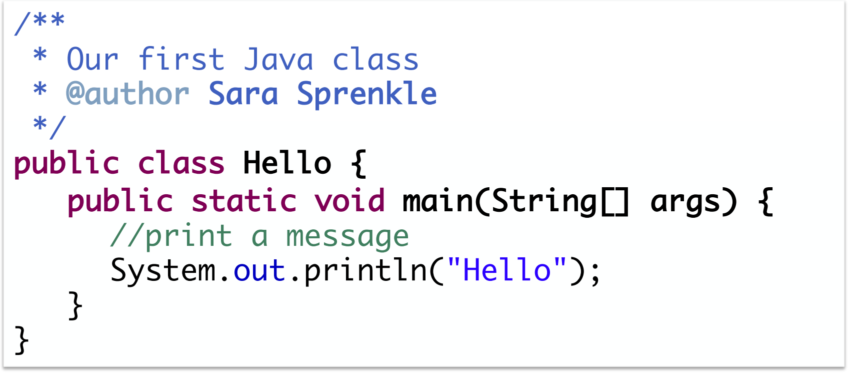
36

Java vs. Python, so far…

- ***Semantics*** the same, ***syntax*** different
  - Blocks of code
  - End statements
- Access modifiers
- Data type declarations
- Class-based programs
- Compiled

We'll see more differences as we go…

Literal Translation to Python Program?



```
/**
 * Our first Java class
 * @author Sara Sprenkle
 */
public class Hello {
   public static void main(String[] args) {
      //print a message
      System.out.println("Hello");
   }
}
```

Translation to Python Program

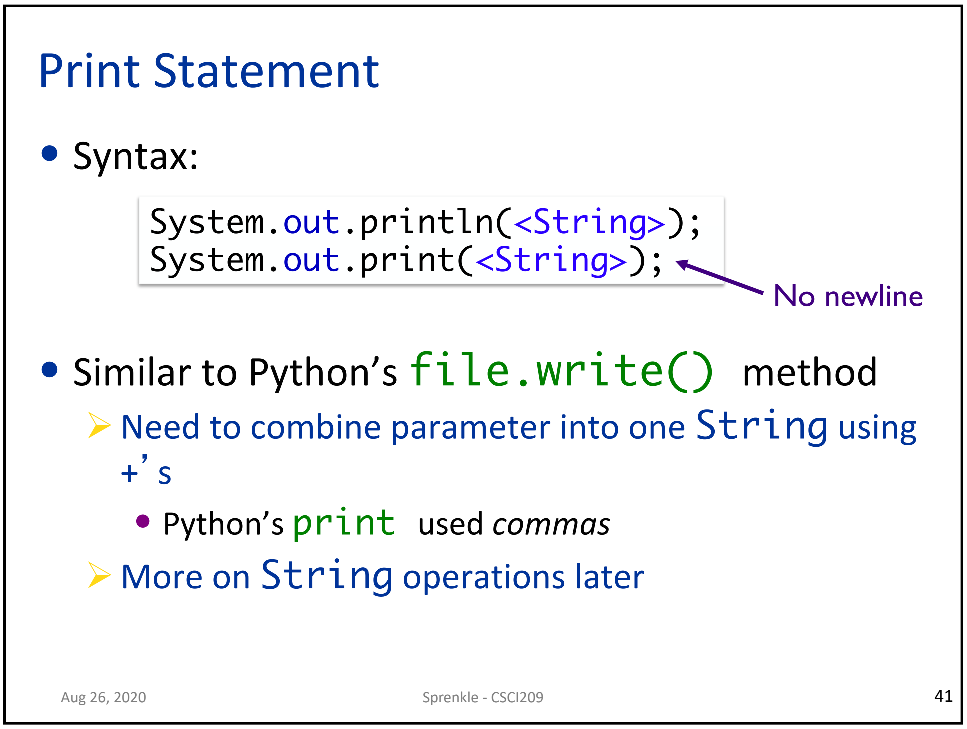
```
class Hello:
   """Our first Python class"""

   def __init__(self):
      # fill in later…

   def main(self):
      print("Hello")
```

Semi-literal translation

JAVA FUNDAMENTALS



Print Statement

- Syntax:

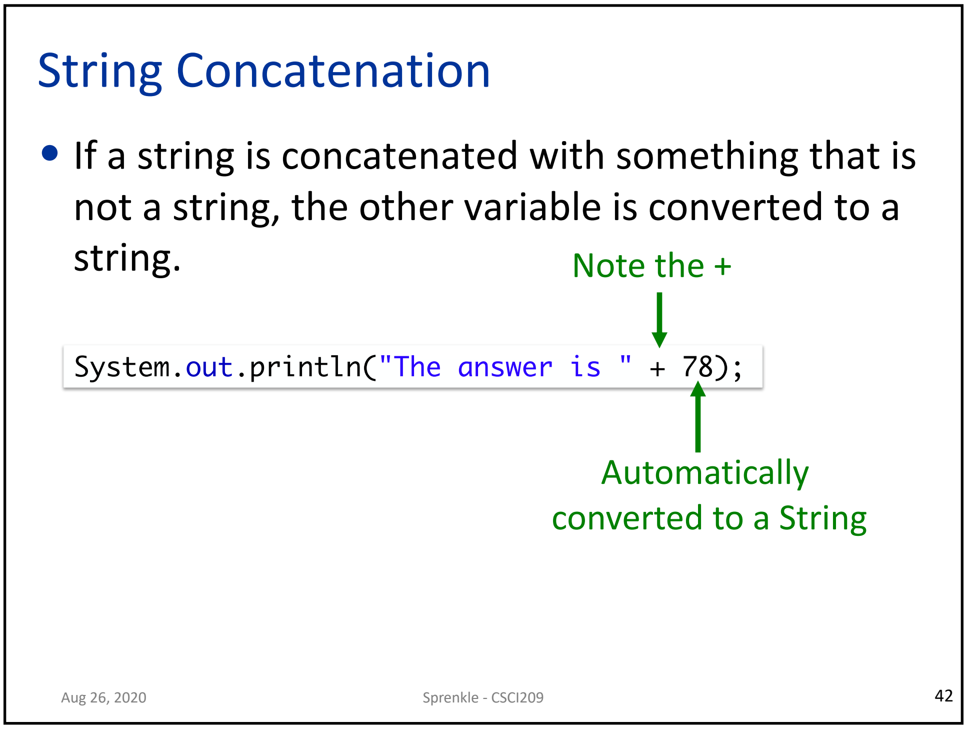
```
System.out.println(<String>);
System.out.print(<String>);
```

No newline

- Similar to Python's `file.write()` method
  - Need to combine parameter into one `String` using +'s
    - Python's `print` used *commas*
  - More on `String` operations later

Aug 26, 2020 Sprenkle - CSCI209 41

41



String Concatenation

- If a string is concatenated with something that is not a string, the other variable is converted to a string.

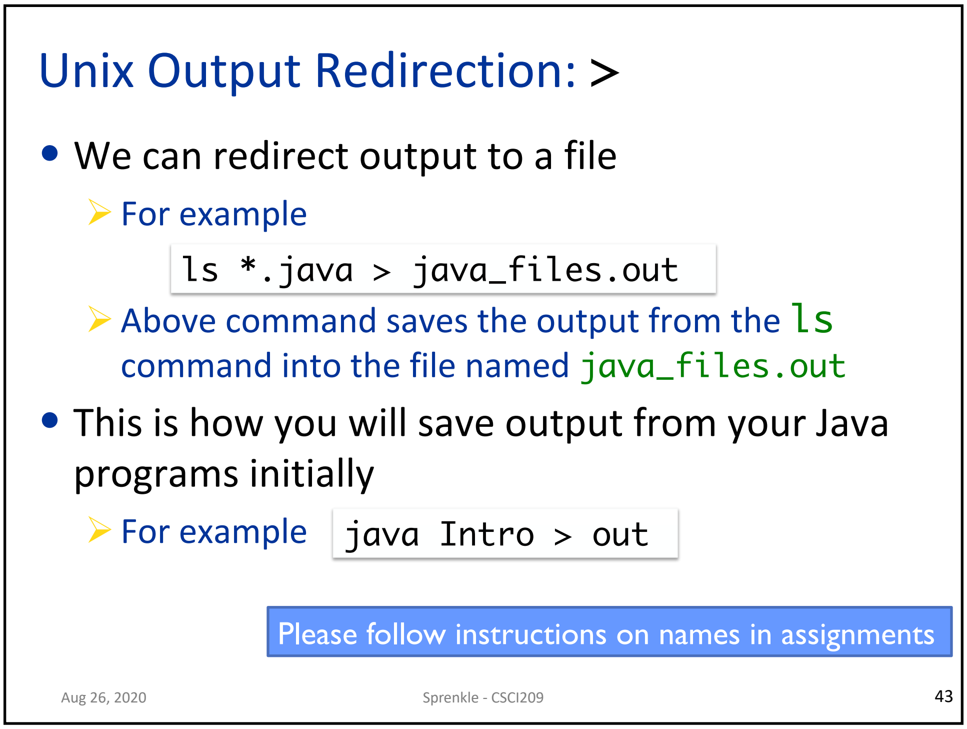
Note the +

```
System.out.println("The answer is " + 78);
```

Automatically converted to a String

Aug 26, 2020 Sprenkle - CSCI209 42

42



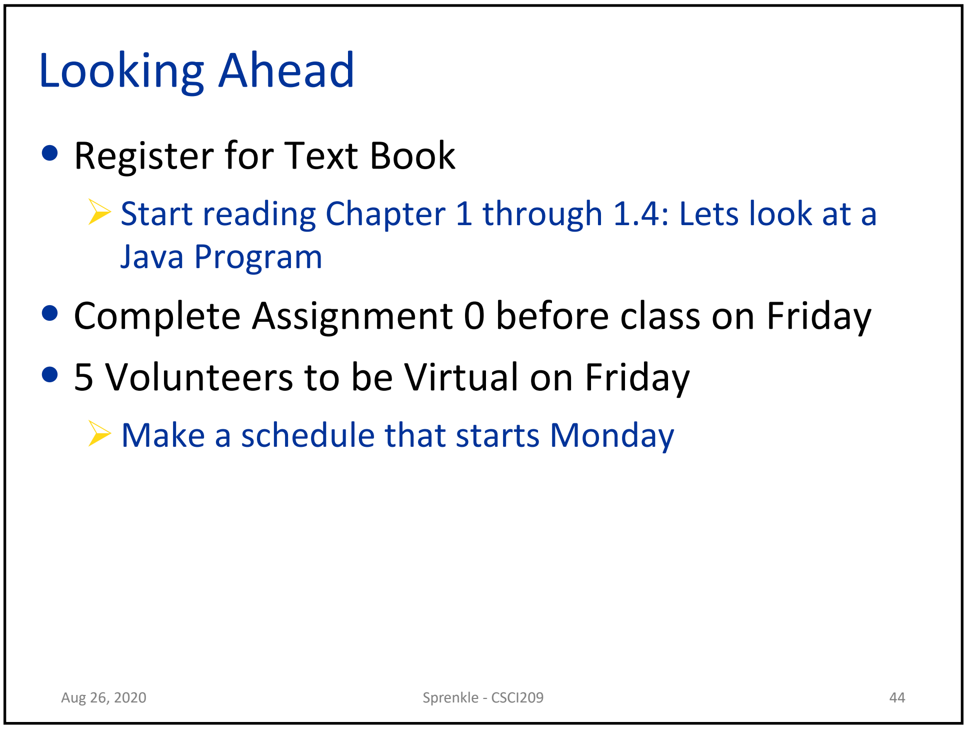
Unix Output Redirection: >

- We can redirect output to a file
  - For example

    ```
    ls *.java > java_files.out
    ```

  - Above command saves the output from the `ls` command into the file named `java_files.out`
- This is how you will save output from your Java programs initially
  - For example `java Intro > out`

Please follow instructions on names in assignments



Looking Ahead

- Register for Text Book
  - Start reading Chapter 1 through 1.4: Lets look at a Java Program
- Complete Assignment 0 before class on Friday
- 5 Volunteers to be Virtual on Friday
  - Make a schedule that starts Monday