Recitation 1

C3: Harsh, Daniel and Viren

Introduction

- Viren Sabhnani
  - Sophomore
  - CS Major
  - vsabhnani@gatech.edu

Recitation Outline

- Announcements
- Basic Java Syntax
- Values & Variables
- Printing to the Console
- Boolean Values, Comparisons & Logical Operators
- If-else Statements
- Checkstyle

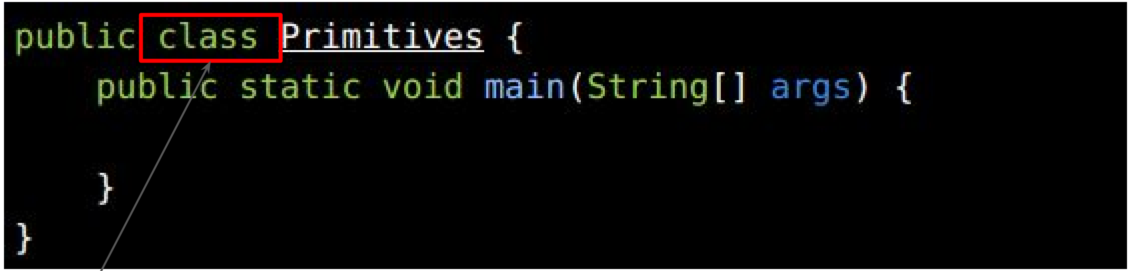
Announcements

- Homework 0
    - Due Thursday (18 Jan) at 11:59PM
    - http://cs1331.gatech.edu/spring2018/hw0/hw0.html
    - Submit on Canvas
    - Redownload your submission to verify
    - Don’t use submission comments on Canvas
- Office hours

Announcements

- Homework 0
- Office hours
  - TAs are signing up this week
  - CoC 107
  - https://cs1331.org/officehours.html
  - Can go to any TA

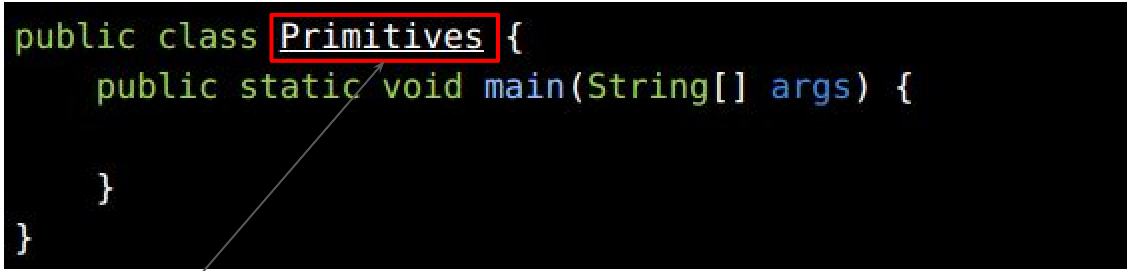
Intro to Java: Anatomy of a Program



```
public class Primitives {
    public static void main(String[] args) {

    }
}
```

Every program is a class in Java

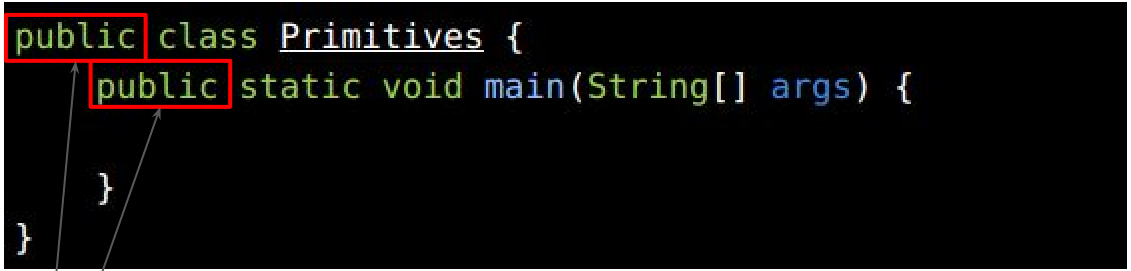
Intro to Java: Anatomy of a Program



```
public class Primitives {
    public static void main(String[] args) {

    }
}
```

Must match filename (excluding .java)

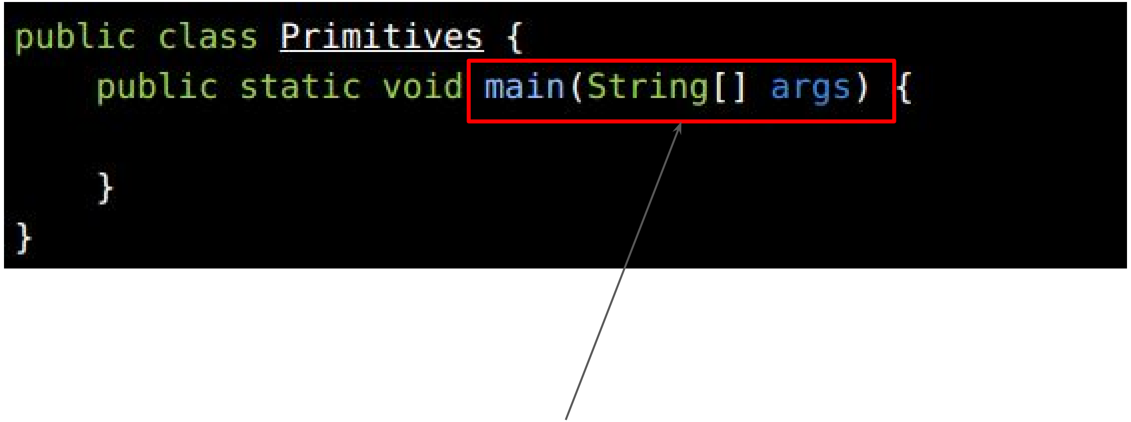
Intro to Java: Anatomy of a Program



```
public class Primitives {
    public static void main(String[] args) {

    }
}
```

Anyone can see this class

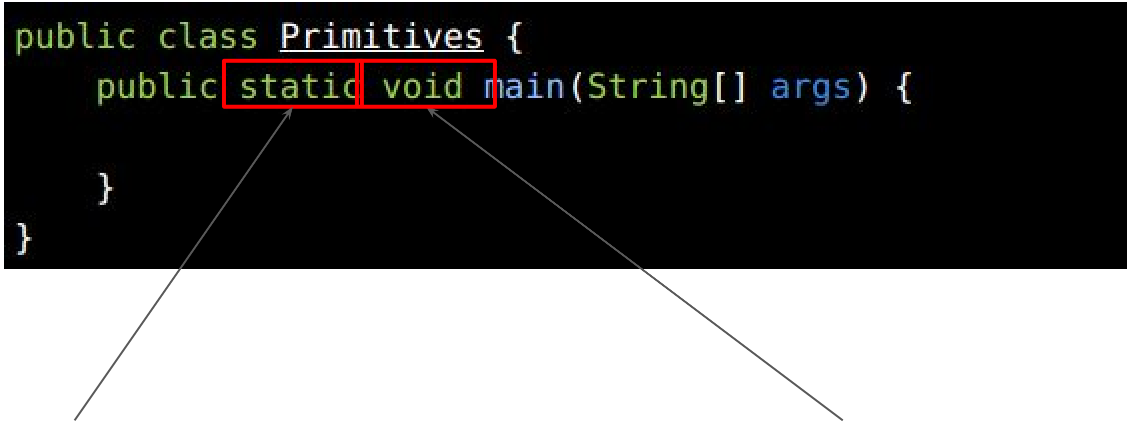
Intro to Java: Anatomy of a Program



```
public class Primitives {
    public static void main(String[] args) {

    }
}
```

A function, called *main*, that takes one parameter, *args*.

Intro to Java: Anatomy of a Program



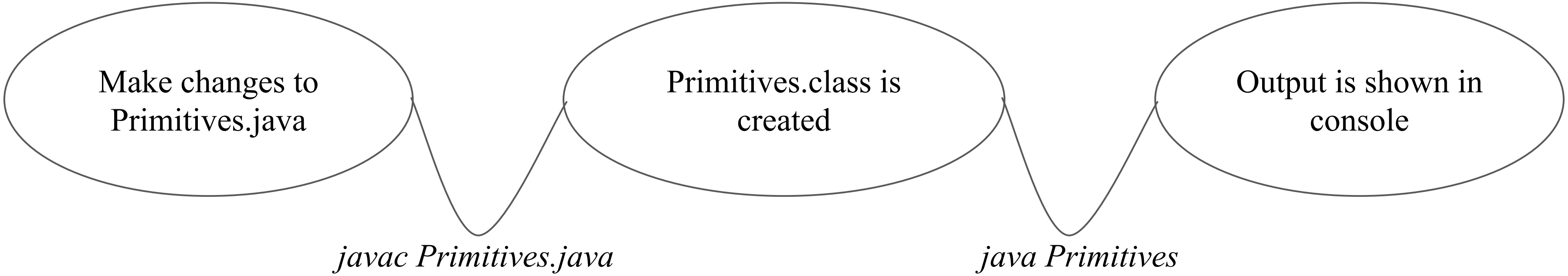
```
public class Primitives {
    public static void main(String[] args) {

    }
}
```

Belongs to the class as a whole

Doesn’t return anything

Intro to Java: Compiling

Compile after **EVERY CHANGE**



Make changes to Primitives.java

javac Primitives.java

Primitives.class is created

java Primitives

Output is shown in console

Variable Declaration and Initialization

- When declaring a variable, you’re saying that the variable exists, but you do not assign a value to it.
- Formatted like this: **[type] [var name];**
    - Ex] **int num; char c; double money;**
- When assigning a value to a variable (aka initialization)…
    - a) You can declare it first, and then assign a value to the variable.
        - Ex] **int num; num = 4; (*notice that ‘=’ means assignment, not value equality*)**
    - b) You can combine the declaration and the assignment into one statement
        - Ex] **int num = 4; char c = ‘c’; double money = 4.50;**

Types - Primitives

- On the previous slide, we spoke about types.
- Each variable gets a type.
- A variable can be primitive
- Important Primitive Types
    - int (an integer)
        - int x = 2; int y = 3; int sum = x + y;
    - char (character)
        - char letter = ‘a’;
    - double (floating point)
        - double money = 5.10;
    - float (floating point too, less precise than double)
        - float y = 3.14f;
    - boolean (true or false)
        - boolean todayIsWednesday = false; boolean todayIsTuesday = true;

Type Conversion

- You can assign to a variable a value that is less precise than the indicated type.
- Example] double x = 4;
- Example] double y = 3.14f;
- You cannot assign to a variable a value that is more precise than the indicated type
- Example] int z = 4.0;
- example] float b = 5.27;
- **Order of precision, from least precise to most precise: int, float, double;**

Types - Strings

- As opposed to primitives, types can be represented by a class.
- String is a class that represents a sequence of characters.
- When using it as a type in variable initialization, the ‘S’ in String is capitalized.
- Why? Because it is a class.
- Example] String str = “Hello”; **(double quotations are necessary)**
- Example] String num = “12345”;
- Example] String newNum = “12.345”;
- The following cannot be done: String character = ‘c’; Why?

String Concatenation

- Let’s say we want to combine Strings.

Example] String s = “Hello”; String x = “World”;

- What would “System.out.println(s + x);” print?
- How about “System.out.println(s + “ ” + x);”

Example] String s = “Hello”; int num = 4; int otherNum = 5;

- What would “System.out.println(s + num + otherNum);” print?
- How about “System.out.println(s + (num + otherNum));” ?

Printing to the Console

- System.out.print
- System.out.println
- System.out.printf

Comparisons and Logical Operators

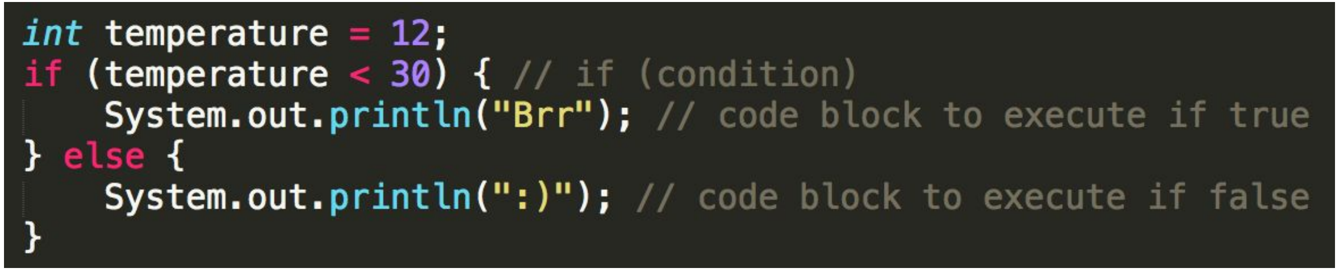
- You can compare two primitives of the same type using:
  - >, <
  - <=, >=
  - ==, != (value equality for primitives, reference equality (aka alias) for objects)
- Logical Combinations
  - [boolean expression] **&&** [boolean expression] → ‘AND’ (if both expressions are true, evaluates to true; false otherwise)
  - [boolean expression] **||** [boolean expression] → ‘OR’ (if at least one expression is true, evaluates to true; false otherwise)
- Examples of Boolean Expressions
  - (2 > 3)
  - todayIsWednesday == false
  - (2 % 2 >= 0)

If-else Statements

- One of Java’s **control flow statements**
  - Will learn more about these later
  - Used to alter regular top-to-bottom flow of execution
- Basic example
- Simplification for **single statements**
- Nested
- Multi-way
- Ternary

If-else Statements

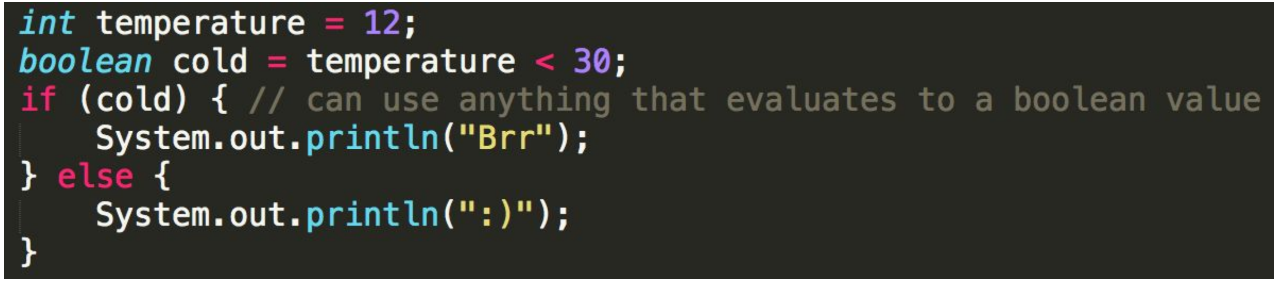
- One of Java’s **control flow statements**
  - Will learn more about these later
  - Used to alter regular top-to-bottom flow of execution
- Basic example
- Simplification for **single statements**
- Nested
- Multi-way
- Ternary



```
int temperature = 12;
if (temperature < 30) { // if (condition)
    System.out.println("Brr"); // code block to execute if true
} else {
    System.out.println(":)"); // code block to execute if false
}
```

If-else Statements

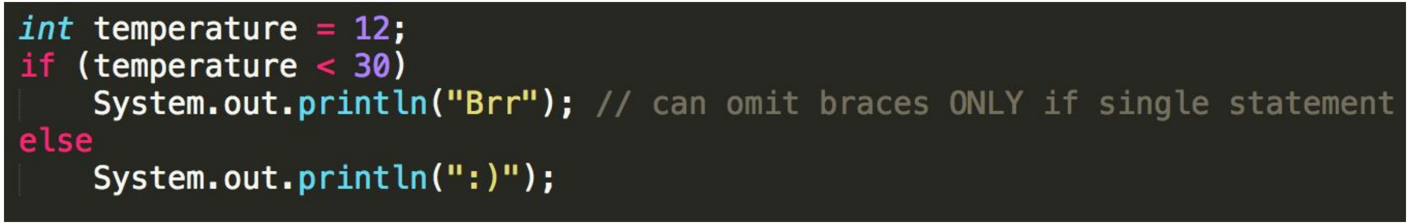
- One of Java’s **control flow statements**
    - Will learn more about these later
    - Used to alter regular top-to-bottom flow of execution
- Basic example
- Simplification for **single statements**
- Nested
- Multi-way
- Ternary



```
int temperature = 12;
boolean cold = temperature < 30;
if (cold) { // can use anything that evaluates to a boolean value
    System.out.println("Brr");
} else {
    System.out.println(":)");
}
```

If-else Statements

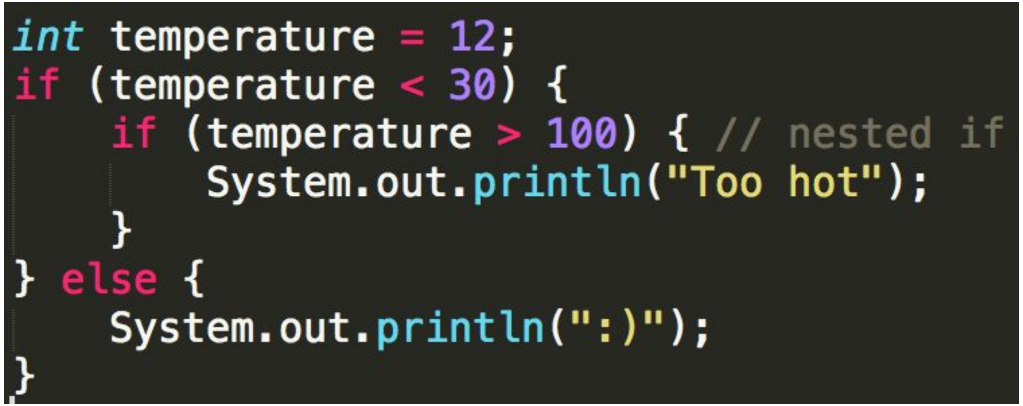
- One of Java’s **control flow statements**
    - Will learn more about these later
    - Used to alter regular top-to-bottom flow of execution
- Basic example
- Simplification for **single statements**
- Nested
- Multi-way
- Ternary



```
int temperature = 12;
if (temperature < 30)
    System.out.println("Brr"); // can omit braces ONLY if single statement
else
    System.out.println(":)");
```

If-else Statements

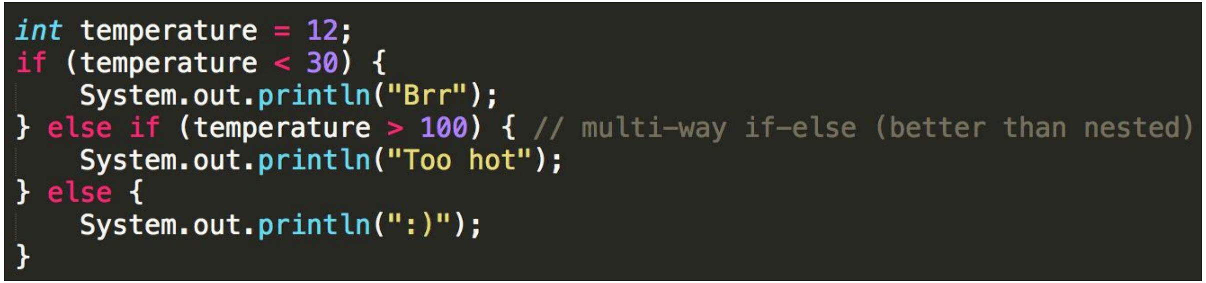
- One of Java’s **control flow statements**
  - Will learn more about these later
  - Used to alter regular top-to-bottom flow of execution
- Basic example
- Simplification for **single statements**
- Nested
- Multi-way
- Ternary



```
int temperature = 12;
if (temperature < 30) {
    if (temperature > 100) { // nested if
        System.out.println("Too hot");
    }
} else {
    System.out.println(":)");
}
```

If-else Statements

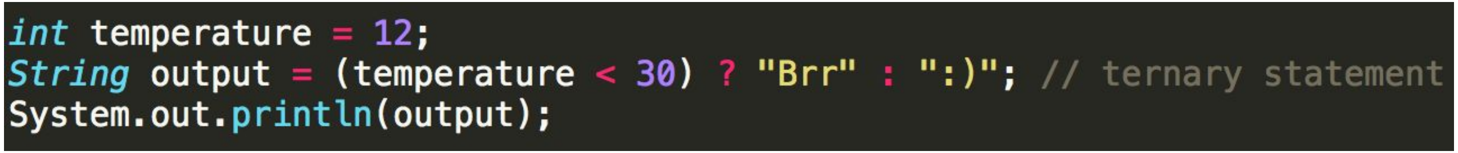
- One of Java’s **control flow statements**
  - Will learn more about these later
  - Used to alter regular top-to-bottom flow of execution
- Basic example
- Simplification for **single statements**
- Nested
- Multi-way
- Ternary



```java
int temperature = 12;
if (temperature < 30) {
    System.out.println("Brr");
} else if (temperature > 100) { // multi-way if-else (better than nested)
    System.out.println("Too hot");
} else {
    System.out.println(":)");
}
```

If-else Statements

- One of Java’s **control flow statements**
  - Will learn more about these later
  - Used to alter regular top-to-bottom flow of execution
- Basic example
- Simplification for **single statements**
- Nested
- Multi-way
- Ternary



```java
int temperature = 12;
String output = (temperature < 30) ? "Brr" : ":)"; // ternary statement
System.out.println(output);
```

Checkstyle

- http://cs1331.gatech.edu/cs1331-style-guide.html
  - Download checkstyle-6.2.2.jar
- How to use:
  - Move it into the directory of the file you want to checkstyle
  - java -jar checkstyle-6.2.2.jar MyJavaFile.java