Methods

CSCI 201L

Jeffrey Miller, Ph.D.
jeffrey.miller@usc.edu

HTTP://WWW-SCF.USC.EDU/~CSCI201
Outline

- Methods
- Program
Method Description

- A method in Java is equivalent to a function in C++
- All methods in Java must exist within a class
- Other than constructors (which have no return type), all methods must return a value or have a return type of `void`
  - If a method has a return type, it must specify that as the word immediately preceding the name of the method (i.e. all other modifiers must be before the return type)
  - The method must also return a value along all paths of execution through it or the code will not compile
- To call a method, you must have an instance of the class in which the method is declared, unless the method is declared static
- If a method calls itself, it is called a recursive method
public class Lemonade {
    private int numLemons;
    public int getNumLemons() {
        return numLemons;
    }
    public void setNumLemons(int numLemons) {
        this.numLemons = numLemons;
    }
    public static void main(String [] args) {
        Lemonade glass = new Lemonade();
        glass.setNumLemons(3);
        System.out.println("lemons = " + glass.getNumLemons());
    }
}
Method/Variable Modifiers

- Static (methods and variables)
  - Only one instance of static variables or methods exist in memory
  - You do not need an instance of a class to access static members of a class
    - The access modifiers still apply though
  - Non-static variables can only be accessed by non-static methods
    - Why?

- Final (methods and variables)
  - A final variable can only be initialized once, either inline or in the constructor
  - A final method cannot be overridden by a subclass

- Abstract (methods and classes)
  - An abstract method must be overridden in a subclass or the subclass becomes abstract

- Synchronized (methods)
  - A synchronized method obtains a lock on the object so no other synchronized method can execute until the first one terminates

- Volatile (variables)
  - A volatile variable will not allow a cached value to be used in threads – all threads will get the same value when accessing a volatile variable

- Transient (variables)
  - A transient variable will not persist when using serialization
Constructors

- Constructors are called when a class is instantiated.
- Constructors typically initialize member variables, though they can do anything.
- Constructors can be overridden.
  › If any constructor is explicitly created in the class, the default constructor is no longer created by the compiler.
- Constructors have no return type.
public class Apple {
    private String color;
    private int numApples;
    public Apple(String color, int numApples) {
        this.color = color;
        this.numApples = numApples;
    }
    public Apple(String color) {
        this.color = color;
        this.numApples = 1;
    }
    public static void main(String[] args) {
        Apple greenApples = new Apple("green", 3);
        Apple bushel = new Apple("red", 126); // medium-sized :)
        Apple yellowApples = new Apple("yellow");
        Apple app = new Apple(); // ?
    }
}
Outline

- Methods
- Program
Write a program that iteratively and recursively finds the Fibonacci number specified by the user when prompted. Here is a sample execution with user input bolded.

```
c:\> java csci201.Fibonacci
What Fibonacci number would you like? 8
Iteratively: Fibonacci number 8 is 21.
Recursively: Fibonacci number 8 is 21.
c:\>
```