

COMP 110-001

Computer Basics

Yi Hong

May 13, 2015

Today

- Hardware and memory
- Programs and compiling
- Your first program

Before Programming

- Need to know basics of a computer
- Understand what your program is doing
- Talk intelligently about computers

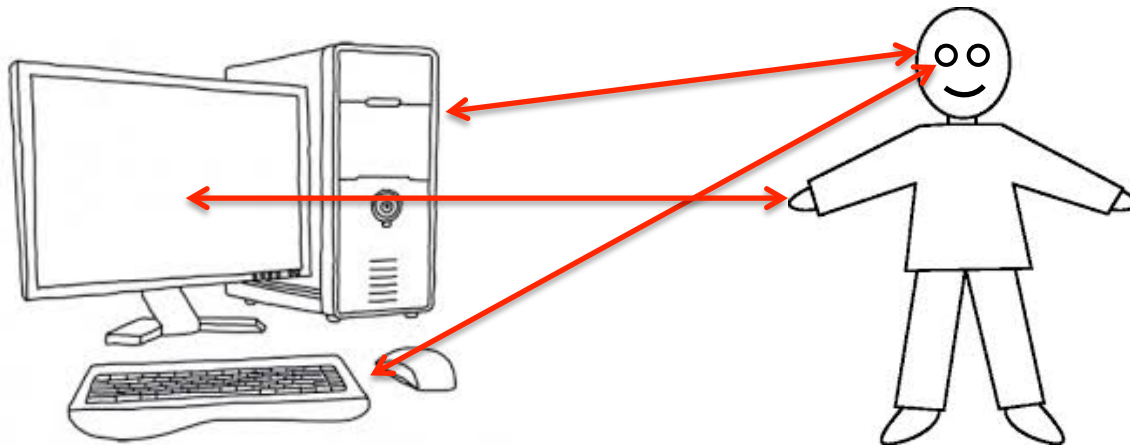
Computer System

- Hardware: Physical components for computation
 - CPU, Memory, Keyboard
- Software: Programs that give instructions to the computer
 - Windows, Office, Games, Eclipse ...



Hardware

- Main components of a computer
 - CPU (Central Processing Unit): Performs the instructions in a program
 - Memory: Holds programs and data
 - Input devices: Provide data to a computer
 - Output devices: Display data carried out by a computer



CPU – the “Brain”

- Central processing unit
 - Clock speed: GHz, how many clock cycles a CPU can perform per second
(1GHz = 1 billion CPU cycles per second)
 - Dual core: Multiple processing units per CPU

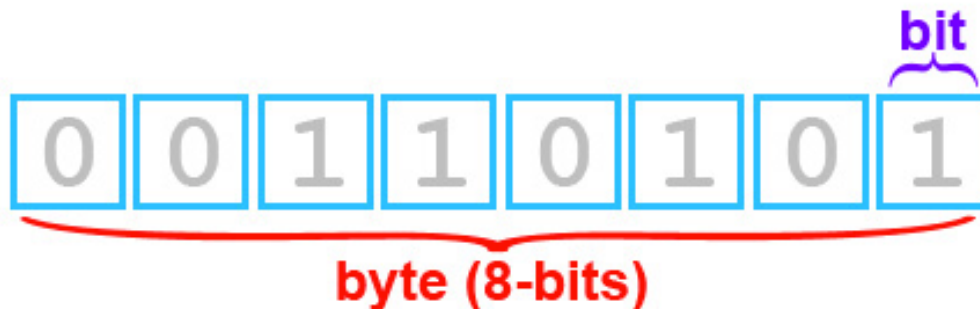


Memory – the Brain

- Holds data for the computer
- Main memory
 - Holds the current program and much of the data that the program is manipulating
 - Volatile, disappears when shutting down the computer
- Auxiliary memory (secondary memory)
 - Hard disk drives, CDs, flash drives ...
 - Exists even when the computer's power is off

RAM (Random Access Memory)

- The main memory
- 4 gigabytes of RAM
 - A bit: the basic unit of information in computing (binary digit, 0 or 1)
 - A byte: A quantity of memory, 8 bits



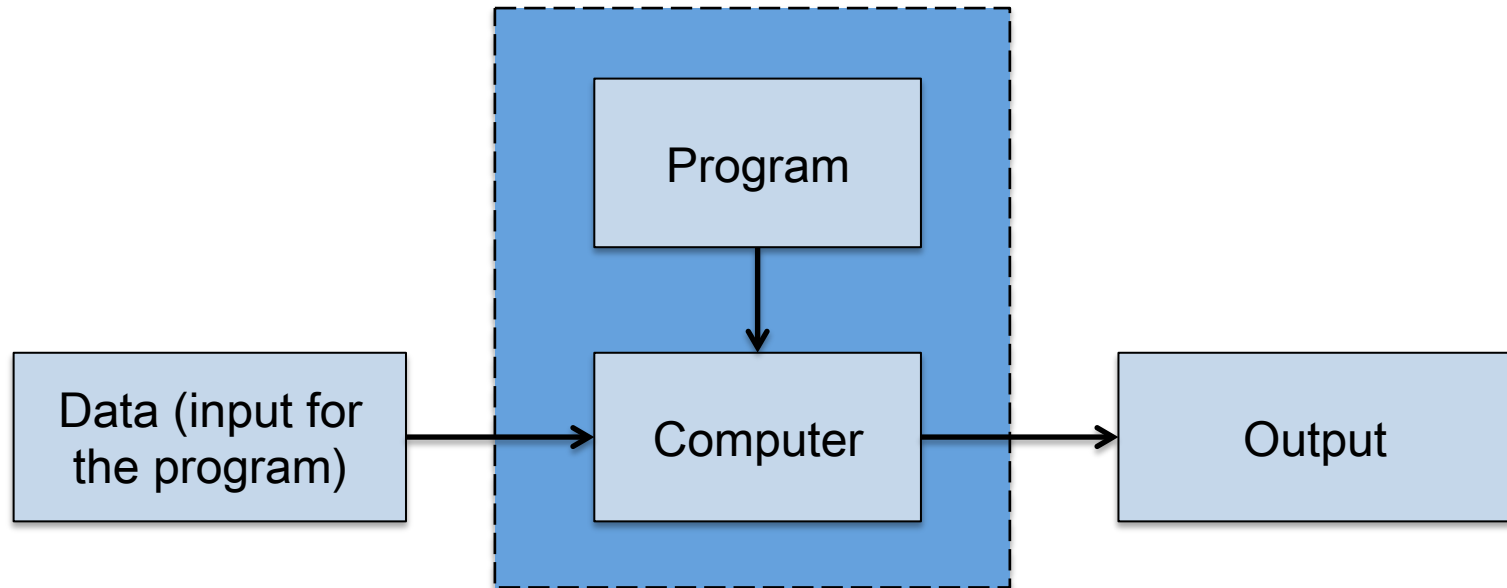
$$2^0 + 2^2 + 2^4 + 2^5 = 53$$

Measuring Memory

- Both main memory and auxiliary memory are measured in bytes
 - 1 byte = 8 bits
 - 1 Kilobyte (KB) = 1024 bytes
 - 1 Megabyte (MB) = 1024 KB = $1024 * 1024$ bytes
 - 1 Gigabyte (GB) = 1024 MB
 - Terabyte (TB), Petabyte (PB) ...

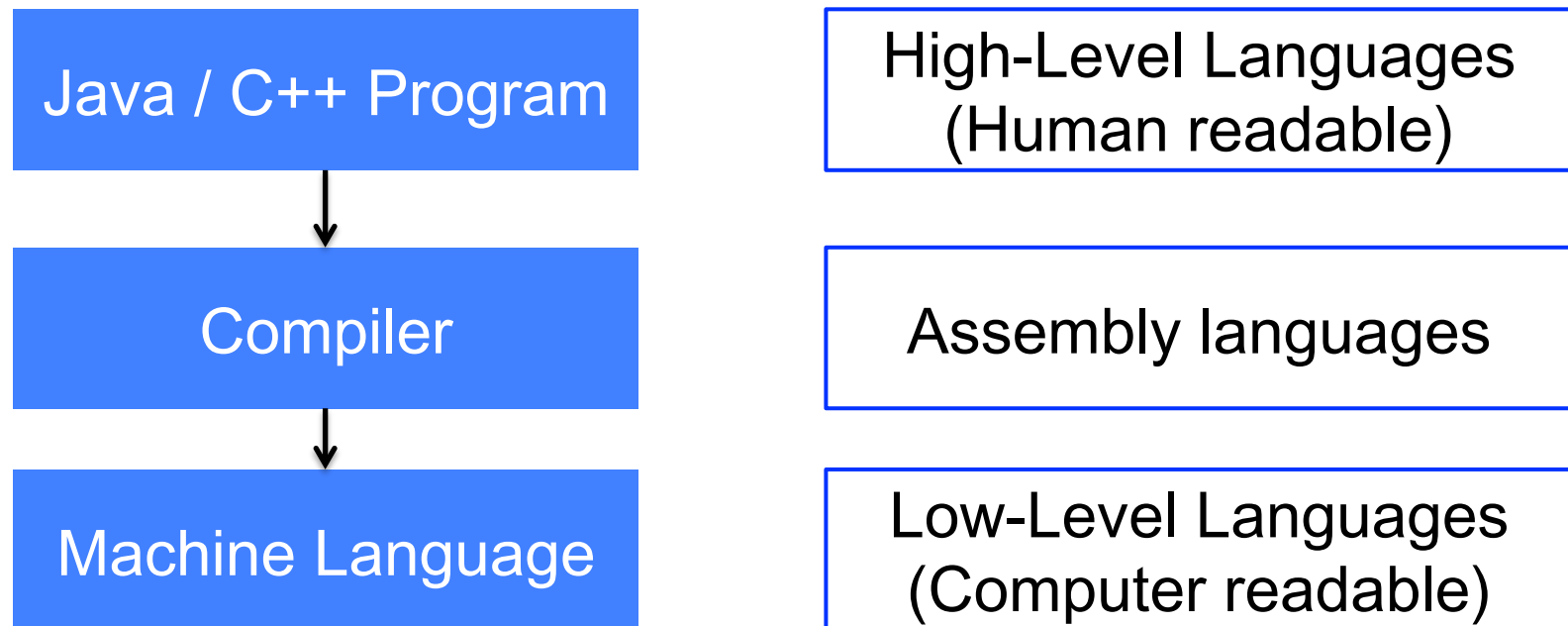
Software

- Program: A set of computer instructions



Programming Languages

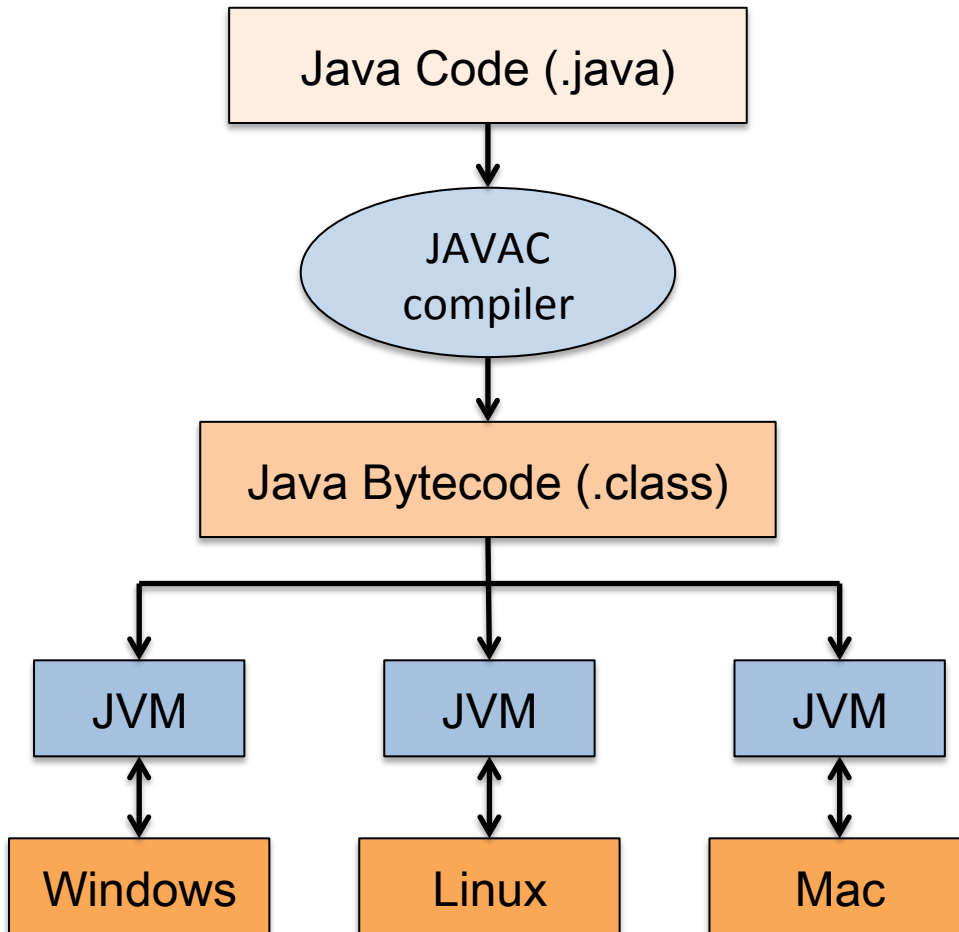
- Different levels



Translation

- A high-level language →? a low-level language
 - Compiler: translate once, run forever
 - Interpreter: translation alternates with execution, directly executes instructions
- Java: combines a compiler and an interpreter

Java Bytecode



A **compiler** translates Java code into bytecode

The Java Virtual Machine (JVM) is an **interpreter** that translates and executes bytecode

Why Using Java Bytecode?

- The bytecode is not the machine language for any particular computers
- Can be easily translated into the machine language of a given computer
- **Portability**
 - Java bytecode runs on any computer has a JVM
 - No need to recompile the Java code

Objects, Methods, and Classes

- Object: a combination of attributes (data) and methods (actions)
 - Yi's Car (has wheels, can start, stop, ...)
- Class: defines a type or kind of object
 - Car (blueprint for defining the objects, e.g., Yi's car, Bob's car ...)
- Methods: actions performed by objects
 - start(), stop(), forward(), back() ...

Invoking a Method

- A Java program uses objects to perform actions that are defined by methods

Yi's car.forward ();

Semicolon at the end of each statement

System.out.println("Welcome to COMP 110");

Object to perform actions

Method of the object System.out

Argument

- Print the string in quotes to screen

First Java Program

- A simple task
 - Print a message: “Welcome to COMP 110”

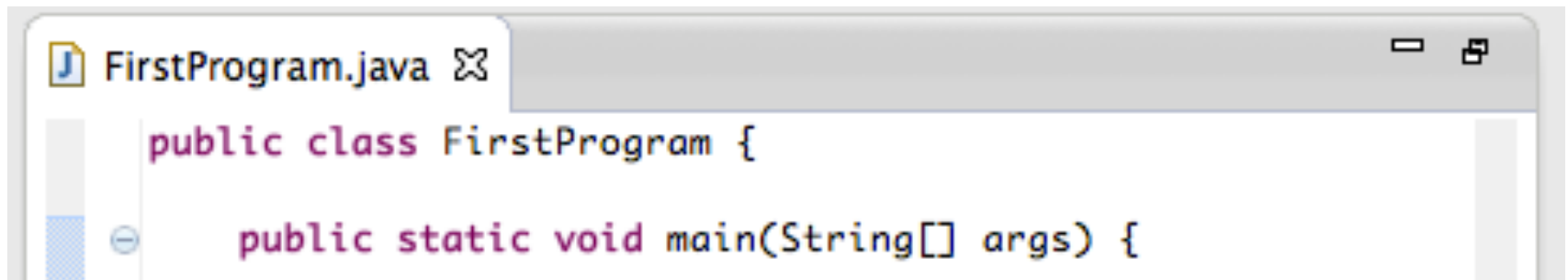
The image shows a screenshot of a Java IDE with a code editor window titled "FirstProgram.java". The code is as follows:

```
public class FirstProgram {  
    public static void main(String[] args) {  
        System.out.println("Welcome to COMP 110!");  
    }  
}
```

Four callout boxes with arrows point to specific parts of the code:

- "Each class is in *.java" points to the filename "FirstProgram.java".
- "A class contains methods" points to the opening curly brace of the class definition.
- "Every application has a main method" points to the `main` method signature.
- "The body of the method" points to the `System.out.println` statement inside the `main` method.

Begin the Program

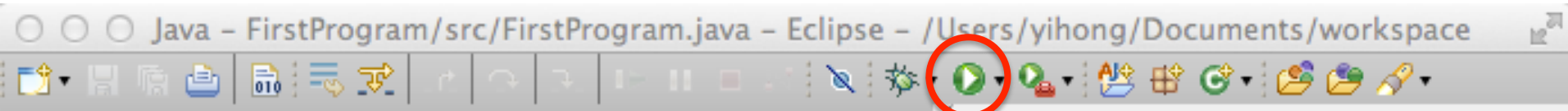


```
FirstProgram.java ✖  
public class FirstProgram {  
    public static void main(String[] args) {
```

- Begin a program named “FirstProgram”
- Program names should make sense
- Capitalize the first letter of each word in the program name

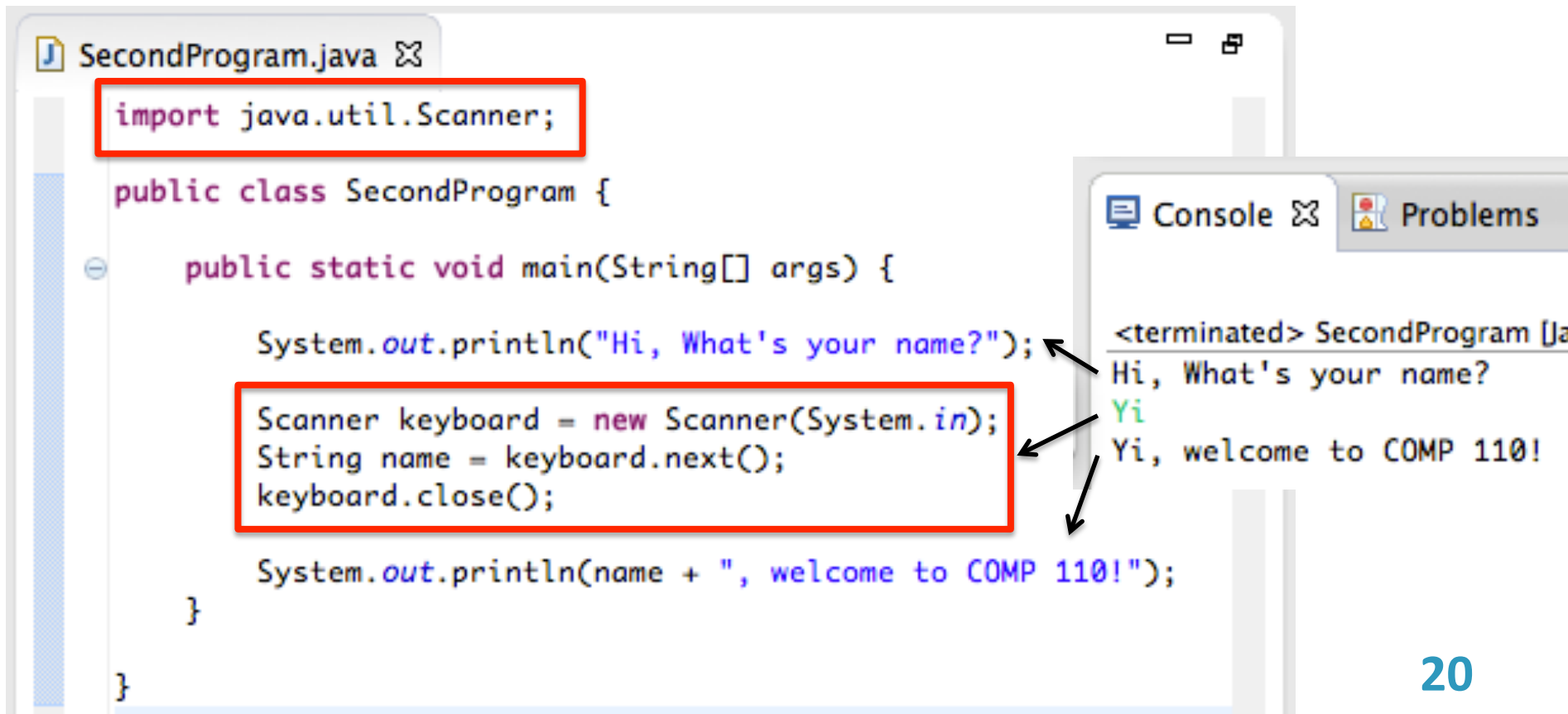
Run the First Program

- Compile: `javac FirstProgram.java`
 - Bytecode is in the file, `FirstProgram.class`
- Execute: `java FirstProgram`
- Or use IDE (integrated development environment)



Second Java Program

- Ask the user to input his/her name, and print a welcome message



The screenshot shows an IDE window titled "SecondProgram.java" with the following code:

```
import java.util.Scanner;

public class SecondProgram {

    public static void main(String[] args) {

        System.out.println("Hi, What's your name?");

        Scanner keyboard = new Scanner(System.in);
        String name = keyboard.next();
        keyboard.close();

        System.out.println(name + ", welcome to COMP 110!");

    }

}
```

The code is annotated with red boxes: one around the import statement and another around the Scanner initialization and input handling lines. To the right, a "Console" window shows the program's execution output:

```
<terminated> SecondProgram [Ja
Hi, What's your name?
Yi
Yi, welcome to COMP 110!
```

Arrows point from the console output to the corresponding code lines: from "Hi, What's your name?" to the first print statement, from "Yi" to the scanner input line, and from "Yi, welcome to COMP 110!" to the second print statement.

What's New (1): Java Package

```
import java.util.Scanner
```

- Gets the **Scanner** class from the package **java.util**
- Package = Library of classes
- Different libraries provide different functionalities
 - Math library: math equations
 - Network library: send / receive packages
 - java.util : allows you to read data from keyboard

What's New (2): Create Objects

```
Scanner keyboard = new Scanner(System.in)
```

- Create an object (i.e., keyboard) of the Scanner class
- Then the object performs actions:

```
String name = keyboard.next();
```

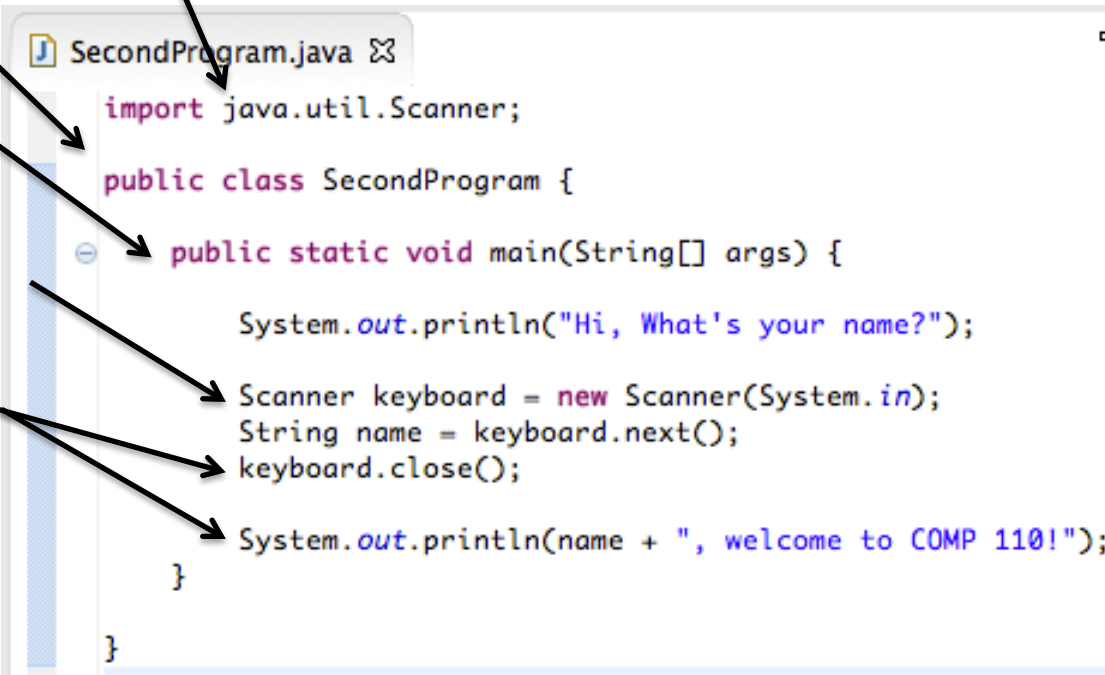
- Read a string from the keyboard

```
Keyboard.close();
```

- Close the keyboard, stop getting data from a user

First & Second Java Programs

- Import a package / class
- Define a class
- A main method
- Create an object
- Invoke methods



```
SecondProgram.java
import java.util.Scanner;
public class SecondProgram {
    public static void main(String[] args) {
        System.out.println("Hi, What's your name?");
        Scanner keyboard = new Scanner(System.in);
        String name = keyboard.next();
        keyboard.close();
        System.out.println(name + ", welcome to COMP 110!");
    }
}
```

The screenshot shows a code editor window titled "SecondProgram.java". The code is as follows: `import java.util.Scanner;`, `public class SecondProgram {`, `public static void main(String[] args) {`, `System.out.println("Hi, What's your name?");`, `Scanner keyboard = new Scanner(System.in);`, `String name = keyboard.next();`, `keyboard.close();`, `System.out.println(name + ", welcome to COMP 110!");`, `}`, `}`. Arrows from the list on the left point to these elements: "Import a package / class" points to the import statement; "Define a class" points to the class declaration; "A main method" points to the method signature; "Create an object" points to the Scanner object creation; "Invoke methods" points to the `next()` and `close()` calls.

Next Class

- Object-Oriented Programming (OOP)
- Primitive data types and variables
- Reading and coding assignments
 - Chapter 1.3, 2.1
 - Try the two sample programs in class