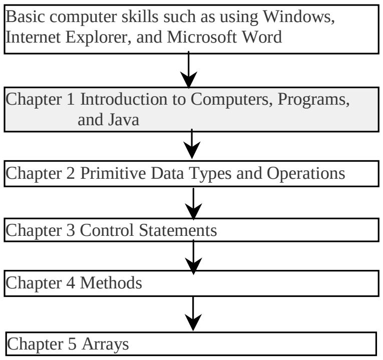
Chapter 1 Introduction to Computers, Programs, and Java

Prerequisites for Part I



Basic computer skills such as using Windows, Internet Explorer, and Microsoft Word

Chapter 1 Introduction to Computers, Programs, and Java

Chapter 2 Primitive Data Types and Operations

Chapter 3 Control Statements

Chapter 4 Methods

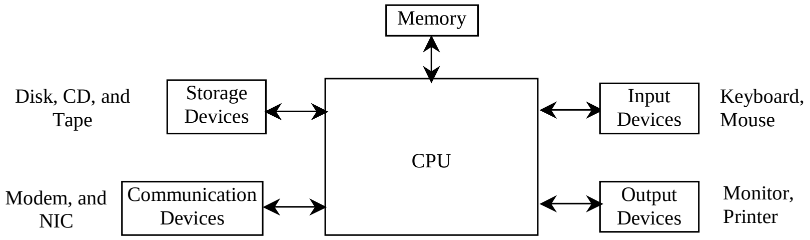
Chapter 5 Arrays

Objectives

- To review computer basics, programs, and operating systems (§1.2-1.4).
- To represent numbers in binary, decimal, and hexadecimal (§1.5 Optional).
- To understand the relationship between Java and the World Wide Web (§1.6).
- To know Java’s advantages (§1.7).
- To distinguish the terms API, IDE, and JDK (§1.8).
- To write a simple Java program (§1.9).
- To create, compile, and run Java programs (§1.10).
- To understand the Java runtime environment (§1.10).
- To know the basic syntax of a Java program (§1.11).

What is a Computer?

A computer consists of a CPU, memory, hard disk, floppy disk, monitor, printer, and communication devices.



Memory

Disk, CD, and Tape

Storage Devices

CPU

Input Devices

Keyboard, Mouse

Modem, and NIC

Communication Devices

Output Devices

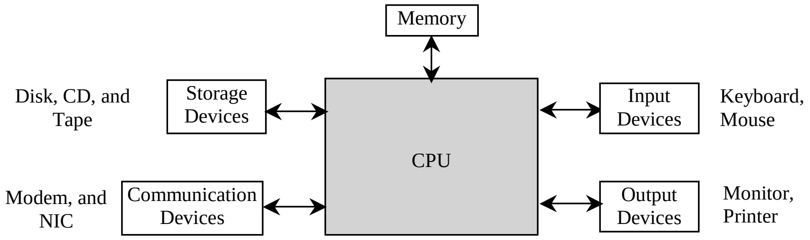
Monitor, Printer

CPU

•The central processing unit (CPU) is the brain of a computer.

•It retrieves instructions from memory and executes them.

•The CPU speed is measured in megahertz (MHz), with 1 megahertz equaling 1 million pulses per second.



Memory

Disk, CD, and Tape

Storage Devices

CPU

Input Devices

Keyboard, Mouse

Modem, and NIC

Communication Devices

Output Devices

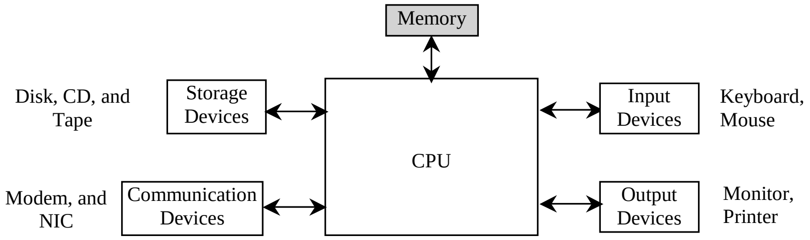
Monitor, Printer

Memory

•*Memory* stores data and program instructions for the CPU to execute.

•A memory unit is an ordered sequence of bytes

•A program and its data must be brought to memory before they can be executed.



Memory

Disk, CD, and Tape

Storage Devices

CPU

Input Devices

Keyboard, Mouse

Modem, and NIC

Communication Devices

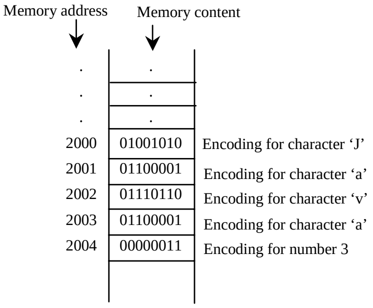
Output Devices

Monitor, Printer

How is Data Stored?

☞Data of various kinds, such as numbers, characters, and strings, are encoded as a series of bits (zeros and ones).

☞For example, character ‘J’ is represented by 01001010 in one byte.



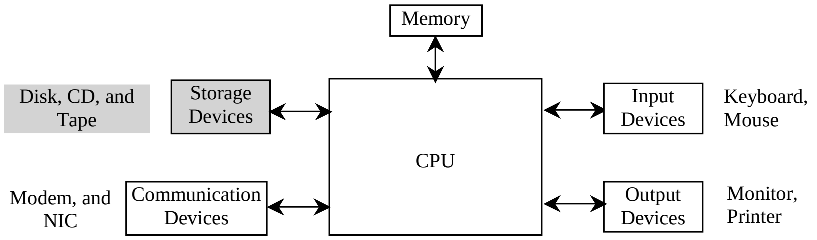
Memory address	Memory content	
.	.	
.	.	
.	.	
2000	01001010	Encoding for character ‘J’
2001	01100001	Encoding for character ‘a’
2002	01110110	Encoding for character ‘v’
2003	01100001	Encoding for character ‘a’
2004	00000011	Encoding for number 3

Storage Devices

•Memory is volatile, because content is lost when the power is off.

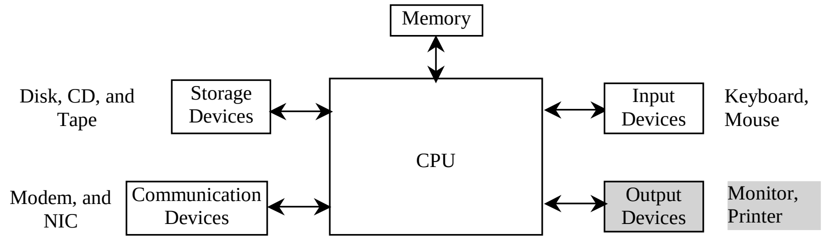
•More permanent type of storage necessary: storage devices

•Three main types of storage devices:Disk drives (hard disks and floppy disks), CD drives (CD-R and CD-RW), and Tape drives.



Output Devices: Monitor

•The monitor displays information (text and graphics). The resolution and dot pitch determine the quality of the display.



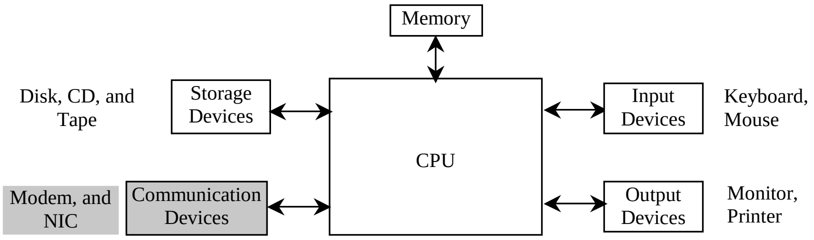
Communication Devices

•A *regular modem* uses a phone line and can transfer data in speeds up to 56,000 bps (bits per second).

•A *DSL* (digital subscriber line) also uses a phone line and can transfer data in speeds 20 times faster than a regular modem.

•A *cable modem* uses the TV cable line maintained by the cable company. A cable modem is as fast as a DSL.

•Network interface card (*NIC*) is a device to connect a computer to a local area network (LAN).



Programs

Computer *programs*, known as *software*, are instructions to the computer.

You tell a computer what to do through programs. Without programs, a computer is an empty machine. Computers do not understand human languages, so you need to use computer languages to communicate with them.

Programs are written using programming languages.

Programming Languages

Machine Language   Assembly Language   High-Level Language

Machine language is a set of primitive instructions built into every computer. The instructions are in the form of binary code, so you have to enter binary codes for various instructions. Program with native machine language is a tedious process. Moreover the programs are highly difficult to read and modify. For example, to add two numbers, you might write the an instruction in binary like this:

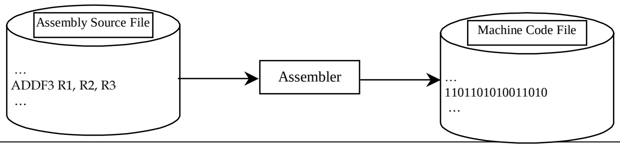
```
1101101010011010
```

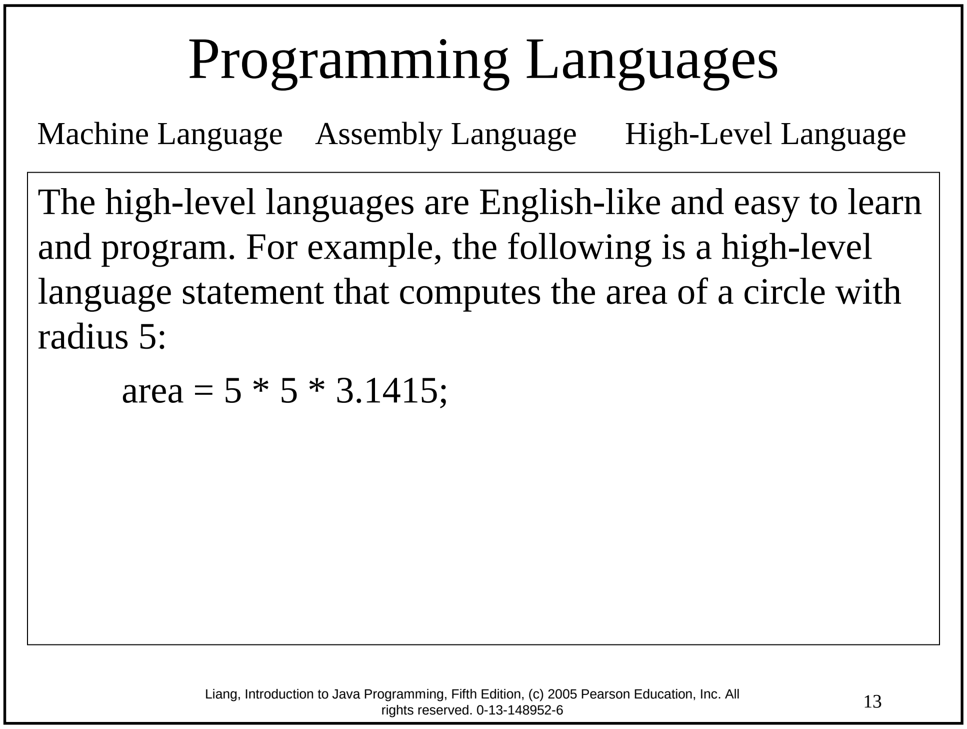
Programming Languages

Machine Language   Assembly Language   High-Level Language

Assembly languages were developed to make programming easy. Since the computer cannot understand assembly language, however, a program called assembler is used to convert assembly language programs into machine code. For example, to add two numbers, you might write an instruction in assembly code like this:

ADDF3 R1, R2, R3



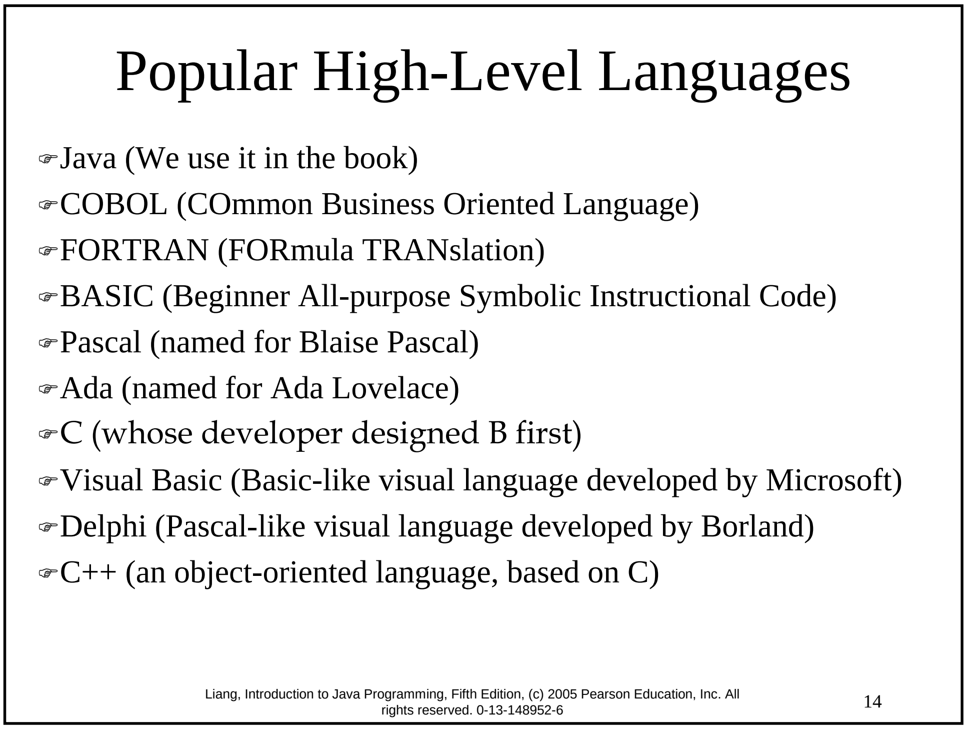


Programming Languages

Machine Language Assembly Language High-Level Language

The high-level languages are English-like and easy to learn and program. For example, the following is a high-level language statement that computes the area of a circle with radius 5:

```
area = 5 * 5 * 3.1415;
```

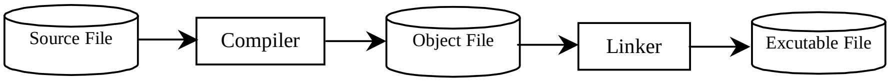


Popular High-Level Languages

- Java (We use it in the book)
- COBOL (COmmon Business Oriented Language)
- FORTRAN (FORmula TRANslation)
- BASIC (Beginner All-purpose Symbolic Instructional Code)
- Pascal (named for Blaise Pascal)
- Ada (named for Ada Lovelace)
- C (whose developer designed B first)
- Visual Basic (Basic-like visual language developed by Microsoft)
- Delphi (Pascal-like visual language developed by Borland)
- C++ (an object-oriented language, based on C)

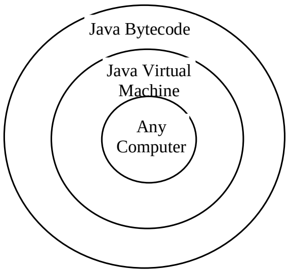
Compiling Source Code

- A program written in a high-level language is called a s*ource program*.
- Source code needs to be *compiled* to translate into a machine language program called an *object program*.
- The object program is often then linked with other supporting library code before the object can be executed on the machine.
  - Generates .exe file on Windows



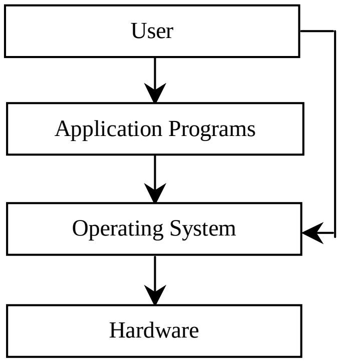
Compiling Source Code

- Source code can be ported across machines with different OSs
- However, it needs to be recompiled on each machine
- With Java, you write the program once, and compile the source program into a special type of object code, known as *bytecode,* which then runs on any computer with a Java Virtual Machine.
- Java Virtual Machine is a software that interprets Java bytecode.



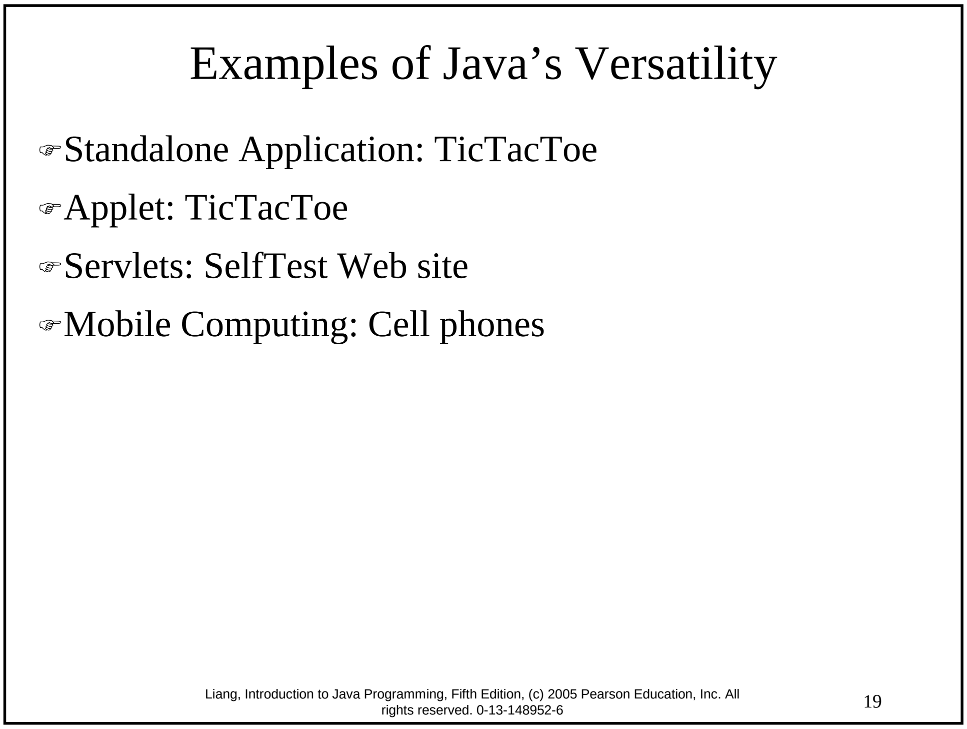
Operating Systems

- ☞The *operating system* (OS) is a program that manages and controls a computer's activities.
- ☞You are probably using Windows 98, NT, 2000, XP, or ME.
  - – Windows is currently the most popular PC operating system.
- ☞Application programs such as an Internet browser and a word processor cannot run without an operating system.



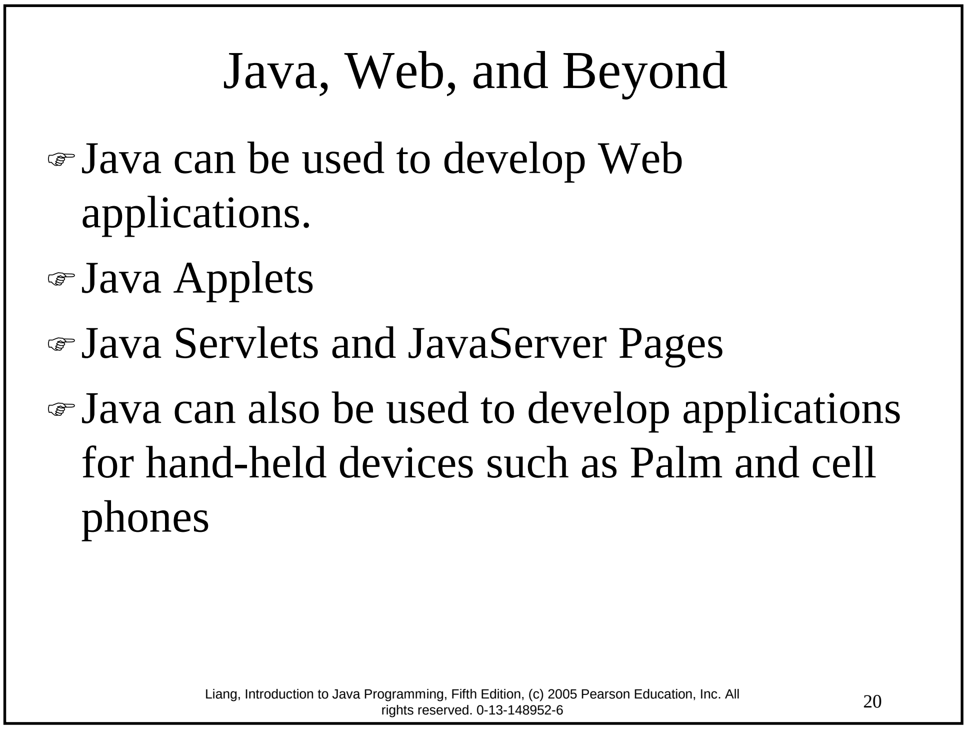
Why Java?

- ☞It supports application development on the Internet for servers, desktop computers, and small hand-held devices.
- ☞Java is a general purpose programming language.
  - ☞ Supports object-oriented programming
- ☞Java is good for programming Internet applications



Examples of Java’s Versatility

- Standalone Application: TicTacToe
- Applet: TicTacToe
- Servlets: SelfTest Web site
- Mobile Computing: Cell phones



Java, Web, and Beyond

- Java can be used to develop Web applications.
- Java Applets
- Java Servlets and JavaServer Pages
- Java can also be used to develop applications for hand-held devices such as Palm and cell phones

Java's History

- James Gosling and Sun Microsystems
- Oak
- Java, May 20, 1995, Sun World
- HotJava
  - The first Java-enabled Web browser
- Early History Website:

http://java.sun.com/features/1998/05/birthday.html

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

JDK Versions

- JDK 1.02 (1995)
- JDK 1.1 (1996)

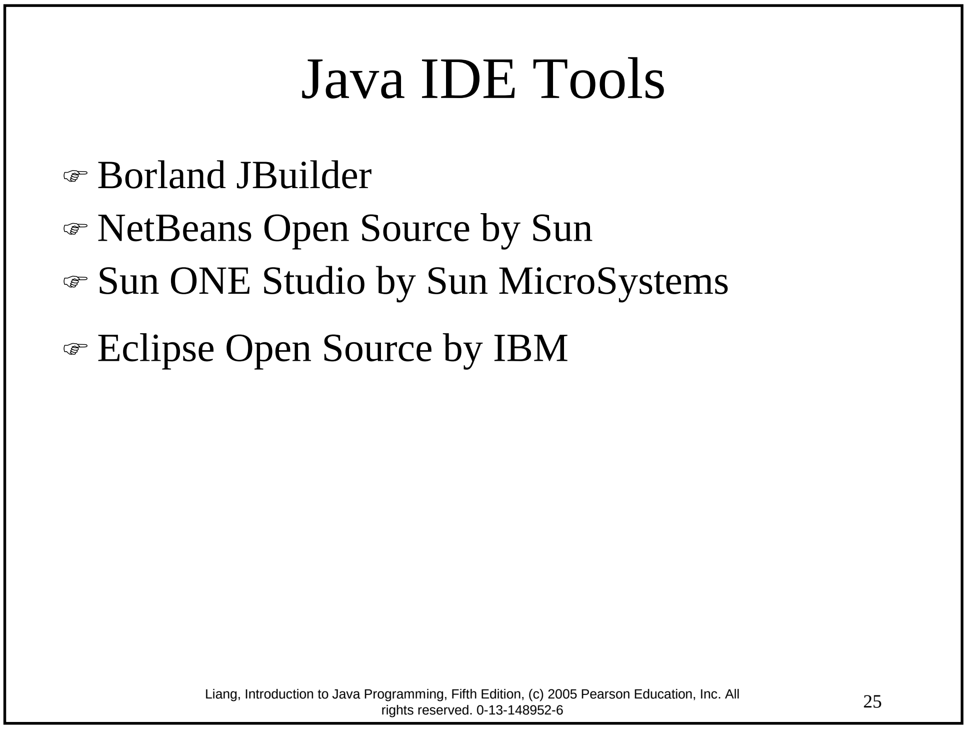
major changes

- Java 2 SDK v 1.2 (a.k.a JDK 1.2, 1998)
- Java 2 SDK v 1.3 (a.k.a JDK 1.3, 2000)
- Java 2 SDK v 1.4 (a.k.a JDK 1.4, 2002)
- Java 2 SDK v 1.5 (a.k.a JDK 1.5, 2004)

JDK Editions

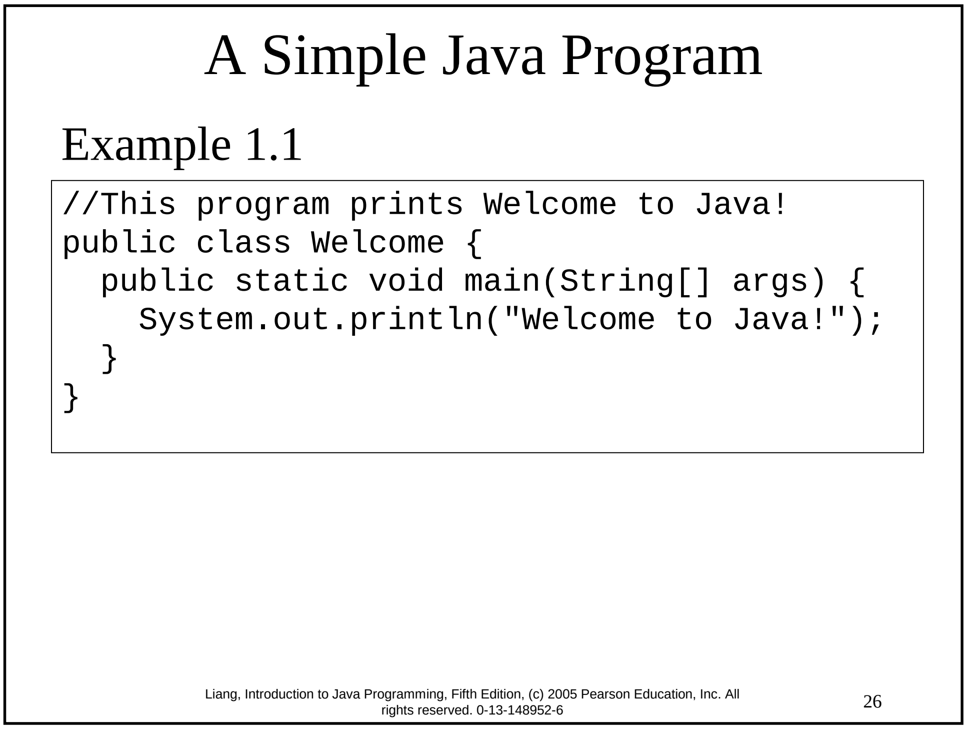
- Java Standard Edition (J2SE)
  - J2SE can be used to develop client-side standalone applications or applets.
- Java Enterprise Edition (J2EE)
  - J2EE can be used to develop server-side applications such as Java servlets and Java ServerPages.
- Java Micro Edition (J2ME).
  - J2ME can be used to develop applications for mobile devices such as cell phones.

This book uses J2SE to introduce Java programming.



Java IDE Tools

- ☞ Borland JBuilder
- ☞ NetBeans Open Source by Sun
- ☞ Sun ONE Studio by Sun MicroSystems
- ☞ Eclipse Open Source by IBM



A Simple Java Program

Example 1.1

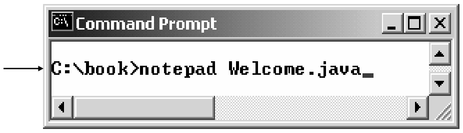
```
//This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Creating and Editing Using NotePad

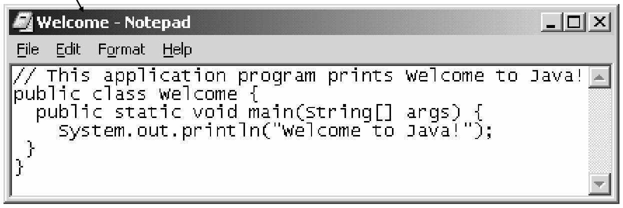
To use NotePad, type

notepad Welcome.java

from the DOS prompt.



```
C:\book>notepad Welcome.java_
```



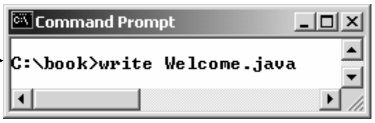
```
// This application program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Creating and Editing Using WordPad

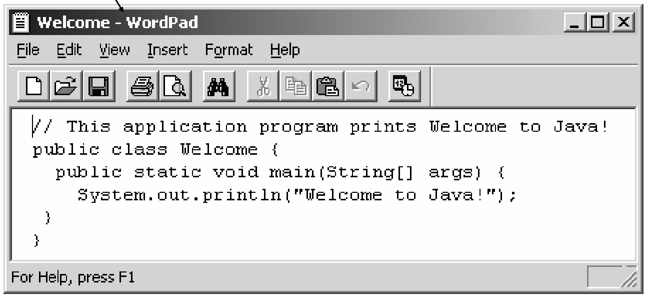
To use WordPad, type

write Welcome.java

from the DOS prompt.

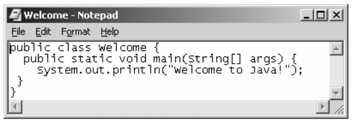


```
C:\book>write Welcome.java
```



```
// This application program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Creating, Compiling, and Running Programs



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

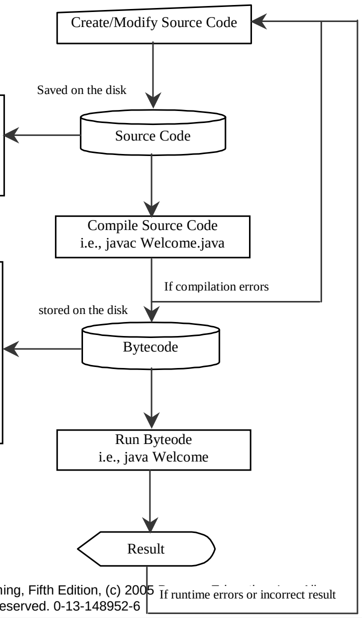
Source code (developed by the programmer)

```
package chapter1;
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```

Byte code (generated by the compiler for JVM to read and interpret, not for you to understand)

```
…
Method Welcome()
  0 aload_0
  …

Method void main(java.lang.String[])
  0 getstatic #2 …
  3 ldc #3 <String "Welcome to
Java!">
  5 invokevirtual #4 …
```



Supplements on the Companion Website

- ☞ See Supplement A for installing and configuring JDK 1.5
- ☞ See Supplement B for compiling and running Java from the command window for details

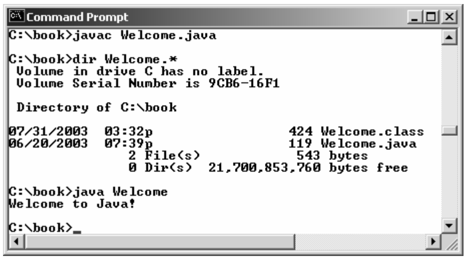
www.prenhall.com/liang

Direct link at

www.cs.armstrong.edu/liang/intro5e.html

Compiling and Running Java from the Command Window

- Set path to JDK bin directory
  - set path=c:\Program Files\java\jdk1.5.0\bin
- Set classpath to include the current directory
  - set classpath=.
- Compile
  - javac Welcome.java
- Run
  - java Welcome



```
C:\book>javac Welcome.java

C:\book>dir Welcome.*
 Volume in drive C has no label.
 Volume Serial Number is 9CB6-16F1

 Directory of C:\book

07/31/2003  03:32p                 424 Welcome.class
06/20/2003  07:39p                 119 Welcome.java
               2 File(s)            543 bytes
               0 Dir(s)  21,700,853,760 bytes free

C:\book>java Welcome
Welcome to Java!

C:\book>_
```

Anatomy of a Java Program

- Comments
- Package
- Reserved words
- Modifiers
- Statements
- Blocks
- Classes
- Methods
- The main method

Comments

- In Java, comments can be of three types
  - preceded by two slashes (//) in a line
  - enclosed between /* and */ in one or multiple lines
  - Enclosed between /** and */ in one or multiple lines

Package

- The second line in the program (package chapter1;) specifies a package name, chapter1, for the class
- Packages are a way to logically organize your programs

Reserved Words

☞Reserved words or keywords are words that have a specific meaning to the compiler and cannot be used for other purposes in the program.

☞Examples include the words class, public, static, and void

Modifiers

☞Java uses certain reserved words called modifiers that specify the properties of the data, methods, and classes and how they can be used.

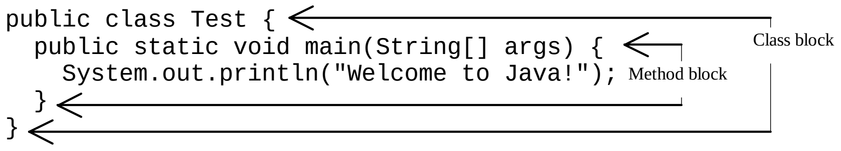
- Examples of modifiers are public and static.
- Other modifiers are private, final, abstract, and protected.

Statements

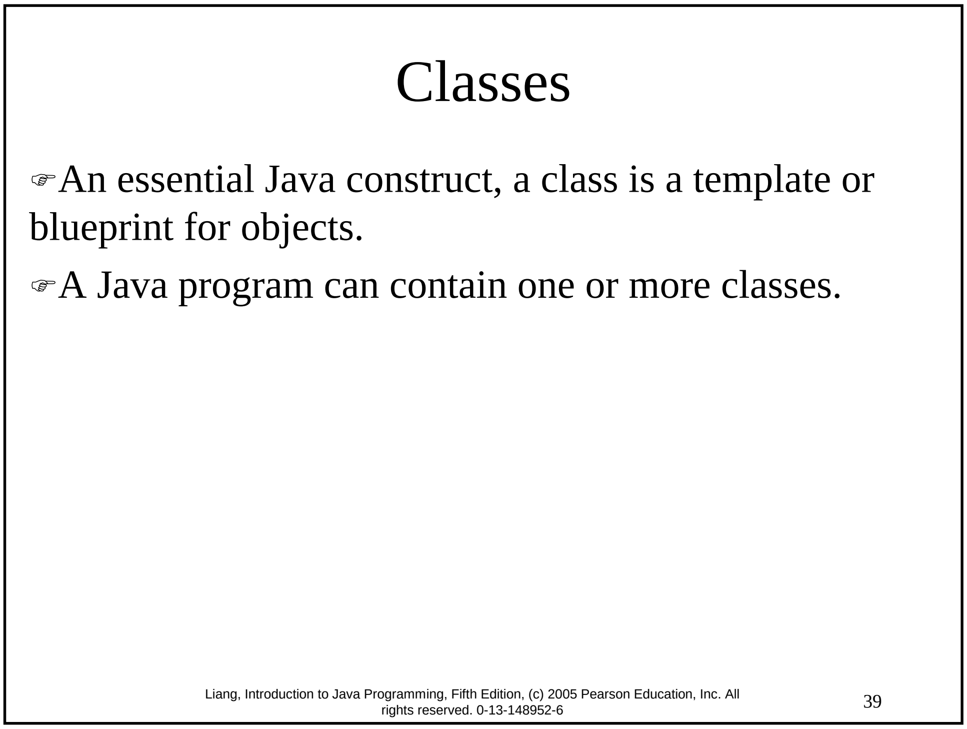
- ☞A statement represents an action or a sequence of actions.
- ☞Every statement in Java ends with a semicolon (;).
  - – Example: System.out.println(“Welcome to Java!”);

Blocks

A pair of braces in a program forms a block that groups components of a program.



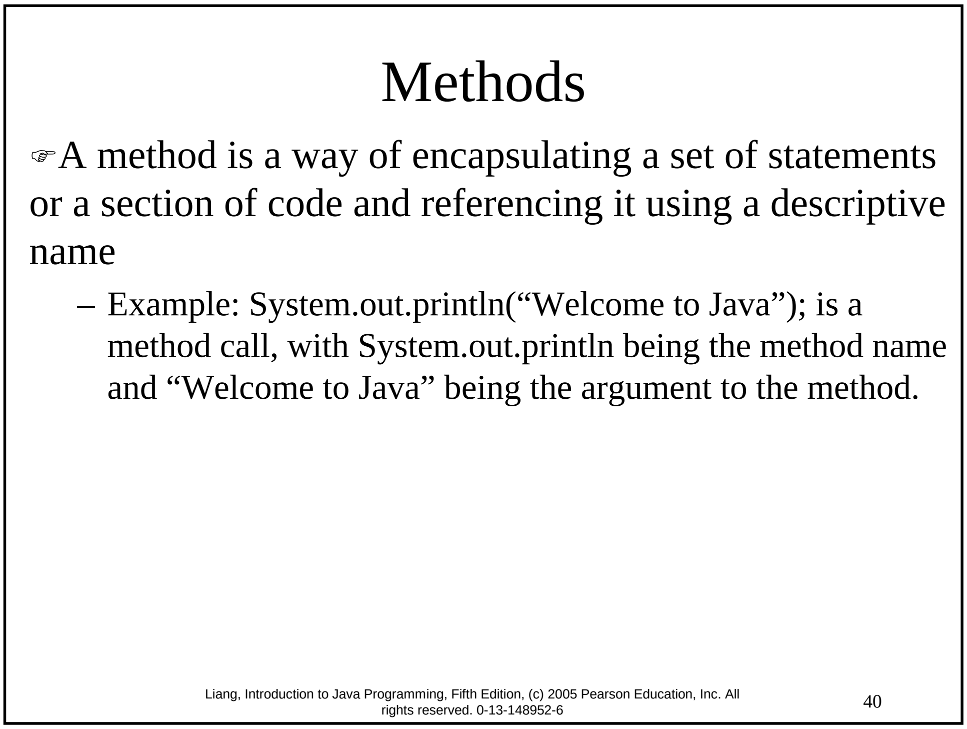
```
public class Test {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```



Classes

☞An essential Java construct, a class is a template or blueprint for objects.

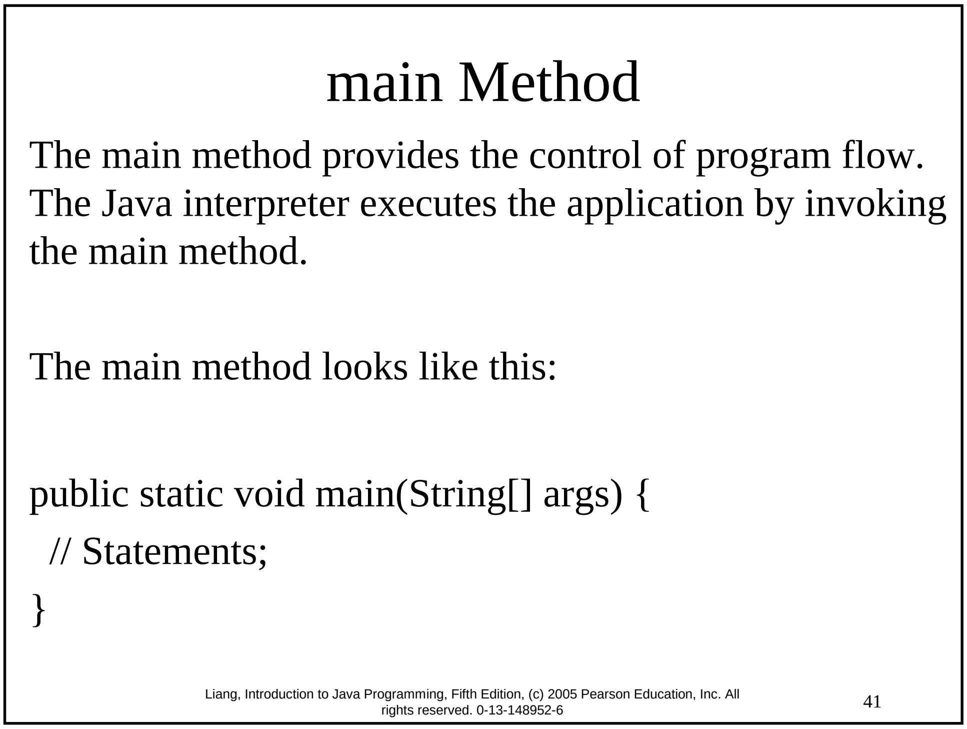
☞A Java program can contain one or more classes.



Methods

☞A method is a way of encapsulating a set of statements or a section of code and referencing it using a descriptive name

- Example: System.out.println(“Welcome to Java”); is a method call, with System.out.println being the method name and “Welcome to Java” being the argument to the method.



main Method

The main method provides the control of program flow. The Java interpreter executes the application by invoking the main method.

The main method looks like this:

```
public static void main(String[] args) {
  // Statements;
}
```