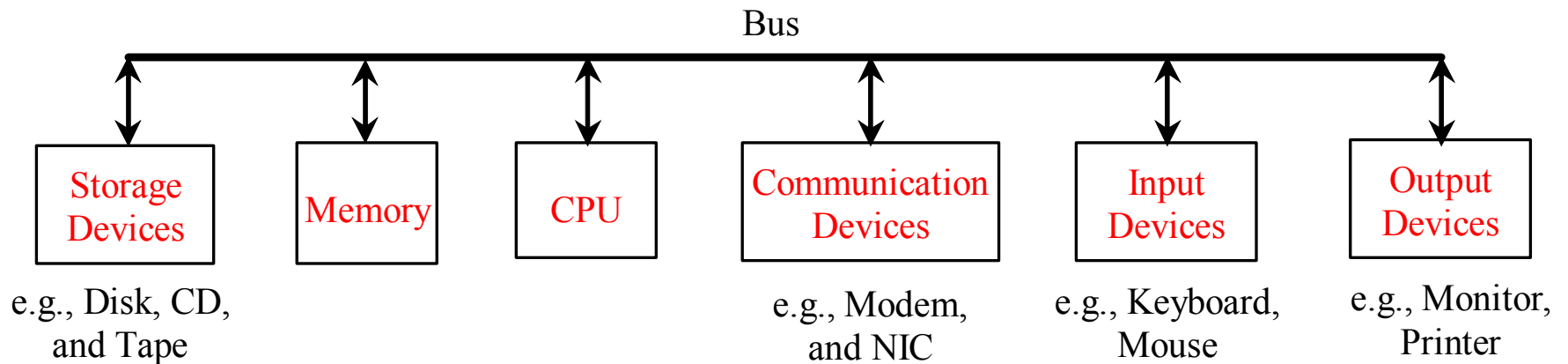# Chapter 1 Introduction to Computers, Programs, and Java

Hu Zheng
huzheng@bupt.edu.cn

What is a Computer?

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

Bus

| Storage Devices | Memory | CPU | Communication Devices | Input Devices | Output Devices |

e.g., Disk, CD, and Tape

e.g., Modem, and NIC

e.g., Keyboard, Mouse

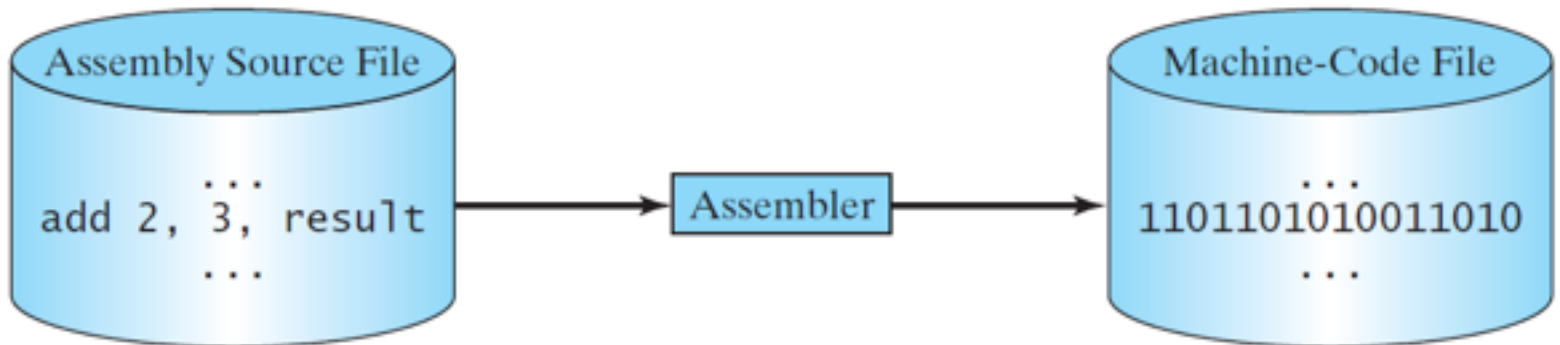e.g., Monitor, Printer

Programs

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

Programming Languages

Machine Language   Assembly Language   High-Level Language

110110101010011010

area = 5 * 5 * 3.1415;

Popular High-Level Languages

| Language	Description
Ada	Named for Ada Lovelace, who worked on mechanical general-purpose computers. The Ada language was developed for the Department of Defense and is used mainly in defense projects.
BASIC	Beginner's All-purpose Symbolic Instruction Code. It was designed to be learned and used easily by beginners.
C	Developed at Bell Laboratories. C combines the power of an assembly language with the ease of use and portability of a high-level language.
C++	C++ is an object-oriented language, based on C.
C#	Pronounced "C Sharp." It is a hybrid of Java and C++ and was developed by Microsoft.
COBOL	COmmon Business Oriented Language. Used for business applications.
FORTRAN	FORmula TRANslation. Popular for scientific and mathematical applications.
Java	Developed by Sun Microsystems, now part of Oracle. It is widely used for developing platform-independent Internet applications.
Pascal	Named for Blaise Pascal, who pioneered calculating machines in the seventeenth century. It is a simple, structured, general-purpose language primarily for teaching programming.
Python	A simple general-purpose scripting language good for writing short programs.
Visual Basic	Visual Basic was developed by Microsoft and it enables the programmers to rapidly develop graphical user interfaces.

Interpreting/Compiling Source Code

A program written in a high-level language is called a *source program* or *source code*.

A source program must be translated into machine code for execution.

A Translation can be done using another programming tool called an *interpreter* or a *compiler*.

Interpreting Source Code

 An interpreter reads one statement from the source code, translates it to the machine code or virtual machine code, and then executes it right away.
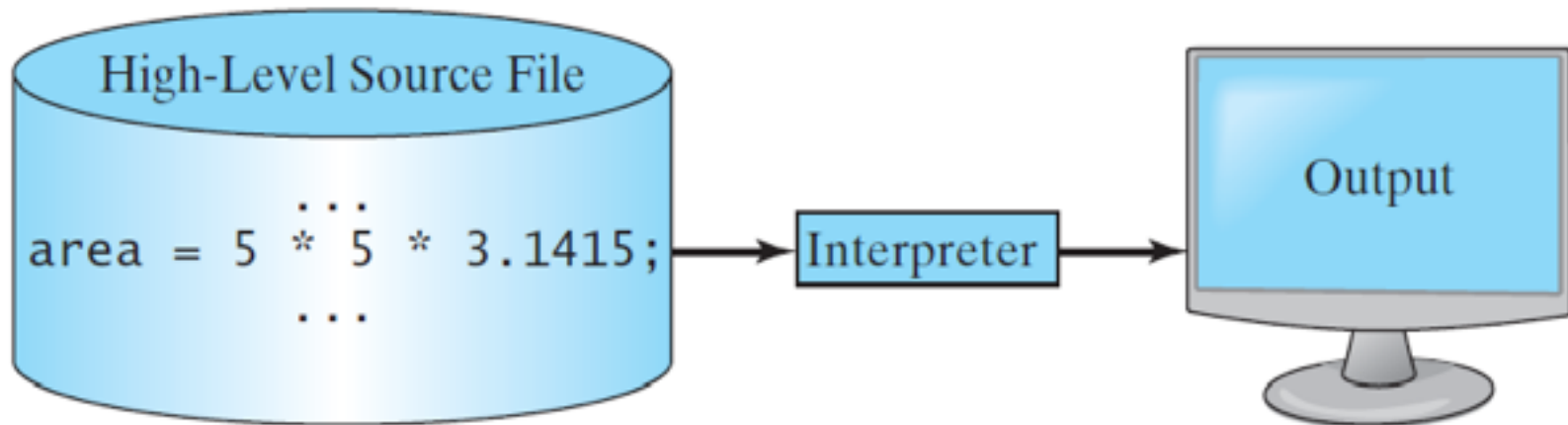
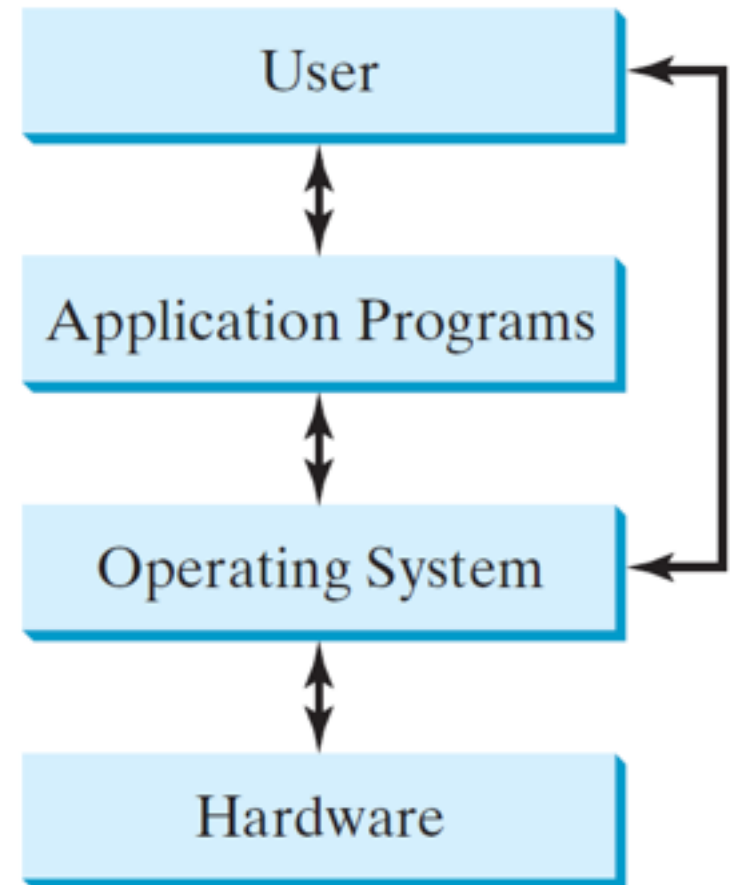 Note: a statement from the source code may be translated into several machine instructions.

Compiling Source Code

A compiler translates the entire source code into a machine-code file, and the machine-code file is then executed.
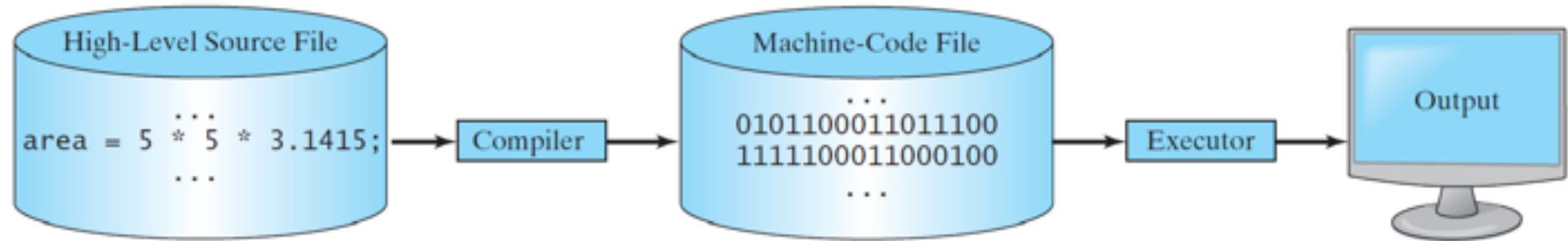
Operating Systems

The *operating system* (OS) is a program that manages and controls a computer's activities.

Why Java?

The future of computing is being profoundly influenced by the Internet

✦ Java is a general purpose programming language.

✦ Java is the Internet programming language.

Java, Web, and Beyond

✦ Java can be used to develop standalone applications.

✦ Java can be used to develop applications running from a browser.

✦ Java can also be used to develop applications for hand-held devices.

✦ Java can be used to develop applications for Web servers.

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://www.java.com/en/javahistory/index.jsp

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

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-N

✦ Java Is Portable

✦ Java's Performance

✦ Java Is Multithreaded

✦ Java Is Dynamic

Java is partially modeled on C++, but greatly simplified and improved. Some people refer to Java as "C++--" because it is like C++ but with more functionality and fewer negative aspects.

What does it mean

Java syntax and the Java API (application programming interface -- a library of Java code) is simple and designed to be easy to learn and use.

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

Java is inherently object-oriented. Java was designed from the start to be object-oriented. Object-oriented programming (OOP) is a popular programming approach that is replacing traditional procedural programming techniques.

One of the central issues in software development is how to reuse code. Object-oriented programming provides great flexibility, modularity, clarity, and reusability through encapsulation, inheritance, and polymorphism.

The idea behind objects is quite simple. Instead of designing a long and complex set of instructions or routines to perform one or more tasks, the programmer can break these into various parts. Each part can perform one or more discrete tasks and contains all the data needed to perform the task. This is an object. Java files define an object or a class of objects.

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

Distributed computing involves several computers working together on a network. Java is designed to make distributed computing easy. Since networking capability is inherently integrated into Java, writing network programs is like sending and receiving data to and from a file.

Java language provides a library of program routines to open and access objects across the Internet via URLs with the same ease as when accessing a local file system. You can send data across the net easily with Java programs.

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

You need an interpreter to run Java programs. The programs are compiled into the Java Virtual Machine code called bytecode. The bytecode is machine-independent and can run on any machine that has a Java interpreter, which is part of the Java Virtual Machine (JVM).

Java source code is translated to a byte code that is interpreted one instruction at a time by the Java Virtual Machine.

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

Java compilers can detect many problems that would first show up at execution time in other languages.

Java has eliminated certain types of error-prone programming constructs found in other languages.

Java has a runtime exception-handling feature to provide programming support for robustness.

The design of the Java platform ensures that the programs run correctly and do not break when the unexpected happens.

Characteristics of Java

✦ Java Is Simple

✦ Java Is Object-Oriented

✦ Java Is Distributed

✦ Java Is Interpreted

✦ Java Is Robust

✦ Java Is Secure

Java implements several security mechanisms to protect your system against harm caused by stray programs.

✦ Java Is Architecture-Neutral

✦ Java Is Portable

✦ Java's Performa

✦ Java Is Multithr

✦ Java Is Dynami

Java programs can be downloaded from a location on the network. Such downloaded Java programs cannot access files or destroy programs on your computer.

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

Java language programs can be written on one machine and run on a machine with a different type of CPU.

Write once, run anywhere

With a Java Virtual Machine (JVM), you can write one program that will run on any platform.

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

Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

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

Java's performance Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

22

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-
- ✦ Java Is Portable
- ✦ Java's Performance
- ✦ Java Is Multithreaded
- ✦ Java Is Dynamic

Multithreading is when multiple Java programs are running simultaneously and sharing data and instructions. Multithreading provides the user the ability to view an animation or video clip, listen to music in a browser, and search for information on the World Wide Web.

Multithread programming is smoothly integrated in Java, whereas in other languages you have to call procedures specific to the operating system to enable multithreading.

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-Neut
- ✦ Java Is Portable
- ✦ Java's Performance
- ✦ Java Is Multithreaded
- ✦ Java Is Dynamic

Dynamic has several meanings as it applies to the Java language. For example, the language has special capabilities that allow the program to call upon resources as needed dynamically when the program is running in memory. Such a resource can be another Java routine. The software development process can also be dynamic. Java allows a programmer to approach a solution to a problem in incremental steps. At each step of the solution, a small number of routines can be built with newer routines using the work of already defined routines. This encourages reuse of code.

Java was designed to adapt to an evolving environment. New code can be loaded on the fly without recompilation. There is no need for developers to create, and for users to install, major new software versions. New features can be incorporated transparently as needed.

JDK Versions

- JDK 1.02 (1995)
- JDK 1.1 (1996)
- JDK 1.2 (1998)
- JDK 1.3 (2000)
- JDK 1.4 (2002)
- JDK 1.5 (2004) a. k. a. JDK 5 or Java 5
- JDK 1.6 (2006) a. k. a. JDK 6 or Java 6
- JDK 1.7 (2011) a. k. a. JDK 7 or Java 7
- JDK 1.8 (2014) a. k. a. JDK 8 or Java 8

JDK 1.1.4	Sparkler
JDK 1.1.5	Pumpkin
JDK 1.1.6	Abigail
JDK 1.1.7	Brutus
JDK 1.1.8	Chelsea
JAVA 1.2	Playground
JAVA1.2.1	none
JAVA 1.2.2	Cricket
JAVA 1.3	Kestrel
JAVA 1.3.1	Ladybird
JAVA 1.4.0	Merlin
JAVA 1.4.1	grasshopper
JAVA 1.4.2	Mantis
JAVA 5.0 (1.5)	Tiger
JAVA 6.0 (1.6)	Mustang
JAVA 7.0 (1.7)	Dolphin
JAVA 8.0 (1.8)	Spider

JDK Editions

✦ **Java Standard Edition (J2SE)**

– J2SE can be used to develop client-side standalone applications or applets.

– This book uses J2SE to introduce Java programming.

✦ **Java Enterprise Edition (J2EE)**

– J2EE can be used to develop server-side applications such as Java servlets, Java Server Pages, and Java ServerFaces.

✦ **Java Micro Edition (J2ME).**

– J2ME can be used to develop applications for mobile devices such as cell phones.

Java API

Application Programming Interface

✦1) Java Core API

✦2) Java Standard Extension API (javax)

✦3) Provided by Vendors or Organizations

Editors and Popular Java IDEs

✦ Common Text Editors
- NotePad
- TextEdit
- Editplus
- UltraEdit

✦ IDE (Integrated Development Environment)
- Eclipse http://www.eclipse.org/
- IntelliJ IDEA
- NetBeans

A Simple Java Program

Listing 1.1

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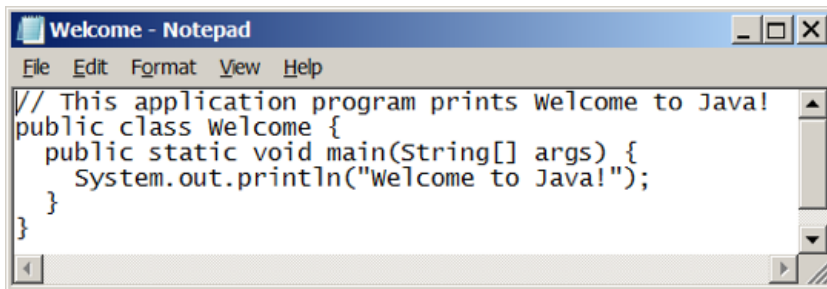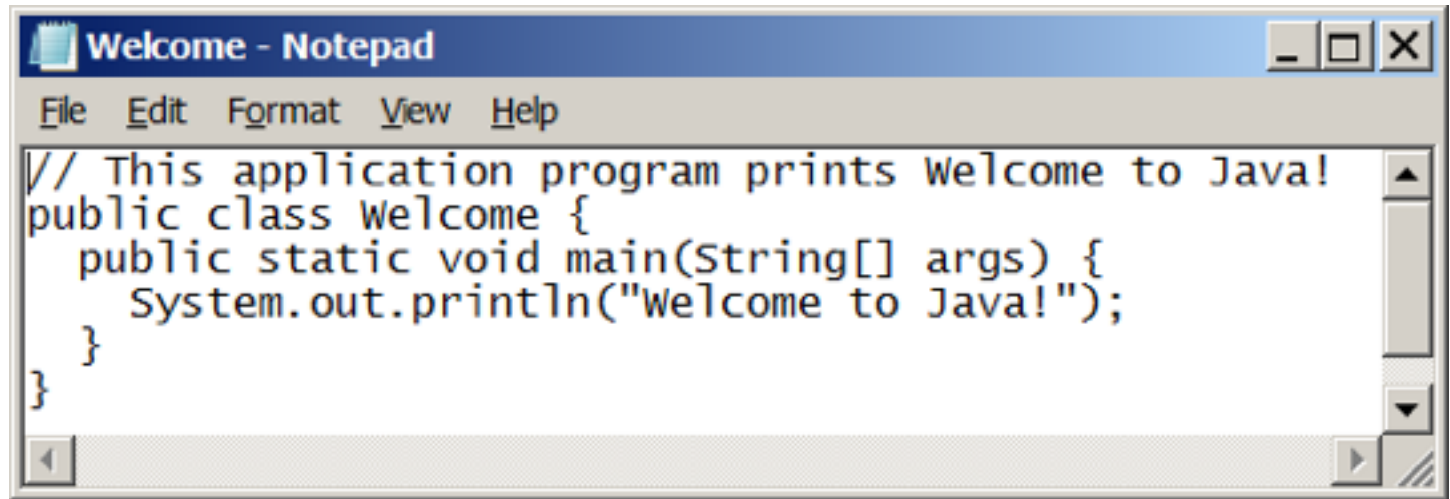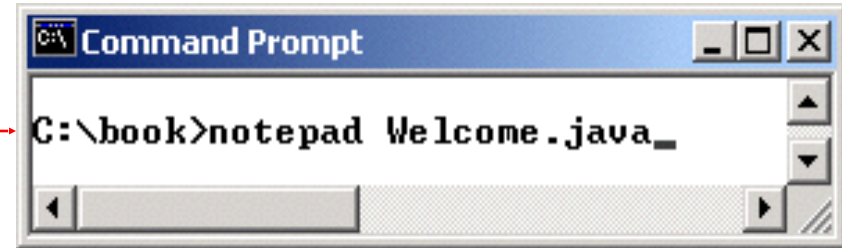
Animation

www.cs.armstrong.edu/liang/intro10e/html/Welcome.html

Creating and Editing Using NotePad

To use NotePad, type
    notepad Welcome.java
from the DOS prompt.

Command Prompt

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

Welcome - Notepad

File Edit Format View Help

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

Welcome - Notepad

File Edit Format View Help

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

Source code (developed by the programmer)

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

Bytecode (generated by the compiler for JVM to read and interpret)
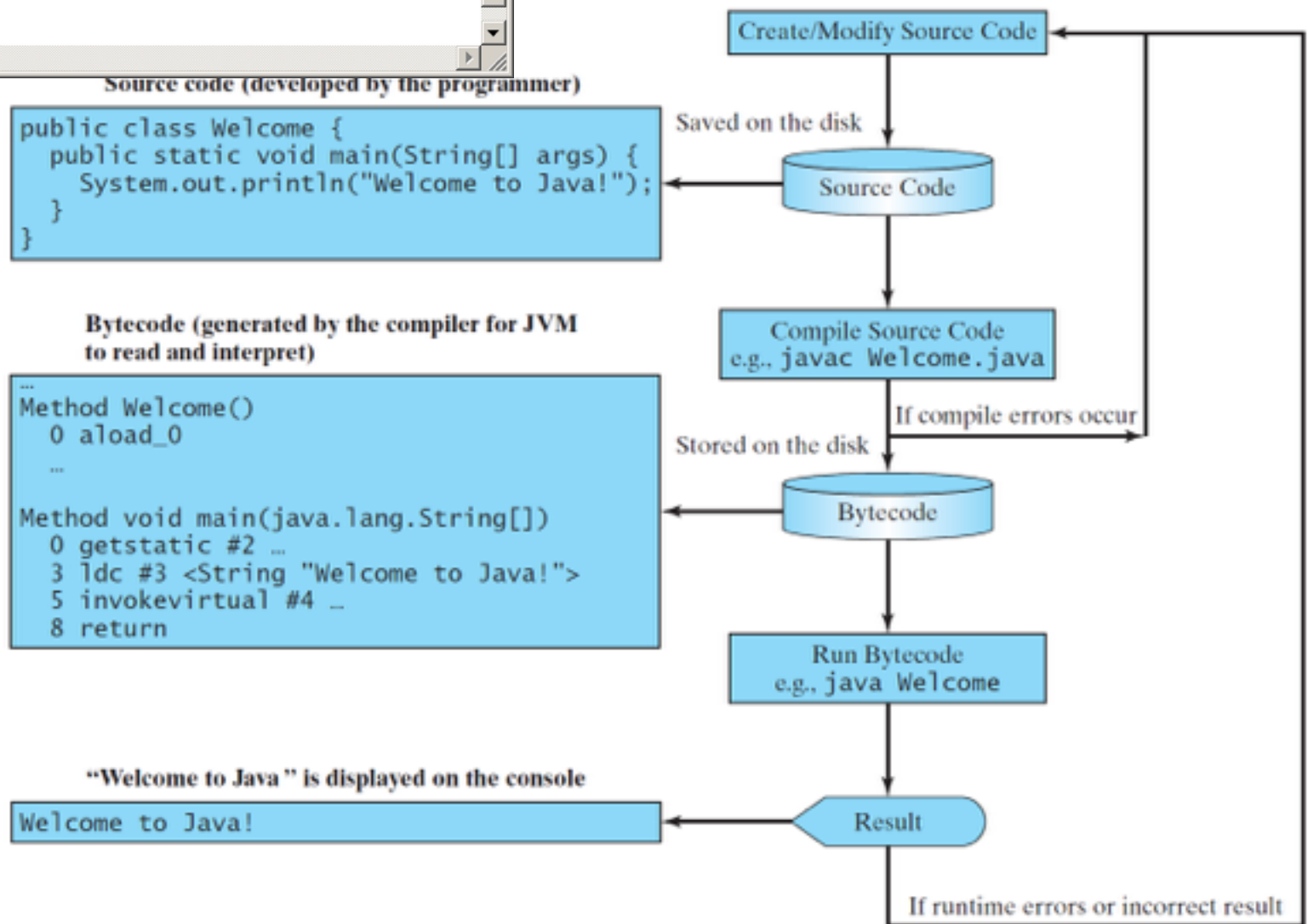
```
...
Method Welcome()
   0 aload_0
   ...

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

"Welcome to Java" is displayed on the console

```
Welcome to Java!
```

Create/Modify Source Code

Saved on the disk

Source Code

Compile Source Code
e.g., javac Welcome.java

If compile errors occur

Stored on the disk

Bytecode

Run Bytecode
e.g., java Welcome

Result

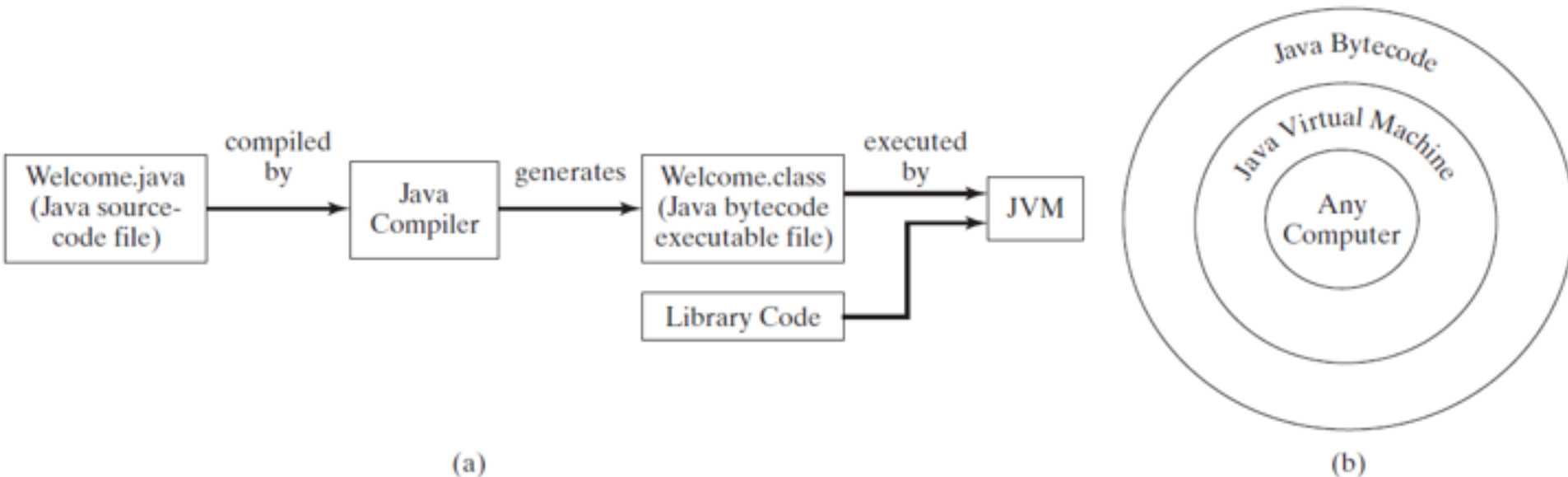If runtime errors or incorrect result
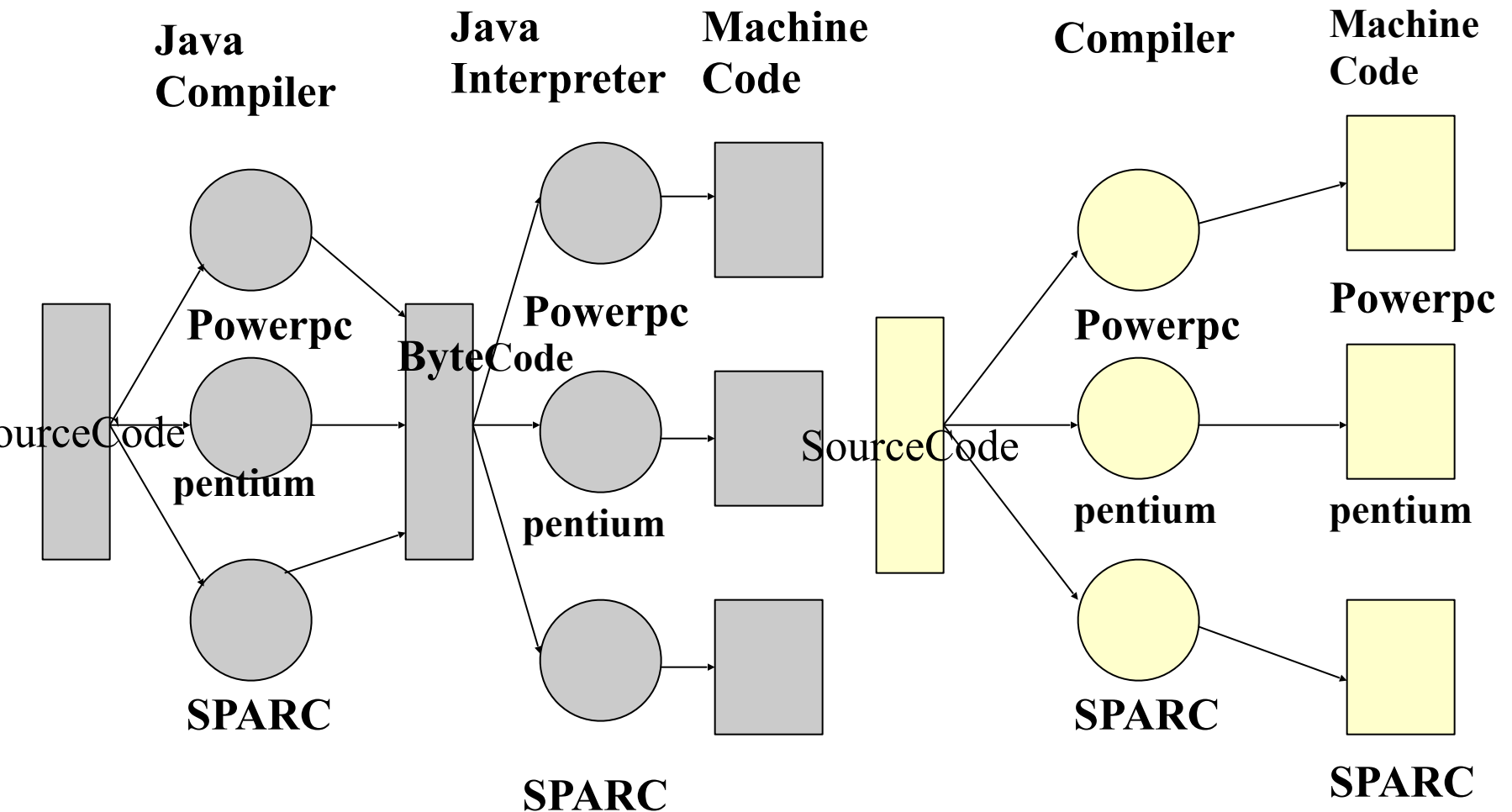
Compiling Java Source Code

Write the program once, and compile the source program into a special type of object code, known as *bytecode*.

The bytecode can then run on any computer with a Java Virtual Machine.

Java Virtual Machine is a software that interprets Java bytecode.

Neutral/Platform Independent

Trace a Program Execution

Enter main method

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

34

Trace a Program Execution

Execute statement

```java
// This program prints Welcome to Java!
public class Welcome {
  public static void main(String[] args) {
    System.out.println("Welcome to Java!");
  }
}
```
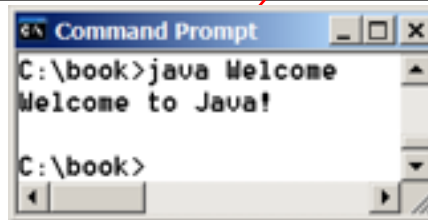
35

Trace a Program Execution

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

```
Command Prompt                    _ □ ×
C:\book>java Welcome
Welcome to Java!

C:\book>
```

print a message to the console

36

Two More Simple Examples

Animation

WelcomeWithThreeMessages

www.cs.armstrong.edu/liang/intro10e/html/
WelcomeWithThreeMessages.html

Animation

ComputeExpression

www.cs.armstrong.edu/liang/intro10e/html/
ComputeExpression.html

Anatomy of a Java Program

✦ Class name

✦ Main method

✦ Statements

✦ Statement terminator

✦ Reserved words

✦ Comments

✦ Blocks

Class Name

Every Java program must have at least one class. Each class has a name.

By convention, class names start with an uppercase letter.

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

Main Method

Line 2 defines the main method. In order to run a class, the class must contain a method named main. The program is executed from the main method.

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

Statement

A statement represents an action or a sequence of actions.

System.out.println("Welcome to Java!")

in Listing 1.1 is to display the greeting "Welcome to Java!".

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

Statement Terminator

Every statement in Java ends with a semicolon (;).

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

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.

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

Special Symbols

Character	Name	Description
{}	Opening and closing braces	Denotes a block to enclose statements.
()	Opening and closing parentheses	Used with methods.
[]	Opening and closing brackets	Denotes an array.
//	Double slashes	Precedes a comment line.
" "	Opening and closing quotation marks	Enclosing a string (i.e., sequence of characters).
;	Semicolon	Marks the end of a statement.

{ ... }

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

(...)

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

.
,

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

// ...

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

"

... "

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

Source Code Structure

- Zero or up to one Package Statement
- Zero or multiple Import Statements
- Zero or multiple Class Declaration
- Zero or multiple Interface Declaration
- Only one Public Class or Interface
- Java source code file name must be as same as the public Class Nname
- Up to one Package Declaration. The First Line of java file
- Import class as specific as possible
  -        import com.abc.dollapp.main.*;
  - Better: import com.abc.dollapp.main.AppMain;

Programming Style and Documentation

✦ Appropriate Comments

✦ Naming Conventions

✦ Proper Indentation and Spacing Lines

✦ Block Styles

Appropriate Comments

Include a summary at the beginning of the program to explain what the program does, its key features, its supporting data structures, and any unique techniques it uses.

Include your name, class section, instructor, date, and a brief description at the beginning of the program.

FileResource.java

Naming Conventions

✦ Choose meaningful and descriptive names.

✦ Class names:
  – Capitalize the first letter of each word in the name. For example, the class name `ComputeExpression`.

Proper Indentation and Spacing

✦ Indentation
  – Indent two spaces.

✦ Spacing
  – Use blank line to separate segments of the code.

Block Styles

Use end-of-line style for braces.

Next-line style

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

End-of-line style

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

Programming Errors

✦ Syntax Errors
– Detected by the compiler

✦ Runtime Errors
– Causes the program to abort

✦ Logic Errors
– Produces incorrect result

Syntax Errors

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

ShowSyntaxErrors

www.cs.armstrong.edu/liang/intro10e/html/ShowSyntaxErrors.html

Runtime Errors

```java
public class ShowRuntimeErrors {
  public static void main(String[] args) {
    System.out.println(1 / 0);
  }
}
```

ShowRuntimeErrors

www.cs.armstrong.edu/liang/intro10e/html/ShowRuntimeErrors.html

Logic Errors

```java
public class ShowLogicErrors {
  public static void main(String[] args) {
    System.out.println("Celsius 35 is Fahrenheit degree ");
    System.out.println((9 / 5) * 35 + 32);
  }
}
```

ShowLogicErrors

www.cs.armstrong.edu/liang/intro10e/html/ShowLogicErrors.html

Exercise 01

1.1 包的使用：示例**DroidApp**（**ch0102.zip**），解压后形成带子目录的文件夹，了解其包含的子目录结构，并尝试用命令行进行带包编译和带包执行，从而掌握目录、包、**classpath**等概念的关系。

1.2 2*2线性方程组求解可以使用**Cramer**规则

$$ax + by = e \qquad x = \frac{ed - bf}{ad - bc} \qquad y = \frac{af - ec}{ad - bc}$$
$$cx + dy = f$$

编写程序，求解一下方程组并显示**x**和**y**的值。

$$3.4x + 50.2y = 44.5$$
$$2.1x + 0.55y = 5.9$$

Submission Deadline: the Next Monday(09/26/2016 9:00am)

HW submission requirement

Printed report

including

1) **Name, studentid, class**

2) **Exercise No.**

3) **Source Code, procedures/results description and results demo such as pictures.**