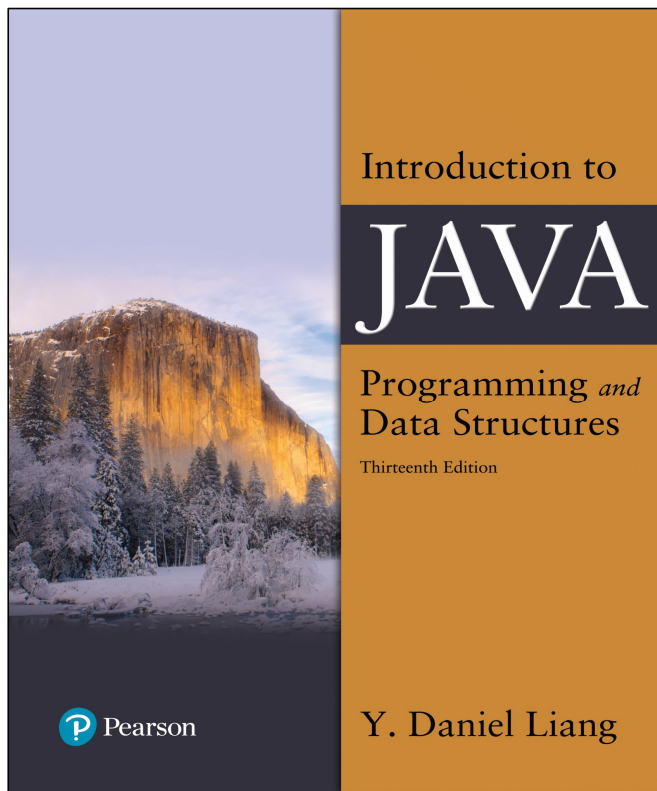


Introduction to Java Programming and Data Structures

Thirteenth Edition



Chapter 1

Introduction to Computers, Programs,
and Java

Bahram Zartoshty

Email: Bahram.zartoshty@csun.edu

COMP 110/L Chapter 1

Bahram Zartoshty

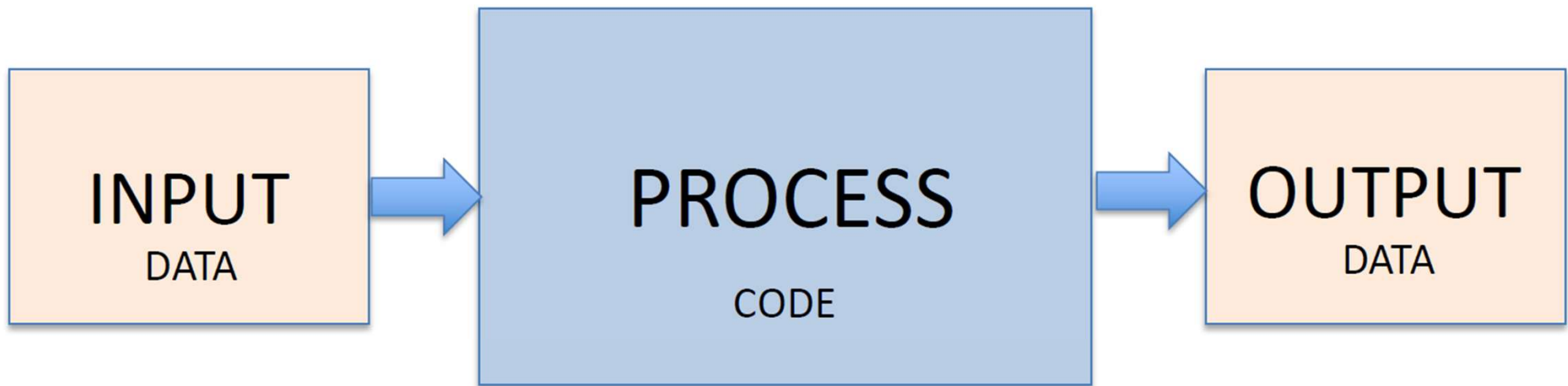
Objectives

- To understand computer basics, programs, and operating systems (§§1.2–1.4).
- To understand the meaning of Java language specification, API, JDK, and IDE (§1.6).
- To write a simple Java program (§1.7).
- To display output on the console (§1.7).
- To explain the basic syntax of a Java program (§1.7).
- To create, compile, and run Java programs (§1.8).
- To use sound Java programming style and document programs properly (§1.9).
- To explain the differences between syntax errors, runtime errors, and logic errors (§1.10).
- develop Java programs using NetBeans (§1.11).

What is Computer Science?

- The fundamental core of Computer Science is problem solving.
- Computers do not solve problems, you do
- Computers are dumb!
- It is up to you, the user, to approach a complex problem, study it, understand it, and develop a solution to it.
- Computers only automate solutions

Computer System (IPO Model)



- Computer Programs

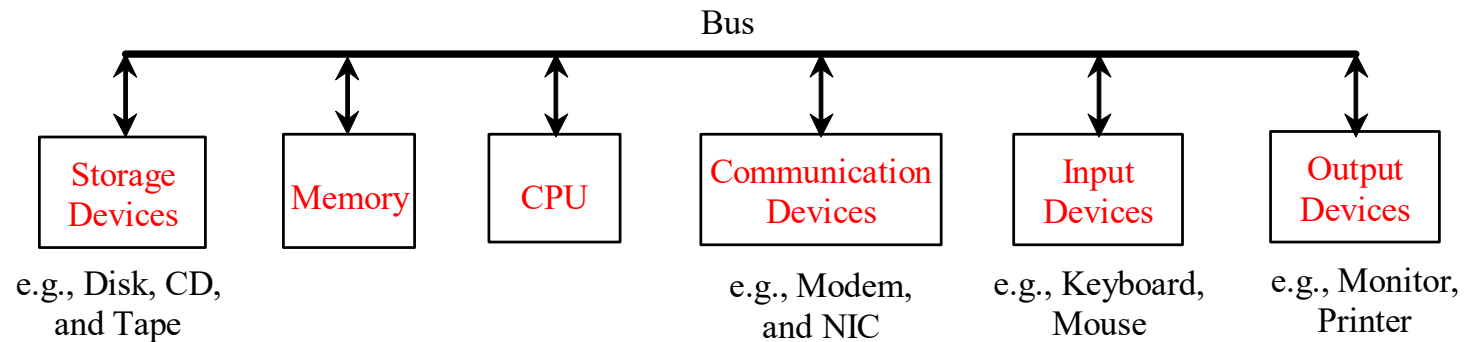
- A computer program is a sequence of instructions in a programming language that a computer can execute or interpret.
 - Java Programming Language
 - Object oriented Programming Language

- *Software*

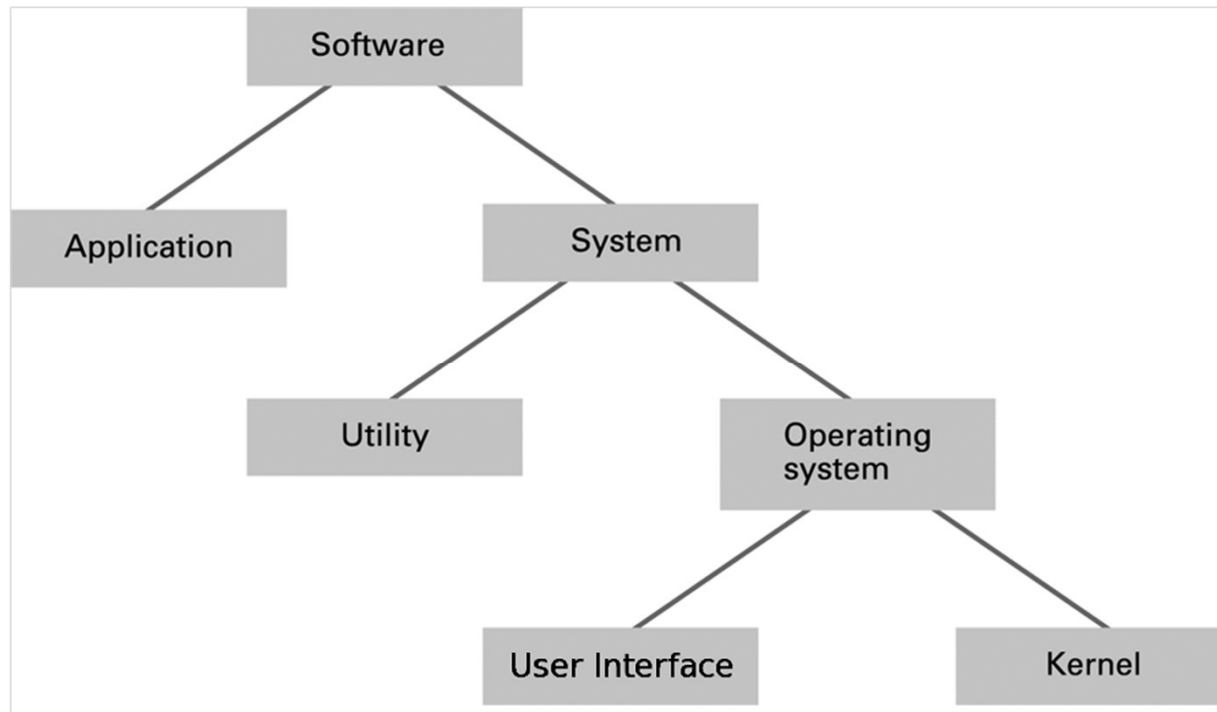
- *Set of programs that tell computer how to solve specific problem*

Hardware

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



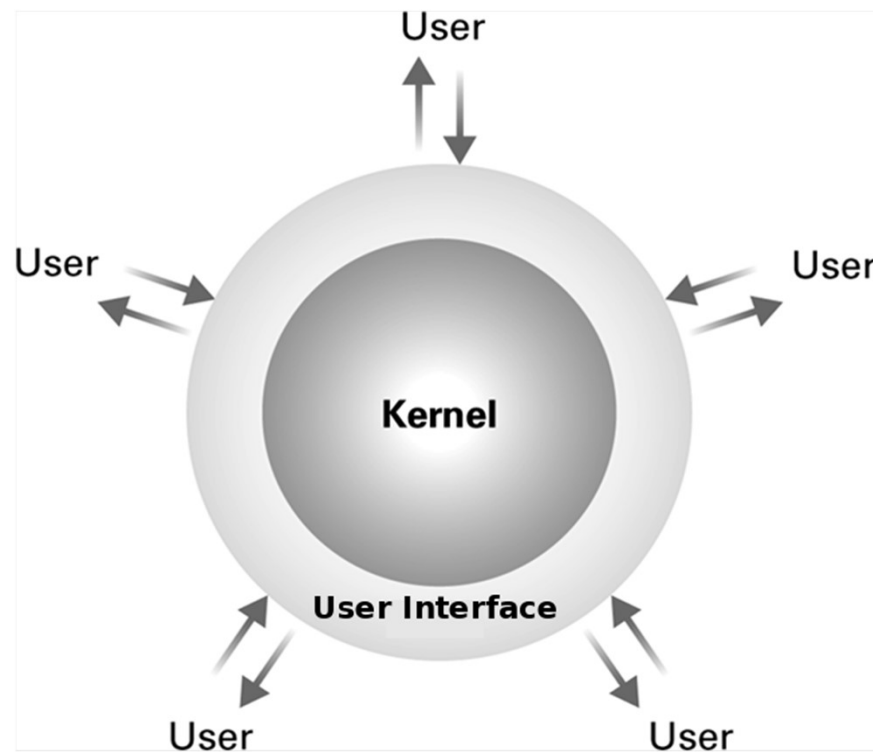
Software classification



Operating System Components

- **User Interface:** Communicates with users
 - Text based (Shell)
 - Graphical user interface (GUI)
- **Kernel:** Performs basic required functions
 - File manager
 - Device drivers
 - Memory manager
 - Scheduler and dispatcher

The user interface act as an intermediary between users and the operating system kernel



Computational Thinking

- Algorithm
 - A process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer

Programming languages are a lot like human languages in that they have syntax rules (grammar of the language) and semantics (meaning of the statement).

Syntax (Grammar)

- Lisa goes to school.



- Goes Lisa school to.



Semantics

Defines what valid sentences *mean*

Lisa goes to the school.



Programming Languages

Machine Language Assembly Language High-Level Language

- Any computer can directly understand only its own **machine language**, *defined by its hardware design*.
 - Generally consist of strings of numbers (ultimately reduced to 1s and 0s) that instruct computers to perform their most elementary operations one at a time.
 - *Machine dependent*—a particular machine language can be used on only one type of computer.
 - For example, to add two numbers, you might write an instruction in binary like this:

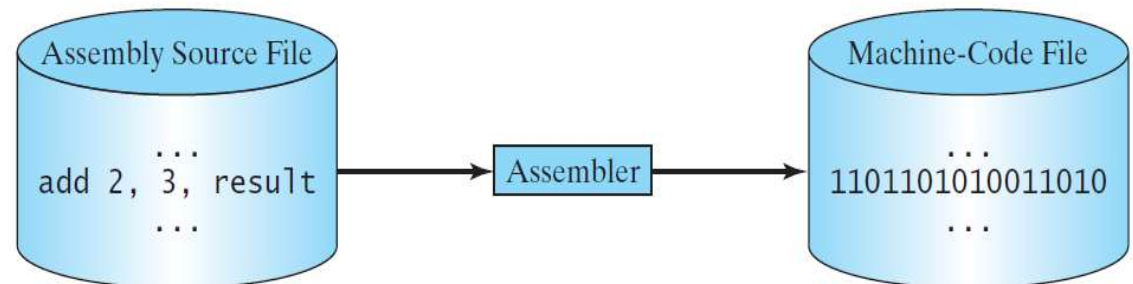
1101101010011010

Programming Languages

Machine Language **Assembly Language** High-Level Language

- Assembly languages were developed to make programming easy.
- English-like abbreviations that represent elementary operations formed the basis of **assembly languages**.
- *Translator programs* called **assemblers** convert early assembly-language programs to machine language.
- 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.
- Single statements accomplish substantial tasks.
- A Java program to calculate the area of a circle might contain a single statement such as
$$\text{area} = \text{radius} * \text{radius} * \text{Math.PI};$$
- High-Level programs must be converted to Machine Language.

Java Assignment Statement syntax

area = radius * radius * Math.PI;



radius * radius * Math.PI = area;

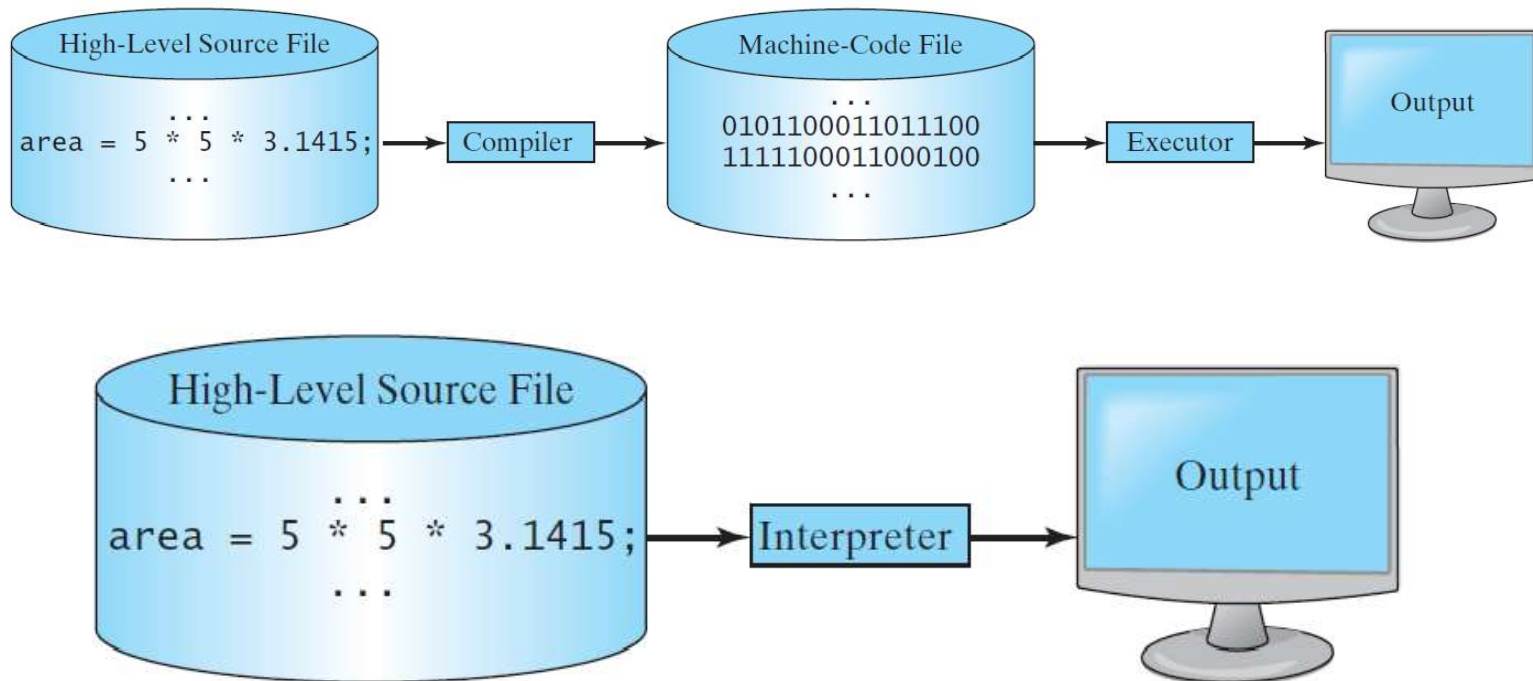


area = radius * radius * Math.PI



Interpreting/Compiling Source Code

- Computer Program: (Source code / Source Program)
- Compiling/Interpreting: (Object Code / Byte Code)

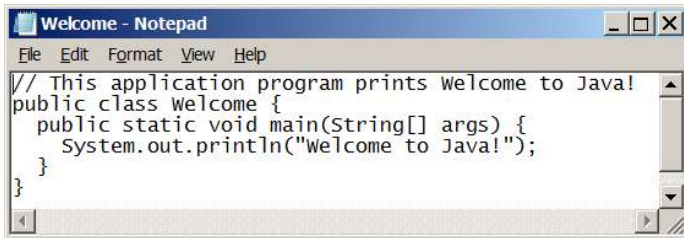


Java (Cont.)

Java Class Libraries

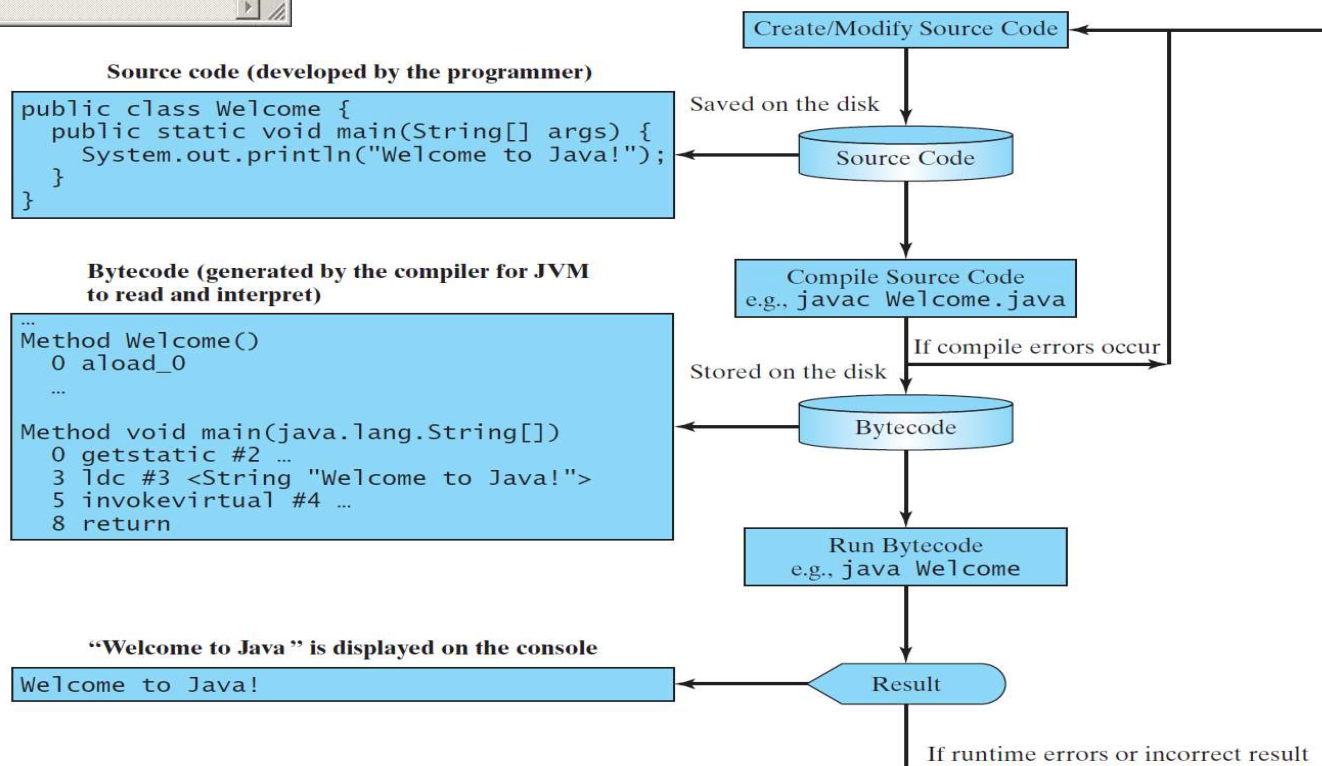
- Rich collections of existing classes and methods
- Also known as the **Java APIs (Application Programming Interfaces)**.

Java Development Environment



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

Edit → Compile → Load → Verify → Execute



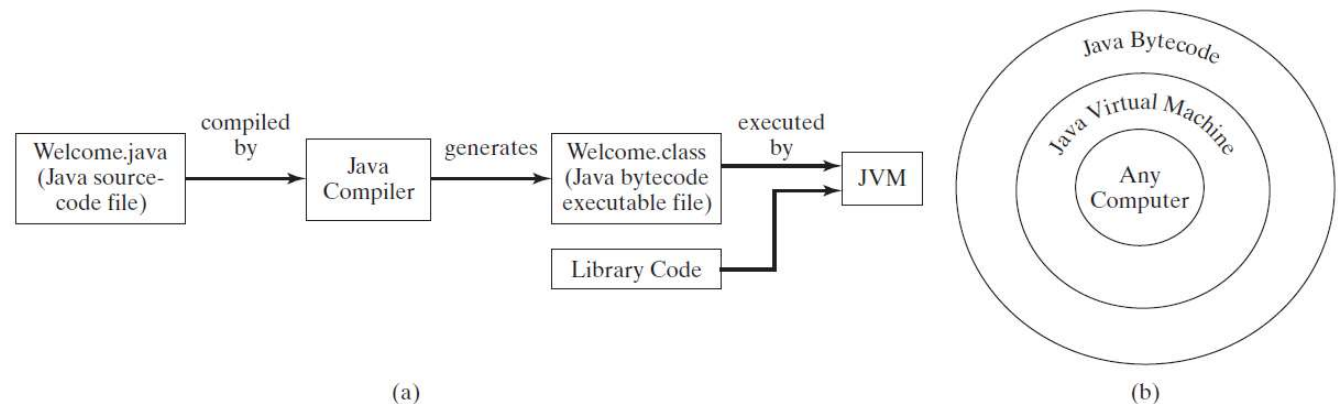
Java SE Development Kit (JDK)

- The JDK is a development environment for building applications, applets, and components using the Java programming language.
- The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

→ [Java SE Development Kit](#)

Java Development Environment (Cont.)

- Bytecodes are **portable** (platform independent)
- The JVM is invoked by the **java** command.
 - `java Welcome`
- The JVM places the program in memory to execute it. (loading **.class** file)
 - As the classes are loaded, the **bytecode verifier** examines their bytecodes
 - Ensures that they're valid and do not violate Java's security restrictions.
- The JVM **executes** the program's bytecodes.



Your First Program in Java: Printing a Line of Text

- Java **application**
 - A computer program that executes when you use the **java command** to launch the Java Virtual Machine (JVM).

```
1. /* File: Welcome.java
2. This program prints Welcome to Java!
3. */
4. public class Welcome {
5.     public static void main(String[] args) {
6.         System.out.println( "Welcome to Java!" );
7.     } // end main method
8. } // end class Welcome
```


Programming Style and Documentation

- Appropriate Comments
- Naming Conventions
- Proper Indentation and Spacing Lines
- 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 Welcome {  
    public static main(String[] args) {  
        System.out.println("Welcome to Java");  
    }  
}
```

Runtime Errors

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

Logic Errors

```
public class Circle {  
    public static void main(String[] args) {  
        System.out.print("Area of a circle with radius 5 is ");  
        System.out.println( 5 * Math.PI );  
    }  
}
```