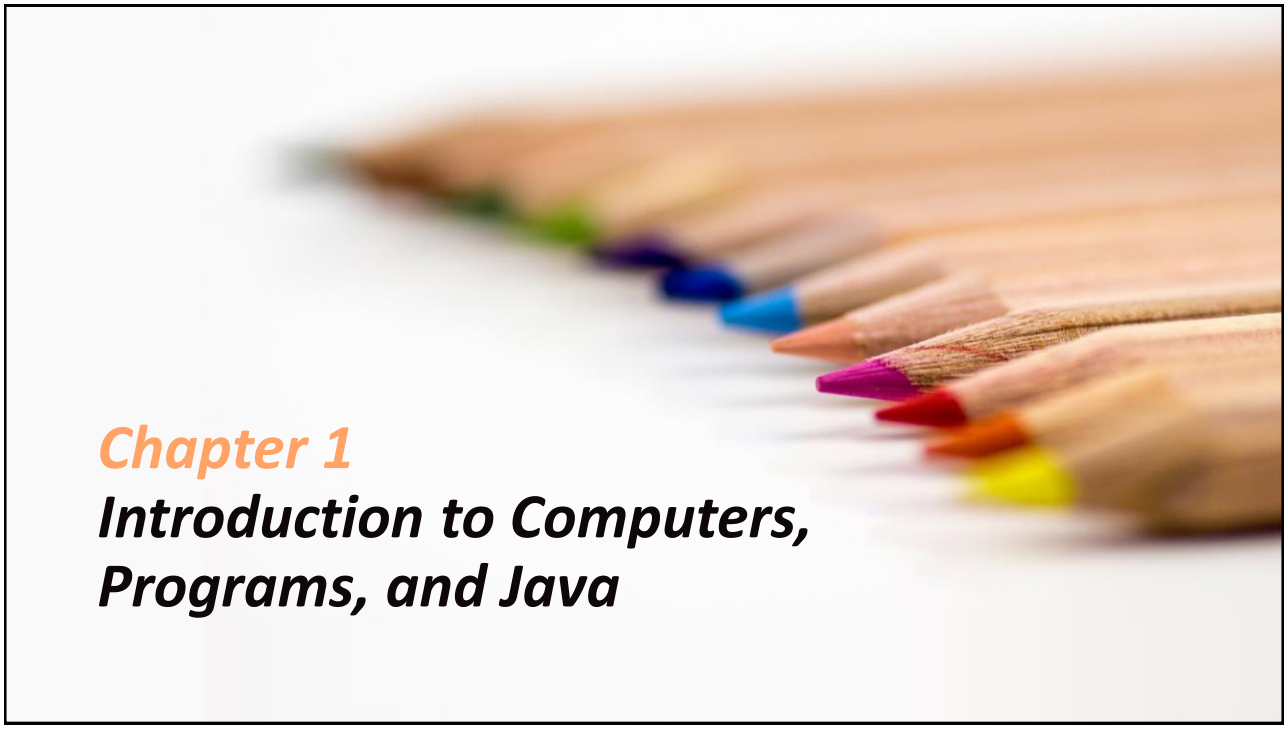
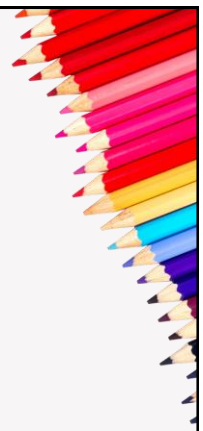


Chapter 1
Introduction to Computers, Programs, and Java

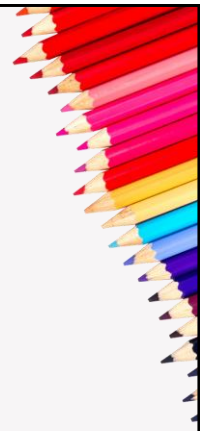
Objectives (1)



- To understand computer basics, programs, and operating systems (§§1.2–1.4).
- To describe the relationship between Java and the World Wide Web (§1.5).
- 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).

Objectives (2)

- 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).
- To develop Java programs using NetBeans (§1.11).
- To develop Java programs using Eclipse (§1.12).



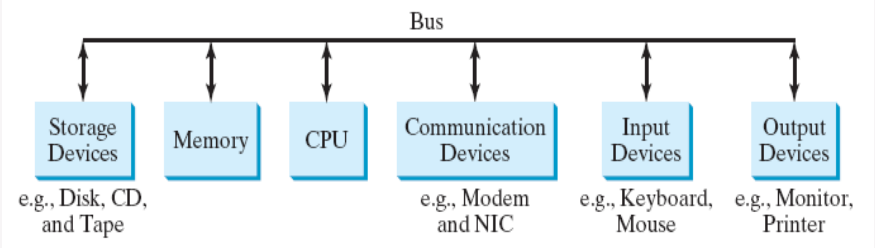
3

1.1 Introduction

- ***The central theme of this book is to learn how to solve problems by writing a program.***
- The term *programming means to* create (or develop) software, which is also called a *program.*
- Software developers create software with the help of powerful tools called *programming languages.*
- This book teaches you how to create programs by using the Java programming language.

1.2 What Is a Computer?

- A *computer* includes both hardware and software.
- In general, *hardware* comprises the visible, physical elements of the computer, and *software* provides the invisible instructions that control the hardware and make it perform specific tasks.
- A computer consists of a CPU, memory, hard disk, floppy disk, monitor, printer, and communication devices.



1.3 Programming Languages

- *Computer programs, known as software, are instructions that tell a computer what to do.*
- A computer's native language, which differs among different types of computers, is its *machine* language—a set of built-in primitive instructions.
- *Assembly language* uses a short descriptive word, known as a *mnemonic,* to represent each of the machine-language instructions.
  - assembler
- High-Level Language

Popular High-Level Programming Languages

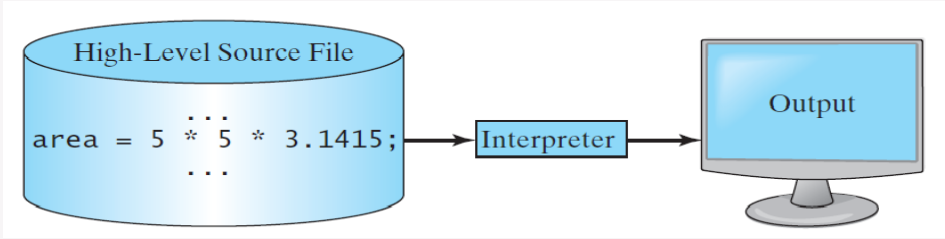
Language	*Description*
Ada	Named for Ada Lovelace, who worked on mechanical general-purpose computers. The Ada language was developed for the Department of Defense and is used mainly in defense projects.
BASIC	Beginner's All-purpose Symbolic Instruction Code. It was designed to be learned and used easily 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***.
- Because a computer cannot understand a source program, a source program must be translated into machine code for execution.
- The translation can be done using another programming tool called an ***interpreter*** or a ***compiler***.

Interpreting Source Code

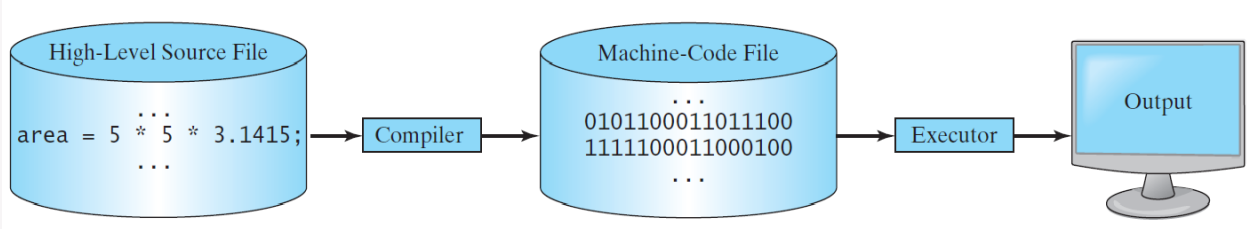
- An interpreter reads one statement from the source code, translates it to the machine code or virtual machine code, and then executes it right away, as shown in the following figure.
  - Note that a statement from the source code may be translated into several machine instructions.



9

Compiling Source Code

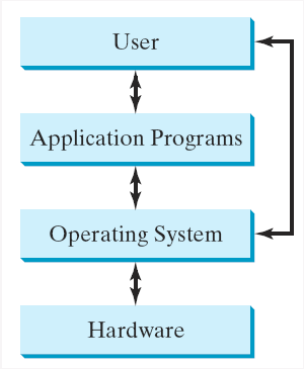
- A compiler translates the entire source code into a machine-code file, and the machine-code file is then executed, as shown in the following figure.



10

1.4 Operating Systems

- *The operating system (OS)* is the most important program that runs on a computer.
- The major tasks of an operating system are as follows:
  - Controlling and monitoring system activities
  - Allocating and assigning system resources
  - Scheduling operations
- Users and applications access the computer's hardware via the operating system.



Operating Systems

- Controlling and Monitoring System Activities
  - Operating systems perform basic tasks, such as recognizing input from the keyboard, sending output to the monitor, keeping track of files and folders on storage devices, and controlling peripheral devices, such as disk drives and printers.
  - An operating system must also ensure that different programs and users working at the same time do not interfere with each other.
  - In addition, the OS is responsible for security, ensuring that unauthorized users and programs are not allowed to access the system.

Operating Systems

- Allocating and Assigning System Resources
  - The operating system is responsible for determining what computer resources a program needs (such as CPU time, memory space, disks, input and output devices) and for allocating and assigning them to run the program.
- Scheduling Operations
  - The OS is responsible for scheduling programs' activities to make efficient use of system resources.
  - Many of today's operating systems support techniques such as *multiprogramming*, *multithreading*, *and multiprocessing* to ***increase system performance***.

Multiprogramming, Multithreading, And Multiprocessing

- Multiprogramming（多道程序设计） allows multiple programs to run simultaneously by sharing the same CPU.
- Multithreading allows a single program to execute multiple tasks at the same time.
- Multiprocessing, or parallel processing, uses two or more processors together to perform subtasks concurrently and then combine solutions of the subtasks to obtain a solution for the entire task

1.5 Java, Web, and Beyond

- Java is a powerful and versatile（多用途的）programming language for developing software running on mobile devices, desktop computers, and servers.
  - Java can be used to develop standalone applications.
  - Java can also be used to develop applications running from a browser, applications for hand-held devices, and applications for Web servers.
  - The software for Android cell phones is developed using Java.
- Java is a general purpose programming language.
- Java is the Internet programming language.

15

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

16

Characteristics of Java

- Java Is Simple
- Java Is ***Object-Oriented***
- Java Is Distributed
- Java Is ***Interpreted***
  - bytecode
  - JVM
- Java Is Robust
- Java Is Secure
- Java Is ***Architecture-Neutral***
  - Write once, run anywhere
  - JVM
- Java Is ***Portable***
- Java's Performance
- Java Is ***Multithreaded***
- Java Is Dynamic

17

1.6 The Java Language Specification, API, JDK, and IDE

- The Java language specification and the Java API define the Java standards.
- *Java's syntax and semantics* are *defined in the **Java language specification**.*
- *The Java library is defined in the **Java API**.*
  - predefined classes and interfaces for developing Java programs
- *The **JDK** is the software for developing and running Java programs.*
- *An **IDE** is an integrated development environment for rapidly developing programs.*

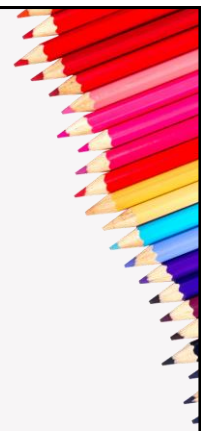
Java SE, EE, and ME

- Java *Standard Edition* (***Java SE***)
  - *to develop client-side applications.*
  - *The applications* can run standalone or as applets running from a Web browser.
- *Java Enterprise Edition* (*Java EE*)
  - *to develop server-side applications, such as Java* servlets, JavaServer Pages (JSP), and JavaServer Faces (JSF).
- *Java Micro Edition* (*Java ME*)
  - *to develop applications for mobile devices, such as* cell phones.
- This book uses ***Java SE*** to introduce Java programming.

Java SE Versions History

Java SE Version	Version Number	Release Date
JDK 1.0 (Oak)	1.0	January 1996
JDK 1.1	1.1	February 1997
J2SE 1.2 (Playground)	1.2	December 1998
J2SE 1.3 (Kestrel)	1.3	May 2000
J2SE 1.4 (Merlin)	1.4	February 2002
J2SE 5.0 (Tiger)	1.5	September 2004
Java SE 6 (Mustang)	1.6	December 2006
Java SE 7 (Dolphin)	1.7	July 2011

Java SE Version	Version Number	Release Date
Java SE 8	1.8	March 2014
Java SE 9	9	September, 21st 2017
Java SE 10	10	March, 20th 2018
Java SE 11	11	September, 25th 2018
Java SE 12	12	March, 19th 2019
Java SE 13	13	September, 17th 2019
Java SE 14	14	March, 17th 2020
Java SE 15	15	September, 15th 2020
Java SE 16	*16*	*Expected on March 2021*



1.7 A Simple Java Program

LISTING 1.1 Welcome.java

```
1  public class Welcome {
2    public static void main(String[] args) {
3      // Display message Welcome to Java! on the console
4      System.out.println("Welcome to Java!");
5    }
6  }
```

```
Welcome to Java!
```

21

Anatomy of a Java Program

- Every Java program must have at least one *class*.
  - Each class has a name.
  - By convention, class names start with an uppercase letter.
- *A Java program is executed from the main method in the class.*

LISTING 1.1 Welcome.java

```
1  public class Welcome {
2    public static void main(String[] args) {
3      // Display message Welcome to Java! on the console
4      System.out.println("Welcome to Java!");
5    }
6  }
```

Anatomy of a Java Program

- A method is a construct that contains *statements*.
- Every statement in Java ends witha semicolon (;), known as the *statement terminator.*
- The statement **System.out.println("Welcome to Java!")** in the program in Listing 1.1 is a statement to display the greeting "Welcome to Java!".

LISTING 1.1 Welcome.java

```
1  public class Welcome {
2    public static void main(String[] args) {
3      // Display message Welcome to Java! on the console
4      System.out.println("Welcome to Java!");
5    }
6  }
```

Anatomy of a Java Program

- *Reserved words, or keywords, have a specific meaning to the compiler and cannot be used* for other purposes in the program.
- In Java, comments are preceded by two slashes (**//**) on a line, called a *line comment,* or enclosed between ***/*** *and* ****/*** on one or several lines, called a *block comment or paragraph comment.*

LISTING 1.1 Welcome.java

```
1  public class Welcome {
2    public static void main(String[] args) {
3      // Display message Welcome to Java! on the console
4      System.out.println("Welcome to Java!");
5    }
6  }
```

Anatomy of a Java Program

- A pair of curly braces in a program forms a *block* that groups the program's components.
- In Java, each block begins with an opening brace ({) and ends with a closing brace (}).
- Every class has a *class block* that groups the data and methods of the class.
- every method has a *method block* that groups the statements in the method.
- Blocks can be nested.

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

Class block

Method block

Two More Simple Examples

LISTING 1.2 WelcomeWithThreeMessages.java

```
1  public class WelcomeWithThreeMessages {
2    public static void main(String[] args) {
3      System.out.println("Programming is fun!");
4      System.out.println("Fundamentals First");
5      System.out.println("Problem Driven");
6    }
7  }
```

WelcomeWithThreeMessages

Run

26

Two More Simple Examples

LISTING 1.3 ComputeExpression.java

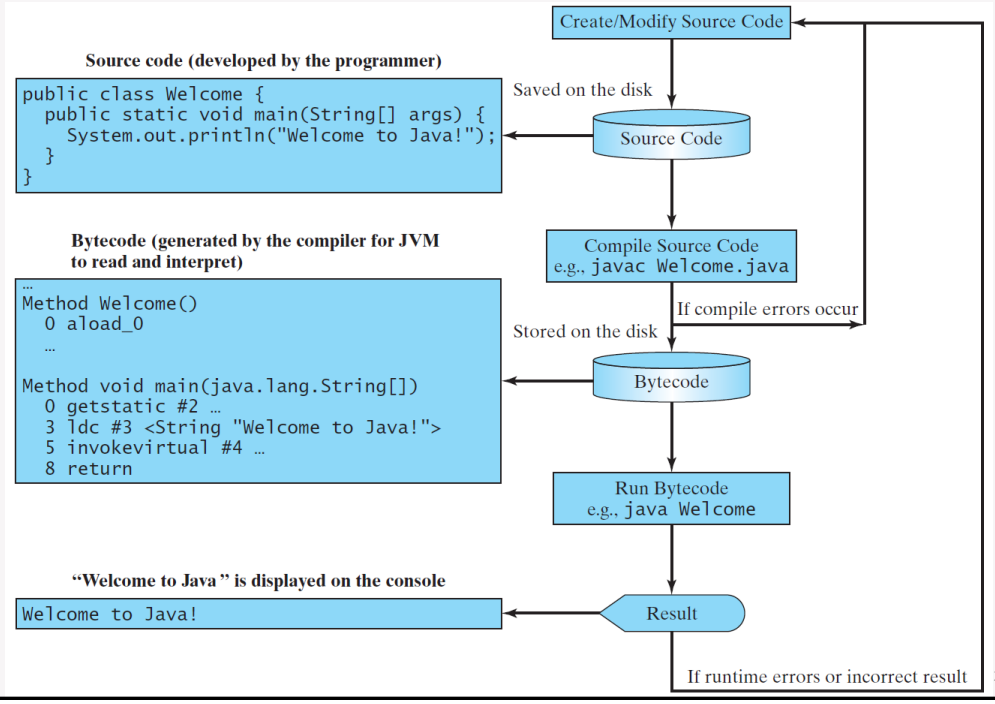
```
1  public class ComputeExpression {
2    public static void main(String[] args) {
3      System.out.println((10.5 + 2 * 3) / (45 - 3.5));
4    }
5  }
```

ComputeExpression

Run

27

Creating, Compiling, and Running Programs



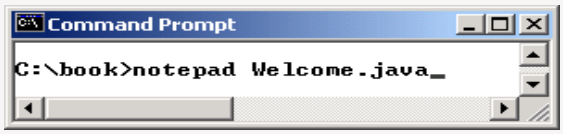
28

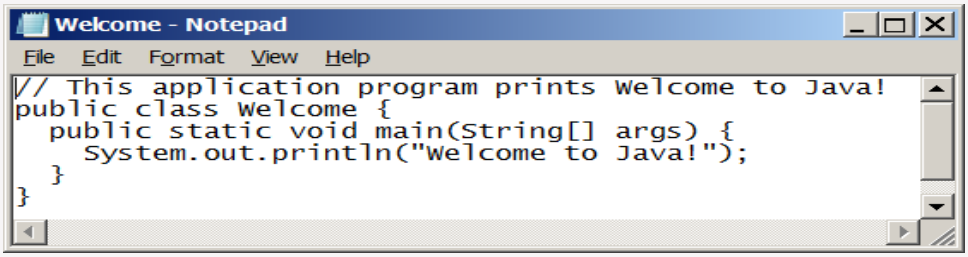
Creating and Editing Using NotePad

To use NotePad, type

notepad Welcome.java

from the DOS prompt.





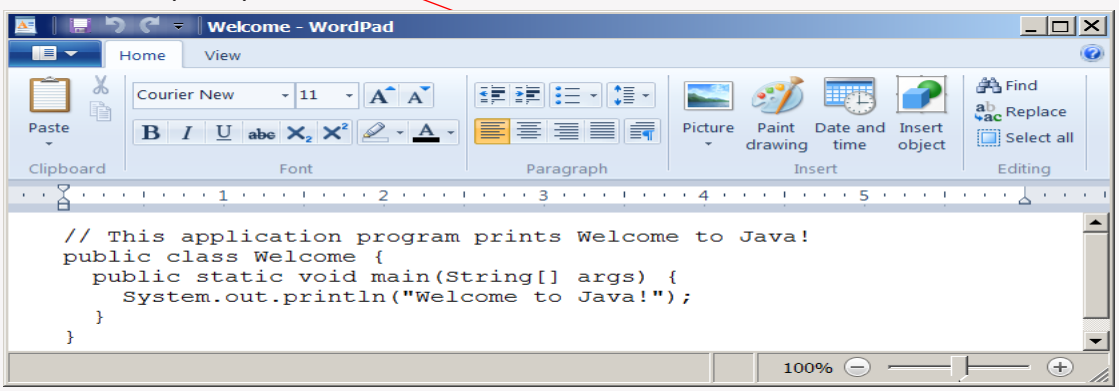
29

Creating and Editing Using WordPad

To use WordPad, type

write Welcome.java

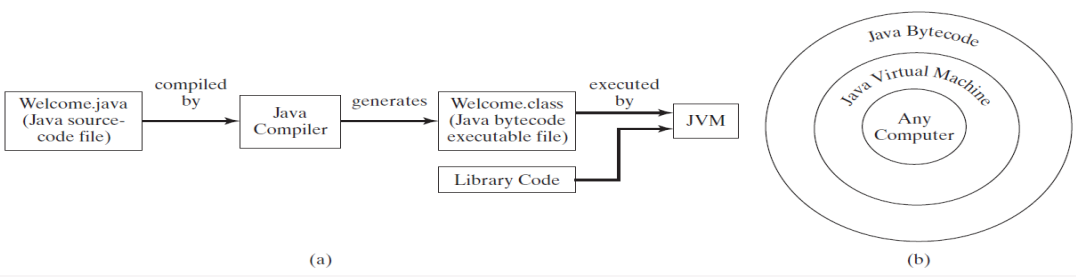
from the DOS prompt.



30

Compiling Java Source Code

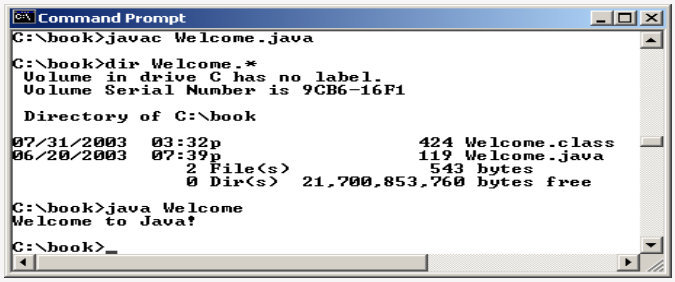
- Java was designed to run object programs on any platform.
- With Java, you write the program once, and compile the source program into a special type of object code, known as *bytecode*.
- The bytecode can then run on any computer with a Java Virtual Machine.
- Java Virtual Machine is a software that interprets Java bytecode.



31

Compiling and Running Java from the Command Window

- *Advanced system settings>System properties>Environment variables*
- Command Prompt
  - Set path to JDK bin directory
    - set path=c:\Program Files\java\jdk1.8.0\bin
  - Set classpath to include the current directory
    - set classpath=.
- Compile
  - javac Welcome.java
- Run
  - java Welcome



Popular Java IDEs

- **1.11** Developing Java Programs Using NetBeans
- **1.12** Developing Java Programs Using Eclipse
- *You can edit, compile, run, and debug Java Programs using Eclipse.*
  - Download and install Java SE JDK 8/ **JDK11(+JavaFX SDK)**
  - https://www.oracle.com/technetwork/java/javase/downloads/index.html
  - Download Eclipse IDE for Java Developers
  - https://www.eclipse.org/downloads/packages/
  - Unzip

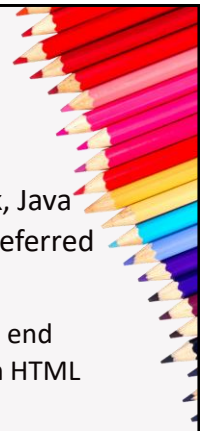
33

1.9 Programming Style and Documentation

- *Good programming style and proper documentation make a program easy to read and help programmers prevent errors.*
- *Programming style deals with what programs look like.*
- *Documentation* is the body of explanatory remarks and comments pertaining to a program.
- Guidelines
  - Appropriate Comments
  - Naming Conventions
  - Proper Indentation and Spacing Lines
  - Block Styles

34

Appropriate Comments and Comments Style



- Include a summary at the beginning of the program that explains what the program does, its key features, and any unique techniques it uses.
- In a long program, you should also include comments that introduce each major step and explain anything that is difficult to read.
- In addition to line comments and block, Java supports comments of a special type, referred to as javadoc comments.
  - javadoc comments begin with **/**** and end with ***/**. They can be extracted into an HTML file using the JDK's *javadoc* command.
- Use javadoc comments (/** ... */) for commenting on an entire class or an entire method.
- For commenting on steps inside a method, use line comments (//).

35

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.
- A single space should be added on both sides of a binary operator.

36

Block Styles

- A block is a group of statements surrounded by braces.
- There are two popular styles, next-line style and end-of-line style.
- Use end-of-line style for braces.

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

Next-line style

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

End-of-line style

37

1.10 Programming Errors

- *Programming errors can be categorized into three types: syntax errors, runtime errors, and logic errors.*
- Syntax Errors
  - Detected by the compiler
- Runtime Errors
  - Causes the program to abort
- Logic Errors
  - Produces incorrect result

38

Syntax Errors

LISTING 1.4 ShowSyntaxErrors.java

```
1  public class ShowSyntaxErrors {
2    public static main(String[] args) {
3      System.out.println("Welcome to Java);
4    }
5  }
```

ShowSyntaxErrors

Run

39

Runtime Errors

LISTING 1.5 ShowRuntimeErrors.java

```
1  public class ShowRuntimeErrors {
2    public static void main(String[] args) {
3      System.out.println(1 / 0);
4    }
5  }
```

ShowRuntimeErrors

Run

40

Logic Errors

LISTING 1.6 ShowLogicErrors.java

```
1  public class ShowLogicErrors {
2    public static void main(String[] args) {
3      System.out.println("Celsius 35 is Fahrenheit degree ");
4      System.out.println((9 / 5) * 35 + 32);
5    }
6  }
```

ShowLogicErrors

Run

41

Common Errors for New Programmers

- Common Error 1: Missing Braces
- Common Error 2: Missing Semicolons
- Common Error 3: Missing Quotation Marks
- Common Error 4: Misspelling Names

```
public class Welcome {

} ←——Type this closing brace right away to match the opening brace
```

```
public static void main(String[] args) {
  System.out.println("Programming is fun!");
  System.out.println("Fundamentals First");
  System.out.println("Problem Driven")
}
                                    Missing a semicolon
```

```
System.out.println("Problem Driven );
                                Missing a quotation mark
```

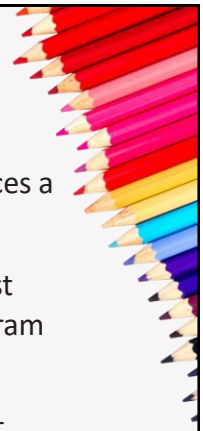
```
1  public class Test {
2    public static void Main(string[] args) {
3      System.out.println((10.5 + 2 * 3) / (45 - 3.5));
4    }
5  }
```

Chapter Summary

Chapter Summary

- Java is platform independent, meaning that you can write a program once and run it on any computer.
- The Java source file name must match the public class name in the program. Java source code files must end with the **.java** extension.
- Every class is compiled into a separate bytecode file that has the same name as the class and ends with the **.class** extension.
- To compile a Java source-code file from the command line, use the **javac** command.
- To run a Java class from the command line, use the **java** command.

Chapter Summary



- Every Java program is a set of class definitions. The keyword **class** introduces a class definition. The contents of the class are included in a block.
- Methods are contained in a class. To run a Java program, the program must have a **main method**. The main method is the entry point where the program starts when it is executed.
- Every statement in Java ends with a semicolon (;), known as the statement terminator.
- Java source programs are **case sensitive**.
- Programming errors can be categorized into three types: ***syntax errors***, ***runtime errors***, and ***logic errors***.

Programming Exercises

教材编程练习题

1, 3, 4, 5, 8, 11, 12, 13