

CHAPTER

1

INTRODUCTION





Chapter Goals

- ❑ To learn about computers and programming
- ❑ To compile and run your first Java program
- ❑ To recognize compile-time and run-time errors
- ❑ To describe an algorithm with pseudocode

In this chapter, you will learn how to write and run your first Java program. You will also learn how to diagnose and fix programming errors, and how to use pseudocode to describe an algorithm.



Contents

- ❑ Computer Programs
- ❑ The Anatomy of a Computer
- ❑ The Java Programming Language
- ❑ Becoming Familiar with your Programming Environment
- ❑ Analyzing Your First Program
- ❑ Errors
- ❑ Problem Solving:
 - Algorithm Design





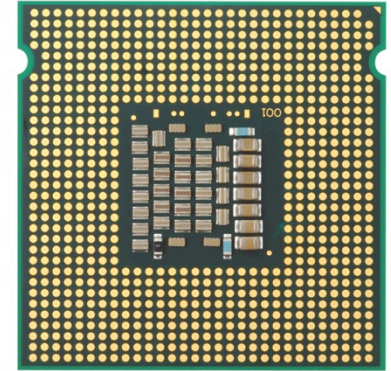
1.1 Computer Programs

- ❑ A **Computer Program** is a sequence of instructions and decisions
- ❑ Computers execute very basic instructions in rapid succession
- ❑ Programming is the act of designing and implementing computer programs



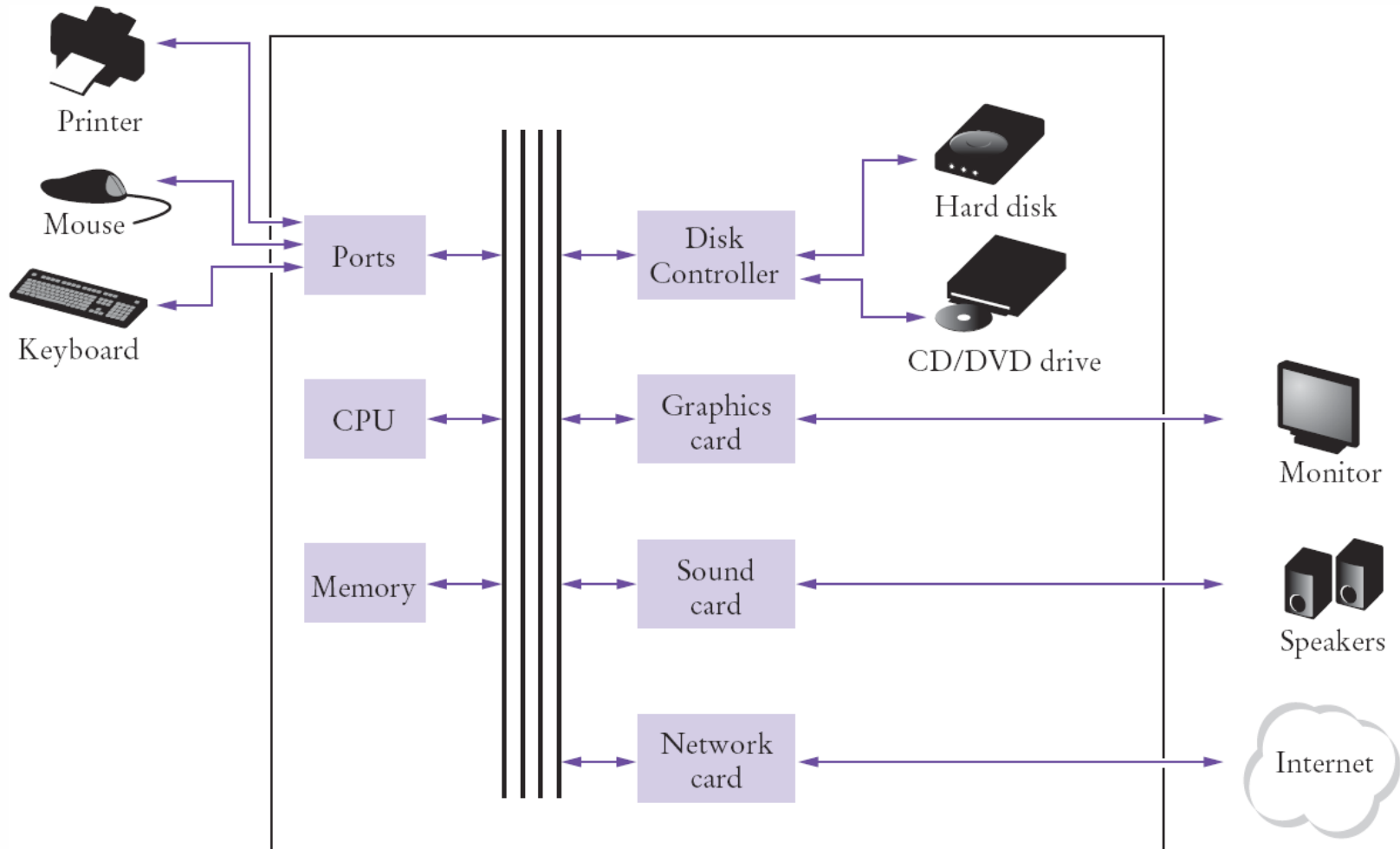
1.2 The Anatomy of a Computer

- ❑ The **central processing unit (CPU)** performs program control and data processing
- ❑ Storage devices include **memory (RAM)** and **secondary storage**
 - Hard disk
 - Flash drives
 - CD/DVD drives
- ❑ **Input/Output devices** allow the user to interact with the computer
 - Mouse, keyboard, printer, screen...





Schematic Design of a PC





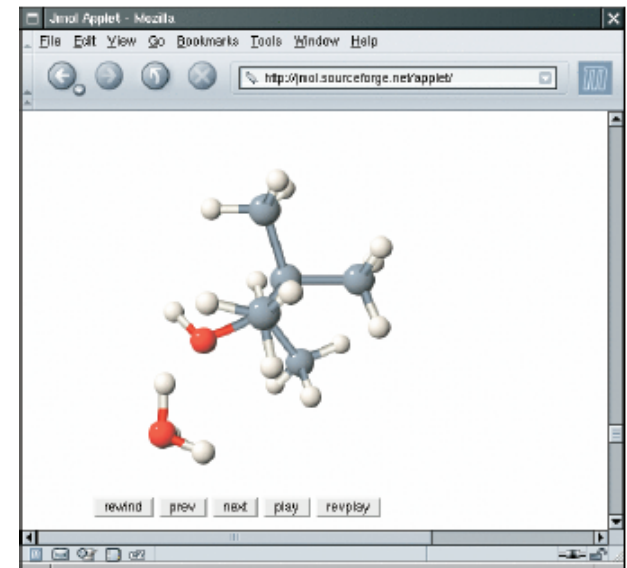
When you ‘run’ a program

- ❑ Program instructions and data (such as text, numbers, audio, or video) are stored on the hard disk, on a compact disk (or DVD), or elsewhere on the network.
- ❑ When a program is started, it is brought into memory, where the CPU can read it.
- ❑ The CPU runs the program one instruction at a time. The program may react to user input
- ❑ As directed by these instructions and the user, the CPU reads data, modifies it, and writes it back to memory, the screen or secondary storage.



1.3 The Java Language

- ❑ In 1991, James Gosling of Sun Microsystems designed what would become the Java programming language
- ❑ Java was originally designed for *programming consumer devices*, but it was first successfully used to *write Internet applets*
 - An applet is typically embedded inside a web page and runs in the context of a browser





Java History

❑ Java Design Goals

- **Safe**: Can be run inside a browser and will not attack your computer
- **Portable**: Run on many Operating Systems
 - Windows
 - Mac OS

❑ Java programs are distributed as instructions for a ‘**virtual machine**,’ making them **platform-independent**

- Virtual machines are available for most Operating Systems. The iPhone is a notable exception



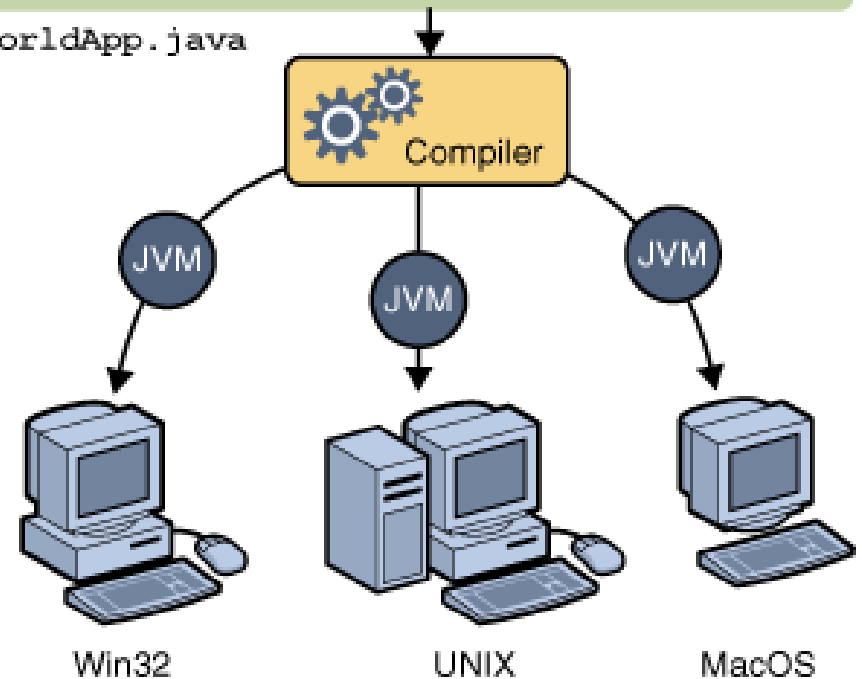
Java Virtual Machines

- ❑ Source code
- ❑ Portable 'byte code'
 - The compiler generates byte code in a 'class' file which can be run on any Java Virtual Machine
- Compiled java programs contain instructions for the Java Virtual machine, a program that simulates a real CPU

Java Program

```
class HelloWorldApp {  
    public static void main(String[] args) {  
        System.out.println("Hello World!");  
    }  
}
```

HelloWorldApp.java





Java Timeline

Version	Year	Important New Features
1.0	1996	
1.1	1997	Inner classes
1.2	1998	Swing, Collections framework
1.3	2000	Performance enhancements
1.4	2002	Assertions, XML support
5	2004	Generic classes, enhanced for loop, auto-boxing, enumerations, annotations
6	2006	Library improvements
7	2011	Small language changes and library improvements

- ❑ Oracle purchased Sun (along with Java) in 2010
 - There are still quite a few references and links to Sun Microsystems which are now re-directed to Oracle



Library Packages

- ❑ The Java language itself is relatively simple, but Java contains a cast set of **library packages** that are required to write useful programs
- ❑ There are packages for graphics, user-interface design, cryptography, networking, sound, database storage, and many other purposes



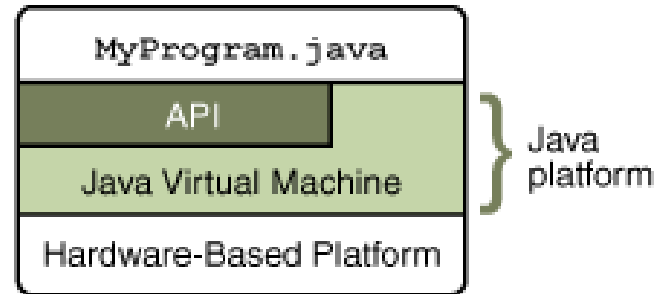
The Java API

- ❑ The Java Platform consists of two parts:

- 1) Java Virtual Machine

- 2) Java API

-- also called libraries



- ❑ The Application Programming Interface (API) is a huge collection of handy software packages that programmers can use:
 - Graphics, user interface, networking, sound, database, math, and many more



The Java SDK

- ❑ You need to install the Java **SDK** (Software Development Kit) to create Java programs
 - Your instructor will suggest one to start with
 - Google ‘Java SDK download,’ Get SE version
 - Location after installed on Windows will be:
 - C:\Program Files\Java\jdk1.7.x
 - The last few numbers may vary with releases
- ❑ The SDK includes programs such as:
 - java.exe (Executes Java applications)
 - javac.exe (Java compiler)
 - javadoc.exe (Javadoc generator)

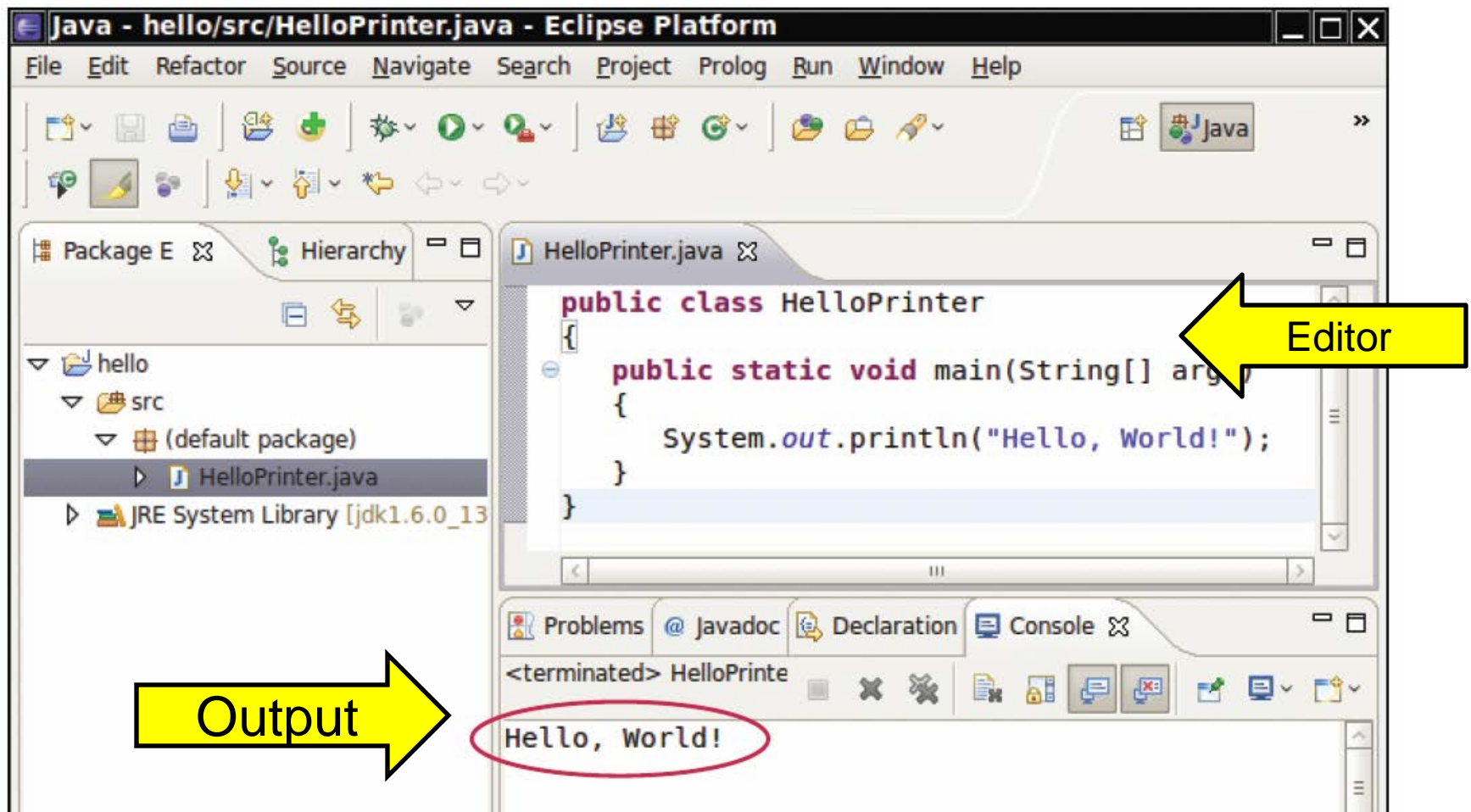


1.4 Programming Environment

- ❑ There are many free programming tools available for Java
 - Your instructor will suggest one to start with
- ❑ Components of an Integrated Development Environment (IDE):
 - Source code editor helps programming by:
 - Listing line numbers of code
 - Color lines of code (comments, text...)
 - Auto-indent source code
 - Output window
 - Debugger



An Example IDE



- Many IDEs are designed specifically for Java programming



Your First Program

❑ Traditional 'Hello World' program in Java

```
1 public class HelloPrinter
2 {
3     public static void main(String[] args)
4     {
5         System.out.println("Hello, World!");
6     }
7 }
```

❑ We will examine this program in the next section

- Typing it into your IDE would be good practice!
- Be careful of spelling
- JaVa iS CaSe SeNsItIvE
- Java uses special characters, e.g. { } () ;



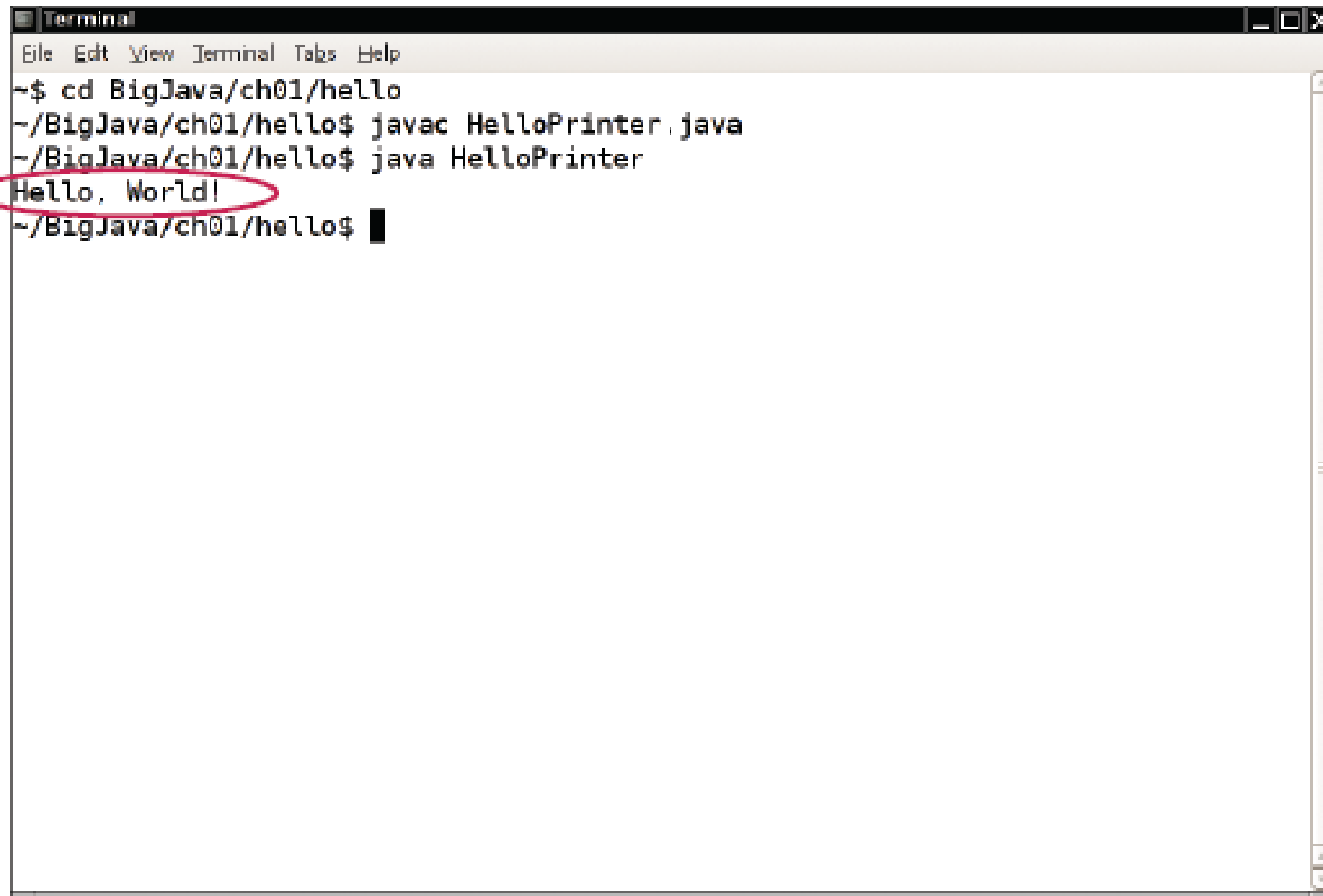
Text Editor Programming

- ❑ You can also use a simple text editor such as Notepad to write your source code
- ❑ Once saved as HelloPrinter.java, you can use a console window to:
 - 1) Compile the program
 - 2) Run the program

```
Administrator: C:\Windows\system32\cmd.exe

D:\temp\hello>javac HelloPrinter.java
D:\temp\hello>java HelloPrinter
Hello, World!
D:\temp\hello>_
```

HelloPrinter in a Console Window



```
Terminal
File Edit View Terminal Tabs Help
~$ cd BigJava/ch01/hello
~/BigJava/ch01/hello$ javac HelloPrinter.java
~/BigJava/ch01/hello$ java HelloPrinter
Hello, World!
~/BigJava/ch01/hello$
```

The image shows a terminal window titled "Terminal" with a menu bar (File, Edit, View, Terminal, Tabs, Help). The command history shows the user navigating to the directory ~/BigJava/ch01/hello, compiling the HelloPrinter.java file using javac, and then running the HelloPrinter class using java. The output "Hello, World!" is displayed and circled in red. The prompt ~/BigJava/ch01/hello\$ is visible at the bottom.

Figure 10 Running the HelloPrinter Program in a Console Window

HelloPrinter in an IDE

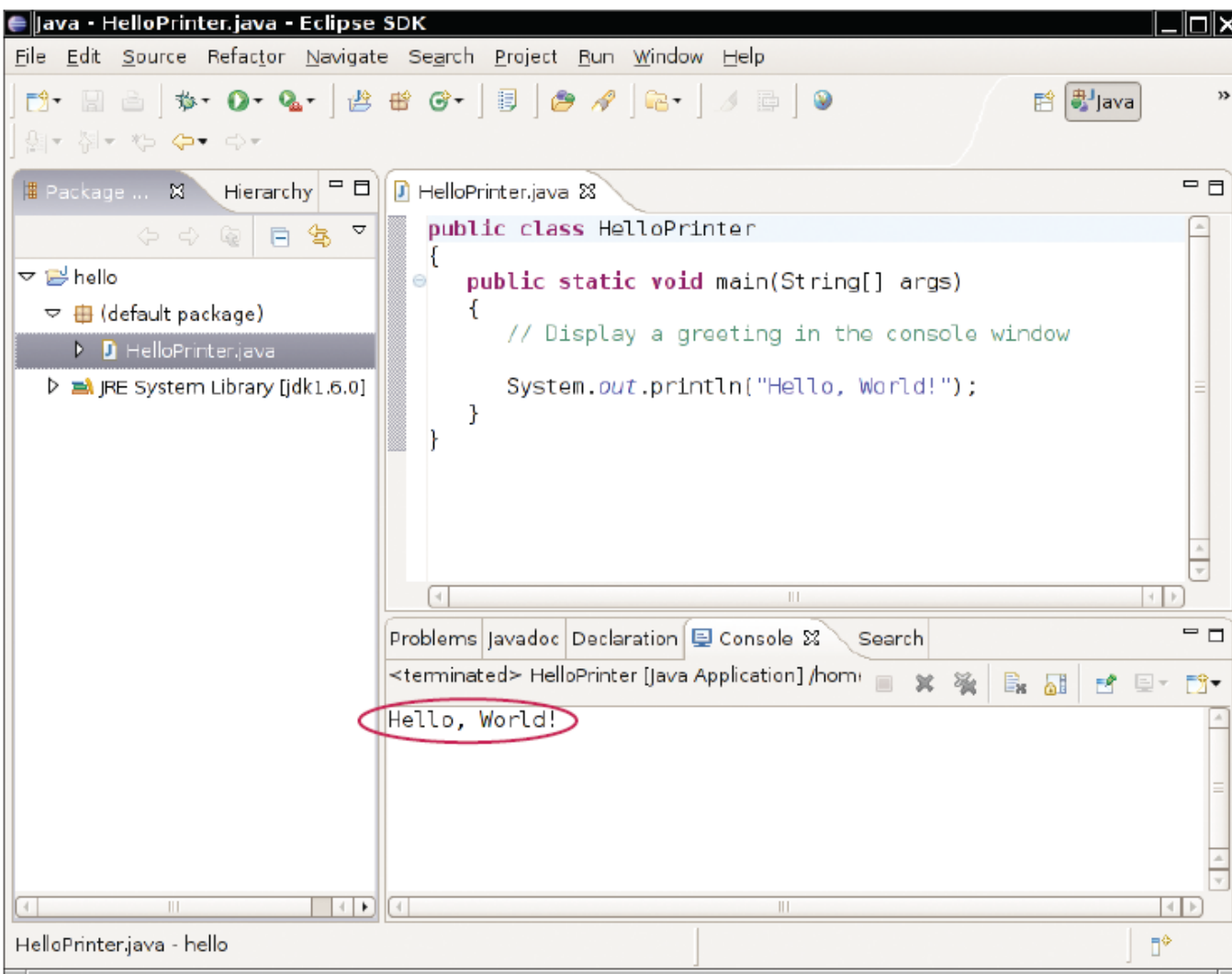


Figure 11

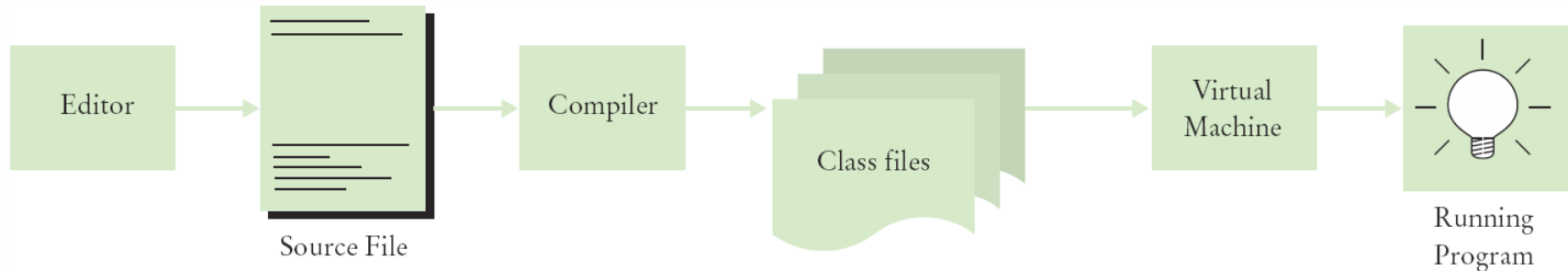
Running the HelloPrinter Program in an Integrated Development Environment

Big Java by Cay Horstmann

Copyright © 2008 by John Wiley & Sons. All rights reserved.



Source Code to Running Program

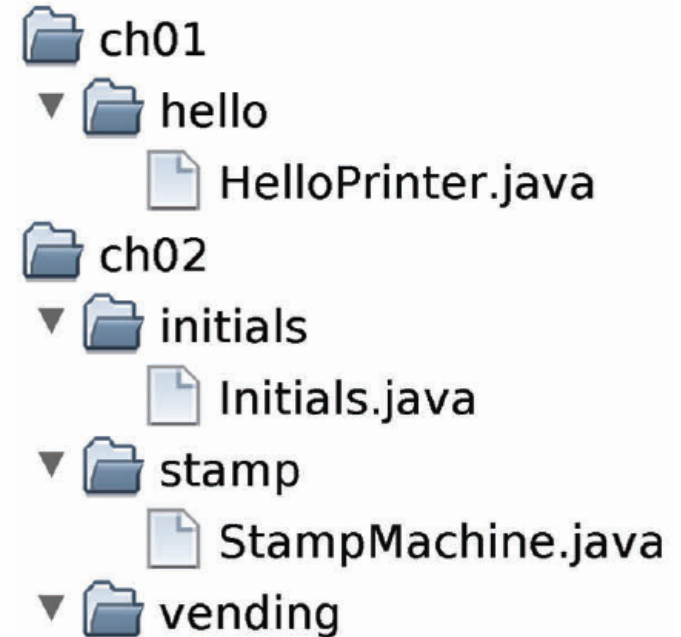


- ❑ The **compiler** generates the **.class** file which contains instructions for the Java Virtual machine
- ❑ Class files contain ‘byte code’ that you cannot edit
 - D:\temp\hello>Type HelloPrinter.class
 - `☪☐||= 2 ↔ ♠ ☀ ▶ ⚡ !! ¶ § —☺ ♠<init>☺ ♥()V☺ ♦Code☺ ☀LineNumberTable☺ ♦main—([Ljava/lang/String;)V☺`
 - Hello, World! elloPrinter.java♀ ⚡♀ ↑ ↓☺



Organize your work

- ❑ Your ‘source code’ is stored in .java files
- ❑ Create one folder per program
 - Can be many .java files
- ❑ Be sure you know where your IDE stores your files!
- ❑ Backup your work!



Backup your work to a Flash Drive, external hard drive, or network drive that is backed up nightly.





1.5 Analyzing your First Program

```
1 public class HelloPrinter
2 {
3     public static void main(String[] args)
4     {
5         System.out.println("Hello, World!");
6     }
7 }
```



1: Declares a ‘class’ called HelloPrinter

-- Every Java program has *one or more classes*.

3: Declares a method called ‘main’

-- Every Java application has exactly one ‘main’ method

-- Entry point where the program starts

5: Method `System.out.println` outputs ‘Hello, World!’

-- A statement must end with a semicolon (;)



- ❑ Every source file contain at most one public class, and the name of the public class must match the name of the file containing the class
 - Ex. The class *HelloPrinter* must be contained in a file named *HelloPrinter.java*



Syntax 1.1: The Java Program

- ❑ Every application has the same basic layout
 - Add your 'code' inside the **main** method

Every Java program contains a main method with this header.

The statements inside the main method are executed when the program runs.

Be sure to match the opening and closing braces.

```
public class HelloPrinter
```

```
{
```

```
public static void main(String[] args)
```

```
{
```

```
System.out.println("Hello, World!");
```

method

argument

Each statement ends in a semicolon.
 See page 14.

Replace this statement when you write your own programs.

Every program contains at least one class. Choose a class name that describes the program action.



Calling Java Library methods

5

```
System.out.println("Hello, World!");
```

- ❑ Line 5 shows how to ‘call’ a ‘method’ from the Java API: `System.out.println`
 - Code that somebody else wrote for you!
 - Notice the dots (periods)
 - Parenthesis surround the arguments that you ‘pass’ to a method `("Hello, World!");`
 - We are passing a String “Hello World”
 - Note the double quotes which denote a String inside
 - You can also print numerical values
 - `System.out.println(3 + 4);`



Getting to know `println`

- ❑ The `println` method prints a string or a number and then starts a new line.

```
System.out.println("Hello");  
System.out.println("World!");
```

```
Hello  
World!
```

- ❑ `println` has a ‘cousin’ method named `print` that does not print a new line.

```
System.out.print("00");  
System.out.println(3+4);
```

```
007
```

A method is called by specifying the method and its arguments



Common Error 1.1



❑ Omitting Semicolons

- In Java, every statement must end in a semicolon. Forgetting to type a semicolon is a common error. It confuses the compiler, because the compiler uses the semicolon to find where one statement ends and the next one starts. For example, the compiler sees this:

```
System.out.println("Hello")  
System.out.println("World!");
```

- As this:

```
System.out.println("Hello") System.out.println("World!");
```

- It doesn't understand this statement, because it does not expect the word `System` following the closing parenthesis after `Hello`.



1.6 Errors

□ The Two Categories of Errors:

1) Compile-time Errors

- **Syntax Errors**

- Spelling, Capitalization, punctuation
- Ordering of statements, matching of braces/parenthesis...

- No .class file is generated by the compiler

- Correct first error listed, then compile again

2) Run-time Errors

- **Logic Errors**

- Program runs, but produces unintended results

- Program may ‘crash’





Syntax Errors

```
1 public class HelloPrinter
2 {
3     public static void main(String[] args)
4     {
5         System.out.println("Hello, World!");
6     }
7 }
```



❑ What happens if you

- Misspell a word: `System.ou.println`
- Don't Capitalize a word: `system.out.println`
- Leave out a word: `void`
- Forget a Semicolon after: `("Hello, World!")`
- Don't match a curly brace? Remove line 6

❑ Try it to see what error messages are generated



Logic Errors

```
1 public class HelloPrinter
2 {
3     public static void main(String[] args)
4     {
5         System.out.println("Hello, World!");
6     }
7 }
```



❑ What happens if you

- Divide by Zero `System.out.println(1/0);`
- Mis-spell output `("Hello, Word!")`
- Forget to output Remove line 5

❑ **Programs will compile and run**

- The output may not be as expected



1.7 Problem Solving: Algorithm Design

- ❑ Algorithms are simply plans
 - Detailed plans that describe the steps to solve a specific problem
- ❑ You already know quite a few
 - Calculate the area of a circle
 - Find the length of the hypotenuse of a triangle
- ❑ Some problems are more complex and require more steps
 - Calculate PI to 100 decimal places
 - Calculate the trajectory of a missile



Text Problem to Algorithm

□ Given the problem:

- You put \$10,000 into a bank account that earns 5 percent interest per year. How many years does it take for the account balance to be double the original?

□ How would you solve it?

- Manual method
 - Make a table
 - Add lines until done
- Use a spreadsheet!
 - Write a formula
 - Per line, based on line above

year	balance
0	10000
1	$10000.00 \times 1.05 = 10500.00$
2	$10500.00 \times 1.05 = 11025.00$
3	$11025.00 \times 1.05 = 11576.25$
4	$11576.25 \times 1.05 = 12155.06$



Text Problem to Algorithm Steps

- You put \$10,000 into a bank account that earns 5 percent interest per year. How many years does it take for the account balance to be double the original?

□ Break it into steps

Start with a year value of 0 and a balance of \$10,000

Repeat the following while the balance is less than \$20,000

Add 1 to the year value

Multiply the balance by 1.05
(5% increase)

year	balance
0	10000

year	balance
0	10000
1	10500

14	19799.32
15	20789.28

Report the final year value as the answer



Text Problem to Pseudocode

❑ Pseudocode

- Half-way between natural language and a programming language

❑ Modified Steps

- **Set** the year value of 0
- **Set** the balance to \$10,000
- **While** the balance is less than \$20,000
 - Add 1 to the year value
 - Multiply the balance by 1.05
- Report the final year value as the answer

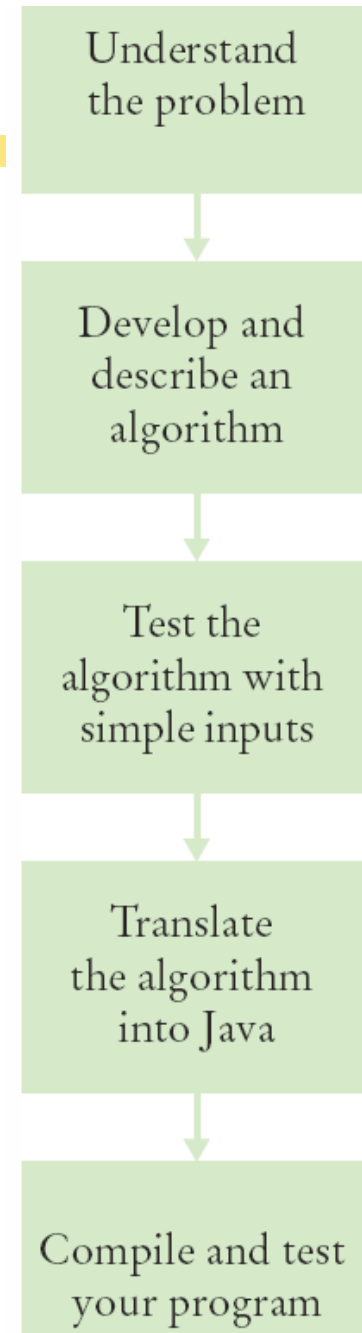
Pseudocode is an informal description of a sequence of steps for solving a problem

❑ This can be translated into Java fairly easily



Algorithm Defined

- ❑ An algorithm describes a sequence of steps that is:
 - Unambiguous
 - Do not require ‘assumptions’
 - Uses precise instructions
 - Executable
 - Can be carried out in practice
 - Terminating
 - Will eventually come to an end





Steps: Algorithm to Pseudocode

You have the choice of buying two cars. One is more fuel efficient than the other, but also more expensive. You know the price and fuel efficiency (in miles per gallon, mpg) of both cars. You plan to keep the car for ten years. Which car is the better deal?

1. Determine the inputs and outputs
From the problem statement:
price1, price2, mpg1, mpg2...
2. Break down the problem into smaller tasks
'Calculate total cost' for each car
3. Describe each subtask as pseudocode
total cost = purchase price + operating cost
4. Test your pseudocode with example input



Summary: Computer Basics

- ❑ Computers execute very basic instructions in rapid succession.
- ❑ A computer program is a sequence of instructions and decisions.
- ❑ Programming is the act of designing and implementing computer programs.
- ❑ The central processing unit (CPU) performs program control and data processing.
- ❑ Storage devices include memory and secondary storage.



Summary: Java

- ❑ Java was originally designed for programming consumer devices, but it was first successfully used to write Internet applets.
- ❑ Java was designed to be safe and portable, benefiting both Internet users and students.
- ❑ Java programs are distributed as instructions for a virtual machine, making them platform-independent.
- ❑ Java has a very large set of libraries. Focus on learning those parts of libraries that you need for your programming projects.



Summary: Java

- ❑ Set aside some time to become familiar with the programming environment that you will use for your class work.
- ❑ An editor is a program for entering and modifying text, such as a Java program.
- ❑ Java is case sensitive. You must be careful about distinguishing between upper and lowercase letters.
- ❑ The Java compiler translates source code into class files that contain instructions for the Java virtual machine.



Summary: Java

- ❑ Classes are the fundamental building blocks of Java programs.
- ❑ Every Java application contains a class with a main method. When the application starts, the instructions in the main method are executed.
- ❑ Each class contains declarations of methods. Each method contains a sequence of instructions.
- ❑ A method is called by specifying the method and its parameters.
- ❑ A string is a sequence of characters enclosed in quotation marks.



Summary: Errors and Pseudocode

- ❑ A compile-time error is a violation of the programming language rules that is detected by the compiler.
- ❑ A run-time error causes a program to take an action that the programmer did not intend.
- ❑ Pseudocode is an informal description of a sequence of steps for solving a problem.
- ❑ An algorithm for solving a problem is a sequence of steps that is unambiguous, executable, and terminating.