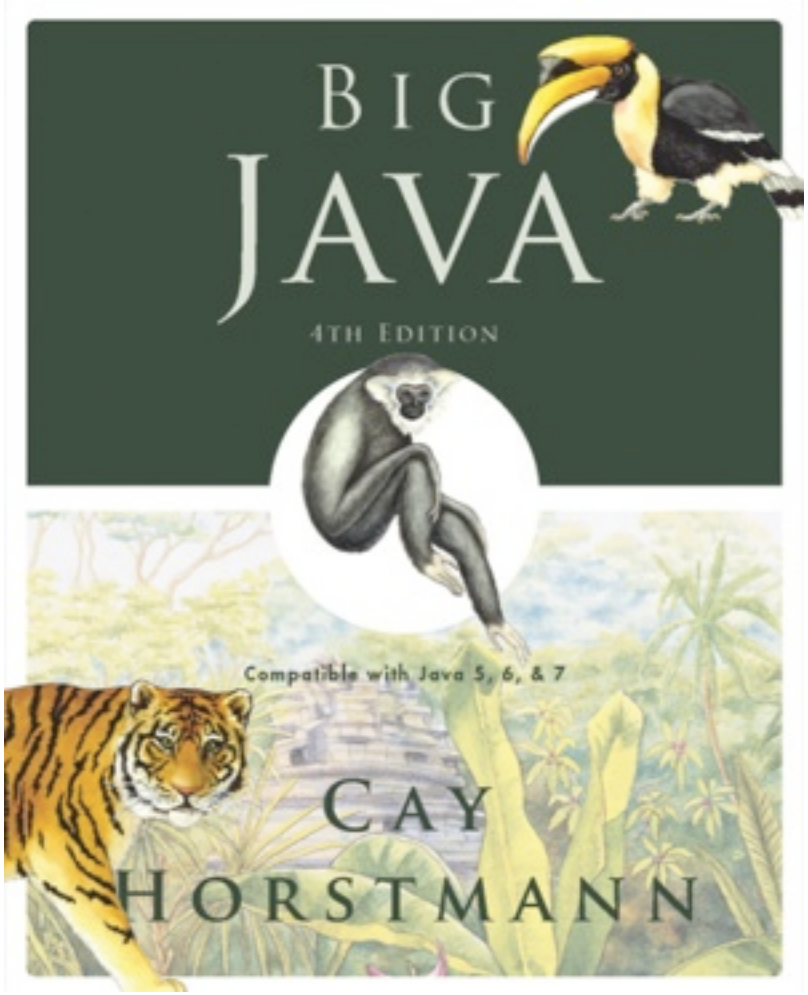


Chapter 1 – Introduction

Chapter Goals

- To understand the activity of programming
- To learn about the architecture of computers
- To learn about machine code and high level programming languages
- To become familiar with your computing environment and your compiler
- To compile and run your first Java program
- To recognize syntax and logic errors
- To write pseudocode for simple algorithms

What Is Programming?

- Computers are programmed to perform tasks
- Different tasks = different programs
- Program
  - *Sequence of basic operations executed in succession*
  - *Contains instruction sequences for all tasks it can execute*
- Sophisticated programs require teams of highly skilled programmers and other professionals

Self Check 1.1

What is required to play a music CD on a computer?

Self Check 1.2

Why is a CD player less flexible than a computer?

Self Check 1.3

Can a computer program develop the initiative to execute tasks in a better way than its programmers envisioned?

The Anatomy of a Computer

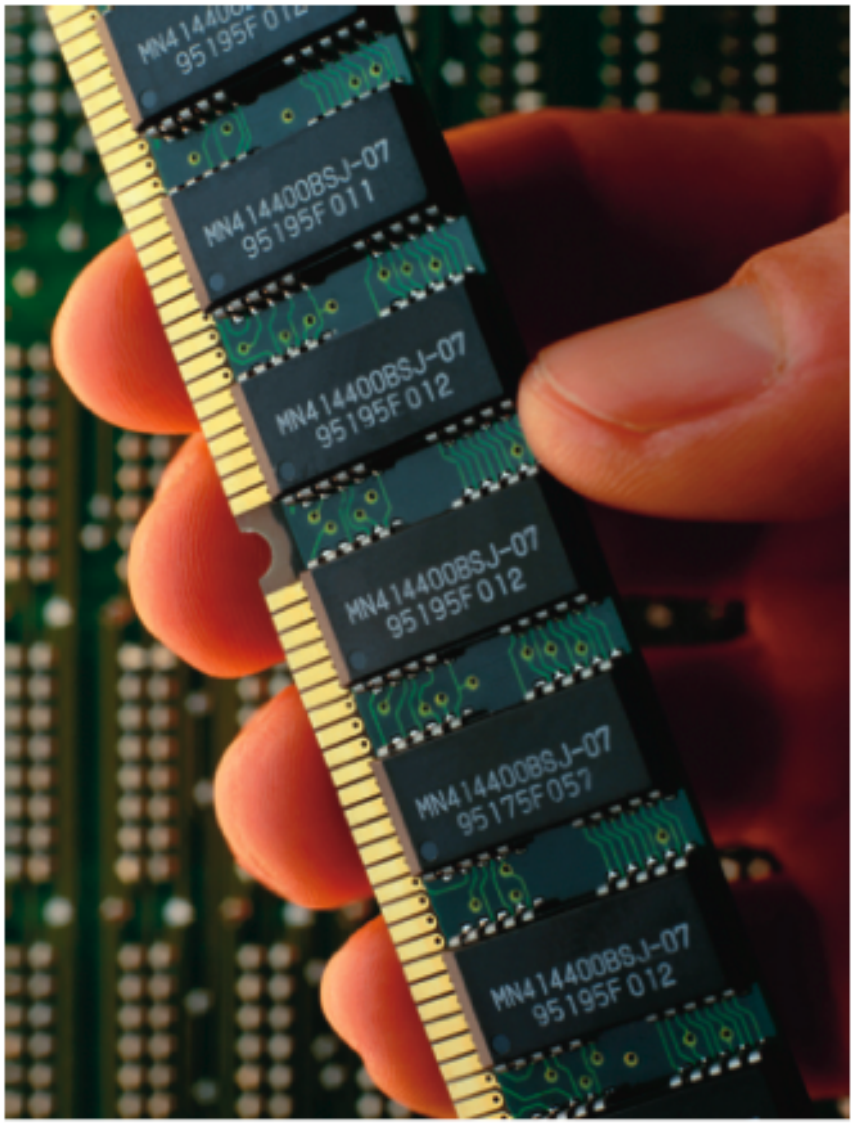
- Central processing unit
  - *Chip*
  - *Transistors*
- Storage
  - *Primary storage: Random-access memory (RAM)*
  - *Secondary storage: e.g. hard disk*
  - *Removable storage devices: e.g.: floppy disks, tapes, CDs*
- Peripherals
- Executes very simple instructions
- Executes instructions very rapidly
- General purpose device

Central Processing Unit

Figure 1
Central Processing Unit

A Memory Module with Memory Chips

Figure 2
A Memory Module with Memory Chips



A Hard Disk

Figure 3 A Hard Disk

A Motherboard

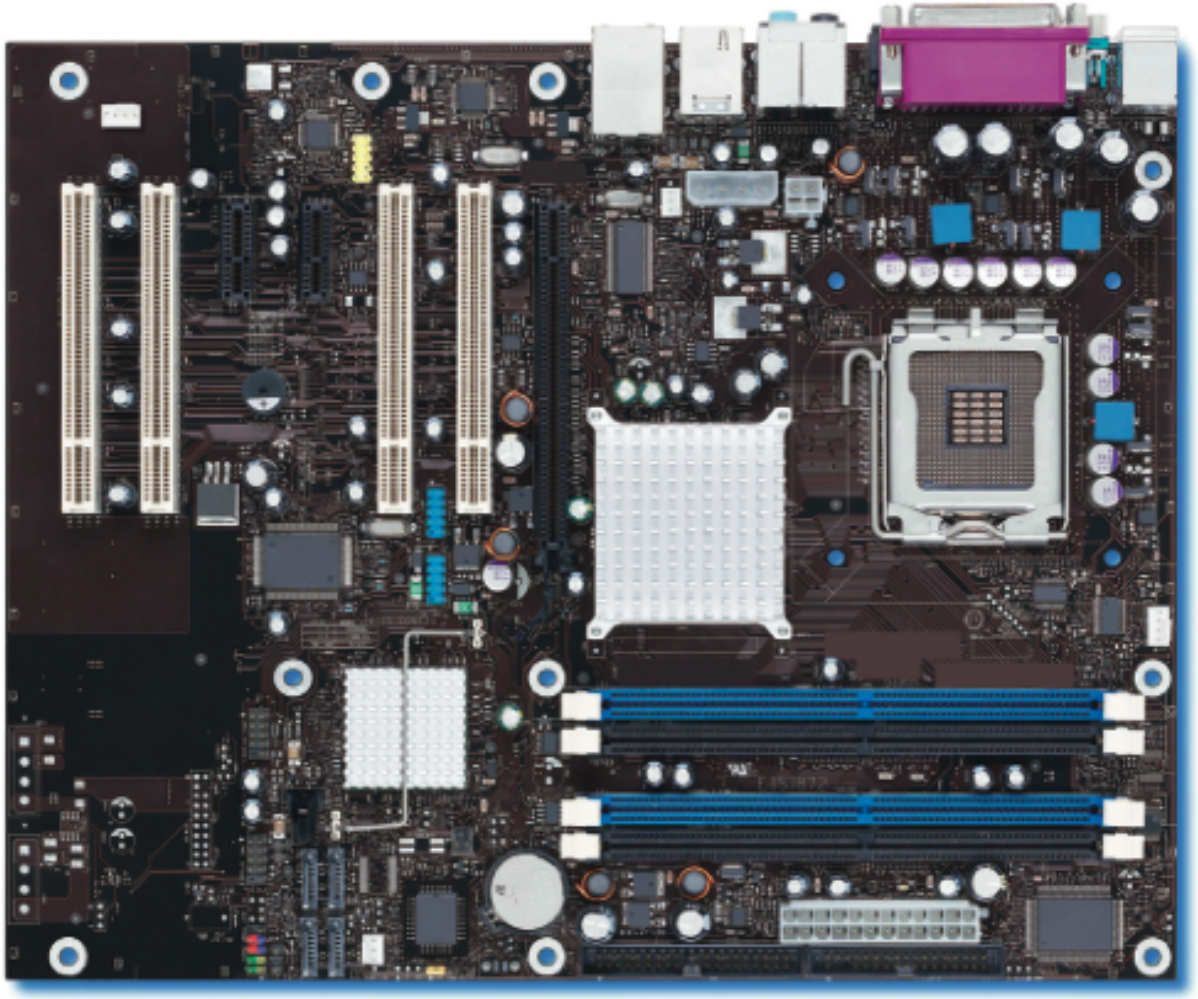


Figure 4 A Motherboard

Schematic Diagram of a Computer

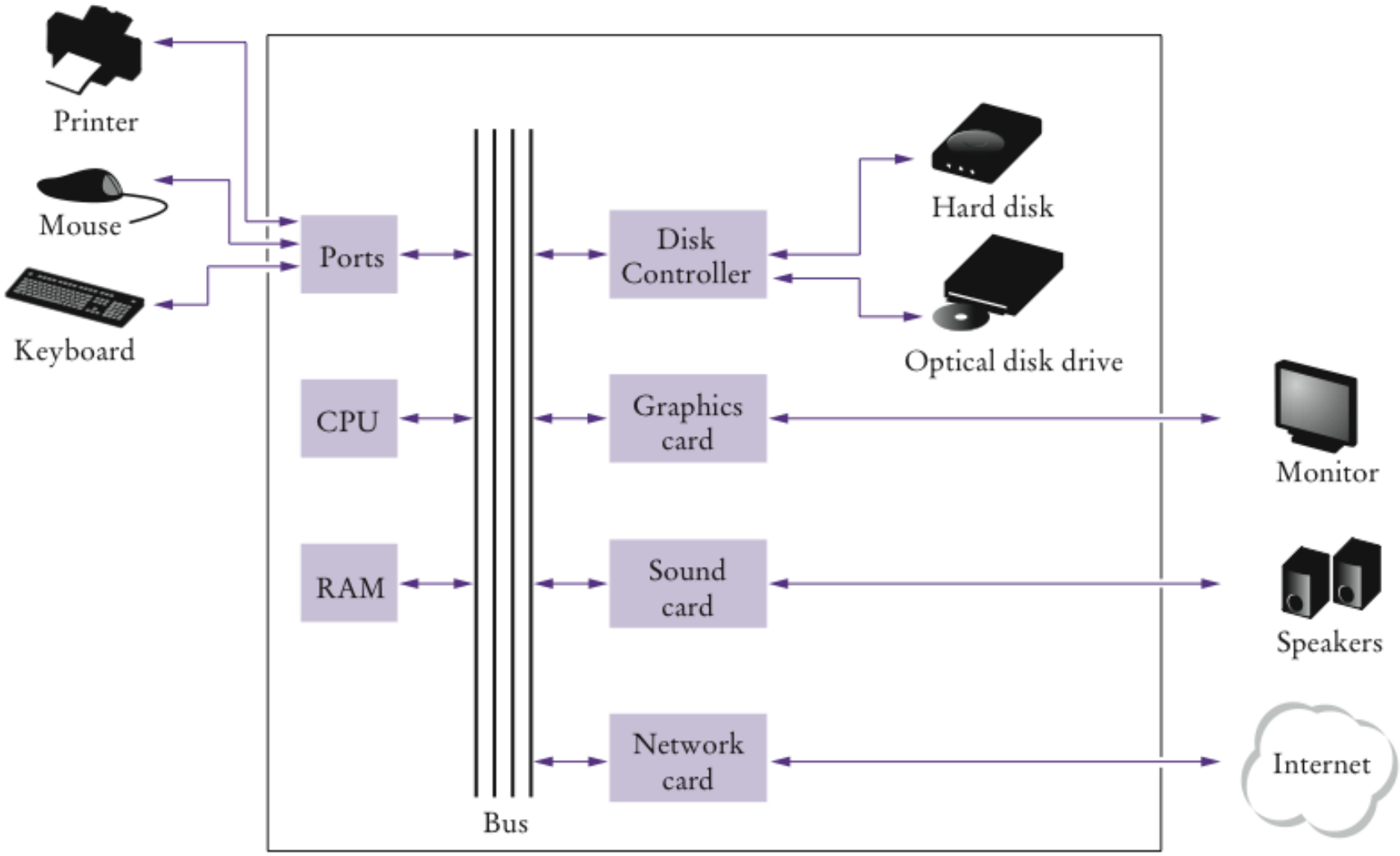
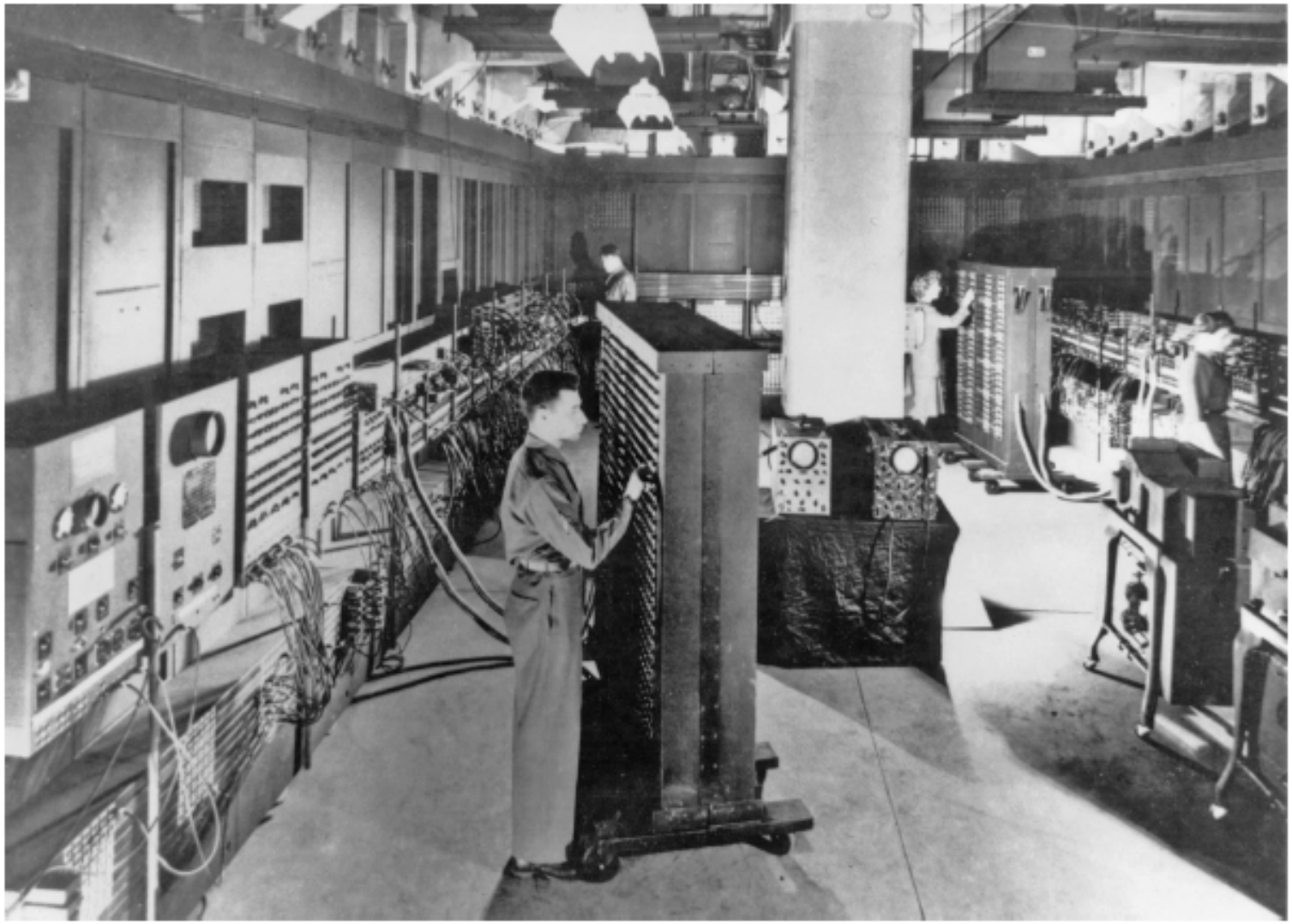


Figure 5 Schematic Diagram of a Computer

The ENIAC



The ENIAC

Self Check 1.4

Where is a program stored when it is not currently running?

Self Check 1.5

Which part of the computer carries out arithmetic operations, such as addition and multiplication?

Machine Code

- Generally, machine code depends on the CPU type
- However, the instruction set of the Java virtual machine (JVM) can be executed on many types of CPU
- Java Virtual Machine (JVM) – a typical sequence of machine instructions is:

1. *Load the contents of memory location 40.*
2. *Load the value 100.*
3. *If the first value is greater than the second value, continue with the instruction that is stored in memory location 240.*

Machine Code

- Machine instructions are encoded as numbers:

```
21 40
16 100
163 240
```

- Compiler translates high-level language to machine code

Self Check 1.6

What is the code for the Java virtual machine instruction “Load the contents of memory location 100”?

Self Check 1.7

Does a person who uses a computer for office work ever run a compiler?

The Java Programming Language

- Simple
- Safe
- Platform-independent ("write once, run anywhere")
- Rich library (packages)
- Designed for the internet

Applet on a Web Page

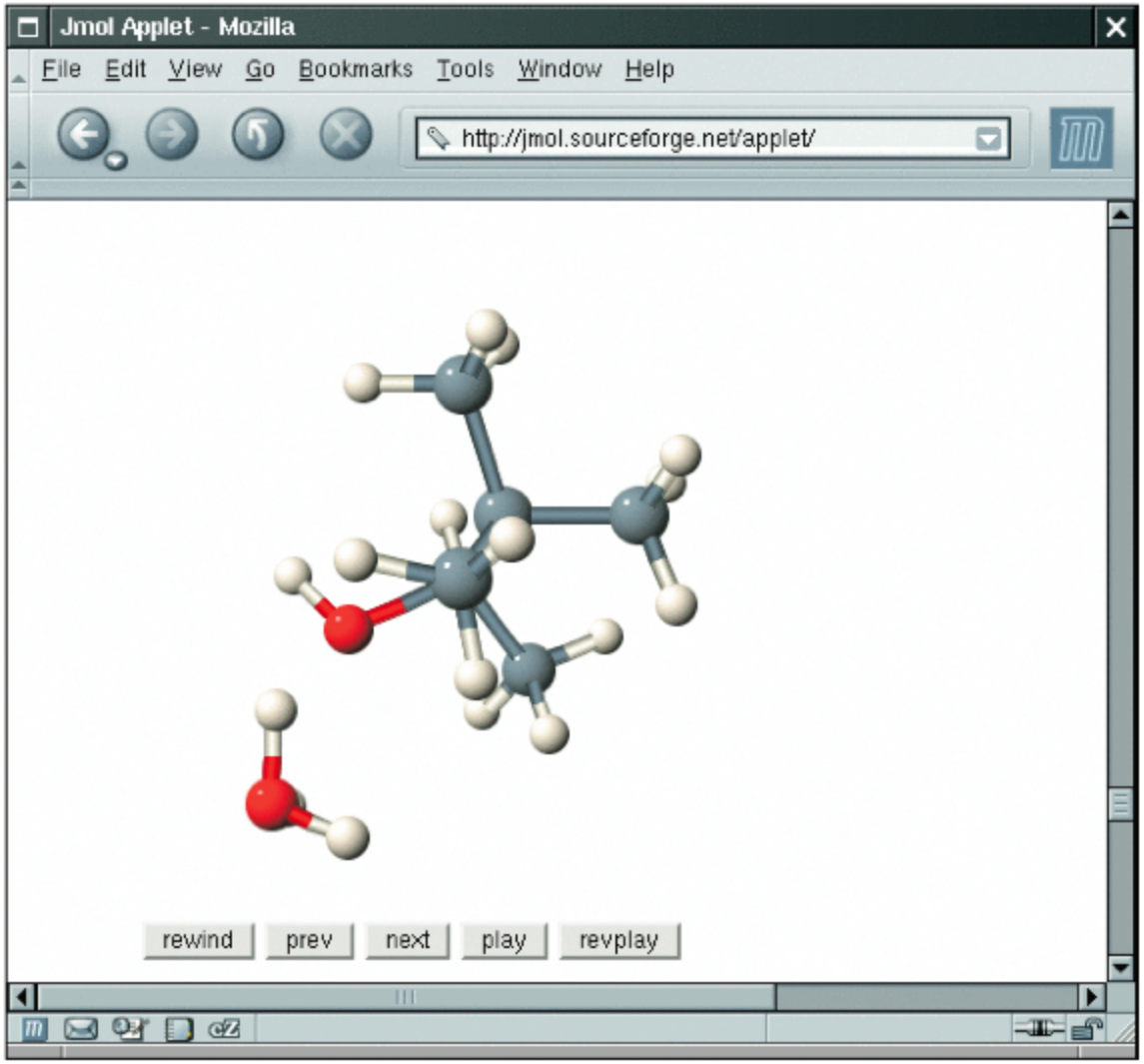


Figure 6 An Applet for Visualizing Molecules Running in a Browser (*http://jmol.sourceforge.net/applet/*)

Java Versions

Version	Year	Important New Features
1.0	1996	
1.1	1997	Inner classes
1.2	1998	Swing, Collections
1.3	2000	Performance enhancements
1.4	2002	Assertions, XML
5	2004	Generic classes, enhanced for loop, auto-boxing, enumerations
6	2006	Library improvements
7	2010	Small language changes and library improvements

Self Check 1.8

What are the two most important benefits of the Java language?

Self Check 1.9

How long does it take to learn the entire Java library?

ch01/hello/HelloPrinter.java

```
1  public class HelloPrinter
2  {
3     public static void main(String[] args)
4     {
5        // Display a greeting in the console window
6
7        System.out.println("Hello, World!");
8     }
9  }
```

Program Run:

```
Hello, World!
```

The Structure of a Simple Program: Class Declaration

- Classes are the fundamental building blocks of Java programs:

```
public class HelloPrinter
```

  starts a new **class**

- Every source file can contain at most one public class
- The name of the public class must match the name of the file containing the class:
  - *Class* `HelloPrinter` *must be contained in a file named* `HelloPrinter.java`

The Structure of a Simple Program: `main` Method

- Every Java application contains a class with a main method
  - *When the application starts, the instructions in the main method are executed*

- ```
  public static void main(String[] args)
  {
     . . .
  }
  ```

  declares a `main` method
  ```

The Structure of a Simple Program: Comments

- The first line inside the main method is a comment:

```
// Display a greeting in the console window
```

- Compiler ignores any text enclosed between `//` and end of the line
- Use comments to help human readers understand your program

The Structure of a Simple Program: Statements

- The body of the main method contains statements inside the curly brackets (`{}`)
- Each statement ends in a semicolon (`;`)
- Statements are executed one by one
- Our method has a single statement:

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

which prints a line of text:

```
Hello, World
```

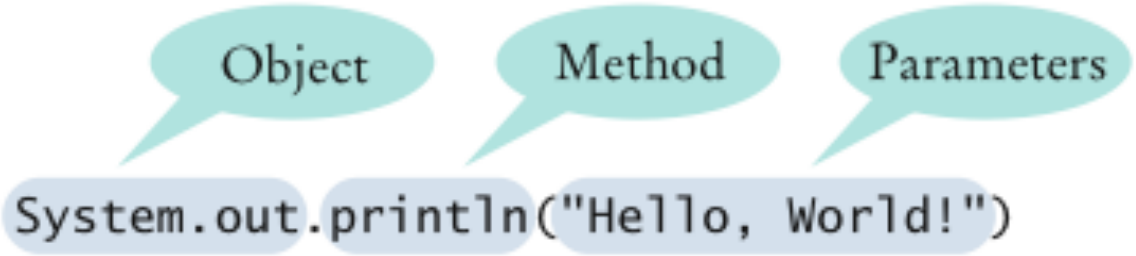
The Structure of a Simple Program: Method Call

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

  is a *method call*

- A method call requires:
  1. *The object that you want to use (in this case, `System.out`)*
  2. *The name of the method you want to use (in this case, `println`)*
  3. ***Parameters** enclosed in parentheses (`()`) containing any other information the method needs (in this case, `"Hello, World!"`)*

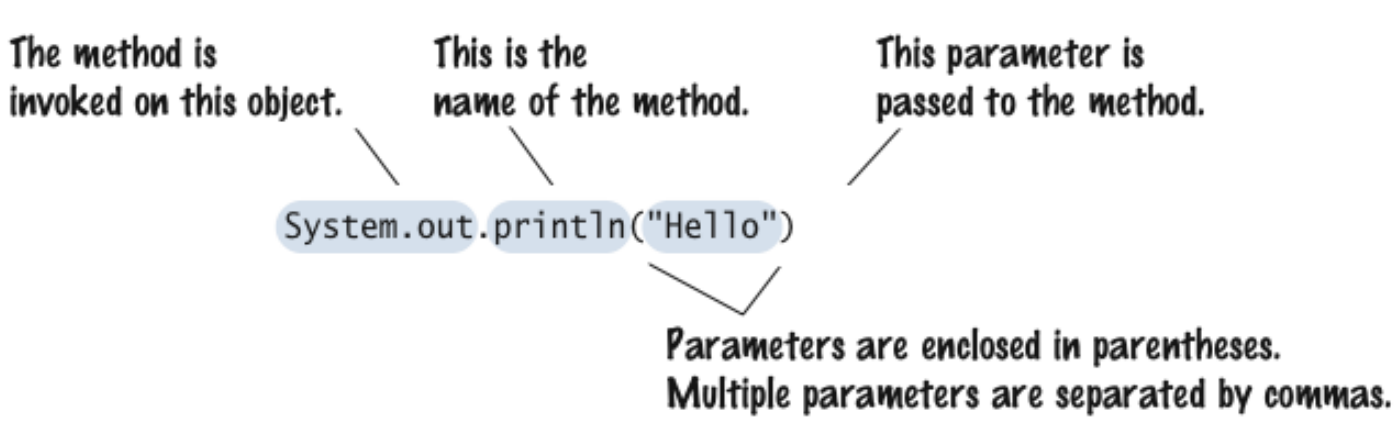
Figure 7
Calling a Method



Syntax 1.1 Method Call

Syntax *object.methodName(parameters)*

Example



The Structure of a Simple Program: Strings

- **String:** a sequence of characters enclosed in double quotation marks:

```
"Hello, World!"
```

Self Check 1.10

How would you modify the `HelloPrinter` program to print the words `"Hello,"` and `"World!"` on two lines?

Self Check 1.11

Would the program continue to work if you omitted the line starting with `//`?

Self Check 1.12

What does the following set of statements print?

```
System.out.print("My lucky number is");
System.out.println(3 + 4 + 5);
```

Editing a Java Program

- Use an editor to enter and modify the program text
- Java is case-sensitive
  - *Be careful to distinguish between upper- and lowercase letters*
- Lay out your programs so that they are easy to read

Compiling and Running a Java Program

- The Java compiler translates source code into class files that contain instructions for the Java virtual machine
- A class file has extension `.class`
- The compiler does not produce a class file if it has found errors in your program
- The Java virtual machine loads instructions from the program's class file, starts the program, and loads the necessary library files as they are required

`HelloPrinter` in a Console Window

Figure 8 Running the `HelloPrinter` Program in a Console Window

`HelloPrinter` in an IDE

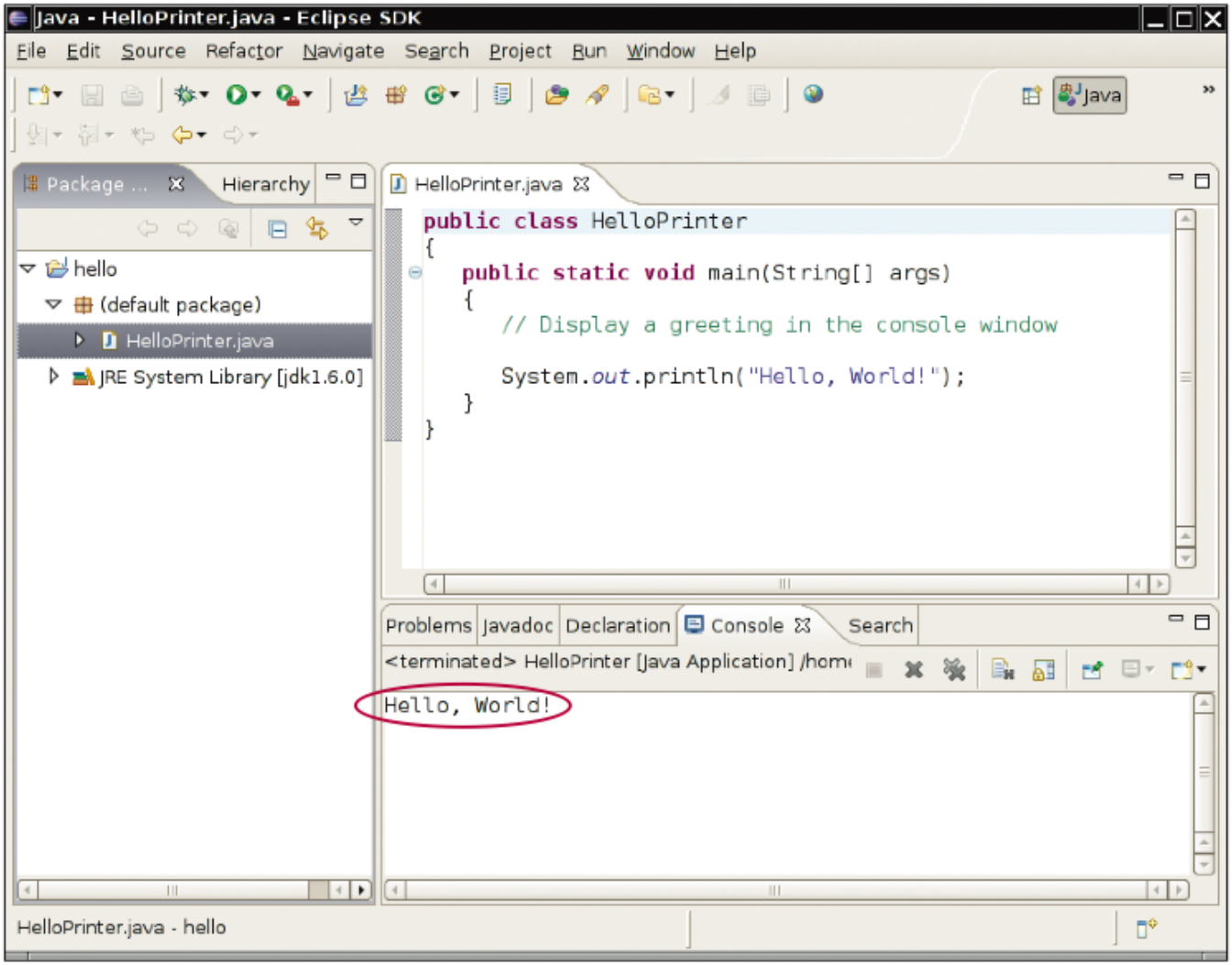


Figure 9 Running the `HelloPrinter` Program in an Integrated Development Environment

From Source Code to Running Program

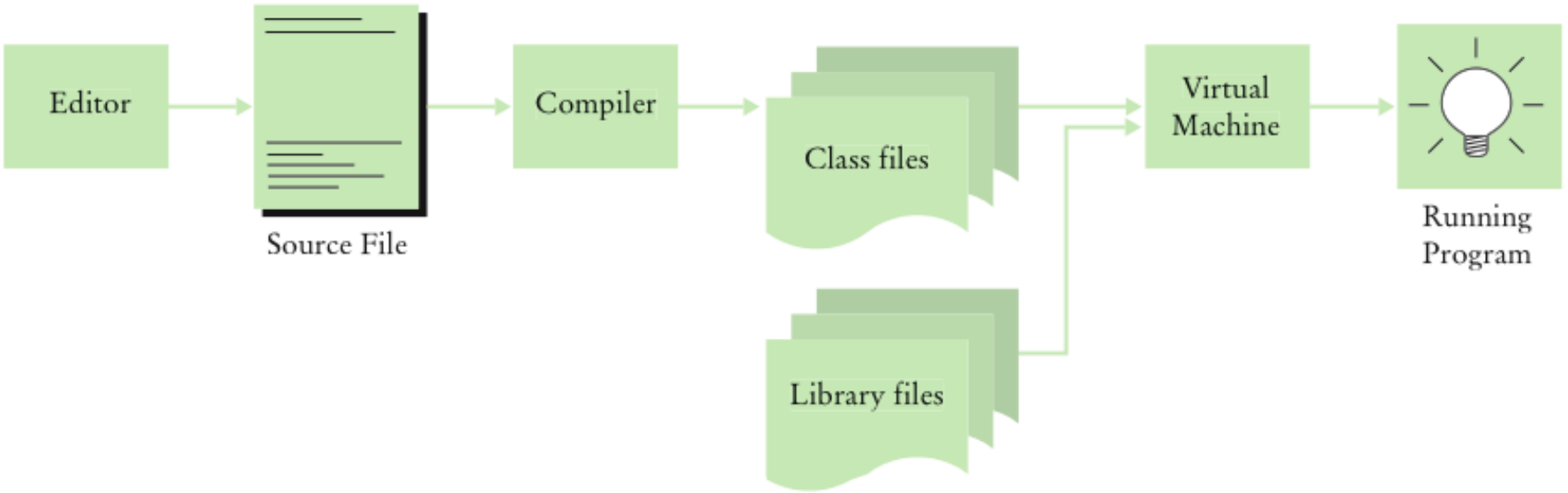


Figure 10 From Source Code to Running Program

Self Check 1.13

Can you use a word processor for writing Java programs?

Self Check 1.14

What do you expect to see when you load a class file into your text editor?

Errors

- **Compile-time error:** A violation of the programming language rules that is detected by the compiler
  - *Example*:

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

  - *Syntax error*
- **Run-time error:** Causes the program to take an action that the programmer did not intend
  - *Examples:*

```
System.out.println("Hello, Word!");
System.out.println(1/0);
```

  - *Logic error*

Error Management Strategy

- Learn about common errors and how to avoid them
- Use defensive programming strategies to minimize the likelihood and impact of errors
- Apply testing and debugging strategies to flush out those errors that remain

Self Check 1.15

Suppose you omit the `//` characters from the `HelloPrinter.java` program but not the remainder of the comment. Will you get a compile-time error or a run-time error?

Self Check 1.16

When you used your computer, you may have experienced a program that “crashed” (quit spontaneously) or “hung” (failed to respond to your input). Is that behavior a compile-time error or a run-time error?

Self Check 1.17

Why can't you test a program for run-time errors when it has compiler errors?

Algorithms

- **Algorithm:** A sequence of steps that is:
  - *unambiguous*
  - *executable*
  - *terminating*
- Algorithm for deciding which car to buy, based on total costs:

```
For each car, compute the total cost as follows:
   annual fuel consumed = annual miles driven / fuel efficiency
   annual fuel cost = price per gallon x annual fuel consumed
   operating cost = 10 x annual fuel cost
   total cost = purchase price + operating cost
If total cost1 < total cost2
   Choose car1
Else
   Choose car2
```

Pseudocode

- **Pseudocode:** An informal description of an algorithm:
  - *Describe how a value is set or changed:*

    total cost = purchase price + operating cost

  - *Describe decisions and repetitions:*

    For each car
    operating cost = 10 x annual fuel cost
    total cost = purchase price + operating cost

    Use indentation to indicate which statements should be selected or repeated

  - *Indicate results:*

    Choose car1

Program Development Process

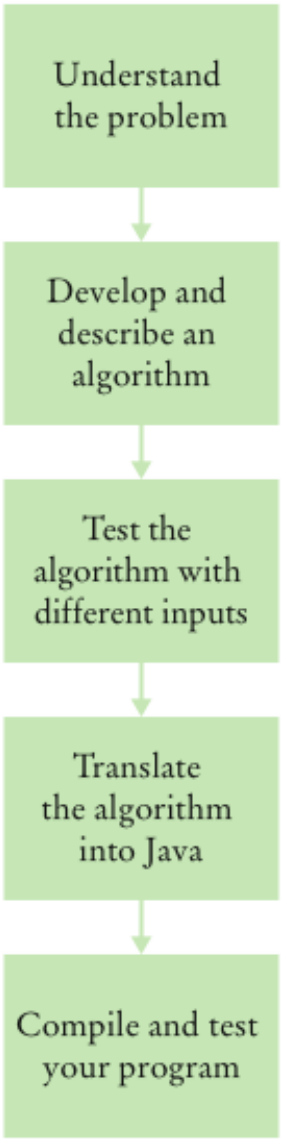


Figure 12
The Program Development Process

Self Check 1.18

Investment 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?

Algorithm:

Start with a year value of 0 and a balance of $10,000.
Repeat the following steps while the balance is less than $20,000.
  Add 1 to the year value.
  Multiply the balance value by 1.05 (a 5 percent increase).

Suppose the interest rate was 20 percent. How long would it take for the investment to double?

Self Check 1.19

Suppose your cell phone carrier charges you $29.95 for up to 300 minutes of calls, and $0.45 for each additional minute, plus 12.5 percent taxes and fees. Give an algorithm to compute the monthly charge for a given number of minutes.