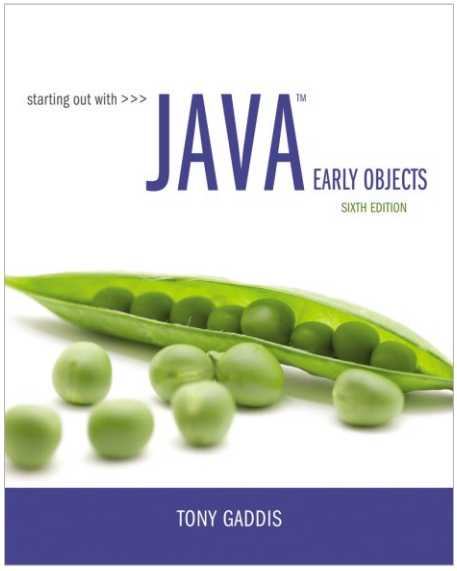
CHAPTER 1

Introduction to Computers and Programming



Topics

Chapter 1 discusses the following main topics:

- Introduction
- Why Program?
- Computer Systems: Hardware and Software
- Programming Languages
- What Is a Program Made Of?
- The Programming Process
- Object-Oriented Programming

Introduction

- **This book teaches programming using Java**
  - Java is a powerful language that runs on practically every type of computer
  - Java can be used to create large applications, small programs, mobile applications, and code that powers a website.
- **Before plunging right into learning Java, however, this chapter will:**
  - Review fundamentals of computer hardware and software
  - Take a broad look at programming in general

Why Program?

- **Computers are tools that can be programmed to perform many functions, such as:**
  - spreadsheets
  - databases
  - word processing
  - games
  - etc.
- **Computers are versatile because they can be programmed.**
- **Computer Programmers implement programs that perform these functions.**

Why Program?

Aspects of a computer program that must be designed:

- The logical flow of the instructions
- The mathematical procedures
- The layout of the programming statements
- The appearance of the screens
- The way information is presented to the user
- The program's "user friendliness"
- Manuals, help systems, and/or other forms of written documentation.

Why Program?

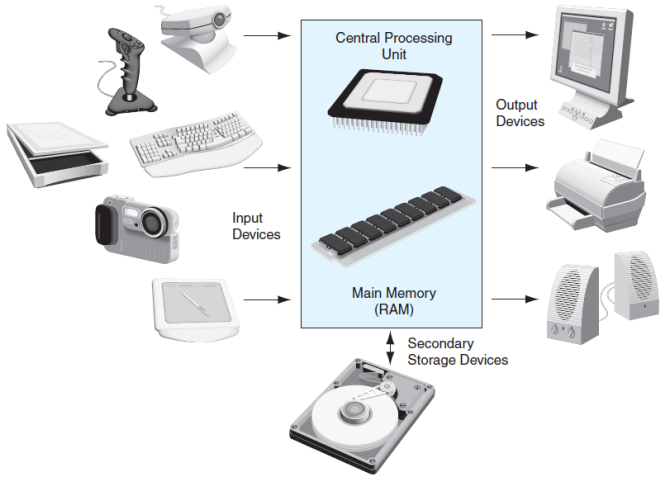
- **Programs must be analytically correct as well.**
- **Programs rarely work the first time they are programmed.**
- **Programmers must perform the following on a continual basis:**
  - analyze,
  - experiment,
  - correct, and
  - redesign.
- **Programming languages have strict rules, known as *syntax*, that must be carefully followed.**

Computer Systems: Hardware

- **Computer hardware components are the physical pieces of the computer.**
- **The major hardware components of a computer are:**
  - The central processing unit (CPU)
  - Main memory
  - Secondary storage devices
  - Input and Output devices

Computer Systems: Hardware

Figure 1-1 The organization of a computer system

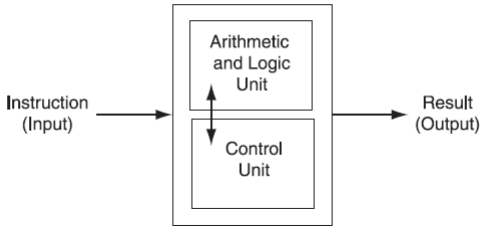


The CPU

- **At the heart of the computer is the *central processing unit*, or CPU**
  - The CPU's job is to fetch instructions, follow instructions, and produce some resulting data
- **Internally, the CPU consists of two parts:**
  - ***Control Unit***
    - Coordinates all of the computer's operations
  - ***Arithmetic and Logic Unit* (ALU)**
    - Performs mathematical operations

The CPU

Figure 1-2 The organization of the CPU



The CPU

- A program is a sequence of instructions stored in the computer's memory
- The CPU performs the ***fetch / decode / execute cycle*** in order to process program information.
  - **Fetch** - the CPU's control unit fetches, from main memory, the next instruction in the sequence of program instructions.
  - **Decode** - the instruction is encoded in the form of a number. The control unit decodes the instruction and generates an electronic signal.
  - **Execute** - the signal is routed to the appropriate component of the computer (such as the ALU, a disk drive, or some other device). The signal causes the component to perform an operation.

Main Memory

- **Commonly known as *random-access memory (RAM)***
- **RAM contains:**
  - currently running programs
  - data used by those programs.
- **RAM is divided into units called *bytes*.**
- **A byte consists of eight *bits* that may be either on or off.**

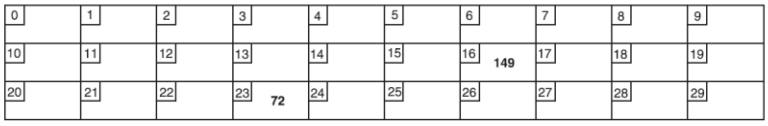
Main Memory

- **A bit is either on or off:**
  - 1 = on
  - 0 = off
- **The bits form a pattern that represents a character or a number.**
- **Each byte in memory is assigned a unique number known as an *address*.**
- **RAM is *volatile*, which means that when the computer is turned off, the contents of RAM are erased.**

Main Memory

- **For example, a series of bytes with their addresses:**
  - The number 149 is stored in the byte at address 16
  - The number 72 is stored in the byte at address 23

Figure 1-3 Memory bytes and their addresses



Secondary Storage

- **Secondary storage devices are capable of storing information for longer periods of time (*non-volatile*).**
- **Common Secondary Storage devices:**
  - Disk drive
  - Solid-state drive
  - USB drive
  - Optical devices
  - CD / DVD

Input Devices

- **Input is any data the computer collects from the outside world.**
- **That data comes from devices known as *input devices*.**
- **Common input devices:**
  - Keyboard
  - Mouse
  - Touchscreen
  - Digital Camera
- Disk drives, optical drives, and USB drives can also be considered input devices when data is read from those devices

Output Devices

- **Output is any data the computer sends to the outside world.**
- **That data is displayed on devices known as *output devices*.**
- **Common output devices:**
  - Monitors
  - Printers
- Disk drives, USB drives, and CD/DVD recorders can also be considered output devices when data is sent to those devices to be saved

Computer Systems: Software

- **Software refers to the programs that run on a computer.**
- **There are two classifications of software:**
  - Operating Systems
  - Application Software

Operating Systems

- An *operating system* is a set of programs that manages the computer's hardware devices and controls their processes.
- Most all modern operating systems are multitasking
  - Capable of running multiple programs at once
  - Through *time sharing*
    - a multitasking system divides the allocation of hardware resources and the attention of the CPU among all the executing programs.

Operating Systems

- **Multitasking Operating Systems include:**
  - UNIX
  - Linux
  - Modern versions of:
    - Windows
    - Mac OS

Application Software

- ***Application software*** **refers to programs that make the computer useful to the user.**
- **Application software provides a more specialized type of environment for the user to work in.**
- **Common application software:**
  - Spreadsheets
  - Word processors
  - Accounting software
  - Tax software
  - Games

What is a Program?

- **A program is a set of instructions a computer follows in order to perform a task.**
- **A programming language is a special language used to write computer programs.**
- **A computer program is a set of instructions that enable the computer to solve a problem or perform a task.**
- **Collectively, these instructions form an *algorithm***

What is a Program?

- **An *algorithm* is a set of well defined steps for performing a task or solving a problem.**
- **The steps in an algorithm are performed sequentially.**
- **A computer needs the algorithm to be written in *machine language*.**
- **Machine language is written using *binary numbers*.**
- **The binary numbering system (base 2) only has two digits (0 and 1).**

What is a Program?

- **The binary numbers are encoded as a machine language.**
- **Example of a machine language instruction:**

  1011010000000101
- **Each CPU has its own machine language.**
  - If you wrote a program for computer A, and then wanted to run it on computer B, with a different CPU, you would have to rewrite the program in computer B's machine language.

What is a Program?

- Programming languages were invented to ease the task of programming
  - Use words instead of numbers
- A program can be written in a programming language and translated into machine language
  - Programmers use software to perform this translation.

Table 1-1 Programming languages

Language	Description
BASIC	Beginners All-purpose Symbolic Instruction Code is a general-purpose, procedural programming language. It was originally designed to be simple enough for beginners to learn.
FORTRAN	FORmula TRANslator is a procedural language designed for programming complex mathematical algorithms.
COBOL	Common Business-Oriented Language is a procedural language designed for business applications.
Pascal	Pascal is a structured, general-purpose, procedural language designed primarily for teaching programming.
C	C is a structured, general-purpose, procedural language developed at Bell Laboratories.
C++	Based on the C language, C++ offers object-oriented features not found in C. C++ was also invented at Bell Laboratories.
C#	Pronounced "C sharp." It is a language invented by Microsoft for developing applications based on the Microsoft .NET platform.
Java	Java is an object-oriented language invented at Sun Microsystems and is now owned by Oracle. It may be used to develop stand-alone applications that operate on a single computer, applications that run over the Internet from a Web server, and applets that run in a Web browser.
JavaScript	JavaScript is a programming language that can be used in a Web site to perform simple operations. Despite its name, JavaScript is not related to Java.
Perl	A general-purpose programming language that is widely used on Internet servers.
PHP	A programming language used primarily for developing Web server applications and dynamic Web pages.
Python	Python is an object-oriented programming language that is used in both business and academia. Many popular Web sites have features that are developed in Python.
Ruby	Ruby is a simple but powerful object-oriented programming language. It can be used for a variety of purposes, from small utility programs to large Web applications.
Visual Basic	Visual Basic is a Microsoft programming language and software development environment that allows programmers to create Windows-based applications quickly.

A History of Java

- 1991 - Green Team started by Sun Microsystems.
- *7 Handheld controller for multiple entertainment systems.
- There was a need for a programming language that would run on various devices.
- Java (first named Oak) was developed for this purpose.

A History of Java

- Java enabled web browser (*HotJava*) demonstrated at 1995 Sun World conference.
- Java incorporated into Netscape shortly after.
- Java is “cross platform”, meaning that it can run on various computer operating systems.

What is a Program Made Of?

Table 1-2 The common elements of a programming language

Language Element	Description
Key Words	These are words that have a special meaning in the programming language. They may be used for their intended purpose only. Key words are also known as *reserved words*.
Operators	Operators are symbols or words that perform operations on one or more operands. An *operand* is usually an item of data, such as a number.
Punctuation	Most programming languages require the use of punctuation characters. These characters serve specific purposes, such as marking the beginning or ending of a statement, or separating items in a list.
Programmer-Defined Names	Unlike key words, which are part of the programming language, these are words or names that are defined by the programmer. They are used to identify storage locations in memory and parts of the program that are created by the programmer. Programmer-defined names are often called *identifiers*.
Syntax	These are rules that must be followed when writing a program. Syntax dictates how key words and operators may be used, and where punctuation symbols must appear.

What is a Program Made Of?

Code Listing 1-1 `Payroll.java`

```
1  public class Payroll
2  {
3     public static void main(String[] args)
4     {
5        int hours = 40;
6        double grossPay, payRate = 25.0;
7
8        grossPay = hours * payRate;
9        System.out.println("Your gross pay is $" + grossPay);
10    }
11 }
```

What is a Program Made Of?

- **Key words in the sample program are:**
  - `public`
  - `class`
  - `static`
  - `void`
- **Key words are lower case (Java is a case sensitive language).**
- **Key words cannot be used as a programmer-defined identifier.**

What is a Program Made Of?

- **Semi-colons are used to end Java statements**
  - However, not all lines of a Java program end a statement.
- **Part of learning Java is to learn where to properly use the punctuation.**

What is a Program Made Of?

- **There are differences between lines and statements when discussing source code.**

```
System.out.println(
    message);
```

- **This is one Java statement written using two lines. Do you see the difference?**
- **A statement is a complete Java instruction that causes the computer to perform an action.**

What is a Program Made Of?

- **Data in a Java program is stored in memory.**
- **Variable names represent a location in memory.**
- **Variables are created by the programmer who assigns it a programmer-defined identifier.**

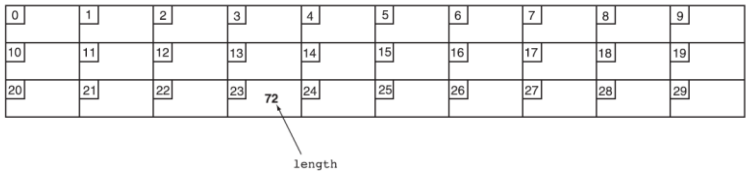
```
int hours = 40;
```

- The variable `hours` is created as an integer (more on this later) and assigned the value of `40`.

What is a Program Made Of?

- **Variables are simply a name given to represent a place in memory.**

Figure 1-4 A variable name represents a location in memory



The Compiler and the Java Virtual Machine

- **A programmer writes Java programming statements for a program.**
- **These statements are known as *source code*.**
- **A *text editor* is used to edit and save a Java *source code file.***
- **Source code files have a `.java` file extension.**
- **A *compiler* is a program that translates source code into an executable form.**

The Compiler and the Java Virtual Machine

- A compiler is run using a source code file as input.
- Syntax errors that may be in the program will be discovered during compilation.
- *Syntax errors* are mistakes that the programmer has made that violate the rules of the programming language.
- The compiler creates another file that holds the translated instructions.

The Compiler and the Java Virtual Machine

- Most compilers translate source code into *executable* files containing *machine code*.
- The Java compiler translates a Java source file into a file that contains *byte code* instructions.
- Byte code instructions are the machine language of the *Java Virtual Machine (JVM)* and cannot be directly executed directly by the CPU.

The Compiler and the Java Virtual Machine

- **Byte code files end with the `.class` file extension.**
- **The JVM is a program that *emulates* a micro-processor.**
- **The JVM executes instructions as they are read.**
- **JVM is often called an *interpreter.***
- **Java is often referred to as an *interpreted language*.**

Figure 1-5 Program development process

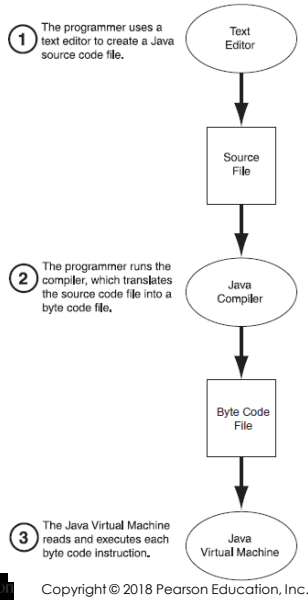
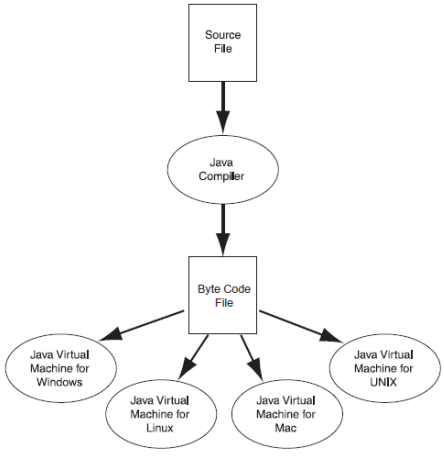


Figure 1-6 Java byte code may be run on any computer with a JVM



The Compiler and the Java Virtual Machine

- ***Portable*** **means that a program may be written on one type of computer and then run on a wide variety of computers, with little or no modification.**
- **Java byte code runs on the JVM and not on any particular CPU; therefore, compiled Java programs are highly portable.**
- **JVMs exist on many platforms:**
  - Windows
  - Mac
  - Linux
  - Unix
  - BSD
  - Etc.

The Compiler and the Java Virtual Machine

- **With most programming languages, portability is achieved by compiling a program for each CPU it will run on.**
- **Java provides an JVM for each platform so that programmers do not have to recompile for different platforms.**

Java Software Editions

- **The software you use to write Java programs is called the Java Development Kit, or JDK.**
- **There are different editions of the JDK:**
  - **Java SE** – *Standard Edition*
    - Provides essential tools for developing applications and applets
  - **Java EE** – *Enterprise Edition*
    - Provides tools for creating large business applications that employ servers and provide services over the Web
  - **Java ME** – *Micro Edition*
    - Provides a small, optimized runtime environment for consumer products such as cell phones, pagers, and appliances

Available for download at `http://java.oracle.com`

Compiling a Java Program

- **The Java compiler is a *command line* utility.**
- **The command to compile a program is:**

```
javac filename.java
```

- **`javac` is the Java compiler.**
- **The `.java` file extension must be used.**

Example: To compile a java source code file named `Payroll.java` you would use the command:

```
javac Payroll.java
```

Running a Java Program

- To run the Java program, you use the java command in the following form:

```
java ClassFilename
```

- *ClassFilename* is the name of the . *class* file that you wish to execute.
  - However, you do not type the . *class* extension.
- For example, to run the program that is stored in the *Payroll.class* file, you would enter the following command:

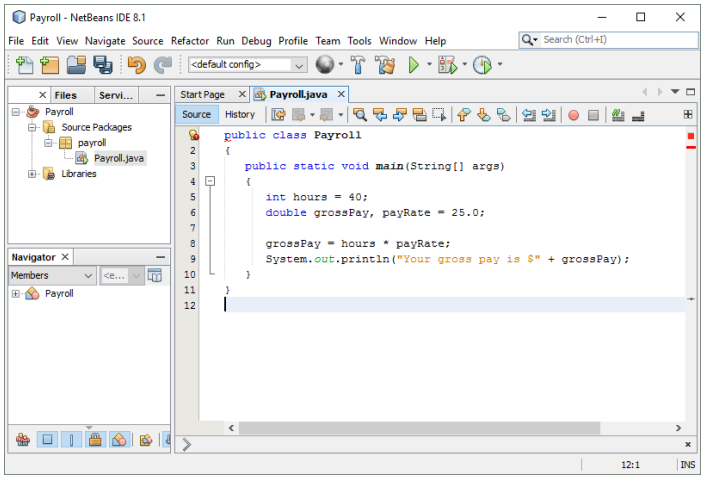
```
java Payroll
```

  - This command runs the Java interpreter (the JVM) and executes the program.

Integrated Development Environments

- In addition to the command prompt programs, there are also several Java integrated development environments (IDEs).
- These environments consist of a text editor, compiler, debugger, and other utilities integrated into a package with a single set of menus.
- A program is compiled and executed with a single click of a button, or by selecting a single item from a menu.

Integrated Development Environments



The Programming Process

1. **Clearly define what the program is to do.**
2. **Visualize the program running on the computer.**
3. **Use design tools to create a model of the program.**
4. **Check the model for logical errors.**
5. **Enter the code and compile it.**
6. **Correct any errors found during compilation.**
   Repeat Steps 5 and 6 as many times as necessary.
7. **Run the program with test data for input.**
8. **Correct any runtime errors found while running the program.**
   Repeat Steps 5 through 8 as many times as necessary.
9. **Validate the results of the program.**

Object-Oriented Programming

- Older programming languages were procedural.
- A *procedure* is a set of programming language statements that, together, perform a specific task.
- Procedures typically operate on data items that are separate from the procedures.
- In a procedural program, the data items are commonly passed from one procedure to another

Object-Oriented Programming

- In procedural programming, procedures are developed to operate on the program's data.
- Data in the program tends to be global to the entire program.
- Data formats might change and thus, the procedures that operate on that data must change.

Object-Oriented Programming

- **Object-oriented programming is centered on creating objects rather than procedures.**
- **Objects are a melding of data and procedures that manipulate that data.**
- **Data in an object are known as *attributes*.**
- **Procedures in an object are known as *methods*.**

Object-Oriented Programming

Figure 1-10 An object contains data and procedures

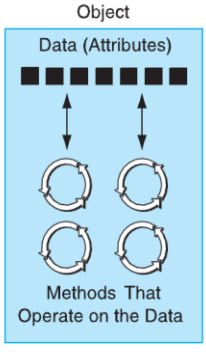
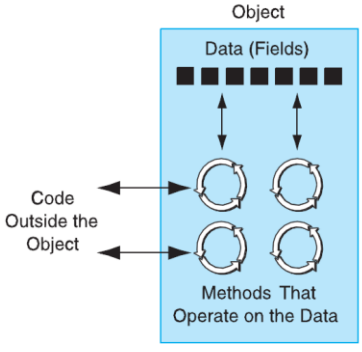


Figure 1-11 Code outside the object interacts with the object's methods



Object-Oriented Programming

- **Data hiding is important for several reasons.**
  - It protects of attributes from accidental corruption by outside objects.
  - It hides the details of how an object works, so the programmer can concentrate on using it.
  - It allows the maintainer of the object to have the ability to modify the internal functioning of the object without "breaking" someone else's code.

Object-Oriented Programming

- **Object-Oriented Programming (OOP) has encouraged component reusability.**
- **A component is a software object contains data and methods that represents a specific concept or service.**
- **Components typically are not stand-alone programs.**
- **Components can be used by programs that need the component's service.**
- **Reuse of code promotes the rapid development of larger software projects.**

An Everyday Example of an Object

- **Data attributes: define the state of an object**
  - Example: clock object would have `second`, `minute`, and `hour` data attributes
- **Public methods: allow external code to manipulate the object**
  - Example: `setTime()`, `setAlarmTime()`
- **Private methods: used for object's inner workings**

Classes and Objects

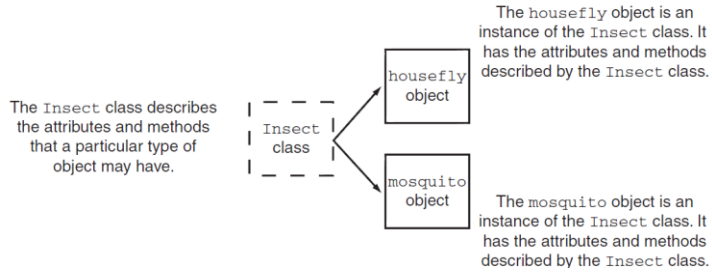
- **Components are objects.**
- **The programmer determines the attributes and methods needed, and then creates a class.**
- **A *class* is a collection of programming statements that define the required object**
- **A class as a "blueprint" that objects may be created from.**
- **An object is the realization (instantiation) of a class in memory**

Classes and Objects

- **Classes can be used to instantiate as many objects as are needed.**
- **Each object that is created from a class is called an *instance* of the class.**
- **A program is simply a collection of objects that interact with each other to accomplish a goal.**

Classes and Objects

Figure 1-12 The `housefly` and `mosquito` objects are instances of the `Insect` class

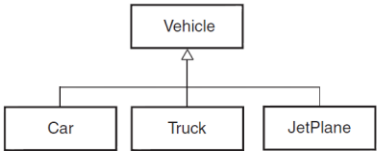


Inheritance

- ***Inheritance*** **is the ability of one class to extend the capabilities of another.**
- **Consider the class `Car`.**
- **A `Car` is a specialized form of the `Vehicle` class.**
- **So, is said that the `Vehicle` class is the base or parent class of the `Car` class.**
- **The `Car` class is the derived or child class of the `Vehicle` class.**

Inheritance

Figure 1-13 An example of inheritance



Software Engineering

- **Encompasses the whole process of crafting computer software.**
- **Software engineers perform several tasks in the development of complex software projects.**
  - Designing
  - Writing
  - Testing
  - Debugging
  - Documenting
  - Modifying
  - Maintaining

Software Engineering

- **Software engineers also use special software designed for testing programs.**
- **Most commercial software applications are large and complex.**
- **Usually a team of programmers, not a single individual, develops them.**
- **Program requirements are thoroughly analyzed and divided into subtasks that are handled by**
  - Individual teams
  - Individuals within a team