Introduction to Computers and Java

Chapter 1
Objectives

• Overview of computer hardware and software
• Introduce program design and object-oriented programming
• Overview of the Java programming language
Hardware and Software

• Computer systems consist of hardware and software.
  ▪ Hardware includes the tangible parts of computer systems.
  ▪ Software includes programs - sets of instructions for the computer to follow.

• Familiarity with hardware basics helps us understand software.
Hardware and Memory

• Most modern computers have similar components including
  ▪ Input devices (keyboard, mouse, etc.)
  ▪ Output devices (display screen, printer, etc.)
  ▪ A processor
  ▪ Two kinds of memory (main memory and auxiliary memory).
The Processor

• Also called the *CPU* (central processing unit) or the *chip* (e.g. Pentium processor)
• The processor **processes** a program’s instructions.
• It can process only very simple instructions.
Memory

• Memory holds
  ▪ programs
  ▪ data for the computer to process
  ▪ the results of intermediate processing.

• Two kinds of memory
  ▪ main memory
  ▪ auxiliary memory
Main memory

• Working memory used to store
  ▪ The current program
  ▪ The data the program is using
  ▪ The results of intermediate calculations

• Usually measured in low gigabytes (e.g. 8 GBs of RAM)
  ▪ RAM is short for *random access memory*
  ▪ A *byte* is a quantity of memory
Auxiliary Memory

• Also called *secondary memory*
• Disk drives, CDs, DVDs, flash drives, etc.
• More or less permanent (nonvolatile)
• Usually measured in high gigabytes (e.g. 500 gigabyte hard drive)
Bits, Bytes, and Addresses

- A *bit* is a digit with a value of either 0 or 1.
- A *byte* consists of 8 bits.
- Each byte in main memory resides at a numbered location called its *address*. 
Main Memory

- Figure 1.1
Storing Data

• Data of all kinds (numbers, letters, strings of characters, audio, video, even programs) are encoded and stored using 1s and 0s.

• When more than a single byte is needed, several adjacent bytes are used.
  ▪ The address of the first byte is the address of the unit of bytes.
Files

• Large groups of bytes in auxiliary memory are called *files*.
• Files have names.
• Files are organized into groups called *directories* or *folders*.
• Java programs are stored in files.
• Programs files are copied from auxiliary memory to main memory in order to be run.
0s and 1s

• Machines with only 2 stable states are easy to make, but programming using only 0s and 1s is difficult.

• Fortunately, the conversion of numbers, letters, strings of characters, audio, video, and programs is done automatically.
Programs

• A *program* is a set of instructions for a computer to follow.
• We use programs almost daily (email, word processors, video games, bank ATMs, etc.).
• Following the instructions is called *running* or *executing* the program.
Running a Program

• Figure 1.2

• Sometimes the computer and the program are considered to be one unit.
  - Programmers typically find this view to be more convenient.
The Operating System

• The *operating system* is a supervisory program that oversees the operation of the computer.

• The operating system retrieves and starts program for you.

• Well-known operating systems including: Microsoft Windows, Apple’s Mac OS, Linux, and UNIX.
Programming Languages

• *High-level languages* are relatively easy to use
  ▪ Java, C#, C++, Visual Basic, Python, Ruby.

• Unfortunately, computer hardware does not understand high-level languages.
  ▪ Therefore, a high-level language program must be translated into a *low-level language*. 
Compilers

• A compiler translates a program from a high-level language to a low-level language the computer can run.

• You compile a program by running the compiler on the high-level-language version of the program called the source program.

• Compilers produce machine- or assembly-language programs called object programs.
Compilers

• Most high-level languages need a different compiler for each type of computer and for each operating system.
• Most compilers are very large programs that are expensive to produce.
Java Byte-Code

• The Java compiler does not translate a Java program into assembly language or machine language for a particular computer.
• Instead, it translates a Java program into byte-code.
  ▪ Byte-code is the machine language for a hypothetical computer (or interpreter) called the Java Virtual Machine.
Portability

• After compiling a Java program into byte-code, that byte-code can be used on any computer with a byte-code interpreter and without a need to recompile.

• Byte-code is portable and can be sent over the Internet and used anywhere in the world.

• This makes Java suitable for many applications including Internet and mobile.
Class Loader

• A Java program typically consists of several pieces called *classes*.
• Each class may have a separate author and each is compiled (translated into byte-code) separately.
• A *class loader* (called a *linker* in other programming languages) automatically connects the classes together.
Compiling and Running a Program

• Figure 1.3