If you can't explain it to a six year old, you don't understand it yourself.
A. Einstein
Focus of the Course

• Object-Oriented Software Development
  – problem solving
  – program design, implementation, and testing
  – object-oriented concepts
    • classes
    • objects
    • encapsulation
    • inheritance
    • polymorphism
  – graphical user interfaces
  – the Java programming language
Introduction

• We start with the fundamentals of computer processing

• Chapter 1 focuses on:
  – components of a computer
  – how computers store and manipulate information
  – computer networks
  – the Internet and the World Wide Web
  – programming and programming languages
  – an introduction to Java
  – an overview of object-oriented concepts
Outline

Computer Processing
Hardware Components
Networks
The Java Programming Language
Program Development
Object-Oriented Programming
Hardware and Software

• Hardware
  – the physical, tangible parts of a computer
  – keyboard, monitor, disks, wires, chips, etc.

• Software
  – programs and data
  – a *program* is a series of instructions

• A computer requires both hardware and software

• Each is essentially useless without the other
CPU and Main Memory

Central Processing Unit

Chip that executes program commands

Main Memory

Primary storage area for programs and data that are in active use

Synonymous with RAM

Copyright © 2012 Pearson Education, Inc.
- Each cell of this matrix stores a bit 0/1
- A cell is made of transistor/capacitor
- Capacitor holds an electric charge
Input / Output Devices

I/O devices facilitate user interaction

Central Processing Unit

Main Memory

Monitor screen
Keyboard
Mouse
Touch screen

Copyright © 2012 Pearson Education, Inc.
Secondary Memory Devices

Information is moved between main and secondary memory as needed.

- Secondary memory devices provide long-term storage.

Central Processing Unit

Main Memory

Hard Disk

USB Flash Drive

Copyright © 2012 Pearson Education, Inc.
Software Categories

• **Operating System**
  – controls all machine activities
  – provides the user interface to the computer
  – manages resources such as the CPU and memory
  – Windows, Mac OS, Unix, Linux,

• **Application program**
  – generic term for any other kind of software
  – word processors, missile control systems, games

• Most operating systems and application programs have a *graphical user interface* (GUI)
Analog vs. Digital

- There are two basic ways to store and manage data:

  - **Analog**
    - continuous, in direct proportion to the data represented
    - music on a record album - a needle rides on ridges in the grooves that are directly proportional to the voltages sent to the speaker

  - **Digital**
    - the information is broken down into pieces, and each piece is represented separately
    - *sampling* – record discrete values of the analog representation
    - music on a compact disc - the disc stores numbers representing specific voltage levels sampled at specific times
Analog Information

Sound wave

Analog signal of the sound wave
Sampling

Information can be lost between samples

Analog signal

Sampling process

Sampled values: 12, 11, 39, 40, 7, 14, 47

Copyright © 2012 Pearson Education, Inc.
Data Transmission

• **Analog transmission of analogue data**
  – The air pressure variations (analog data) are converted (microphone) into an electrical analog signal in which either the instantaneous voltage or current is directly proportional to the instantaneous air pressure and then transmitted (e.g., traditional phone or radio)

• **Analog transmission of digital data**
  – The electric analogue signal is digitized, or converted to a digital signal, through an Analog-to-Digital converter and then modulated into analogue signals and transmitted (e.g., digital phones as GSM).
Quick Check

• What are the two primary functions of an operating system?
Quick Check

• What are the two primary functions of an operating system?

The operating system provides a user interface and efficiently coordinates the use of resources such as main memory and CPU.
Digital Information

- Computers store all information digitally:
  - numbers
  - text
  - graphics and images
  - audio
  - video
  - program instructions

- In some way, all information is *digitized* - broken down into pieces and represented as numbers
Representing Text Digitally

• For example, every character is stored as a number, including spaces, digits, and punctuation

• Corresponding upper and lower case letters are separate characters

Hi, Heather.
Binary Numbers

- Once information has been digitized, it is represented and stored in memory using the *binary number system*

- A single binary digit (0 or 1) is called a *bit (unit symbol is b)*

- *A byte is 8 bits (unit symbol is B)*

- Devices that store and move information are cheaper and more reliable if they have to represent only two states

- A single bit can represent two possible states, like a light bulb that is either on (1) or off (0)

- Permutations of bits are used to store values
## Bit Permutations

<table>
<thead>
<tr>
<th>1 bit</th>
<th>2 bits</th>
<th>3 bits</th>
<th>4 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>000</td>
<td>0000</td>
</tr>
<tr>
<td>1</td>
<td>01</td>
<td>001</td>
<td>0001</td>
</tr>
<tr>
<td>10</td>
<td>010</td>
<td>0010</td>
<td>1001</td>
</tr>
<tr>
<td>11</td>
<td>011</td>
<td>0011</td>
<td>1011</td>
</tr>
<tr>
<td>100</td>
<td>0100</td>
<td>1100</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>0101</td>
<td>1101</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>0110</td>
<td>1110</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>0111</td>
<td>1111</td>
<td></td>
</tr>
</tbody>
</table>

Each additional bit doubles the number of possible permutations.

Copyright © 2012 Pearson Education, Inc.
Bit Permutations

- Each permutation can represent a particular item
- There are $2^N$ permutations of N bits
- Therefore, N bits are needed to represent $2^N$ unique items

How many items can be represented by

<table>
<thead>
<tr>
<th>Bits</th>
<th>2^n</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$2^1$</td>
<td>2 items</td>
</tr>
<tr>
<td>2</td>
<td>$2^2$</td>
<td>4 items</td>
</tr>
<tr>
<td>3</td>
<td>$2^3$</td>
<td>8 items</td>
</tr>
<tr>
<td>4</td>
<td>$2^4$</td>
<td>16 items</td>
</tr>
<tr>
<td>5</td>
<td>$2^5$</td>
<td>32 items</td>
</tr>
</tbody>
</table>

Copyright © 2012 Pearson Education, Inc.
Quick Check

How many bits would you need to represent each of the 50 United States using a unique permutation of bits?
How many bits would you need to represent each of the 50 United States using a unique permutation of bits?

Five bits wouldn't be enough, because \(2^5 = 32\).

**Six bits** would give us 64 permutations, and some wouldn't be used.
Place value

- In **decimal numbers** we use 10 digits (0-9) – base 10
- Each digit used in a number has a *place value*
- Ex: $8427 = 8 \times 10^3 + 4 \times 10^2 + 2 \times 10^1 + 7 \times 10^0 = 8000 + 400 + 20 + 7 = 8427$

- In **binary numbers** we use 2 digits (0-1)
- Each digit in a binary number has a place value but the *base* is now 2
- Ex: $1101 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 0 + 1 = 13$
- Using the above definition you convert from binary to decimal.
Hexadecimal numbers

• In **Hexadecimal** number the base is **16**
• The digits are: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
• **Ex:** 2A8E hex = \(2 \times 16^3 + A \times 16^2 + 8 \times 16^1 + E \times 16^0 = 2 \times 16^3 + 10 \times 16^2 + 8 \times 16^1 + 14 \times 16^0 = 2 \times 4096 + 10 \times 256 + 8 \times 16 + 14 \times 1 = 10,893 \) decimal

• **Exercise:**
  – 10 hex = ?? decimal
  – 10 hex = ?? binary

• Note that the first **16** hex number can be written with **4 bits** (F hex = 1111 binary)
## Table conversion

<table>
<thead>
<tr>
<th>Binary</th>
<th>Hex</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0001</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0010</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0011</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>0100</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>0101</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>0110</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>0111</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>1000</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>1001</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>1010</td>
<td>A</td>
<td>10</td>
</tr>
<tr>
<td>1011</td>
<td>B</td>
<td>11</td>
</tr>
<tr>
<td>1100</td>
<td>C</td>
<td>12</td>
</tr>
<tr>
<td>1101</td>
<td>D</td>
<td>13</td>
</tr>
<tr>
<td>1110</td>
<td>E</td>
<td>14</td>
</tr>
<tr>
<td>1111</td>
<td>F</td>
<td>15</td>
</tr>
</tbody>
</table>
Converting Decimal to Binary

180 decimal

180/2^7 = 180/128 = 1 and remainder 52
52/2^6 = 52/64 = 0 and remainder 52
52/2^5 = 52/32 = 1 and remainder 20
20/2^4 = 20/16 = 1 and remainder 4
4/2^3 = 4/8 = 0 and remainder 4
4/2^2 = 4/4 = 1 and remainder 0
0/2^1 = 0/2 = 0 and remainder 0
0/2^0 = 0/1 = 0 and remainder 0

= 1011 0100 binary
Converting Hexadecimal to Binary

- 16 decimal = $2^4$ decimal = 10000 binary
- Example:
  A1 hexadecimal = $10*16^1 + 1*16^0$ decimal
  = $(1*2^3 + 0*2^2 + 1*2^1 + 0*2^0 )*16 + (1*2^0)*1$ decimal
  = 1010 * 10000 + 1*1 binary
  = 1010 0000 + 1 = 1010 0001
- The trick is to convert each hex digit and the concatenate
- EX: 40C6 hex = 0100 0000 1100 0110
  - Because: 4 hex = $2^2$ decimal = 0100 ; 0 hex = 0000 ; C hex = 12 decimal = 1100 ; 6 hex = 6 decimal = 0110
Quick Check

We denote with $x_b$ a number $x$ in base $b$. 
*For instance $101_2 = 5_{10}$*

In which base $b$ we have the following:

$11_b = 7_{10}$

$10_b = 1010_2$

$11111_b = 5_{10}$
Quick Check

We denote with \( x_b \) a number \( x \) in base \( b \).

*For instance \( 101_2 = 5_{10} \)

In which base \( b \) we have the following:

\[
\begin{align*}
11_b &= 7_{10} & b=6 \\
10_b &= 1010_2 & b=10 \\
11111_b &= 5_{10} & b=1 \quad \smile \text{ It is not a good base for number }
\end{align*}
\]
Quick Check

• Imagine that we use the genetic code system made of 4 digits/nucleobases: A (Adenine), C (Cytosine), G (Guanine), T (Thymine)

• A is the first digit and T is the last digit – we have 4 digits

• Which numbers (decimal) correspond to the following DNA sequences?:
  – CTGG
  – TTACG
Quick Check

• Imagine that we use the genetic code system made of 4 digits/nucleobases: A (Adenine), C (Cytosine), G (Guanine), T (Thymine)

• Which numbers correspond to the following DNA sequences?:
  - CTGG = C*4³ + T*4² + G*4¹ + G*4⁰ = 1*64 + 3*16 + 2*4 + 2*1 = 122
  - TTACG = T*4⁴ + T*4³ + A*4² + C*4¹ + G*4⁰ = 3 * 256 + 3 * 64 + 0 * 16 + 1*4 + 2*1 = 966
Exercise

• Convert the following (decimal) numbers to binary format:
  – 129
  – 63
  – 9
  – 1025

• *Hint: they are close to powers of twos!*
Exercise

- Convert the following numbers to binary format:

  - $129 = 2^7 + 1 = 1000\ 0001_2$
  - $63 = 2^6 - 1 = 11\ 1111_2$
  - $9 = 2^3 + 1 = 1001$
  - $1025 = 2^{10} + 1 = 100\ 0000\ 0001$
A Computer Specification

• Consider the following specification for a personal computer:
  – 3.07 GHz Intel Core i7 processor
  – 8 GB RAM
  – 750 GB Hard Disk
  – 16x Blu-ray / HD DVD-ROM & 16x DVD+R DVD Burner
  – 17” Flat Screen Video Display with 1280 x 1024 resolution
  – Network Card
Main memory is divided into many memory locations (or *cells*).

Each memory cell has a numeric *address*, which uniquely identifies it.
Large values are stored in consecutive memory locations.

Each memory cell stores a set number of bits (usually 8 bits, or one byte).
### Storage Capacity

- Every memory device has a *storage capacity*, indicating the number of bytes it can hold.
- Capacities are expressed in various units:

<table>
<thead>
<tr>
<th>Unit</th>
<th>Symbol</th>
<th>Number of Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>kilobyte</td>
<td>KB</td>
<td>$2^{10} = 1024$</td>
</tr>
<tr>
<td>megabyte</td>
<td>MB</td>
<td>$2^{20}$ (over one million)</td>
</tr>
<tr>
<td>gigabyte</td>
<td>GB</td>
<td>$2^{30}$ (over one billion)</td>
</tr>
<tr>
<td>terabyte</td>
<td>TB</td>
<td>$2^{40}$ (over one trillion)</td>
</tr>
<tr>
<td>petabyte</td>
<td>PB</td>
<td>$2^{50}$ (a whole bunch)</td>
</tr>
</tbody>
</table>

Copyright © 2012 Pearson Education, Inc.
### Decimal vs. Binary

The software and computer industries often use binary estimates of the SI-prefixed quantities. Producers of computer storage devices prefer the SI values. This is the reason for specifying computer hard drive capacities of, say, 100 GB, when it contains 93 GiB of storage space.

<table>
<thead>
<tr>
<th>Prefixes for bit and byte multiples</th>
<th>Decimal</th>
<th>SI</th>
<th></th>
<th>Binary</th>
<th>IEC</th>
<th>JEDEC</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Value</strong></td>
<td><strong>k</strong></td>
<td>kilo</td>
<td></td>
<td><strong>Value</strong></td>
<td><strong>Ki</strong></td>
<td><strong>kibi</strong></td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td></td>
<td></td>
<td>1024</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$1000^2$</td>
<td><strong>M</strong></td>
<td>mega</td>
<td></td>
<td>$1024^2$</td>
<td><strong>Mi</strong></td>
<td>mebi</td>
</tr>
<tr>
<td>$1000^3$</td>
<td><strong>G</strong></td>
<td>giga</td>
<td></td>
<td>$1024^3$</td>
<td><strong>Gi</strong></td>
<td>gibi</td>
</tr>
<tr>
<td>$1000^4$</td>
<td><strong>T</strong></td>
<td>tera</td>
<td></td>
<td>$1024^4$</td>
<td><strong>Ti</strong></td>
<td>tebi</td>
</tr>
<tr>
<td>$1000^5$</td>
<td><strong>P</strong></td>
<td>peta</td>
<td></td>
<td>$1024^5$</td>
<td><strong>Pi</strong></td>
<td>pebi</td>
</tr>
<tr>
<td>$1000^6$</td>
<td><strong>E</strong></td>
<td>exa</td>
<td></td>
<td>$1024^6$</td>
<td><strong>Ei</strong></td>
<td>exbi</td>
</tr>
<tr>
<td>$1000^7$</td>
<td><strong>Z</strong></td>
<td>zetta</td>
<td></td>
<td>$1024^7$</td>
<td><strong>Zi</strong></td>
<td>zebi</td>
</tr>
<tr>
<td>$1000^8$</td>
<td><strong>Y</strong></td>
<td>yotta</td>
<td></td>
<td>$1024^8$</td>
<td><strong>Yi</strong></td>
<td>yobi</td>
</tr>
</tbody>
</table>
Memory

• Main memory is volatile - stored information is lost if the electric power is removed

• Secondary memory devices are nonvolatile

• Main memory and disks are direct access devices - information can be reached directly

• The terms direct access and random access often are used interchangeably

• A magnetic tape is a sequential access device since its data is arranged in a linear order - you must get by the intervening data in order to access other information
Hard Disk Drive

Inside of Hard Drive video
RAM vs. ROM

- **RAM** - Random Access Memory (direct access)
- **ROM** - Read-Only Memory

The terms RAM and main memory are basically interchangeable

- ROM could be a set of memory chips, or a separate device, such as a CD ROM
- Both RAM and ROM are random (direct) access devices!
- RAM probably should be called Read-Write Memory
Compact Discs

- A CD-ROM is portable read-only memory
- A microscopic pit on a CD represents a binary 1 and a smooth area represents a binary 0
- A low-intensity laser reflects strongly from a smooth area and weakly from a pit (destructive interference of light – pit is $\lambda/4$ deep)
- A CD-Recordable (CD-R) drive can be used to write information to a CD once
- A CD-Rewritable (CD-RW) can be erased and reused
- The speed of a CD drive indicates how fast (max) it can read and write information to a CD

Colors wavelengths
DVDs

• A DVD is the same physical size as a CD, but can store much more information

• The format of a DVD stores more bits per square inch

• A CD can store 650 MB, while a standard DVD can store 4.7 GB
  – A double sided DVD can store 9.4 GB
  – Other advanced techniques can bring the capacity up to 54.0 GB (Blue-Ray – why is blue?)

• Like CDs, there are DVD-R and DVD-RW discs
Quick Check

Which memory capacity is the largest?

Binary units (not decimal) – Transform all of them in bytes in order to compare them

$$1 \text{ gigabyte} = 2^{30} \text{ bytes}$$

• A) 1,500,000,000,000 bytes
• B) 100 gigabytes
• C) 3,500,000 kilobytes
• D) 10 terabyte
• E) 12,000,000 megabytes
Quick Check

Which memory capacity is the largest?

• **A)** 1,500,000,000,000 bytes = $1.5 \times 10^{12}$ bytes

• **B)** 100 gigabytes = $100 \times 2^{30}$ bytes = $1.07 \times 10^{11}$ bytes

• **C)** 3,500,000 kilobytes = $3.5 \times 10^6 \times 2^{10}$ bytes = $3.58 \times 10^9$ bytes

• **D)** 10 terabyte = $10 \times 2^{40}$ bytes = $1.09 \times 10^{13}$ bytes

• **E)** 12,000,000 megabytes = $1.2 \times 10^7 \times 2^{20}$ = $1.25 \times 10^{13}$ bytes
The Central Processing Unit

- A CPU is on a chip called a microprocessor.
- It continuously follows the fetch-decode-execute cycle:
  - **fetch**: Retrieve an instruction from main memory
  - **decode**: Determine what the instruction is
  - **execute**: Carry out the instruction
The Central Processing Unit

- Arithmetic / Logic Unit: Performs calculations and makes decisions
- Control Unit: Coordinates processing steps
- Registers: Small storage areas

Copyright © 2012 Pearson Education, Inc.
The Central Processing Unit

- The speed of a CPU is controlled by the system clock
- The system clock generates an electronic pulse at regular intervals
- The pulses coordinate the activities of the CPU
- The speed is usually measured in gigahertz (GHz)
Quick Check

Volatility is a property of

A) RAM
B) ROM
C) disk
D) software
E) computer networks
Quick Check

Volatility is a property of

A) RAM
B) ROM
C) disk
D) software
E) computer networks

Volatility means that the contents of memory are lost if the electrical power is shut off. This is true of RAM (Random Access Memory), but not ROM (Read Only Memory) or disk. Software and computer networks are not forms of memory.
Quick Check

Which phase of the fetch-decode-execute cycle might use a circuit in the arithmetic-logic unit?

A) fetch
B) decode
C) execute
D) during fetch or execute, but not decode
E) could be used in fetch, decode or execute phase
Quick Check

Which phase of the fetch-decode-execute cycle might use a circuit in the arithmetic-logic unit?

A) fetch
B) decode
C) execute
D) during fetch or execute, but not decode
E) could be used in fetch, decode or execute phase

The fetch phase retrieves (fetches) the next program instruction from memory. The decode phase determines which circuit(s) needs to be used to execute the instruction. The instruction is executed during the execute phase. If the instruction is either an arithmetic operation (like add or multiply) or a logical operation (like comparing two values), then it is carried out by the ALU.
Monitor

- The size of a monitor (17") is measured diagonally, like a television screen.

- A monitor has a certain maximum resolution, indicating the number of picture elements, called pixels, that it can display (such as 1280 by 1024).

- High resolution (more pixels per inch) produces sharper pictures.
Typical Hardware Assumptions

- Symbol: statistic
- **s**: average seek time **5 ms = 5 x 10^{-3} s**
- **b**: transfer time per byte **0.02 μs = 2 x 10^{-8} s**
- **p**: processor’s clock rate **10^9 s^{-1} = 10^9 Hz**
- **p**: low-level operation (e.g., compare & swap a word) **0.01 μs = 10^{-8} s**
- size of main memory **several GB**
- size of disk space **1 TB or more**

- Example: Reading 1GB from disk
  - If stored in contiguous blocks: **2 x 10^{-8} s x 10^9 = 20s**
  - If stored in 1M chunks of 1KB: **20s + 10^6 x 5 x 10^{-3} s = 5020 s = 1.4 h**
Computer Networks

- A **computer network** is two or more computers connected together using a telecommunication system for the purpose of communicating and sharing resources.
- Why they are interesting?
  - Overcome geographic limits
  - Access remote data
  - Separate clients and server
- Goal: Universal Communication (any to any)
Networks

• A *network* is two or more computers that are connected so that data and resources can be shared

• Most computers are connected to some kind of network

• Each computer has its own *network address*, which uniquely identifies it among the others

• A *file server* is a network computer dedicated to storing programs and data that are shared among network users
Network Connections

- Each computer in a network could be directly connected to every other computer in the network.
- These are called *point-to-point* connections.

Adding a computer requires a new communication line for each computer already in the network.

This technique is not practical for more than a few close machines.
Network Connections

- Most networks share a single communication line
- Adding a new computer to the network is relatively easy

Network traffic must take turns using the line, which introduces delays

Often information is broken down in parts, called packets, which are sent to the receiving machine and then reassembled
A Computer Network
Local-Area Networks

A *Local-Area Network* (LAN) covers a small distance and a small number of computers.

A LAN often connects the machines in a single room or building.
A Wide-Area Network (WAN) connects two or more LANs, often over long distances.
The Internet

- The *Internet* is a WAN which spans the planet
- The word Internet comes from the term *internetworking*
- It started as a United States government project, sponsored by the Advanced Research Projects Agency (ARPA)
  - originally it was called the ARPANET
- The Internet grew quickly throughout the 1980s and 90s
A protocol is a set of rules that determine how things communicate with each other.

The software that manages Internet communication follows a suite of protocols called **TCP/IP**.

The *Internet Protocol* (IP) deals with packets and connects independent networks to transport the packets across network boundaries.

The *Transmission Control Protocol* (TCP) is responsible for maintaining end-to-end communications across the network. TCP handles communications between hosts and provides flow control, multiplexing and reliability.
IP and Internet Addresses

• Each computer on the Internet has a unique IP address, such as (32 bits):

  204.192.116.2

• Most computers also have a unique Internet name, which also is referred to as an Internet address:

  hector.vt.edu
  kant.gestalt-llc.com

• The first part indicates a particular computer (hector)

• The rest is the domain name, indicating the organization (vt.edu)
Domain Names

- The last part of a domain name, called a top-level domain (TLD), supposedly indicates the type of organization:

  - **edu** educational institution
  - **com** commercial entity
  - **org** non-profit organization
  - **net** network-based organization

Sometimes the suffix indicates the country:

- **uk** United Kingdom
- **au** Australia
- **ca** Canada
- **se** Sweden

Additional TLDs have been added:

- **biz**, **info**, **tv**, **name**
Domain Names

- A domain name can have several parts
- Unique domain names mean that multiple sites can have individual computers with the same local name
- When used, an Internet address is translated to an IP address by software called the Domain Name System (DNS)
- There is no one-to-one correspondence between the sections of an IP address and the sections of an Internet address
The World Wide Web

- The *World Wide Web* allows many different types of information to be accessed using a common interface.

- A *browser* is a program which accesses network resources and presents them.
  - Popular browsers: Internet Explorer, Safari, Firefox, Chrome.

- Resources presented include:
  - text, graphics, video, sound, audio, executable programs.

- A Web document usually contains *links* to other Web documents, creating a *hypermedia* environment.

- The term Web comes from the fact that information is not organized in a linear fashion.
The World Wide Web

- Web documents are often defined using the *HyperText Markup Language* (HTML)

- Information on the Web is found using a *Uniform Resource Locator* (URL):
  
  http://www.cnn.com
  
  http://www.vt.edu/student_life/index.html
  
  ftp://java.sun.com/applets/animation.zip

- A URL specifies a protocol (http), a domain, and possibly specific documents
When you click on a http://www.unibz.it

- The browser determines the URL (sees what is selected)
- The browser asks DNS for the IP address of www.unibz.it
- **DNS replies with 46.18.24.134**
- The browser makes a TCP connection to port 80 on 46.18.24.134
- It sends over a request asking for path "/" and default filename
- The www.unibz.it server sends the file /index.html
- The TCP connection is released
- The browser displays all the text in index.html (formatting the text according to the instructions contained in the page).
Quick Check

A URL (Universal Resource Locator) specifies the address of
A) a computer on any network
B) a computer on the Internet
C) a local area network (LAN) on the Internet
D) a document or other type of file on the Internet
E) a Java program on the Internet
Quick Check

A URL (Universal Resource Locator) specifies the address of
A) a computer on any network
B) a computer on the Internet
C) a local area network (LAN) on the Internet
D) a document or other type of file on the Internet
E) a Java program on the Internet

URLs are used to locate documents (or other types of files such as an image or sound file) anywhere on the Internet. An URL contains the address of the LAN or WAN and the specific computer from which the file is to be retrieved; it specifies the file's address, not just the computer's address.
Algorithms

- When we write a computer program, we are generally implementing method that has been devised previously to solve some problem
- **Algorithm for making buttered toast (1.0)**
  - step 1: Get a loaf of bread
  - step 2: Cut one slice from end of the loaf of bread
  - step 3: Put slice of bread into toaster
  - step 4: Turn toaster on
  - step 5: Wait for toaster to finish
  - step 6: Put toasted bread onto a plate
  - step 7: Spread butter on toast.
Euclid’s Algorithm

**English-language description**

Compute the greatest common divisor of two nonnegative integers \( p \) and \( q \) as follows: If \( q \) is 0, the answer is \( p \). If not, divide \( p \) by \( q \) and take the remainder \( r \). The answer is the greatest common divisor of \( q \) and \( r \).

**Java-language description**

```java
public static int gcd(int p, int q) {
    if (q == 0) return p;
    int r = p % q;
    return gcd(q, r);
}
```

Euclid’s algorithm
Algorithms

- *Algorithm*: a finite, deterministic, and effective problem-solving method suitable for implementation as a computer program

- Most algorithms of interest involve organizing the data (structures) involved in the computation
  - *In the example before we have variables*

- In an Object Oriented Programming language data structures are implemented with classes and objects (and with primitive data types).
Java

- The Java programming language was created by Sun Microsystems, Inc. (James Gosling)

- It was introduced in 1995 and its popularity has grown quickly since

- A *programming language* specifies the **words** and **symbols** that we can use to write a program

- A *programming language* employs a set of **rules** that dictate how the words and symbols can be put together to form valid *program statements*
Java Program Structure

• In the Java programming language:
  – A program is made up of one or more classes
  – A class contains one or more methods
  – A method contains program statements

• These terms will be explored in detail throughout the course

• A Java application always contains a method called main

• See Lincoln.java
public class Lincoln
{
    // Prints a presidential quote.
    public static void main (String[] args)
    {
        System.out.println ("A quote by Abraham Lincoln:");
        System.out.println ("Whatever you are, be a good one.");
    }
}
public class Lincoln {
    // Prints a presidential quote.
    public static void main (String[] args) {
        System.out.println ("A quote by Abraham Lincoln: ");
        System.out.println ("Whatever you are, be a good one.");
    }
}

Output
A quote by Abraham Lincoln:
Whatever you are, be a good one.
Java Program Structure

// comments about the class
public class MyProgram
{
  // class body

Comments can be placed almost anywhere
}
Java Program Structure

// comments about the class
public class MyProgram
{
    // comments about the method
    public static void main (String[] args)
    {
        // method body
        }
    }
}

Hello Word in several languages
Comments

• Comments should be included to explain the purpose of the program and describe processing steps

• They do not affect how a program works

• Java comments can take three forms:

  // this comment runs to the end of the line

  /* this comment runs to the terminating symbol, even across line breaks */

  /** this is a javadoc comment */
Quick Check

The instruction: `System.out.println("Hello World");`
might best be commented as

A) // prints "Hello World" to the screen
B) // prints a message
C) // used to demonstrate an output message
D) //
E) // meaningless instruction
Quick Check

The instruction: `System.out.println("Hello World");` might best be commented as

A) // prints "Hello World" to the screen
B) // prints a message
C) // used to demonstrate an output message
D) //
E) // meaningless instruction

Comments in A and B state the obvious while the comments in D and E are meaningless. The comment in C explains **why** the instruction appears in the program.
Identifiers

- *Identifiers* are the "words" in a program

- A Java identifier can be made up of **letters**, **digits**, the **underscore** character ( _ ), and the **dollar** $ sign

- Identifiers cannot begin with a digit

- Java is **case sensitive**: Total, total, and TOTAL are different identifiers

- By convention, programmers use different case styles for different types of identifiers, such as
  
  - **title case** for class names - Lincoln
  - **upper case** for constants - MAXIMUM
Identifiers

• Sometimes the programmer chooses the identifier (such as Lincoln)

• Sometimes we are using another programmer's code, so we use the identifiers that he or she chose (such as println)

• Often we use special identifiers called reserved words that already have a predefined meaning in the language

• A reserved word cannot be used in any other way
Reserved Words

- The Java reserved words:

<table>
<thead>
<tr>
<th>abstract</th>
<th>else</th>
<th>interface</th>
<th>switch</th>
</tr>
</thead>
<tbody>
<tr>
<td>assert</td>
<td>enum</td>
<td>long</td>
<td>synchronized</td>
</tr>
<tr>
<td>boolean</td>
<td>extends</td>
<td>native</td>
<td>this</td>
</tr>
<tr>
<td>break</td>
<td>false</td>
<td>new</td>
<td>throw</td>
</tr>
<tr>
<td>byte</td>
<td>final</td>
<td>null</td>
<td>throws</td>
</tr>
<tr>
<td>case</td>
<td>finally</td>
<td>package</td>
<td>transient</td>
</tr>
<tr>
<td>catch</td>
<td>float</td>
<td>private</td>
<td>true</td>
</tr>
<tr>
<td>char</td>
<td>for</td>
<td>protected</td>
<td>try</td>
</tr>
<tr>
<td>class</td>
<td>goto*</td>
<td>public</td>
<td>void</td>
</tr>
<tr>
<td>const*</td>
<td>if</td>
<td>implements</td>
<td>volatile</td>
</tr>
<tr>
<td>continue</td>
<td>implements</td>
<td>protected</td>
<td>while</td>
</tr>
<tr>
<td>default</td>
<td>import</td>
<td>public</td>
<td></td>
</tr>
<tr>
<td>do</td>
<td>instanceof</td>
<td>short</td>
<td></td>
</tr>
<tr>
<td>double</td>
<td>int</td>
<td>static</td>
<td></td>
</tr>
</tbody>
</table>

‘*’ means reserved but not currently used
Quick Check

Which of the following are valid Java identifiers?

grade
quizGrade
NetworkConnection
frame2
3rdTestScore
MAXIMUM
MIN_CAPACITY
student#
Shelves1&2
Which of the following are valid Java identifiers?

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Validity</th>
</tr>
</thead>
<tbody>
<tr>
<td>grade</td>
<td>Valid</td>
</tr>
<tr>
<td>quizGrade</td>
<td>Valid</td>
</tr>
<tr>
<td>NetworkConnection</td>
<td>Valid</td>
</tr>
<tr>
<td>frame2</td>
<td>Valid</td>
</tr>
<tr>
<td>3rdTestScore</td>
<td>Invalid – cannot begin with a digit</td>
</tr>
<tr>
<td>MAXIMUM</td>
<td>Valid</td>
</tr>
<tr>
<td>MIN_CAPACITY</td>
<td>Valid</td>
</tr>
<tr>
<td>student#</td>
<td>Invalid – cannot contain the '#' character</td>
</tr>
<tr>
<td>Shelves1&amp;2</td>
<td>Invalid – cannot contain the '&amp;' character</td>
</tr>
</tbody>
</table>
White Space

• Spaces, blank lines, and tabs are called *white space*

• White space is used to separate words and symbols in a program

• Extra white space is ignored

• A valid Java program can be formatted many ways

• Programs should be formatted to enhance readability, using consistent indentation

• See `Lincoln2.java` and `Lincoln3.java`
public class Lincoln3 {
    public static void main (String [] args) {
        System.out.println ("A quote by Abraham Lincoln:" + )
        System.out.println ("Whatever you are, be a good one." + )
    }
}
Quick Check

The main method for a Java program is defined by a header like this:

• A) public static void main( )
• B) public static void main(String[ ] args);
• C) public static void main(String[ ] args)
• D) private static void main(String[ ] args)
• E) the main method could be defined as in A, C or D but not B
Quick Check

The main method for a Java program is defined by …
• A) public static void main()
• B) public static void main(String[] args);
• C) public static void main(String[] args)
• D) private static void main(String[] args)
• E) the main method could be defined as in A, C or D but not B

In A, the parameter is missing. The parameters are defined later in the text, but in effect, they allow the user to run the program and include some initial arguments if the program calls for it. In B, the semicolon at the end of the statement is not allowed. In D, "private" instead of "public" would make the program non-executable by anyone and thus makes the definition meaningless.
Program Development

• The mechanics of developing a program include several activities:
  – writing the program in a specific programming language (such as Java)
  – translating the program into a form that the computer can execute
  – investigating and fixing various types of errors that can occur

• Software tools can be used to help with all parts of this process
Language Levels

- There are four programming language levels:
  - machine language
  - assembly language
  - high-level language
  - fourth-generation language (PowerBuilder, PHP, FoxPro, etc.)

- Each type of CPU has its own specific *machine language*

- The other levels were created to make it easier for a human being to read and write programs
## High-Level Expression Translation

<table>
<thead>
<tr>
<th>High-Level Language</th>
<th>Assembly Language</th>
<th>Machine Language</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a + b$</td>
<td><code>ld [%fp-20], %0</code></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td><code>ld [%fp-24], %1</code></td>
<td>1101 0000 0000 0111</td>
</tr>
<tr>
<td></td>
<td><code>add %0, %1, %0</code></td>
<td>1011 1111 1110 1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1101 0010 0000 0111</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1011 1111 1110 1000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1001 0000 0000 0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
Programming Languages

• Each type of CPU executes only a particular *machine language*

• A program must be translated into machine language before it can be executed

• A *compiler* is a software tool which translates *source code* into a specific target language

• Often, that target language is the machine language for a particular CPU type

• The Java approach is somewhat different
Java Translation

• The Java compiler translates Java source code into a special representation called bytecode.

• Java bytecode is not the machine language for any traditional CPU.

• Another software tool, called an interpreter, translates bytecode into machine language and executes it.

• Therefore the Java compiler is not tied to any particular machine.

• Java is considered to be architecture-neutral.
Java Translation

Java source code -> Java bytecode

Java compiler

Bytecode interpreter

Bytecode compiler

Machine code

https://www.youtube.com/watch?v=_C5AHaS1mOA
Development Environments

• There are many programs that support the development of Java software, including:
  – Java Development Kit (JDK)
  – Eclipse
  – NetBeans
  – BlueJ
  – jGRASP

• Though the details of these environments differ, the basic compilation and execution process is essentially the same
Syntax and Semantics

• The *syntax rules* of a language define how we can put together symbols, reserved words, and identifiers to make a valid program.

• The *semantics* of a program statement define what that statement means (its purpose or role in a program).

• A program that is syntactically correct is not necessarily logically (semantically) correct.

• A program will always do what we tell it to do, not what we meant to tell it to do.
Errors

• A program can have three types of errors
• The compiler will find syntax errors and other basic problems (*compile-time errors*)
  – If compile-time errors exist, an executable version of the program is not created
• A problem can occur during program execution, e.g. trying to divide by zero, which causes a program to terminate abnormally (*run-time errors*)
• A program may run, but produce incorrect results, perhaps using an incorrect formula (*logical errors*)
Basic Program Development

1. Edit and save program
2. Compile program
3. Execute program and evaluate results

If errors are found, go back to the previous step.
Quick Check

An error in a program that results in the program outputting $100 instead of the correct answer, $250 is

A) a programmer error
B) a syntax error
C) a run-time error
D) a logical error
E) a snafu
An error in a program that results in the program outputting $100 instead of the correct answer, $250 is

A) a programmer error
B) a syntax error
C) a run-time error
D) a logical error
E) a snafu

While this is an error (answer A), programmers classify the type of error in order to more easily solve the problem. Syntax errors are caught by the compiler and the program cannot run without fixing all syntax errors. Run-time errors arise during program execution and cause the program to stop running. Logical errors are errors whereby the program can run to completion, but gives the wrong answer.
Quick Check

Mistyping "println" as "printn" will result in

A) a syntax error
B) a run-time error
C) a logical error
D) no error at all
E) converting the statement into a comment
Quick Check

Mistyping "println" as "printn" will result in

A) a syntax error
B) a run-time error
C) a logical error
D) no error at all
E) converting the statement into a comment

If the Java compiler cannot make sense of a command, the compiler cannot convert it and responds with a syntax error. While "println" is recognized as a command, "printn" is not, and so the compiler provides a syntax error.
Outline

Computer Processing
Hardware Components
Networks
The Java Programming Language
Program Development

Object-Oriented Programming
Problem Solving

• The purpose of writing a program is to solve a problem

• Solving a problem consists of multiple activities:
  – Understand the problem
  – Design a solution
  – Consider alternatives and refine the solution
  – Implement the solution
  – Test the solution

• These activities are not purely linear – they overlap and interact
Problem Solving

• The key to designing a solution is breaking it down into manageable pieces

• When writing software, we design separate pieces that are responsible for certain parts of the solution

• An object-oriented approach lends itself to this kind of solution decomposition

• We will dissect our solutions into pieces called objects and classes
Object-Oriented Programming

- Java is an object-oriented programming language
- As the term implies, an object is a fundamental entity in a Java program
- Objects can be used effectively to represent real-world entities
- For instance, an object might represent a particular employee in a company
- Each employee object handles the processing and data management related to that employee
Objects

• An object has:
  – *state* - descriptive characteristics
  – *behaviors* - what it can do (or what can be done to it)

• The state of a bank account includes its account number and its current balance

• The behaviors associated with a bank account include the ability to make deposits and withdrawals

• Note that the behavior of an object might change its state
A Software Object
Classes

- An object is defined by a *class*
- A class is the blueprint of an object
- The class uses methods to define the behaviors of the object
- The class that contains the *main* method of a Java program represents the entire program
- A class represents a concept, and an object represents the embodiment of that concept
- Multiple objects can be created from the same class
A Class

class Bicycle {
    int cadence = 0;
    int speed = 0;
    int gear = 1;

    void changeCadence(int newValue) {
        cadence = newValue;
    }
    ...

    void printStates() {
        System.out.println("cadence:" + 
                          cadence + " speed:" + 
                          speed + " gear:" + gear);
    }
}

Copyright © 2012 Pearson Education, Inc.
Class = Blueprint

• One blueprint to create several similar, but different, houses:
Objects and Classes

A class (the concept)

Bank Account

Multiple objects from the same class

An object (the realization)

John’s Bank Account
Balance: $5,257

Bill’s Bank Account
Balance: $1,245,069

Mary’s Bank Account
Balance: $16,833

Copyright © 2012 Pearson Education, Inc.
Panda
Inheritance

- One class can be used to derive another via inheritance
- Classes can be organized into hierarchies
Quick Check

What is wrong with the following class definition?

```java
public class Program2
    public static void main(String[ ] args)
    {
        System.out.println("My 2nd Java program");
    }
```
Quick Check

What is wrong with the following class definition?

```java
public class Program2
    public static void main(String[ ] args)
    {
        System.out.println("My 2nd Java program");
    }
```

The definition of a class is placed within { } statements, which are missing here.
Summary

• Chapter 1 focused on:
  – components of a computer
  – how those components interact
  – how computers store and manipulate information
  – computer networks
  – the Internet and the World Wide Web
  – programming and programming languages
  – an introduction to Java
  – an overview of object-oriented concepts
Exercises

• Why do we use the binary number system to store information on a computer?

• If a picture is made up of 128 possible colors, how many bits would be needed to store each pixel of the picture? Why?

• Explain the difference between random-access memory (RAM) and read-only memory (ROM).

• What is the total number of communication lines needed for a fully connected point-to-point network of eight computers? Nine computers? Ten computers? What is a general formula for determining this result?
Exercises

• What is a Java Virtual Machine? Explain its role.
• Categorize each of the following situations as a compile-time error, run-time error, or logical error.
  – multiplying two numbers when you meant to add them
  – dividing by zero
  – forgetting a semicolon at the end of a programming statement
  – spelling a word wrong in the output
  – producing inaccurate results
  – typing a { when you should have typed (