Introduction to Computers and Programming
Lecture Notes

These all slide pages are selected from “C How to Program”, 5/e and 7/e

Asst.Prof.Dr. Mahmut YALCIN
1.1 Introduction

- The core of the book emphasizes effective software engineering through the proven methodologies of structured programming in C and object-oriented programming in C++.
- All of these example programs may be downloaded from our website www.deitel.com/books/chtp7/.
- You’ll learn how to command computers to perform tasks.
  - **Software** (i.e., the instructions you write to command computers to perform **actions** and make **decisions**) controls computers (often referred to as **hardware**).
1.2 Computers and the Internet in Industry and Research

- Figure 1.1 provides a few examples of how computers are used in industry and research.
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Electronic health records</td>
<td>These might include a patient’s medical history, prescriptions, immunizations, lab results, allergies, insurance information and more. Making this information available to health care providers across a secure network improves patient care, reduces the probability of error and increases overall efficiency of the health care system.</td>
</tr>
<tr>
<td>Human Genome Project</td>
<td>The Human Genome Project was founded to identify and analyze the 20,000+ genes in human DNA. The project used computer programs to analyze complex genetic data, determine the sequences of the billions of chemical base pairs that make up human DNA and store the information in databases which have been made available over the Internet to researchers in many fields.</td>
</tr>
<tr>
<td>AMBER™ Alert</td>
<td>The AMBER (America’s Missing: Broadcast Emergency Response) Alert System is used to find abducted children. Law enforcement notifies TV and radio broadcasters and state transportation officials, who then broadcast alerts on TV, radio, computerized highway signs, the Internet and wireless devices. AMBER Alert recently partnered with Facebook, whose users can “Like” AMBER Alert pages by location to receive alerts in their news feeds.</td>
</tr>
</tbody>
</table>

**Fig. 1.1 | A few uses for computers. (Part 1 of 7.)**
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>World Community Grid</td>
<td>People worldwide can donate their unused computer processing power by installing a free secure software program that allows the World Community Grid (<a href="http://www.worldcommunitygrid.org">www.worldcommunitygrid.org</a>) to harness unused capacity. This computing power, accessed over the Internet, is used in place of expensive supercomputers to conduct scientific research projects that are making a difference—providing clean water to third-world countries, fighting cancer, growing more nutritious rice for regions fighting hunger and more.</td>
</tr>
<tr>
<td>Medical imaging</td>
<td>X-ray computed tomography (CT) scans, also called CAT (computerized axial tomography) scans, take X-rays of the body from hundreds of different angles. Computers are used to adjust the intensity of the X-rays, optimizing the scan for each type of tissue, then to combine all of the information to create a 3D image. MRI scanners use a technique called magnetic resonance imaging, also to produce internal images non-invasively.</td>
</tr>
<tr>
<td>One Laptop Per Child (OLPC)</td>
<td>One Laptop Per Child (one.laptop.org) is providing low-power, inexpensive, Internet-enabled laptops to children in third-world countries—enabling learning and reducing the digital divide.</td>
</tr>
</tbody>
</table>

**Fig. 1.1 |** A few uses for computers. (Part 2 of 7.)
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cloud computing</td>
<td><strong>Cloud computing</strong> allows you to use software, hardware and information stored in the “cloud”—i.e., accessed on remote computers via the Internet and available on demand—rather than having it stored on your personal computer. These services allow you to increase or decrease resources to meet your needs at any given time, so they can be more cost effective than purchasing expensive hardware to ensure that you have enough storage and processing power to meet your needs at their peak levels. Business applications often are expensive, and require significant hardware to run them and knowledgeable support staff to ensure that they’re running properly and securely. Using cloud computing services shifts the burden of managing these applications from the business to the service provider, saving businesses money.</td>
</tr>
</tbody>
</table>

**Fig. 1.1** | A few uses for computers. (Part 3 of 7.)
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPS</td>
<td>Global Positioning System (GPS) devices use a network of satellites to retrieve location-based information. Multiple satellites send time-stamped signals to the GPS device, which calculates the distance to each satellite based on the time the signal left the satellite and the time the signal arrived. This information is used to determine the exact location of the device. GPS devices can provide step-by-step directions and help you locate nearby businesses (restaurants, gas stations, etc.) and points of interest. GPS is used in numerous location-based Internet services such as check-in apps to help you find your friends (e.g., Four-square and Facebook), exercise apps such as RunKeeper that track the time, distance and average speed of your outdoor jog, dating apps that help you find a match nearby and apps that dynamically update changing traffic conditions.</td>
</tr>
</tbody>
</table>

**Fig. 1.1** | A few uses for computers. (Part 4 of 7.)
<table>
<thead>
<tr>
<th><strong>Name</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Robots</td>
<td>Robots can be used for day-to-day tasks (e.g., iRobot's Roomba vacuuming robot), entertainment (e.g., robotic pets), military combat, deep sea and space exploration (e.g., NASA's Mars rover) and more. RoboEarth (<a href="http://www.roboearth.org">www.roboearth.org</a>) is “a World Wide Web for robots.” It allows robots to learn from each other by sharing information and thus improving their abilities to perform tasks, navigate, recognize objects and more.</td>
</tr>
<tr>
<td>E-mail, Instant Messaging, Video Chat and FTP</td>
<td>Internet-based servers support all of your online messaging. E-mail messages go through a mail server that also stores the messages. Instant Messaging (IM) and Video Chat apps, such as AIM, Skype, Yahoo! Messenger and others allow you to communicate with others in real time by sending your messages and live video through servers. FTP (file transfer protocol) allows you to exchange files between multiple computers (e.g., a client computer such as your desktop and a file server) over the Internet.</td>
</tr>
</tbody>
</table>

**Fig. 1.1** A few uses for computers. (Part 5 of 7.)
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internet TV</td>
<td>Internet TV set-top boxes (such as Apple TV, Google TV and TiVo) allow you to access an enormous amount of content on demand, such as games, news, movies, television shows and more, and they help ensure that the content is streamed to your TV smoothly.</td>
</tr>
</tbody>
</table>

**Fig. 1.1** | A few uses for computers. (Part 6 of 7.)
Name | Description
--- | ---
Game programming | Analysts expect global video game revenues to reach $91 billion by 2015 (www.vg247.com/2009/06/23/global-industry-analysts-predicts-gaming-market-to-reach-91-billion-by-2015/). The most sophisticated games can cost as much as $100 million to develop. Activision’s *Call of Duty: Black Ops*—one of the best-selling games of all time—earned $360 million in just one day (www.forbes.com/sites/insertcoin/2011/03/11/call-of-duty-black-ops-now-the-best-selling-video-game-of-all-time/)! Online *social gaming*, which enables users worldwide to compete with one another over the Internet, is growing rapidly. Zynga—creator of popular online games such as *Farmville* and *Mafia Wars*—was founded in 2007 and already has over 200 million monthly users. To accommodate the growth in traffic, Zynga is adding nearly 1,000 servers each week (techcrunch.com/2010/09/22/zynga-moves-1-petabyte-of-data-daily-adds-1000-servers-a-week/).

**Fig. 1.1** | A few uses for computers. (Part 7 of 7.)
1.2 What is a Computer?

- **Computer**
  - Device capable of performing computations and making logical decisions
  - Computers process data under the control of sets of instructions called computer programs

- **Hardware**
  - Various devices comprising a computer
  - Keyboard, screen, mouse, disks, memory, CD-ROM, and processing units

- **Software**
  - Programs that run on a computer
1.3 Computer Organization

- Six logical units in every computer:
  1. Input unit
     - Obtains information from input devices (keyboard, mouse)
  2. Output unit
     - Outputs information (to screen, to printer, to control other devices)
  3. Memory unit
     - Rapid access, low capacity, stores input information
  4. Arithmetic and logic unit (ALU)
     - Performs arithmetic calculations and logic decisions
  5. Central processing unit (CPU)
     - Supervises and coordinates the other sections of the computer
  6. Secondary storage unit
     - Cheap, long-term, high-capacity storage
     - Stores inactive programs
1.4 Early Operating Systems

- Batch processing
  - Do only one job or task at a time
- Operating systems
  - Manage transitions between jobs
  - Increased throughput
    - Amount of work computers process
- Multitasking
  - Computer resources are shared by many jobs or tasks
- Timesharing
  - Computer runs a small portion of one user’s job then moves on to service the next user
1.5 Personal Computing, Distributed Computing, and Client/Server Computing

- Personal computers
  - Economical enough for individual
- Distributed computing
  - Computing distributed over networks
- Client/server computing
  - Sharing of information across computer networks between file servers and clients (personal computers)
1.6 Machine Languages, Assembly Languages, and High-level Languages

Three types of programming languages

1. Machine languages
   - Strings of numbers giving machine specific instructions
   - Example:
     +1300042774
     +1400593419
     +1200274027

2. Assembly languages
   - English-like abbreviations representing elementary computer operations (translated via assemblers)
   - Example:
     LOAD BASEPAY
     ADD OVERPAY
     STORE GROSSPAY
1.6 Machine Languages, Assembly Languages, and High-level Languages

Three types of programming languages (continued)

3. High-level languages
   - Codes similar to everyday English
   - Use mathematical notations (translated via compilers)
   - Example:
     - grossPay = basePay + overTimePay
1.7 Fortran, COBOL, Pascal and Ada

- **Fortran**
  - developed by IBM Corporation in the 1950s
  - used for scientific and engineering applications that require complex mathematical computations

- **COBOL**
  - developed in 1959 by computer manufacturers, the government and industrial computer users
  - used for commercial applications that require precise and efficient manipulation of large amounts of data
1.7 Fortran, COBOL, Pascal and Ada

- Pascal
  - Developed by Professor Niklaus Wirth in 1971
  - Designed for teaching structured programming

- Ada
  - Developed under the sponsorship of the U.S. Department of Defense (DOD) during the 1970s and early 1980s
  - Able to perform multitasking
1.8 History of C

- **C**
  - Evolved by Ritchie from two previous programming languages, BCPL and B
  - Used to develop UNIX
  - Used to write modern operating systems
  - Hardware independent (portable)
  - By late 1970's C had evolved to "Traditional C"

- **Standardization**
  - Many slight variations of C existed, and were incompatible
  - Committee formed to create a "unambiguous, machine-independent" definition
  - Standard created in 1989, updated in 1999
Portability Tip 1.1

- Because C is a hardware-independent, widely available language, applications written in C can run with little or no modifications on a wide range of different computer systems.
1.9 C Standard Library

- C programs consist of pieces/modules called functions
  - A programmer can create his own functions
    - Advantage: the programmer knows exactly how it works
    - Disadvantage: time consuming
  - Programmers will often use the C library functions
    - Use these as building blocks
  - Avoid re-inventing the wheel
    - If a pre-made function exists, generally best to use it rather than write your own
    - Library functions carefully written, efficient, and portable
Performance Tip 1.1

- Using Standard C library functions instead of writing your own comparable versions can improve program performance, because these functions are carefully written to perform efficiently.
Using Standard C library functions instead of writing your own comparable versions can improve program portability, because these functions are used in virtually all Standard C implementations.
1.13 Key Software Trend: Object Technology

- **Objects**
  - Reusable software components that model items in the real world
  - Meaningful software units
    - Date objects, time objects, paycheck objects, invoice objects, audio objects, video objects, file objects, record objects, etc.
    - Any noun can be represented as an object
  - Very reusable
  - More understandable, better organized, and easier to maintain than procedural programming
  - Favor modularity
1.14 Typical C Program Development Environment

• Phases of C++ Programs:
  • Edit
  • Preprocess
  • Compile
  • Link
  • Load
  • Execute

Fig. 1.1 | Typical C development environment
Errors like division-by-zero occur as a program runs, so these errors are called runtime errors or execution-time errors. Divide-by-zero is generally a fatal error, i.e., an error that causes the program to terminate immediately without successfully performing its job. Nonfatal errors allow programs to run to completion, often producing incorrect results. [Note: *On some systems, divide-by-zero is not a fatal error. Please see your system documentation.*]
1.15 Hardware Trends

Every year or two the following approximately double:

- Amount of memory in which to execute programs
- Amount of secondary storage (such as disk storage)
  - Used to hold programs and data over the longer term
- Processor speeds
  - The speeds at which computers execute their programs
1.16 History of the Internet

- The Internet enables
  - Quick and easy communication via e-mail
  - International networking of computers
- Packet switching
  - The transfer of digital data via small packets
  - Allows multiple users to send and receive data simultaneously
- No centralized control
  - If one part of the Internet fails, other parts can still operate
- TCP/IP
- Bandwidth
  - Information carrying capacity of communications lines
1.17 History of the World Wide Web

- World Wide Web
  - Locate and view multimedia-based documents on almost any subject
  - Makes information instantly and conveniently accessible worldwide
  - Possible for individuals and small businesses to get worldwide exposure
  - Changing the way business is done
1.18 General Notes About C and This Book

- Program clarity
  - Programs that are convoluted are difficult to read, understand, and modify
- C is a portable language
  - Programs can run on many different computers
  - However, portability is an elusive goal
- We will do a careful walkthrough of C
  - Some details and subtleties are not covered
  - If you need additional technical details
    - Read the C standard document
    - Read the book by Kernigan and Ritchie
Good Programming Practice

1.1

- Write your C programs in a simple and straightforward manner. This is sometimes referred to as KIS ("keep it simple"). Do not "stretch" the language by trying bizarre usages.
Portability Tip 1.3

- Although it is possible to write portable programs, there are many problems between different C compilers and different computers that make portability difficult to achieve. Simply writing programs in C does not guarantee portability. The programmer will often need to deal directly with complex computer variations.
1.3 Computers: Hardware and Software

- In use today are more than a billion general-purpose computers, and billions more *embedded* computers are used in cell phones, smartphones, tablet computers, home appliances, automobiles and more.
- Computers can perform computations and make logical decisions phenomenally faster than human beings can.
- Today’s personal computers can perform billions of calculations in one second—more than a human can perform in a lifetime.
- *Supercomputers* are already performing *thousands of trillions* (*quadrillions*) of instructions per second!
- Computers process data under the control of sets of instructions called *computer programs*.
- These programs guide the computer through ordered actions specified by people called computer *programmers*. 
1.3 Computers: Hardware and Software (Cont.)

- The programs that run on a computer are referred to as software.

- You’ll learn key programming methodology that are enhancing programmer productivity, thereby reducing software-development costs—structured programming (in C) and object-oriented programming in C++.

- A computer consists of various devices referred to as hardware
  - (e.g., the keyboard, screen, mouse, hard disks, memory, DVD drives and processing units).

- Computing costs are dropping dramatically, owing to rapid developments in hardware and software technologies.
1.3 Computers: Hardware and Software (Cont.)

- Computers that might have filled large rooms and cost millions of dollars decades ago are now inscribed on silicon chips smaller than a fingernail, costing perhaps a few dollars each.
- Silicon-chip technology has made computing so economical that more than computers have become a commodity.
1.3.1 Moore’s Law

- For many decades, hardware costs have fallen rapidly.
- Every year or two, the capacities of computers have approximately doubled inexpensively.
- This remarkable trend often is called Moore’s Law, named for the person who identified it, Gordon Moore, co-founder of Intel.
1.2 Computers: Hardware and Software (Cont.)

- Moore’s Law and related observations apply especially to the amount of memory that computers have for programs, the amount of secondary storage (such as disk storage) they have to hold programs and data over longer periods of time, and their processor speeds—the speeds at which computers execute their programs (i.e., do their work).
- Similar growth has occurred in the communications field.
1.2 Computers: Hardware and Software (Cont.)

- Costs have plummeted as enormous demand for communications bandwidth (i.e., information-carrying capacity) has attracted intense competition.
- Such phenomenal improvement is fostering the Information Revolution.
1.3.2 Computer Organization

- Regardless of differences in physical appearance, computers can be envisioned as divided into various logical units or sections (Fig. 1.2).
<table>
<thead>
<tr>
<th>Logical unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input unit</td>
<td>This “receiving” section obtains information (data and computer programs) from <strong>input devices</strong> and places it at the disposal of the other units for processing. Most information is entered into computers through keyboards, touch screens and mouse devices. Other forms of input include receiving voice commands, scanning images and barcodes, reading from secondary storage devices (such as hard drives, DVD drives, Blu-ray Disc™ drives and USB flash drives—also called “thumb drives” or “memory sticks”), receiving video from a webcam and having your computer receive information from the Internet (such as when you download videos from YouTube™ or e-books from Amazon). Newer forms of input include position data from a GPS device, and motion and orientation information from an accelerometer in a smartphone or game controller (such as Microsoft® Kinect™, Wii™ Remote and PlayStation® Move).</td>
</tr>
</tbody>
</table>

**Fig. 1.2**  | Logical units of a computer. (Part 1 of 4.)
<table>
<thead>
<tr>
<th>Logical unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output unit</td>
<td>This “shipping” section takes information that the computer has processed and places it on various output devices to make it available for use outside the computer. Most information that’s output from computers today is displayed on screens, printed on paper, played as audio or video on PCs and media players (such as Apple’s popular iPods) and giant screens in sports stadiums, transmitted over the Internet or used to control other devices, such as robots and “intelligent” appliances.</td>
</tr>
<tr>
<td>Memory unit</td>
<td>This rapid-access, relatively low-capacity “warehouse” section retains information that has been entered through the input unit, making it immediately available for processing when needed. The memory unit also retains processed information until it can be placed on output devices by the output unit. Information in the memory unit is volatile—it’s typically lost when the computer’s power is turned off. The memory unit is often called either memory or primary memory. Typical main memories on desktop and notebook computers contain between 1 and 8 GB (GB stands for gigabytes; a gigabyte is approximately one billion bytes).</td>
</tr>
</tbody>
</table>

*Fig. 1.2 | Logical units of a computer. (Part 2 of 4.)*
<table>
<thead>
<tr>
<th><strong>Logical unit</strong></th>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic and logic unit (ALU)</td>
<td>This “manufacturing” section performs calculations, such as addition, subtraction, multiplication and division. It also contains the decision mechanisms that allow the computer, for example, to compare two items from the memory unit to determine whether they’re equal. In today’s systems, the ALU is usually implemented as part of the next logical unit, the CPU.</td>
</tr>
<tr>
<td>Central processing unit (CPU)</td>
<td>This “administrative” section coordinates and supervises the operation of the other sections. The CPU tells the input unit when information should be read into the memory unit, tells the ALU when information from the memory unit should be used in calculations and tells the output unit when to send information from the memory unit to certain output devices. Many of today’s computers have multiple CPUs and, hence, can perform many operations simultaneously. A multi-core processor implements multiple processors on a single integrated-circuit chip—a dual-core processor has two CPUs and a quad-core processor has four CPUs. Today’s desktop computers have processors that can execute billions of instructions per second.</td>
</tr>
</tbody>
</table>

Fig. 1.2 | Logical units of a computer. (Part 3 of 4.)
<table>
<thead>
<tr>
<th>Logical unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secondary storage unit</td>
<td>This is the long-term, high-capacity “warehousing” section. Programs or data not actively being used by the other units normally are placed on secondary storage devices (e.g., your hard drive) until they’re again needed, possibly hours, days, months or even years later. Information on secondary storage devices is <em>persistent</em>—it’s preserved even when the computer’s power is turned off. Secondary storage information takes much longer to access than information in primary memory, but the cost per unit of secondary storage is much less than that of primary memory. Examples of secondary storage devices include CD drives, DVD drives and flash drives, some of which can hold up to 512 GB. Typical hard drives on desktop and notebook computers can hold up to 2 TB (TB stands for terabytes; a terabyte is approximately one trillion bytes).</td>
</tr>
</tbody>
</table>

**Fig. 1.2** | Logical units of a computer. (Part 4 of 4.)
1.4 Data Hierarchy

- Data items processed by computers form a data hierarchy that becomes larger and more complex in structure as we progress from bits to characters to fields, and so on.
- Figure 1.3 illustrates a portion of the data hierarchy.
- Figure 1.4 summarizes the data hierarchy’s levels.
<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits</td>
<td>The smallest data item in a computer can assume the value 0 or the value 1. Such a data item is called a bit (short for “binary digit”—a digit that can assume one of two values). It’s remarkable that the impressive functions performed by computers involve only the simplest manipulations of 0s and 1s—examining a bit’s value, setting a bit’s value and reversing a bit’s value (from 1 to 0 or from 0 to 1).</td>
</tr>
<tr>
<td>Characters</td>
<td>It’s tedious for people to work with data in the low-level form of bits. Instead, they prefer to work with decimal digits (0–9), letters (A–Z and a–z), and special symbols (e.g., $, @, %, &amp;, *, (, ), –, +, ”, ;, : and /). Digits, letters and special symbols are known as characters. The computer’s character set is the set of all the characters used to write programs and represent data items. Computers process only 1s and 0s, so every character is represented as a pattern of 1s and 0s. The Unicode character set contains characters for many of the world’s languages. C supports several character sets, including 16-bit Unicode® characters</td>
</tr>
</tbody>
</table>

**Fig. 1.4** | Levels of the data hierarchy. (Part 1 of 4.)
<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Characters</td>
<td>that are composed of two <strong>bytes</strong>, each composed of eight bits. See Appendix B for more information on the <strong>ASCII (American Standard Code for Information Interchange)</strong> character set—the popular subset of Unicode that represents uppercase and lowercase letters, digits and some common special characters.</td>
</tr>
<tr>
<td>Fields</td>
<td>Just as characters are composed of bits, <strong>fields</strong> are composed of characters or bytes. A field is a group of characters or bytes that conveys meaning. For example, a field consisting of uppercase and lowercase letters could be used to represent a person’s name, and a field consisting of decimal digits could represent a person’s age.</td>
</tr>
</tbody>
</table>

**Fig. 1.4** | Levels of the data hierarchy. (Part 2 of 4.)
<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Records</td>
<td>Several related fields can be used to compose a record. In a payroll system, for example, the record for an employee might consist of the following fields (possible types for these fields are shown in parentheses):</td>
</tr>
<tr>
<td></td>
<td>• Employee identification number (a whole number)</td>
</tr>
<tr>
<td></td>
<td>• Name (a string of characters)</td>
</tr>
<tr>
<td></td>
<td>• Address (a string of characters)</td>
</tr>
<tr>
<td></td>
<td>• Hourly pay rate (a number with a decimal point)</td>
</tr>
<tr>
<td></td>
<td>• Year-to-date earnings (a number with a decimal point)</td>
</tr>
<tr>
<td></td>
<td>• Amount of taxes withheld (a number with a decimal point)</td>
</tr>
<tr>
<td></td>
<td>Thus, a record is a group of related fields. In the preceding example, all the fields belong to the same employee. A company might have many employees and a payroll record for each one.</td>
</tr>
</tbody>
</table>

**Fig. 1.4**  | Levels of the data hierarchy. (Part 3 of 4.)
<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Files</td>
<td>A file is a group of related records. [Note: More generally, a file contains arbitrary data in arbitrary formats. In some operating systems, a file is viewed simply as a sequence of bytes—any organization of the bytes in a file, such as organizing the data into records, is a view created by the application programmer.] It’s not unusual for an organization to have many files, some containing billions, or even trillions, of characters of information.</td>
</tr>
<tr>
<td>Database</td>
<td>A database is an electronic collection of data that’s organized for easy access and manipulation. The most popular database model is the relational database in which data is stored in simple tables. A table includes records and fields. For example, a table of students might include first name, last name, major, year, student ID number and grade point average. The data for each student is a record, and the individual pieces of information in each record are the fields. You can search, sort and manipulate the data based on its relationship to multiple tables or databases. For example, a university might use data from the student database in combination with databases of courses, on-campus housing, meal plans, etc.</td>
</tr>
</tbody>
</table>

**Fig. 1.4** | Levels of the data hierarchy. (Part 4 of 4.)
1.5 Programming Languages

- Programmers write instructions in various programming languages, some directly understandable by computers and others requiring intermediate translation steps.
- Any computer can directly understand only its own machine language, defined by its hardware design.
- Machine languages generally consist of numbers (ultimately reduced to 1s and 0s). Such languages are cumbersome for humans.
- Programming in machine language—the numbers that computers could directly understand—was simply too slow and tedious for most programmers.
- Instead, they began using Englishlike abbreviations to represent elementary operations.
- These abbreviations formed the basis of assembly languages.
- Translator programs called assemblers were developed to convert assembly-language programs to machine language.
1.5 Types of Programming Languages (cont.)

- Although assembly-language code is clearer to humans, it’s incomprehensible to computers until translated to machine language.
- To speed the programming process even further, high-level languages were developed in which single statements could be written to accomplish substantial tasks.
- High-level languages allow you to write instructions that look almost like everyday English and contain commonly used mathematical expressions.
- Translator programs called compilers convert high-level language programs into machine language.
- Interpreter programs were developed to execute high-level language programs directly, although more slowly than compiled programs.
- Scripting languages such as JavaScript and PHP are processed by interpreters.
Performance Tip 1.1

Interpreters have an advantage over compilers in Internet scripting. An interpreted program can begin executing as soon as it’s downloaded to the client’s machine, without needing to be compiled before it can execute. On the downside, interpreted scripts generally run slower than compiled code.
1.6 The C Programming Language

- C evolved from two previous languages, BCPL and B.
- BCPL was developed in 1967 by Martin Richards as a language for writing operating-systems software and compilers.
- Ken Thompson modeled many features in his B language after their counterparts in BCPL, and in 1970 he used B to create early versions of the UNIX operating system at Bell Laboratories.
1.6 The C Programming Language (Cont.)

- The C language was evolved from B by Dennis Ritchie at Bell Laboratories and was originally implemented in 1972.
- C initially became widely known as the development language of the UNIX operating system.
- Many of today's leading operating systems are written in C and/or C++.
- C is mostly hardware independent.
- With careful design, it’s possible to write C programs that are portable to most computers.
1.6 The C Programming Language (Cont.)

*Built for Performance*

- C is widely used to develop systems that demand performance, such as operating systems, embedded systems, real-time systems and communications systems (Figure 1.5).
<table>
<thead>
<tr>
<th>Application</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating systems</td>
<td>C’s portability and performance make it desirable for implementing operating systems, such as Linux and portions of Microsoft’s Windows and Google’s Android. Apple’s OS X is built in Objective-C, which was derived from C. We discuss some key popular desktop/notebook operating systems and mobile operating systems in Section 1.12.</td>
</tr>
</tbody>
</table>

**Fig. 1.5** | Some popular performance-oriented C applications.
<table>
<thead>
<tr>
<th>Application</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real-time systems</td>
<td>Real-time systems are often used for “mission-critical” applications that require nearly instantaneous response times. For example, an air-traffic-control system must constantly monitor the positions and velocities of the planes and report that information to air-traffic controllers without delay so that they can alert the planes to change course if there’s a possibility of a collision.</td>
</tr>
<tr>
<td>Communications systems</td>
<td>Communications systems need to route massive amounts of data to their destinations quickly to ensure that things such as audio and video are delivered smoothly and without delay.</td>
</tr>
</tbody>
</table>

**Fig. 1.5** | Some popular performance-oriented C applications.
By the late 1970s, C had evolved into what is now referred to as “traditional C.” The publication in 1978 of Kernighan and Ritchie’s book, *The C Programming Language*, drew wide attention to the language.

**Standardization**

- The rapid expansion of C over various types of computers (sometimes called hardware platforms) led to many variations that were similar but often incompatible.
- In 1989, the C standard was approved; this standard was updated in 1999 and is often referred to as C99.
Portability Tip 1.1

Because C is a hardware-independent, widely available language, applications written in C often can run with little or no modification on a range of different computer systems.
1.7 C Standard Library

- As you’ll learn in Chapter 5, C programs consist of pieces called functions.
- You can program all the functions you need to form a C program, but most C programmers take advantage of the rich collection of existing functions called the C Standard Library.
1.7 C Standard Library (Cont.)

- Avoid reinventing the wheel.
- Instead, use existing pieces—this is called software reuse.
- When programming in C you’ll typically use the following building blocks:
  - C Standard Library functions
  - Functions you create yourself
  - Functions other people (whom you trust) have created and made available to you
1.7 C Standard Library (Cont.)

- The advantage of creating your own functions is that you’ll know exactly how they work. You’ll be able to examine the C code.
- The disadvantage is the time-consuming effort that goes into designing, developing and debugging new functions.
Performance Tip 1.2

Using Standard C library functions instead of writing your own comparable versions can improve program performance, because these functions are carefully written to perform efficiently.
Portability Tip 1.2

Using Standard C library functions instead of writing your own comparable versions can improve program portability, because these functions are used in virtually all Standard C implementations.
1.8 C++ and Other C-Based Languages

- C++ was developed by Bjarne Stroustrup at Bell Laboratories.
- It has its roots in C, providing a number of features that “spruce up” the C language.
- More important, it provides capabilities for object-oriented programming.
- Objects are essentially reusable software components that model items in the real world.
- Using a modular, object-oriented design and implementation approach can make software development groups more productive.
1.9 Object Technology

- *Objects*, or more precisely the *classes* objects come from, are essentially *reusable* software components.
- Almost any *noun* can be reasonably represented as a software object in terms of *attributes* (e.g., name, color and size) and *behaviors* (e.g., calculating, moving and communicating).
- Software developers are discovering that using a modular, object-oriented design and implementation approach can make software-development groups much more productive than was possible with earlier techniques—object-oriented programs are often easier to understand, correct and modify.
1.9 Object Technology

The Automobile as an Object

- Suppose you want to drive a car and make it go faster by pressing its accelerator pedal.
- Before you can drive a car, someone has to design it.
- A car typically begins as engineering drawings, similar to the blueprints that describe the design of a house.
- These drawings include the design for an accelerator pedal.
1.9 Object Technology (cont.)

- The pedal *hides* from the driver the complex mechanisms that actually make the car go faster, just as the brake pedal hides the mechanisms that slow the car, and the steering wheel *hides* the mechanisms that turn the car.

- This enables people with little or no knowledge of how engines, braking and steering mechanisms work to drive a car easily.

- Before you can drive a car, it must be *built* from the engineering drawings that describe it.

- A completed car has an *actual* accelerator pedal to make the car go faster, but even that’s not enough—the car won’t accelerate on its own (hopefully!), so the driver must *press* the pedal to accelerate the car.
1.10 Typical C Program Development Environment

- C systems generally consist of several parts: a program development environment, the language and the C Standard Library.
- C programs typically go through six phases to be executed (Fig. 1.7).
- These are: **edit, preprocess, compile, link, load and execute**.
1.10.1 Phase 1: Creating a Program

- Phase 1 consists of editing a file.
- This is accomplished with an editor program.
- Two editors widely used on Linux systems are `vi` and `emacs`.
- Software packages for the C/C++ integrated program development environments such as Eclipse and Microsoft Visual Studio have editors that are integrated into the programming environment.
- You type a C program with the editor, make corrections if necessary, then store the program on a secondary storage device such as a hard disk.
- C program file names should end with the `.c` extension.
1.10.2 Phases 2 and 3: Preprocessing and Compiling a C Program

- In Phase 2, you give the command to compile the program.
- The compiler translates the C program into machine language-code (also referred to as object code).
- In a C system, a preprocessor program executes automatically before the compiler’s translation phase begins.
- The C preprocessor obeys special commands called preprocessor directives, which indicate that certain manipulations are to be performed on the program before compilation.
1.10.2 Phases 2 and 3: Preprocessing and Compiling a C Program (Cont.)

- These manipulations usually consist of including other files in the file to be compiled and performing various text replacements.
- The most common preprocessor directives are discussed in the early chapters; a detailed discussion of preprocessor features appears in Chapter 13.
- In Phase 3, the compiler translates the C program into machine-language code.
- A syntax error occurs when the compiler cannot recognize a statement because it violates the rules of the language.
- Syntax errors are also called compile errors, or compile-time errors.
Fig. 1.7 | Typical C development environment. (Part 1 of 3.)
**Fig. 1.7**  |  Typical C development environment. (Part 2 of 3.)
Phase 6:
CPU takes each instruction and executes it, possibly storing new data values as the program executes.

**Fig. 1.7** | Typical C development environment. (Part 3 of 3.)
1.10.3 Phase 4: Linking

- The next phase is called **linking**.
- C programs typically contain references to functions defined elsewhere, such as in the standard libraries or in the private libraries of groups of programmers working on a particular project.
- The object code produced by the C compiler typically contains “holes” due to these missing parts.
- A **linker** links the object code with the code for the missing functions to produce an **executable image** (with no missing pieces).
- On a typical Linux system, the command to compile and link a program is called **gcc** (the GNU compiler).
1.10.4 Phase 5: Loading

- The next phase is called **loading**.
- Before a program can be executed, the program must first be placed in memory.
- This is done by the **loader**, which takes the executable image from disk and transfers it to memory.
- Additional components from shared libraries that support the program are also loaded.
1.10.5 Phase 6: Execution

Finally, the computer, under the control of its CPU, executes the program one instruction at a time.
1.10.6 Problems That May Occur at Execution Time

- Programs do not always work on the first try.
- Each of the preceding phases can fail because of various errors that we’ll discuss.
- For example, an executing program might attempt to divide by zero (an illegal operation on computers just as in arithmetic).
- This would cause the computer to display an error message.
- You would then return to the edit phase, make the necessary corrections and proceed through the remaining phases again to determine that the corrections work properly.
**Common Programming Error 1.1**

Errors such as division-by-zero occur as a program runs, so they are called runtime errors or execution-time errors. Divide-by-zero is generally a fatal error, i.e., one that causes the program to terminate immediately without successfully performing its job. Nonfatal errors allow programs to run to completion, often producing incorrect results.
1.12 Operating Systems

- **Operating systems** are software systems that make using computers more convenient for users, application developers and system administrators.
- Operating systems provide services that allow each application to execute safely, efficiently and *concurrently* (i.e., in parallel) with other applications.
- The software that contains the core components of the operating system is called the **kernel**.
- Popular desktop operating systems include Linux, Windows 7 and Mac OS X.
- Popular mobile operating systems used in smartphones and tablets include Google’s Android, Apple’s iOS (for iPhone, iPad and iPod Touch devices), BlackBerry OS and Windows Phone 7.
1.12.1 Windows—A Proprietary Operating System

- In the mid-1980s, Microsoft developed the Windows operating system, consisting of a graphical user interface built on top of DOS—an enormously popular personal-computer operating system of the time that users interacted with by typing commands.

- Windows borrowed from many concepts (such as icons, menus and windows) developed by Xerox PARC and popularized by early Apple Macintosh operating systems.

- Windows is a proprietary operating system—it’s controlled by Microsoft exclusively.
The Linux operating system is perhaps the greatest success of the open-source movement.

Open-source software departs from the proprietary software development style that dominated software's early years.

With open-source development, individuals and companies contribute their efforts in developing, maintaining and evolving software in exchange for the right to use that software for their own purposes, typically at no charge.

Rapid improvements to computing and communications, decreasing costs and open-source software have made it much easier and more economical to create a software-based business now than just a decade ago.

A great example is Facebook, which was launched from a college dorm room and built with open-source software.
1.12.2 Linux—An Open-Source Operating System

- The Linux kernel is the core of the most popular open-source, freely distributed, full-featured operating system.
- It’s developed by a loosely organized team of volunteers and is popular in servers, personal computers and embedded systems.
- Unlike that of proprietary operating systems like Microsoft’s Windows and Apple’s Mac OS X, Linux source code (the program code) is available to the public for examination and modification and is free to download and install.
- Linux has become extremely popular on servers and in embedded systems, such as Google’s Android-based smartphones.
1.12.3 Apple’s Mac OS X; Apple’s iOS for iPhone®, iPad® and iPod Touch® Devices

- In 1979, Steve Jobs and several Apple employees visited Xerox PARC (Palo Alto Research Center) to learn about Xerox’s desktop computer that featured a graphical user interface (GUI).
- That GUI served as the inspiration for the Apple Macintosh, launched with much fanfare in a memorable Super Bowl ad in 1984.
- The Objective-C programming language, created by Brad Cox and Tom Love at Stepstone in the early 1980s, added capabilities for object-oriented programming (OOP) to the C programming language.
1.12.3 Apple’s Mac OS X; Apple’s iOS for iPhone®, iPad® and iPod Touch® Devices

- Steve Jobs left Apple in 1985 and founded NeXT Inc. In 1988, NeXT licensed Objective-C from StepStone and developed an Objective-C compiler and libraries which were used as the platform for the NeXTSTEP operating system’s user interface and Interface Builder—used to construct graphical user interfaces.

- Jobs returned to Apple in 1996 when Apple bought NeXT. Apple’s Mac OS X operating system is a descendant of NeXTSTEP.

- Apple’s proprietary iPhone operating system, iOS, is derived from Apple’s Mac OS X and is used in the iPhone, iPad and iPod Touch devices.
1.12.4 Google’s Android

- **Android**—the fastest growing mobile and smartphone operating system—is based on the Linux kernel and Java.
- One benefit of developing Android apps is the openness of the platform. The operating system is open source and free.
- The Android operating system is used in numerous smartphones, e-reader devices, tablet computers, in-store touch-screen kiosks, cars, robots, multimedia players and more.
1.13 The Internet and the World Wide Web

- The Internet—a global network of computers—was made possible by the convergence of computing and communications technologies.
- In the late 1960s, ARPA (the Advanced Research Projects Agency) rolled out blueprints for networking the main computer systems of about a dozen ARPA-funded universities and research institutions.
- ARPA proceeded to implement the ARPANET, which eventually evolved into today’s Internet.
1.13 The Internet and the World Wide Web (Cont.)

**Packet Switching**

- A primary goal for ARPANET was to allow multiple users to send and receive information simultaneously over the same communications paths (e.g., phone lines).
- The network operated with a technique called packet switching, in which digital data was sent in small bundles called packets.
- The packets contained address, error-control and sequencing information.
  - The address information allowed packets to be routed to their destinations.
  - The sequencing information helped in reassembling the packets—which, because of complex routing mechanisms, could actually arrive out of order—into their original order for presentation to the recipient.
1.13 The Internet and the World Wide Web (Cont.)

- The network was designed to operate without centralized control.
- If a portion of the network failed, the remaining working portions would still route packets from senders to receivers over alternative paths for reliability.

**TCP/IP**

- The protocol (i.e., set of rules) for communicating over the ARPANET became known as TCP—the Transmission Control Protocol.
- TCP ensured that messages were properly routed from sender to receiver and that they arrived intact.
As the Internet evolved, organizations worldwide were implementing their own networks. One challenge was to get these different networks to communicate.

ARPA accomplished this with the development of IP—the Internet Protocol, truly creating a network of networks, the current architecture of the Internet.

The combined set of protocols is now commonly called TCP/IP.
1.13 The Internet and the World Wide Web (Cont.)

**World Wide Web, HTML, HTTP**

- The **World Wide Web** allows you to execute web-based applications and to locate and view multimedia-based documents on almost any subject over the Internet.
- In 1989, Tim Berners-Lee of CERN (the European Organization for Nuclear Research) began to develop a technology for sharing information via hyperlinked text documents. Berners-Lee called his invention the **HyperText Markup Language (HTML)**.
- He also wrote communication protocols to form the backbone of his new information system, which he called the World Wide Web.
- In particular, he wrote the **Hypertext Transfer Protocol (HTTP)**—a communications protocol used to send information over the web.
1.13 The Internet and the World Wide Web (Cont.)

- The **URL (Uniform Resource Locator)** specifies the address (i.e., location) of the web page displayed in the browser window.
- Each web page on the Internet is associated with a unique URL.
- **Hypertext Transfer Protocol Secure (HTTPS)** is the standard for transferring encrypted data on the web.
<table>
<thead>
<tr>
<th>Version</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha</td>
<td>An <em>alpha</em> version is the earliest release of a software product that’s still under active development. Alpha versions are often buggy, incomplete and unstable and are released to a relatively small number of developers for testing new features, getting early feedback, etc.</td>
</tr>
<tr>
<td>Beta</td>
<td><em>Beta</em> versions are released to a larger number of developers later in the development process after most major bugs have been fixed and new features are nearly complete. Beta software is more stable, but still subject to change.</td>
</tr>
<tr>
<td>Release candidates</td>
<td><em>Release candidates</em> are generally <em>feature complete</em> and (supposedly) bug free and ready for use by the community, which provides a diverse testing environment—the software is used on different systems, with varying constraints and for a variety of purposes. Any bugs that appear are corrected, and eventually the final product is released to the general public. Software companies often distribute incremental updates over the Internet.</td>
</tr>
<tr>
<td>Continuous beta</td>
<td>Software that’s developed using this approach generally does not have version numbers (for example, Google search or Gmail). The software, which is hosted in the cloud (not installed on your computer), is constantly evolving so that users always have the latest version.</td>
</tr>
</tbody>
</table>

*Fig. 1.32 | Software product-release terminology.*
1.14 Keeping Up-to-Date with Information Technologies

- Figure 1.33 lists key technical and business publications that will help you stay up-to-date with the latest news and trends in technology.
<table>
<thead>
<tr>
<th>Publication</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACM TechNews</td>
<td>technews.acm.org/</td>
</tr>
<tr>
<td>ACM Transactions on Accessible Computing</td>
<td><a href="http://www.gccis.rit.edu/taccess/index.html">www.gccis.rit.edu/taccess/index.html</a></td>
</tr>
<tr>
<td>ACM Transactions on Internet Technology</td>
<td>toit.acm.org/</td>
</tr>
<tr>
<td>Bloomberg BusinessWeek</td>
<td><a href="http://www.businessweek.com">www.businessweek.com</a></td>
</tr>
<tr>
<td>CNET</td>
<td>news.cnet.com</td>
</tr>
<tr>
<td>Communications of the ACM</td>
<td>cacm.acm.org/</td>
</tr>
<tr>
<td>Computer World</td>
<td><a href="http://www.computerworld.com">www.computerworld.com</a></td>
</tr>
<tr>
<td>Engadget</td>
<td><a href="http://www.engadget.com">www.engadget.com</a></td>
</tr>
<tr>
<td>eWeek</td>
<td><a href="http://www.eweek.com">www.eweek.com</a></td>
</tr>
<tr>
<td>Fast Company</td>
<td><a href="http://www.fastcompany.com/">www.fastcompany.com/</a></td>
</tr>
<tr>
<td>Fortune</td>
<td>money.cnn.com/magazines/fortune/</td>
</tr>
<tr>
<td>IEEE Computer</td>
<td><a href="http://www.computer.org/portal/web/computer">www.computer.org/portal/web/computer</a></td>
</tr>
</tbody>
</table>

**Fig. 1.33** | Technical and business publications. (Part 1 of 2.)
<table>
<thead>
<tr>
<th>Publication</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>IEEE Internet Computing</td>
<td><a href="http://www.computer.org/portal/web/internet/home">www.computer.org/portal/web/internet/home</a></td>
</tr>
<tr>
<td>InfoWorld</td>
<td><a href="http://www.infoworld.com">www.infoworld.com</a></td>
</tr>
<tr>
<td>Mashable</td>
<td><a href="http://mashable.com">mashable.com</a></td>
</tr>
<tr>
<td>PCWorld</td>
<td><a href="http://www.pcworld.com">www.pcworld.com</a></td>
</tr>
<tr>
<td>SD Times</td>
<td><a href="http://www.sdtimes.com">www.sdtimes.com</a></td>
</tr>
<tr>
<td>Slashdot</td>
<td><a href="http://slashdot.org/">slashdot.org/</a></td>
</tr>
<tr>
<td>Smarter Technology</td>
<td><a href="http://www.smartertechnology.com">www.smartertechnology.com</a></td>
</tr>
<tr>
<td>Technology Review</td>
<td><a href="http://technologyreview.com">technologyreview.com</a></td>
</tr>
<tr>
<td>Techcrunch</td>
<td><a href="http://techcrunch.com">techcrunch.com</a></td>
</tr>
<tr>
<td>Wired</td>
<td><a href="http://www.wired.com">www.wired.com</a></td>
</tr>
</tbody>
</table>

**Fig. 1.33** | Technical and business publications. (Part 2 of 2.)
/* Fig. 2.1: fig02_01.c */
A first program in C */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
    printf( "Welcome to C!\n" );

    return 0; /* indicate that program ended successfully */
}

/* end function main */

Welcome to C!
2.2 A Simple C Program: Printing a Line of Text

Comments

- Text surrounded by /* and */ is ignored by computer
- Used to describe program

```
#include <stdio.h>
```

- Preprocessor directive
  - Tells computer to load contents of a certain file
- `<stdio.h>` allows standard input/output operations
Common Programming Error

2.1

- Forgetting to terminate a comment with */.
Starting a comment with the characters */ or ending a comment with the characters */.
2.2 A Simple C Program: Printing a Line of Text

- `int main()`
  - C++ programs contain one or more functions, exactly one of which must be `main`
  - Parenthesis used to indicate a function
  - `int` means that `main` "returns" an integer value
  - Braces ({ and }) indicate a block
    - The bodies of all functions must be contained in braces
Good Programming Practice

2.1

• Every function should be preceded by a comment describing the purpose of the function.
2.2 A Simple C Program: Printing a Line of Text

- `printf( "Welcome to C!\n" );`
  - Instructs computer to perform an action
    - Specifically, prints the string of characters within quotes (" ")
  - Entire line called a statement
    - All statements must end with a semicolon (;)
  - Escape character (\)
    - Indicates that printf should do something out of the ordinary
    - \n is the newline character
<table>
<thead>
<tr>
<th>Escape sequence</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\n</td>
<td>Newline. Position the cursor at the beginning of the next line.</td>
</tr>
<tr>
<td>\t</td>
<td>Horizontal tab. Move the cursor to the next tab stop.</td>
</tr>
<tr>
<td>\a</td>
<td>Alert. Sound the system bell.</td>
</tr>
<tr>
<td>\   \</td>
<td>Backslash. Insert a backslash character in a string.</td>
</tr>
<tr>
<td>&quot;</td>
<td>Double quote. Insert a double-quote character in a string.</td>
</tr>
</tbody>
</table>

**Fig. 2.2** | Some common escape sequences.
• Typing the name of the output function `printf` as `print` in a program.
2.2 A Simple C Program: Printing a Line of Text

- `return 0;`
  - A way to exit a function
  - `return 0`, in this case, means that the program terminated normally
- Right brace `}
  - Indicates end of `main` has been reached
- Linker
  - When a function is called, linker locates it in the library
  - Inserts it into object program
  - If function name is misspelled, the linker will produce an error because it will not be able to find function in the library
• Add a comment to the line containing the right brace, }, that closes every function, including main.
• The last character printed by a function that displays output should be a newline (\n). This ensures that the function will leave the screen cursor positioned at the beginning of a new line. Conventions of this nature encourage software reusability—a key goal in software development environments.
2.4

- Indent the entire body of each function one level of indentation (we recommend three spaces) within the braces that define the body of the function. This indentation emphasizes the functional structure of programs and helps make programs easier to read.
Set a convention for the size of indent you prefer and then uniformly apply that convention. The tab key may be used to create indents, but tab stops may vary. We recommend using three spaces per level of indent.
/* Fig. 2.3: fig02_03.c */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
    printf( "Welcome ");
    printf( "to C!\n" );
    return 0; /* indicate that program ended successfully */
}

/* end function main */

welcome to C!
/* Fig. 2.4: fig02_04.c 
Printing multiple lines with a single printf */
#include <stdio.h>

/* function main begins program execution */
int main( void )
{
    printf( "Welcome\nto\nC!\n" );
    return 0; /* indicate that program ended successfully */
}

/* end function main */

Welcome
to
C!