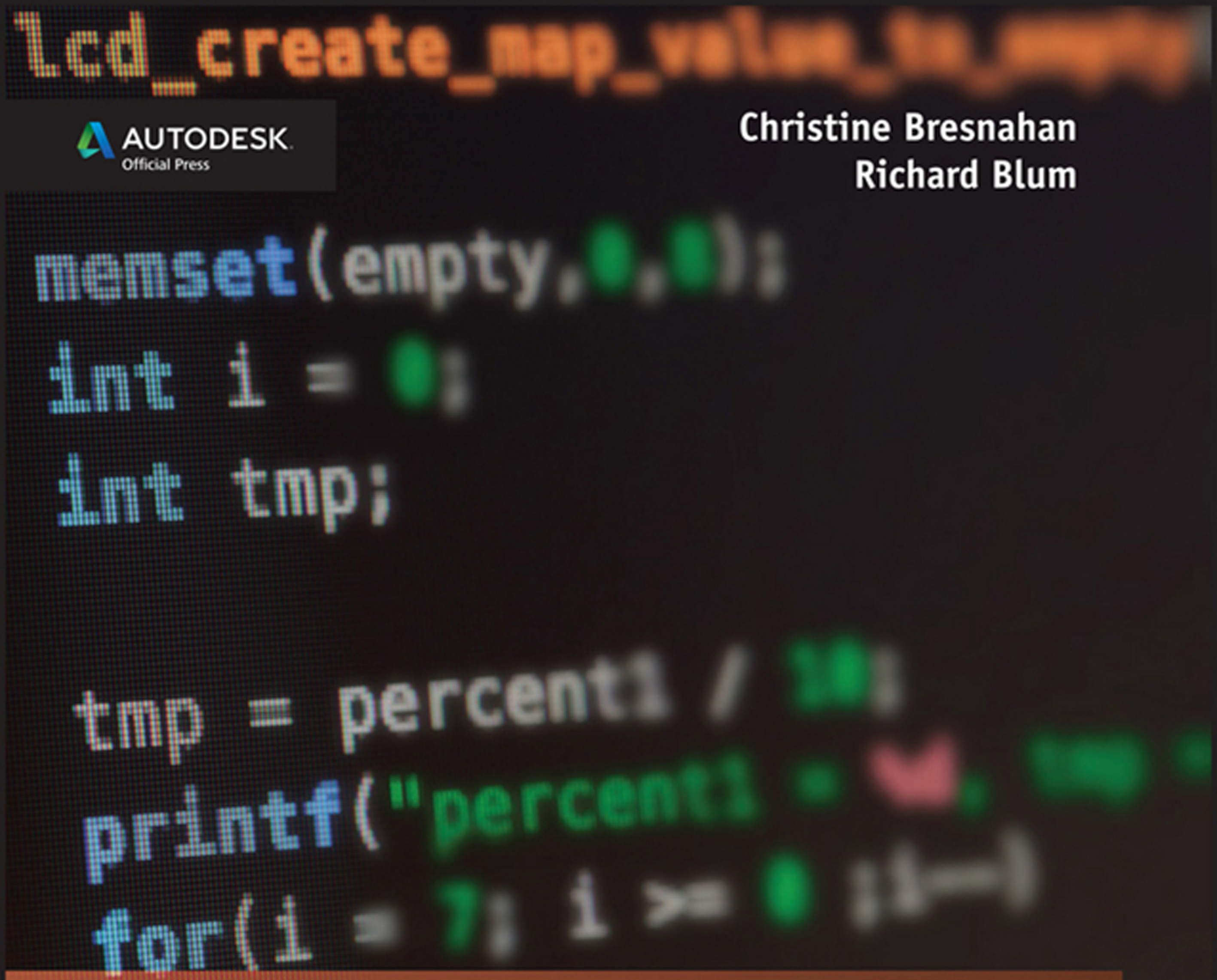


```
lcd_create_map_value_to_[illegible]
memset(empty,0,0);
int i = 0;
int tmp;

tmp = percent1 / 10;
printf("percent1 = %d, tmp =
for(i = 7; i >= 0 ;i--)
```

AUTODESK
Official Press

Christine Bresnahan
Richard Blum

Linux

ESSENTIALS

Second Edition

SYBEX
A Wiley Brand

LINUX®

ESSENTIALS

Second Edition

Christine Bresnahan
Richard Blum

SYBEX®
A Wiley Brand

Senior Acquisitions Editor: Kenyon Brown
Development Editor: Gary Schwartz
Technical Editor: Kevin E. Ryan
Production Editor: Christine O'Connor
Copy Editor: Sharon Wilkey
Editorial Manager: Mary Beth Wakefield
Production Manager: Kathleen Wisor
Associate Publisher: Jim Minatel
Book Designer: Happenstance Type-O-Rama
Proofreader: Sarah Kaikini, Word One New York
Indexer: Robert Swanson
Project Coordinator, Cover: Brent Savage
Cover Designer: Wiley
Cover Image: Daniel Anez, theanez.com

Published simultaneously in Canada
ISBN: 978-1-119-09206-3
ISBN: 978-1-119-09219-3 (ebk.)
ISBN: 978-1-119-09214-8 (ebk.)

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or Web site may provide or recommendations it may make. Further, readers should be aware that Internet Web sites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at (877) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at `http://booksupport.wiley.com`. For more information about Wiley products, visit `www.wiley.com`.

Library of Congress Control Number: 2015947137

10 9 8 7 6 5 4 3 2 1

To those looking to further their knowledge of Linux.
Let the journey begin!
“A wise man is full of strength, and a man of knowledge
enhances his might” Prov 24:5 (ESV)

Acknowledgments

First, all glory and praise go to God, who through His Son, Jesus Christ, makes all things possible and gives us the gift of eternal life.

Many thanks go to the fantastic team at Sybex for their outstanding work on this project. Thanks to acquisitions editor Kenyon Brown for offering us the opportunity to work on this book. Also thanks to developmental editor Gary Schwartz for keeping things on track and making this book more presentable. Thanks, Gary, for all your hard work and diligence. Technical editor Kevin E. Ryan did a wonderful job of double-checking all of the work in the book, in addition to making suggestions to improve the content. Thanks also goes to the young and talented Daniel Anez (theanez.com) for his illustration and cover image work. We would also like to thank Carole Jelen at Waterside Productions for arranging this opportunity for us and for helping us in our writing careers.

Christine would particularly like to thank her husband, Timothy, for his encouragement, patience, and willingness to listen, even when he has no idea what she is talking about.

Rich would like to thank his wife, Barbara, for her love and support of his late-night writing habit, and thank Christine and Timothy for their continued friendship all these years.

About the Authors

Christine Bresnahan, CompTIA Linux+, LPIC-1, Linux Essentials, started working with computers more than 30 years ago in the IT industry as a systems administrator. Christine is an adjunct professor at Ivy Tech Community College, where she teaches Linux certification and Python programming classes. She also writes books and produces instructional resources for the classroom.

Richard Blum, CompTIA Linux+, LPIC-1, Linux Essentials, has worked in the IT industry for more than 25 years as both a systems and network administrator, and he has published numerous Linux and open source books. Rich is an online instructor for Linux and web programming courses that are used by colleges and universities across the United States. When he is not being a computer nerd, Rich enjoys spending time with his wife, Barbara, and his two daughters, Katie Jane and Jessica.

Contents at a Glance

Contents

Introduction

This book you hold in your hands provides a solid introduction to the Linux operating system. As its title suggests, it will give you the essential knowledge to begin using and managing this powerful operating system (OS), which is an important one in today's computing world.

The following pages explain why you should care about Linux, describe the purpose of the Linux Professional Institute's Linux Essentials certification, offer advice on who should buy this book, detail how the book is organized, and explain some of the typographical and organizational elements within the book's chapters.

What Is Linux?

Linux is a clone of the Unix OS, which has been popular in academic and business environments for years. Linux consists of a *kernel*, which is the core control software, and many libraries and utilities that rely on the kernel to provide features with which users interact. The OS is available in many *distributions*, which are collections of a specific kernel with specific support programs. Popular Linux distributions include Arch, CentOS, Debian, Fedora, Gentoo, Mandriva, openSUSE, Red Hat, Slackware, SUSE Enterprise, and Ubuntu, but there are hundreds—if not thousands—of other Linux distributions. This book focuses on tools and techniques that are used in most, if not all, distributions, although from time to time it demonstrates some distribution-specific tools.

Linux has several characteristics that make it worth learning and using:

- Linux is open source software: the files used to create the working programs that make up Linux are freely available and may be modified and redistributed. If you dislike something about the way Linux works, you can change it yourself! (You may need modest-to-considerable programming skills to do so, however.)
- Linux is available free of charge. Although some distributions require payment, most can be downloaded from the Internet and used without paying a cent. This is a great boon for students, businesses on a shoestring budget, or anybody wanting to save money. Those who want to pay for greater support can do so by hiring consultants or by purchasing service contracts.

- As a clone of the older Unix OS, Linux has inherited a great deal of Unix software, including many important Internet server programs, databases, programming languages, and more.
- Linux is highly scalable—it runs on everything from mobile devices to supercomputers. The Linux versions described in the greatest detail in this book run on a more limited range of hardware, but they can run on systems that are several years old or on the very latest hardware. Linux can make good use of systems that are too old for the latest version of Windows or Mac OS X.
- Many businesses and nonprofit organizations rely on Linux. Although desktop systems still usually run Windows, Linux is often used to run the organizations' websites, route their Internet traffic, and do other critical behind-the-scenes tasks. In some cases, Linux is used as the desktop OS too. Thus learning Linux will help your employment prospects.

You can install Linux on almost any system on which you normally run Windows or Mac OS X. You can install Linux by itself or side by side with another OS, so you can learn Linux without losing your ability to get work done in your regular OS.

What Is the Linux Essentials Certification?

The Linux Professional Institute, or LPI (`www.lpi.org`), offers a series of Linux certifications. These certifications aim to provide proof of skill levels for employers; if you've passed a particular certification, you should be competent to perform certain tasks on Linux computers. The LPI exams include Linux Essentials, LPIC-1, LPIC-2, and LPIC-3. As the name implies, the Linux Essentials exam is the lowest level of the four exams, covering the most basic tasks of using and administering a Linux computer. Its specific objectives can be found at `http://wiki.lpi.org/wiki/LinuxEssentials`. This book covers all of these topics, although not in the exact order in which they appear on the LPI website.

Who Should Read This Book?

You may have been assigned this book for a class that you're taking, but if not, it can still have value for self-study or as a supplement to other resources. If you're new to Linux, this book covers the material that you will need to learn the OS from the beginning. You can pick up this book and learn from it even if you've never used Linux before. If you're already familiar with Linux, you'll have a leg up on many of the topics described in these pages.

This book is written with the assumption that you know at least a little about computers generally, such as how to use a keyboard, how to insert a disc into an optical drive, and so on. Chances are that you have used computers in a substantial way in the past—perhaps even Linux, as an ordinary user—or maybe you have used Windows or Mac OS X. We do *not* assume that you have knowledge of Linux system administration.

System Requirements

As a practical matter, you'll need a Linux system on which to practice and learn in a hands-on way. You can install Linux in several ways:

- Alone as the only OS on the computer
- Side by side with another OS
- In an emulated computer environment provided by a virtualization program such as VMware (`www.vmware.com`) or VirtualBox (`www.virtualbox.org`)

If you're taking a course on Linux, you may be able to use Linux in a lab environment. However, if you're using this book in a self-study manner, you should plan to install Linux yourself. Although you can learn something just by reading this book, no amount of reading can substitute for hands-on experience with Linux!

You can use any popular Linux distribution with this book, although if you're new to Linux, you'll probably be happiest with one of the more user-friendly distributions, such as CentOS, Fedora, openSUSE, or Ubuntu. This book does not include instructions for how to install Linux; you should consult distribution-specific documentation to help with this task.

To install Linux and use all of its GUI tools, your computer should meet the following requirements:

CPU 400 MHz Pentium Pro or better

Minimum RAM 640 MiB

Recommended RAM At least 1,152 MiB

Hard Disk Space At least 9 GiB in unpartitioned space

Some distributions can work on less-powerful computers than these, and others may require better hardware to take full advantage of all of their features. Consult your distribution's documentation to fine-tune these requirements.

How This Book Is Organized

This book consists of 15 chapters plus this introduction. The chapters are organized as follows:

Chapter 1: Selecting an Operating System This chapter provides a birds-eye view of the world of operating systems. The chapter will help you understand exactly what Linux is and the situations in which you might want to use it.

Chapter 2: Understanding Software Licensing This chapter describes copyright law and the licenses that both Linux and non-Linux OSs use to expand or restrict users' rights to use and copy software.

Chapter 3: Investigating Linux's Principles and Philosophy This chapter covers Linux's history and the ways in which Linux, and other OSs, are commonly used.

Chapter 4: Using Common Linux Programs This chapter covers the major categories of Linux software, and it provides pointers to some of the most popular Linux programs.

Chapter 5: Managing Hardware This chapter provides advice on how to select and use hardware in Linux. Specific topics range from the central processing unit (CPU) to device drivers.

Chapter 6: Getting to Know the Command Line This chapter tackles using typed commands to control Linux. Although many new users find this topic intimidating, command-line control of Linux is important.

Chapter 7: Managing Files This chapter describes how to move, rename, delete, and edit files. Directories are just a special type of file, so they are covered here as well.

Chapter 8: Searching, Extracting, and Archiving Data This chapter summarizes the tools that you can use to find data on your computer, as well as how you can manipulate data archive files for data transport and backup purposes.

Chapter 9: Exploring Processes and Process Data This chapter describes how to install programs in Linux and how to adjust the priority of running programs or terminate selected programs.

Chapter 10: Editing Files This chapter introduces the topic of editing text files. This includes the basic features of the `pico`, `nano`, and `vi` text-mode text editors, as well as some common configuration file and formatted text file conventions.

Chapter 11: Creating Scripts This chapter describes how to create simple scripts, which are programs that can run other programs. You can use scripts to help automate otherwise tedious manual tasks, thus improving your productivity.

Chapter 12: Understanding Basic Security This chapter introduces the concepts that are critical to understanding Linux's multiuser nature. It also covers the `root` account, which Linux uses for most administrative tasks.

Chapter 13: Creating Users and Groups This chapter covers the software and procedures you use to create, modify, and delete accounts and groups, which define who may use the computer.

Chapter 14: Setting Ownership and Permissions This chapter describes how to control which users may access files and in what ways they may do so. In conjunction with users and groups, ownership and permissions control your computer's security.

Chapter 15: Managing Network Connections This chapter covers the critical topic of telling Linux how to use a network, including testing the connection and some basic network security measures.

Broadly speaking, the chapters are arranged in order of increasing complexity in terms of the tasks and systems described. The book begins with background information on Linux and the philosophies that drive its development. Subsequent chapters describe basic user tasks, such as moving files around. The book concludes with the tasks that are of most interest to system administrators, such as account management and network configuration.

Each chapter begins with a list of the topics covered in that chapter. At the end of each chapter, you'll find a few elements that summarize the material and encourage you to go further:

The Essentials and Beyond This is a one-paragraph summary of the material covered in the chapter. If something sounds unfamiliar when you read it, go back and review the relevant section of the chapter!

Suggested Exercises Each chapter includes two to four exercises that you should perform to give yourself more hands-on experience with Linux. These exercises do not necessarily have "correct" answers; instead, they're intended to promote exploration and discovery of your own computer and of Linux.

Review Questions Each chapter concludes with a series of 10 review questions, in multiple-choice, true/false, or fill-in-the-blank format. (Answers to review questions appear in Appendix A.) These questions can help you test your knowledge and prepare you for the Linux Essentials exam. Note, however, that these questions are *not* taken from LPI's exam. You should *not* memorize the answers to these questions and assume that doing so will enable you to pass the exam. Instead, study the text of the book and *use Linux*.

To get the most out of this book, you should read each chapter from start to finish, perform the suggested exercises, and answer the review questions. Even if you're already familiar with a topic, you should skim the chapter; Linux is complex enough that there are often multiple ways to accomplish a task; you may learn something even if you're already competent in a given area.

Conventions Used in This Book

This book uses certain typographic styles in order to help you quickly identify important information and to avoid confusion over the meaning of words such as onscreen prompts. In particular, look for the following styles:

- *Italicized text* indicates key terms that are described or defined for the first time in a chapter. (Italics are also used for emphasis.)
- A `monospaced font` indicates the contents of configuration files, messages displayed at a text-mode Linux shell prompt, filenames, text-mode command names, and Internet URLs.
- *`Italicized monospaced text`* indicates a variable—information that differs from one system or command run to another, such as the name of a client computer or the name of a user's data file.
- **`Bold monospaced text`** is information that you're to type into the computer, usually at a Linux shell prompt. This text can also be italicized to indicate that you should substitute an appropriate value for your system. When isolated on their own lines, commands are preceded by nonbold monospaced `$` or `#` command prompts, denoting regular user or system administrator use, respectively.

In addition to these text conventions, which can apply to individual words or to entire paragraphs, a few conventions highlight segments of text.

A margin note identifies additional information that may be relevant to the principal point of the accompanying paragraph, but that isn't critical to its basic understanding. This could be a cross-reference to information in another chapter, an interesting but noncritical minor fact, or a warning about a rare pitfall of a procedure.

As a general rule, margin notes are best read after the paragraphs to which they refer.

Sidebars

A *sidebar* is an extended description of a topic that's related to the main text but that doesn't fit neatly into the flow of the surrounding paragraphs. It may expand on a point to provide added context or suggest an alternative way of doing things from the method emphasized in the main text.

Many chapters of this book describe both GUI and text-mode methods of accomplishing tasks. Because you're likely to be more familiar with GUI tools, most chapters begin with them; however, in most cases, Linux's text-mode tools are more powerful. Furthermore, the Linux Essentials certification covers mainly text-mode tools. Therefore, be sure to learn the text-mode tools. As you gain proficiency with Linux, you're likely to find yourself using the text-mode tools more often than the GUI tools because of the added flexibility that the text-mode tools provide. Furthermore, the GUI tools tend to vary a lot between distributions, whereas the text-mode tools vary much less.

CHAPTER 1

Selecting an Operating System

The fact that you're reading this book means you want to learn about the Linux operating system (OS). To begin this journey, you must first understand what an OS is and what type of OS Linux is. This chapter is therefore devoted to these basic issues.

In this chapter, we describe what an OS is, how users interact with an OS, how Linux compares with other OSs with which you may be familiar, and how specific Linux implementations vary. Understanding these issues will help you find your way as you learn about Linux and switch between Linux-based and other systems.

- **What is an OS?**
- **Investigating user interfaces**
- **Where does Linux fit in the OS world?**
- **What is a distribution?**

What Is an OS?

An *operating system*, or OS, provides all of the most fundamental features of a computer, at least from a software point of view. An OS enables you to use the computer's hardware devices, defines the user interface standards, and provides the basic tools that begin to make the computer useful. Ultimately, many of these features trace their way back to the OS's kernel, which is described in more detail next. Other OS features are owed to additional programs that run atop the kernel, as described later in this chapter.

What Is a Kernel?

An OS *kernel* is a software component that's responsible for managing various low-level features of the computer, including the following:

- Interfacing with hardware devices (network adapters, hard drives, and so on)
- Allocating memory to individual programs
- Allocating CPU time to individual programs
- Enabling programs to interact with each other

When you use a program (say, a web browser), it relies on the kernel for many of its basic functions. The web browser can communicate with the outside world only by using network functions provided by the kernel. The kernel allocates memory and CPU time to the web browser, without which it couldn't run. The web browser may rely on plug-ins to display multimedia content; such programs are launched by and interact with the web browser through kernel services. Similar comments apply to any program that you run on a computer, although the details vary from one OS to another and from one program to another.

In sum, the kernel is the software "glue" that holds the computer together. Without a kernel, a modern computer can do very little.

Kernels are not interchangeable; the Linux kernel is different from the Mac OS X kernel or the Windows kernel. Each of these kernels uses a different internal design and provides different software interfaces for programs to use. Thus, each OS is built from the kernel up and uses its own set of programs that further define each OS's features.

Many programs run on multiple kernels, but most need OS-specific tweaks. Programmers create *binaries*—the program files for a particular processor and kernel—for each OS.

Linux uses a kernel called *Linux*—in fact, technically speaking, the word Linux refers *only* to the kernel. Nonkernel programs provide other features that you might associate with Linux, most of which are available on other platforms, as described next in "What Else Identifies an OS."

A student named Linus Torvalds created the Linux kernel in 1991. Linux has evolved considerably since that time. Today it runs on a wide variety of CPUs and other hardware. The easiest way to learn about Linux is to use it on a desktop or laptop PC, so that's the type of configuration that's emphasized in this book. The Linux kernel, however, runs on everything from tiny cell phones to powerful supercomputers.

What Else Identifies an OS?

Certification Objective

The kernel is at the core of any OS, but it's a component that most users don't directly manipulate. Instead, most users interact with a number of other software components, many of which are closely associated with particular OSs. Such programs include the following:

Command-Line Shells Years ago, users interacted with computers exclusively by typing commands in a program (known as a *shell*) that accepted such commands. The commands would rename files, launch programs, and so on. Although many computer users today don't use text-mode shells, they're still important for intermediate and advanced Linux users, so we describe them in more detail in Chapter 6, "Getting to Know the Command Line," and subsequent chapters rely heavily on your ability to use a text-mode shell. Many shells are available, and which shells are available and popular vary from one OS to another. In Linux, a shell known as the Bourne Again Shell (`bash` or Bash) is popular.

Certification Objective

Graphical User Interfaces A graphical user interface (GUI) is an improvement on a text-mode shell, at least from the perspective of a beginning user. Instead of typing in commands, GUIs rely on icons, menus, and a mouse pointer. The Windows and Mac OS both have their own OS-specific GUIs. Linux relies on a GUI known as the X Window System, or X for short. X is a basic GUI, so Linux also uses *desktop environment* program suites, such as GNOME or the K Desktop Environment (KDE), to provide a more complete user experience. It's the differences among Linux desktop environments and the GUIs in Windows or OS X that will probably strike you most when you first begin using Linux.

You can search for Linux equivalents to popular OS X or Windows programs on websites such as `www.linuxrsp.ru/win-lin-soft/table-eng` or `www.linuxalt.com`.

Utility Programs Modern OSs invariably ship with a wide variety of simple utility programs—calculators, calendars, text editors, disk maintenance tools, and so on. These programs differ from one OS to another. Indeed, even the names and methods of launching these programs can differ between OSs. Fortunately, you can usually find the programs you want by perusing menus in the main desktop environment.

In addition to software that runs on an OS, several other features can distinguish between OSs, such as the details of user accounts, rules for naming disk files, and technical details of how the computer starts up. These features are all controlled by software that's part of the OS, of course—sometimes by the kernel and sometimes by nonkernel software.

Libraries Unless you're a programmer, you're unlikely to need to work with libraries directly; nonetheless, we include them in this list because they provide critical services to programs. Libraries are collections of programming functions that can be used by a variety of programs. In Linux, for instance, most programs rely on a library called `libc`. Other libraries provide features associated with the GUI or that help programs parse options passed to them on the command line. Many libraries exist for Linux, which helps enrich the Linux software landscape.

Productivity Programs Major productivity programs—web browsers, word processors, graphics editors, and so on—are the usual reason for using a computer. Although such programs are often technically separate from the OS, they are sometimes associated with certain OSs. Even when a program is available on many OSs, it may have a different "feel" on each OS because of the different GUIs and other OS-specific features.

Investigating User Interfaces

Earlier, we noted the distinction between text-mode and graphical user interfaces. Although most end users favor GUIs because of their ease of use, Linux retains a strong text-mode tradition. Chapter 6 describes Linux's text-mode tools in more detail, and Chapter 4, "Using Common Linux Programs," covers basic principles of Linux GUI operations. It's important that you have some grounding in the basic principles of both text-mode and graphical user interfaces now, since user interface issues crop up from time to time in intervening chapters.

Using a Text-Mode User Interface

Certification Objective

In the past, and even sometimes today, Linux computers booted in text mode. Once the system had completely booted, the screen would display a simple text-mode login prompt, which might look like this:

```
Fedora release 21 (Twenty One)
Kernel 3.18.6-200.fc21.x86_64 on an x86_64 (tty1)

essentials login:
```

The details of such a login prompt vary from one system to another. This example includes several pieces of information:

- The OS name and version—Fedora Linux 21
- The computer's name—`essentials`
- The name of the hardware device being used for the login—`tty1`
- The login prompt itself—`login:`

To try a text-mode login, you must first install Linux on a computer. Neither the Linux Essentials exam nor this book covers Linux installation. Consult your distribution's documentation to learn more about installing Linux.

To log in to such a system, you must type your username at the `login:` prompt. The system then prompts you for a password, which you must also type. If you entered a valid username and password, the computer is likely to display a login message, followed by a shell prompt:

```
[rich@essentials:~]$
```

If you see a GUI login prompt, you can obtain a text-mode prompt by pressing Ctrl+Alt+F1 or Ctrl+Alt+F2. To return to the GUI login prompt, press Alt+F1 or Alt+F7.

In this book, we omit most of the prompts from example commands when they appear on their own lines. We keep the dollar sign ($) prompt, though, for ordinary user commands. Some commands must be entered as `root`, which is the Linux administrative user. We change the prompt to a hash mark (#) for such commands, since most Linux distributions make a similar change to their prompts for the `root` user.

The details of this shell prompt vary from one installation to another, but you can type text-mode commands at the shell prompt. For instance, you could type `ls` (short for *list*) to see a list of files in the current directory. Removing vowels, and sometimes consonants, shortens the most basic commands, in order to minimize the amount of typing required to execute a command. This has the unfortunate effect of making many commands rather obscure.

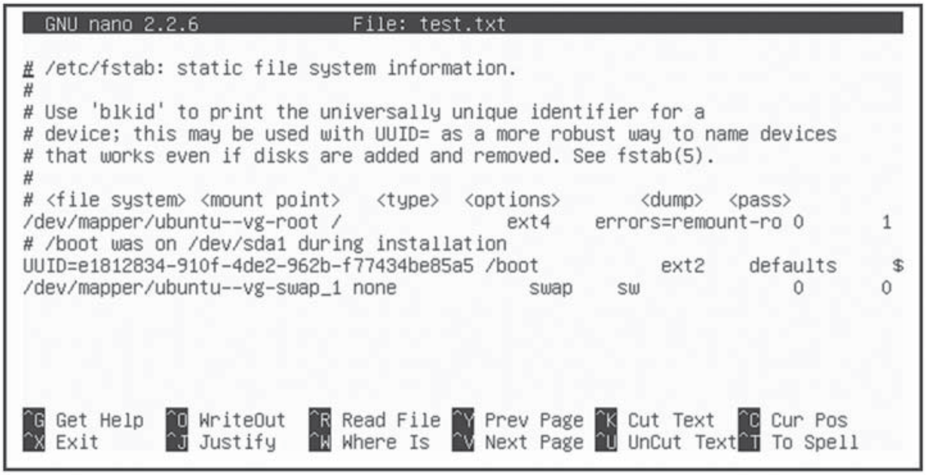
Chapter 13, "Creating Users and Groups," describes Linux accounts, including the root account, in more detail.

Some commands display no information, but most produce some type of output. For instance, the `ls` command produces a list of files:

```
$ ls
106792c01.doc  f0101.tif
```

This example shows two files in the current directory: `106792c01.doc` and `f0101.tif`. You can use additional commands to manipulate these files, such as `cp` to copy them or `rm` to remove (delete) them. Chapter 6 and Chapter 7 ("Managing Files") describe some common file-manipulation commands.

Some text-mode programs take over the display in order to provide constant updates or to enable you to interact with data in a flexible manner. Figure 1.1, for instance, shows the `nano` text editor, which is described in more detail in Chapter 10, "Editing Files." Once `nano` is working, you can use your keyboard's arrow keys to move the cursor around, add text by typing, and so on.



```
  GNU nano 2.2.6              File: test.txt

# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
/dev/mapper/ubuntu--vg-root /               ext4    errors=remount-ro 0       1
# /boot was on /dev/sda1 during installation
UUID=e1812834-910f-4de2-962b-f77434be85a5 /boot           ext2    defaults        $
/dev/mapper/ubuntu--vg-swap_1 none            swap    sw              0       0



^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

FIGURE 1.1 Some text-mode programs take over the entire display.

Even if you use a graphical login, you can use a text-mode shell inside a window, known as a *terminal*. Common Linux GUIs provide the ability to launch a terminal program, which delivers a shell prompt and the means to run text-mode programs.

Using a Graphical User Interface

Certification Objective

Most users are more comfortable with GUIs than with text-mode commands. Thus many modern Linux systems start up in GUI mode by default, presenting a login screen similar to the one shown in Figure 1.2. You can select your username from a list or type it, followed by typing your password, to log in.

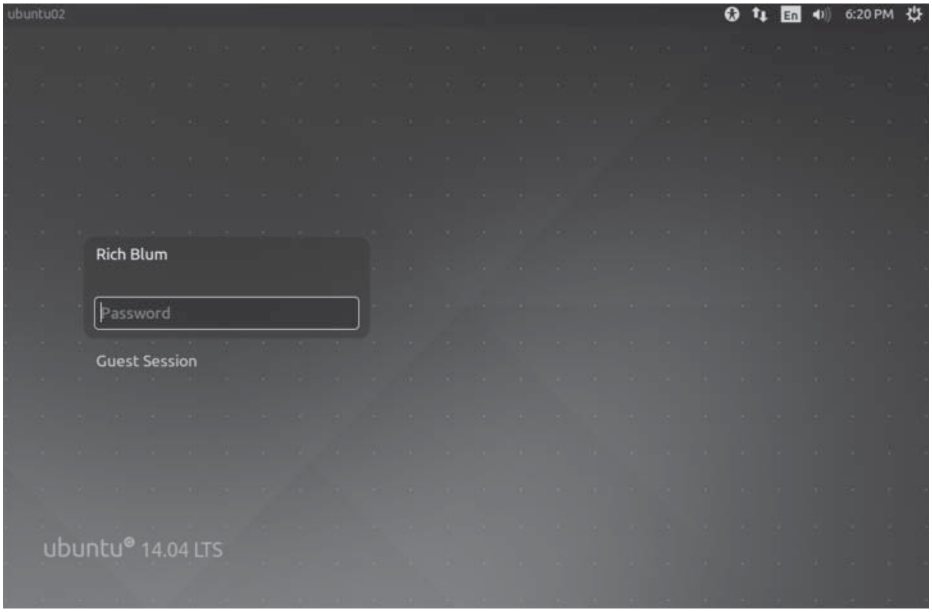


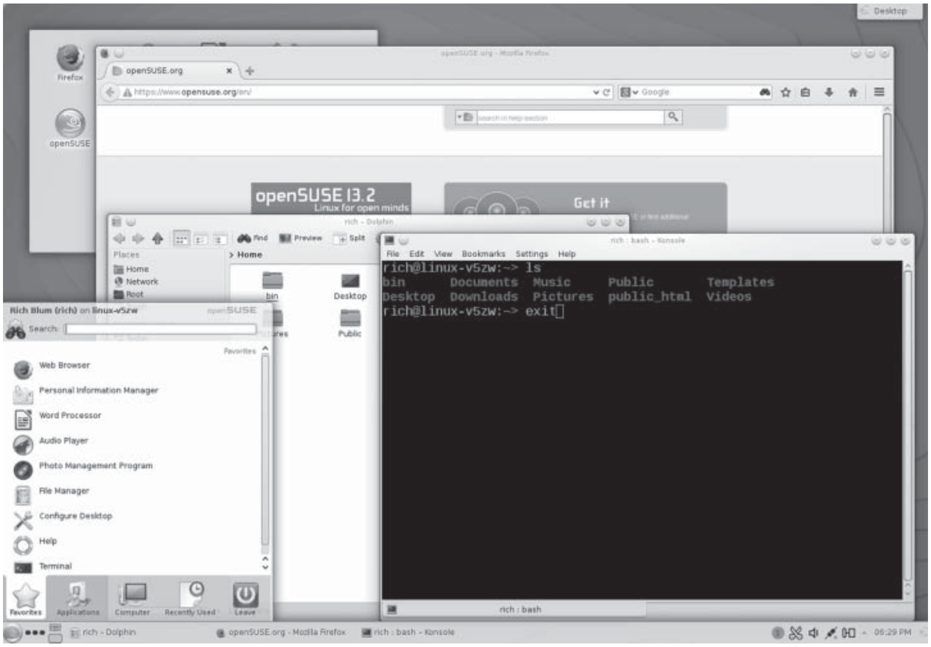
FIGURE 1.2 **Graphical login screens on Linux are similar to those for Windows or OS X.**

Some Linux GUI login screens don't prompt you for a password until after you've entered a valid username.

Unlike Windows and OS X, Linux provides a number of desktop environments. Which one you use depends on the specific variety of Linux you're using, what software options you selected at installation time, and your own personal preferences. Common choices include GNOME, KDE, Xfce, and Unity. Many other options are available as well. Many graphical desktops have assistive technology features built in. In Figure 1.2, the person icon in the top-right corner of the Ubuntu login window lets you select an assistive technology, such as a screen reader or onscreen keyboard, to assist you with the login entry.

Linux desktop environments can look quite different from one another, but they all provide similar functionality. Figure 1.3 shows the default KDE on an openSUSE 13.2 installation with a couple of programs running.

Chapter 4 describes common desktop environments and their features in more detail, but for now you should know that they all provide features such as these:



```
rich@linux-v5zw:~> ls
bin      Documents  Music     Public       Templates
Desktop  Downloads  Pictures  public_html  Videos
rich@linux-v5zw:~> exit
```

FIGURE 1.3 **Linux desktop environments provide the types of GUI controls that most users expect.**

Program Launchers You can launch programs by selecting them from menus or lists. Typically, one or more menus exist along the top, bottom, or side of the screen. In Figure 1.3, you can click the openSUSE gecko icon in the bottom-left corner of the screen to produce the menu that appears in that figure.

File Managers Linux provides GUI file managers similar to those in Windows or OS X. A window for one of these is open in the center of Figure 1.3.

Window Controls You can move windows by clicking and dragging their title bars, resize them by clicking and dragging their edges, and so on.

Multiple Desktops Most Linux desktop environments enable you to keep multiple virtual desktops active, each with their own set of programs. This feature is handy to keep the screen uncluttered while you run many programs simultaneously. Typically, an icon in one of the menus enables you to switch between virtual desktops.

Logout Options You can log out of your Linux session, which enables you to shut down the computer or let another user log in.

Logging out is important in public computing environments. If you fail to log out, a stranger might come along and use your account for malicious purposes.

You may need to install extra desktop environments to use them. This topic is not covered in this book.

As you learn more about Linux, you'll discover that its GUI environments are quite flexible. If you find that you don't like the default environment for your distribution, you can change it. Although they all provide similar features, some people have strong preferences about desktop environments. Linux gives you a choice in the matter that is not available in Windows or OS X, so feel free to try multiple desktop environments.

As described later, in "What Is a Distribution?," Linux can be considered a family of OSs. Thus you can compare one Linux version to another one.

Where Does Linux Fit in the OS World?

This chapter's title implies a comparison, and as this book is about Linux, the comparison must be with non-Linux OSs. Thus we compare Linux to three other OSs or OS families: Unix, Mac OS X, and Microsoft Windows.

Comparing Linux to Unix

Certification Objective

If you were to attempt to draw a "family tree" of OSs, you would end up scratching your head a lot. This is because OS designers often mimic each other's features, and sometimes even incorporate each other's ideas into their OSs' workings. The result can be a tangled mess of similarities between OSs, with causes ranging from coincidence to code "borrowing." Attempting to map these influences can be difficult. In the case of Linux and Unix, though, a broad statement is possible: Linux is modeled after Unix.

You can not only run *open source software*, but also modify and redistribute it yourself. Chapter 3, "Investigating Linux's Principles and Philosophy," covers the philosophy and legal issues concerning open source software.

Unix was created in 1969 at AT&T's Bell Labs. Unix's history is complex and involves multiple *forks* (that is, splitting of the code into two or more independent projects) and even entirely separate code rewrites. Modern Linux systems are, by and large, the product of open source projects that clone Unix programs, or of original open source code projects for Unix generally. These projects include the following:

The Linux Kernel Linus Torvalds created the Linux kernel as a hobby programming project in 1991, but it soon grew to be much more than that. The Linux kernel was designed to be compatible with other Unix kernels, in the sense that it used the same software interfaces in source code. This made using open source programs for other Unix versions with the Linux kernel easy.

GNU is an example of a recursive acronym—an acronym whose expansion includes the acronym itself. This is an example of geek humor.

The GNU Project The GNU's Not Unix (GNU) project is an effort by the Free Software Foundation (FSF) to develop open source replacements for all of the core elements of a Unix OS. In 1991, the FSF had already released the most important of such tools, with the notable exception of the kernel. (The GNU HURD kernel is now available, but is not as popular as the Linux kernel.)

Alternatives to the GNU tools include proprietary commercial tools and open source tools developed for the Berkeley Software Distribution (BSD) Unix variants. The tools used on a Unix-like OS can influence its overall "flavor," but all of these tool sets are similar enough to give any Unix variety a similar feel when compared to a non-Unix OS.

Xorg-X11 The X Window System is the GUI environment for most Unix OSs. Most Linux distributions today use the Xorg-X11 variety of X. As with the basic text-mode tools provided by the GNU project, choice of an X server can affect some features of a Unix-like OS, such as the types of fonts it supports.

Desktop Environments GNOME, KDE, Unity, Xfce, and other popular open source desktop environments have largely displaced commercial desktop environments, even on commercial versions of Unix. Thus you won't find big differences between Linux and Unix in this area.

Mac OS X, described shortly, is a commercial Unix-based OS that eschews both X and the desktop environments that run on it in favor of Apple's own GUI.

Server Programs Historically, Unix and Linux have been popular as server OSs—organizations use them to run web servers, email servers, file servers, and so on. Linux runs the same popular server programs as do commercial Unix versions and the open source BSDs.

User Productivity Programs In this realm, as in server programs, Linux runs the same software as do other Unix-like OSs. In a few cases, Linux runs more programs, or runs them better. This is mostly because of Linux's popularity and the vast array of hardware drivers that Linux offers. If a program needs advanced video card support, for example, it's more likely to find that support on Linux than on a less popular Unix-like OS.

On the whole, Linux can be thought of as a member of the family of Unix-like OSs. Although Linux is technically *not* a Unix OS, it's similar enough that the differences are unimportant compared to the differences between this family as a whole and other OSs, such as Windows. Because of its popularity, Linux offers better hardware support, at least on commodity PC hardware. Some Unix varieties offer specific features that Linux lacks, though. For instance, the Zettabyte File System (ZFS), available on Solaris, FreeBSD, and some other OSs, provides advanced filesystem features that aren't yet fully implemented in Linux.

A ZFS add-on for Linux is available, but it's not fully integrated into the OS. A Linux filesystem known as Btrfs offers many ZFS features, and it is gaining in popularity among various Linux distributions.

Comparing Linux to Mac OS X

Certification Objective

Mac OS X is a commercial Unix-based OS that borrows heavily from the BSDs and discards the usual Unix GUI (namely X) in favor of its own user interface. This makes OS X both very similar to Linux and quite different from it.

Code Types

Programmers write programs in a form known as *source code*. Although source code can seem arcane to the uninitiated, it's crystal clear compared to the form a program must take for a computer to run it—that is, *binary code*. A program known as a *compiler* translates source code to binary code. (Alternatively, some programming languages rely on an *interpreter*, which converts source code to binary code "on the fly," eliminating the need to compile source code.)

The term *open source* refers to the availability of source code. This is generally withheld from the public in the case of commercial software programs and OSs. Conversely, a programmer with access to a program's source code through open source software can fix bugs, add features, and otherwise alter how the program operates.

You can open an OS X Terminal window and type many of the same commands described in this book to achieve similar ends. If a command described in this book isn't present, you may be able to install it in one way or another. OS X ships with some popular Unix server programs, so you can configure it to work much like Linux or another Unix-like OS as a network server computer.

The X in X server is a letter X, but the X in OS X is a Roman numeral (10), denoting the tenth version of Mac OS.

OS X differs from Linux in its user interface, though. The OS X user interface is known as *Cocoa* from a programming perspective, or *Aqua* from a user's point of view. It includes elements that are roughly equivalent to both X and a desktop environment in Linux. Because Cocoa isn't compatible with X from a programming perspective, applications developed for OS X can't be run directly on Linux (or on other Unix-like OSs), and porting them (that is, modifying the source code and recompiling them) for Linux is a nontrivial undertaking. Thus native OS X applications seldom make the transition to Linux.

OS X includes an implementation of X that runs under Aqua. This makes the transfer of GUI Linux and Unix programs to OS X relatively straightforward. The resulting programs don't entirely conform to the Aqua user interface, though. They may have buttons, menus, and other features that look out of place compared to the usual appearance of OS X equivalents.

Apple makes OS X available for its own computers. Its license terms forbid installation on non-Apple hardware and, putting aside licensing issues, installing OS X on non-Apple hardware is a nontrivial undertaking. A variant of OS X, known as iOS, runs on Apple's iPad and iPhone devices, and it is equally nonportable to other devices. Thus OS X is largely limited to Apple hardware. Linux,