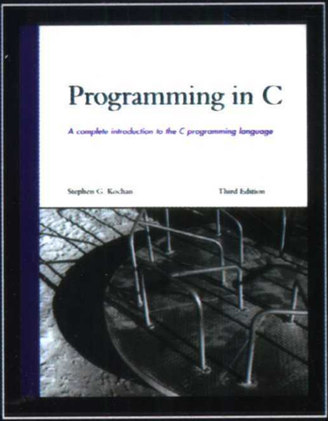
TURING 图灵原版计算机科学系列

SAMS

# Programming in C
## Third Edition

# C语言程序设计
## （英文版·第3版）

［美］Stephen G. Kochan 著



人民邮电出版社
POSTS & TELECOM PRESS

TURING 图灵原版计算机科学系列

# Programming in C

## Third Edition

# C语言程序设计

## （英文版·第3版）

江苏工业学院图书馆
藏书章

[美] Stephen G. Kochan 著

人民邮电出版社
POSTS & TELECOM PRESS

**图书在版编目（CIP）数据**

C 语言程序设计：第 3 版/（美）科汉著. —北京：人民邮电出版社，2006.5

（图灵原版计算机科学系列）

ISBN 7-115-14763-9

Ⅰ. C…　Ⅱ. 科…　Ⅲ. C 语言— 程序设计— 教材— 英文　Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2006）第 043585 号

## 内 容 提 要

本书主要讲解如何用C语言编程。书中涵盖了C语言的所有特性，包括ANSI C99标准最新增加的特性。本书通过示例讲解C语言，使用完整的程序来阐释每个新概念，而且对所有C函数都提供了详细的说明。此外，每章结尾都配有习题，附录还提供了详细的C语言的小结和标准C语言库，便于快速参考。读者既可从本书中学到语言的基础，又可以学到好的编程实践。

无论是课堂讲解还是自学，本书都是理想的C语言教材。

图灵原版计算机科学系列

**C 语言程序设计（英文版·第 3 版）**

◆ 著　　[美] Stephen G. Kochan
　责任编辑　杨海玲

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
　邮编　100061　　电子函件　315@ptpress.com.cn
　网址　http://www.ptpress.com.cn
　北京顺义振华印刷厂印刷
　新华书店总店北京发行所经销

◆ 开本：800×1000　1/16
　印张：34.5
　字数：675 千字　　　　2006 年 5 月第 1 版
　印数：1－3 000 册　　　2006 年 5 月北京第 1 次印刷
　著作权合同登记号　图字：01-2006-1719 号

ISBN 7-115-14763-9/TP・5391

定价：55.00 元

**读者服务热线：(010)88593802　印装质量热线：(010)67129223**

人民邮电出版社
POSTS & TELECOM PRESS

TURING
图灵教育

# 引进国外经典教材 服务中国高等教育

人民邮电出版社图灵公司专注于引进国外经典教材与优秀科技图书，出版国内高校教师编写的专业教材，专业方向包括：计算机、数学、统计学和电子电气等。合作伙伴包括Prentice-Hall、Addison-Wesley、Wiley、McGraw-Hill、剑桥大学出版社、Elsevier、IEEE Press、Wrox、SIAM等多家世界知名出版公司，具有丰富的选题资源和雄厚的出版力量。

书　　名：UML面向对象建模与设计(英文版·第2版)
原 书 名：Object-Oriented Modeling and Design with UML,2E
作　　者：James R.Rumbaugh, Michael R.Blaha等
书　　号：7-115-14076-6
定　　价：55.00元
出版时间：2005年11月

书　　名：现代编译原理—C 语言描述(英文版)
原 书 名：Modern Compiler Implementation in C
作　　者：Andrew W.Appel, Maia Ginsburg
书　　号：7-115-13771-4
定　　价：59.00元
出版时间：2005年9月

书　　名：数据结构与算法分析—C 语言描述(英文版·第2版)
原 书 名：Data Structures and Algorithm Analysis in C
作　　者：Mark Allen Weiss
改　　编：陈越
书　　号：7-115-13984-9/TP·4957
定　　价：49.00元
出版时间：2005年8月

书　　名：数据挖掘导论（英文版）
原 书 名：Introduction to Data Mining
作　　者：Pang-Ning Tan,Michael Steinbach,Vipin Kumar
书　　号：7-115-14144-4
定　　价：59.00元
出版时间：2005年12月

书　　名：数理逻辑（英文版·第2版）
原 书 名：A Mathematical Introduction to Logic
作　　者：Herbert B.Enderton
书　　号：7-115-14145-2
定　　价：39.00元
出版时间：2005年12月

书　　名：UNIX环境高级编程（英文版·第2版）
原 书 名：Advanced Programming in the UNIX Environment
作　　者：W. Richard Stevens Stephen A. Rago
书　　号：7-115-14484
定　　价：99.00元
出版时间：2006年3月

书　　名：C++ Primer中文版（第4版）
原 书 名：C++ Primer,Fourth Edition
作　　者：Stanley B.Lippman, Josée Lajoie等
定　　价：99.00元
出版时间：2006年3月

书　　名：C++程序设计(英文版·第5版)
原 书 名：Small C++ How to Program
作　　者：H.M.Deitel, P.J.Deitel
书　　号：7-115-14151-7
定　　价：59.00元
出版时间：2005年12月

北京图灵文化发展有限公司 www.turingbook.com
地址：北京市海淀区西四环北路140号京鼎原商务楼405室　邮编：100089
电话：010-88592340，88593802　传真：010-88593803　E-mail：contact@turingbook.com

# 版权声明

Original edition, entitled *Programming in C, Third Edition*, 0672326663 by Stephen G. Kochan, published by Pearson Education, Inc, publishing as Sams, Copyright © 2005 by Sams Publishing.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

China edition published by POSTS & TELECOM PRESS Copyright © 2006.

This edition is manufactured in the People's Republic of China, and is authorized for sale only in the People's Republic of China excluding Hong Kong, Macao and Taiwan.

本书英文版由Pearson Education Asia Ltd.授权人民邮电出版社独家出版。未经出版者书面许可，不得以任何方式复制或抄袭本书内容。

仅限于中华人民共和国境内（香港、澳门特别行政区和台湾地区除外）销售发行。

本书封面贴有Pearson Education（培生教育出版集团）激光防伪标签，无标签者不得销售。

版权所有，侵权必究。

此为试读,需要完整PDF请访问: www.ertongbook.com

*To my mother and father*

# 前　言

真是难以置信，我撰写本书第1版已经是20多年前的事了！那时，Kernighan和Ritchie的*The C Programming Language*是市面上唯一的一本C语言的书。现在时代已经变了！

回想20世纪80年代ANSI C标准出现时，本书被分成了两本书：最初的那本书仍然称为*Programming in C*，而涵盖ANSI C的那本书名改成了*Programming in ANSI C*。这样做是因为编译器供应商花费几年的时间才发布了ANSI C编译器并使编译器得到广泛应用。当时要在一本教材中既包含ANSI C又包含非ANSI C让我感觉非常棘手，所以有这样的决定。

自1989年发布第一个标准以来，ANSI C标准已经修订过几次。最新一版（C99）的ANSI C标准是编写本版书的主要原因。本版阐述了这一标准对C语言的修改。

除了涵盖C99特性之外，本书还包括两章新的内容：一章讨论C程序的调试，另一章简要概述了面向对象编程（OOP），增加这一章是因为有几种面向对象编程语言是基于C语言的，包括C++、Java和Objective-C。

我非常感谢这些年来一直使用本书的人，能收到读者的反馈让我感到非常满足，这也是我今天还坚持写作的动力！

非常欢迎新读者的加入，希望本书能够满足你的需要。

## 致谢

我要感谢在本书各个版本的准备过程中给予帮助的人们，他们是Douglas McCormick、Jim Scharf、Henry Tabickman、Dick Fritz、Steve Levy、Tony Ianinno和Ken Brown。我也要感谢纽约大学的Henry Mulish，他在如何写作方面教了我很多，并将我带入出版行业。

我要感谢本书的开发编辑Mark Renfrow和项目编辑Dan Knott。我还要感谢本书的文字编辑Karen Annett和技术编辑Bradley Jones。最后，我要感谢Sams出版社的所有参与本书出版的人员，尽管我并未直接与他们一起工作。

Stephen Kochan
2004年6月
steve@kochan-wood.com

# Contents At a Glance

1 Introduction 1

2 Some Fundamentals 5

3 Compiling and Running Your First Program 11

4 Variables, Data Types, and Arithmetic Expressions 21

5 Program Looping 43

6 Making Decisions 65

7 Working with Arrays 95

8 Working with Functions 119

9 Working with Structures 165

10 Character Strings 195

11 Pointers 235

12 Operations on Bits 279

13 The Preprocessor 299

14 More on Data Types 321

15 Working with Larger Programs 333

16 Input and Output Operations in C 347

17 Miscellaneous and Advanced Features 373

18 Debugging Programs 389

19 Object-Oriented Programming 411

A C Language Summary 425

B The Standard C Library 467

C Compiling Programs with gcc 493

D Common Programming Mistakes 497

E Resources 501

Index 505

# Table of Contents

1 **Introduction 1**

2 **Some Fundamentals 5**
- Programming 5
- Higher-Level Languages 6
- Operating Systems 6
- Compiling Programs 7
- Integrated Development Environments 10
- Language Interpreters 10

3 **Compiling and Running Your First Program 11**
- Compiling Your Program 11
- Running Your Program 12
- Understanding Your First Program 13
- Displaying the Values of Variables 15
- Comments 17
- Exercises 19

4 **Variables, Data Types, and Arithmetic Expressions 21**
- Working with Variables 21
- Understanding Data Types and Constants 23
  - The Basic Integer Type `int` 23
  - The Floating Number Type `float` 24
  - The Extended Precision Type `double` 25
  - The Single Character Type `char` 25
  - The Boolean Data Type `_Bool` 26
  - Type Specifiers: `long`, `long long`, `short`, `unsigned`, and `signed` 28
- Working with Arithmetic Expressions 30
  - Integer Arithmetic and the Unary Minus Operator 33
  - The Modulus Operator 35
  - Integer and Floating-Point Conversions 36
  - Combining Operations with Assignment: The Assignment Operators 38
  - Types `_Complex` and `_Imaginary` 39
- Exercises 40

5 **Program Looping 43**
- The `for` Statement 44
  - Relational Operators 46
  - Aligning Output 50
  - Program Input 51
  - Nested `for` Loops 53
  - `for` Loop Variants 54
- The `while` Statement 56
- The `do` Statement 60
  - The `break` Statement 62
  - The `continue` Statement 62
- Exercises 63

6 **Making Decisions 65**
- The `if` Statement 65
  - The `if-else` Construct 69
  - Compound Relational Tests 72
  - Nested `if` Statements 75
  - The `else if` Construct 76
- The `switch` Statement 84
- Boolean Variables 87
- The Conditional Operator 91
- Exercises 93

7 **Working with Arrays 95**
- Defining an Array 96
  - Using Array Elements as Counters 100
  - Generating Fibonacci Numbers 103
  - Using an Array to Generate Prime Numbers 104
- Initializing Arrays 106
- Character Arrays 108
  - Base Conversion Using Arrays 109
  - The `const` Qualifier 111
- Multidimensional Arrays 113

Variable-Length Arrays 115
Exercises 117

8 **Working with Functions 119**
Defining a Function 119
Arguments and Local Variables 122
Function Prototype Declaration 124
Automatic Local Variables 124
Returning Function Results 126
Functions Calling Functions Calling... 131
Declaring Return Types and Argument Types 134
Checking Function Arguments 135
Top-Down Programming 137
Functions and Arrays 137
Assignment Operators 142
Sorting Arrays 143
Multidimensional Arrays 146
Global Variables 152
Automatic and Static Variables 156
Recursive Functions 159
Exercises 162

9 **Working with Structures 165**
A Structure for Storing the Date 166
Using Structures in Expressions 168
Functions and Structures 171
A Structure for Storing the Time 177
Initializing Structures 180
Compound Literals 181
Arrays of Structures 182
Structures Containing Structures 185
Structures Containing Arrays 187
Structure Variants 190
Exercises 191

10 **Character Strings 195**
Arrays of Characters 196
Variable-Length Character Strings 198
Initializing and Displaying Character Strings 201
Testing Two Character Strings for Equality 204
Inputting Character Strings 206
Single-Character Input 208
The Null String 213
Escape Characters 216
More on Constant Strings 218
Character Strings, Structures, and Arrays 219
A Better Search Method 222
Character Operations 227
Exercises 230

11 **Pointers 235**
Defining a Pointer Variable 235
Using Pointers in Expressions 239
Working with Pointers and Structures 240
Structures Containing Pointers 243
Linked Lists 244
The Keyword `const` and Pointers 253
Pointers and Functions 254
Pointers and Arrays 259
A Slight Digression About Program Optimization 263
Is It an Array or Is It a Pointer? 264
Pointers to Character Strings 266
Constant Character Strings and Pointers 267
The Increment and Decrement Operators Revisited 268
Operations on Pointers 272
Pointers to Functions 273
Pointers and Memory Addresses 274
Exercises 276

12 **Operations on Bits 279**
Bit Operators 280
The Bitwise AND Operator 281
The Bitwise Inclusive-OR Operator 283
The Bitwise Exclusive-OR Operator 284
The Ones Complement Operator 285
The Left Shift Operator 287
The Right Shift Operator 287
A Shift Function 288
Rotating Bits 290
Bit Fields 292

Exercises 297

**13 The Preprocessor 299**

The #define Statement 299

Program Extendability 303

Program Portability 305

More Advanced Types of Definitions 306

The # Operator 312

The ## Operator 313

The #include Statement 313

System Include Files 316

Conditional Compilation 316

The #ifdef, #endif, #else, and #ifndef Statements 316

The #if and #elif Preprocessor Statements 318

The #undef Statement 319

Exercises 320

**14 More on Data Types 321**

Enumerated Data Types 321

The typedef Statement 325

Data Type Conversions 327

Sign Extension 329

Argument Conversion 329

Exercises 330

**15 Working with Larger Programs 333**

Dividing Your Program into Multiple Files 333

Compiling Multiple Source Files from the Command Line 334

Communication Between Modules 336

External Variables 336

Static Versus Extern Variables and Functions 339

Using Header Files Effectively 341

Other Utilities for Working with Larger Programs 342

The make Utility 343

The cvs Utility 344

Unix Utilities: ar, grep, sed, and so on 345

**16 Input and Output Operations in C 347**

Character I/O: getchar and putchar 348

Formatted I/O: printf and scanf 348

The printf Function 348

The scanf Function 355

Input and Output Operations with Files 359

Redirecting I/O to a File 359

End of File 361

Special Functions for Working with Files 363

The fopen Function 363

The getc and putc Functions 365

The fclose Function 365

The feof Function 367

The fprintf and fscanf Functions 368

The fgets and fputs Functions 368

stdin, stdout, and stderr 369

The exit Function 370

Renaming and Removing Files 371

Exercises 371

**17 Miscellaneous and Advanced Features 373**

Miscellaneous Language Statements 373

The goto Statement 373

The null Statement 374

Working with Unions 375

The Comma Operator 378

Type Qualifiers 378

The register Qualifier 378

The volatile Qualifier 379

The restrict Qualifier 379

Command-Line Arguments 380

Dynamic Memory Allocation 383

The calloc and malloc Functions 384

The sizeof Operator 385

The free Function 387

**18 Debugging Programs 389**

Debugging with the Preprocessor 389

Debugging Programs with gdb 395

Working with Variables 398

Source File Display 399

Controlling Program Execution 400

Getting a Stack Trace 405

此为试读，需要完整PDF请访问：www.ertongbook.com

Calling Functions and Setting Arrays and Structures 405
Getting Help with gdb Commands 406
Odds and Ends 408

**19 Object-Oriented Programming 411**
What Is an Object Anyway? 411
Instances and Methods 412
Writing a C Program to Work with Fractions 413
Defining an Objective-C Class to Work with Fractions 414
Defining a C++ Class to Work with Fractions 419
Defining a C# Class to Work with Fractions 422

**A C Language Summary 425**
1.0 Digraphs and Identifiers 425
1.1 Digraph Characters 425
1.2 Identifiers 425
2.0 Comments 426
3.0 Constants 427
3.1 Integer Constants 427
3.2 Floating-Point Constants 427
3.3 Character Constants 428
3.4 Character String Constants 429
3.5 Enumeration Constants 430
4.0 Data Types and Declarations 430
4.1 Declarations 430
4.2 Basic Data Types 430
4.3 Derived Data Types 432
4.4 Enumerated Data Types 438
4.5 The `typedef` Statement 438
4.6 Type Modifiers `const`, `volatile`, and `restrict` 439
5.0 Expressions 439
5.1 Summary of C Operators 440
5.2 Constant Expressions 442
5.3 Arithmetic Operators 443
5.4 Logical Operators 444
5.5 Relational Operators 444
5.6 Bitwise Operators 445
5.7 Increment and Decrement Operators 445
5.8 Assignment Operators 446
5.9 Conditional Operators 446
5.10 Type Cast Operator 446
5.11 `sizeof` Operator 447
5.12 Comma Operator 447
5.13 Basic Operations with Arrays 447
5.14 Basic Operations with Structures 448
5.15 Basic Operations with Pointers 448
5.16 Compound Literals 450
5.17 Conversion of Basic Data Types 451
6.0 Storage Classes and Scope 452
6.1 Functions 452
6.2 Variables 452
7.0 Functions 454
7.1 Function Definition 454
7.2 Function Call 455
7.3 Function Pointers 456
8.0 Statements 456
8.1 Compound Statements 456
8.2 The `break` Statement 456
8.3 The `continue` Statement 457
8.4 The `do` Statement 457
8.5 The `for` Statement 457
8.6 The `goto` Statement 458
8.7 The `if` Statement 458
8.8 The `null` Statement 458
8.9 The `return` Statement 459
8.10 The `switch` Statement 459
8.11 The `while` Statement 460
9.0 The Preprocessor 460
9.1 Trigraph Sequences 460
9.2 Preprocessor Directives 461
9.3 Predefined Identifiers 466

**B The Standard C Library 467**
Standard Header Files 467
`<stddef.h>` 467
`<limits.h>` 468
`<stdbool.h>` 469
`<float.h>` 469
`<stdint.h>` 469
String Functions 470

Memory Functions 472

Character Functions 473

I/O Functions 473

In-Memory Format Conversion Functions 478

String-to-Number Conversion 479

Dynamic Memory Allocation Functions 481

Math Functions 482

Complex Arithmetic 488

General Utility Functions 490

**C Compiling Programs with** gcc **493**

General Command Format 493

Command-Line Options 494

**D Common Programming Mistakes 497**

**E Resources 501**

Answers to Exercises, Errata, etc. 501

The C Programming Language 501

Books 501

Web Sites 502

Newsgroups 502

C Compilers and Integrated Development Environments 502

gcc 502

MinGW 502

CygWin 502

Visual Studio 503

CodeWarrior 503

Kylix 503

Miscellaneous 503

Object-Oriented Programming 503

The C++ Language 503

The C# Language 503

The Objective-C Language 503

Development Tools 504

**Index 505**

# 1 Introduction

THE C PROGRAMMING LANGUAGE WAS pioneered by Dennis Ritchie at AT&T Bell Laboratories in the early 1970s. It was not until the late 1970s, however, that this programming language began to gain widespread popularity and support. This was because until that time C compilers were not readily available for commercial use outside of Bell Laboratories. Initially, C's growth in popularity was also spurred on in part by the equal, if not faster, growth in popularity of the Unix operating system. This operating system, which was also developed at Bell Laboratories, had C as its "standard" programming language. In fact, well over 90% of the operating system itself was written in the C language!

The enormous success of the IBM PC and its look-alikes soon made MS-DOS the most popular environment for the C language. As C grew in popularity across different operating systems, more and more vendors hopped on the bandwagon and started marketing their own C compilers. For the most part, their version of the C language was based on an appendix found in the first C programming text—*The C Programming Language*—by Brian Kernighan and Dennis Ritchie. Unfortunately, this appendix did not provide a complete and unambiguous definition of C, meaning that vendors were left to interpret some aspects of the language on their own.

In the early 1980s, a need was seen to standardize the definition of the C language. The American National Standards Institute (ANSI) is the organization that handles such things, so in 1983 an ANSI C committee (called X3J11) was formed to standardize C. In 1989, the committee's work was ratified, and in 1990, the first official ANSI standard definition of C was published.

Because C is used around the world, the International Standard Organization (ISO) soon got involved. They adopted the standard, where it was called ISO/IEC 9899:1990. Since that time, additional changes have been made to the C language. The most recent standard was adopted in 1999. It is known as ANSI C99, or ISO/IEC 9899:1999. It is this version of the language upon which this book is based.

C is a "higher-level language," yet it provides capabilities that enable the user to "get in close" with the hardware and deal with the computer on a much lower level. This is because, although C is a general-purpose structured programming language, it was originally designed with systems programming applications in mind and, as such, provides the user with an enormous amount of power and flexibility.

This book proposes to teach you how to program in C. It assumes no previous exposure to the language and was designed to appeal to novice and experienced programmers alike. If you have previous programming experience, you will find that C has a unique way of doing things that probably differs from other languages you have used.

Every feature of the C language is treated in this text. As each new feature is presented, a small *complete* program example is usually provided to illustrate the feature. This reflects the overriding philosophy that has been used in writing this book: to teach by example. Just as a picture is worth a thousand words, so is a properly chosen program example. If you have access to a computer facility that supports the C programming language, you are strongly encouraged to download and run each program presented in this book and to compare the results obtained on your system to those shown in the text. By doing so, not only will you learn the language and its syntax, but you will also become familiar with the process of typing in, compiling, and running C programs.

You will find that program readability has been stressed throughout the book. This is because I strongly believe that programs should be written so that they can be easily read—either by the author or by somebody else. Through experience and common sense, you will find that such programs are almost always easier to write, debug, and modify. Furthermore, developing programs that are readable is a natural result of a true adherence to a structured programming discipline.

Because this book was written as a tutorial, the material covered in each chapter is based on previously presented material. Therefore, maximum benefit will be derived from this book by reading each chapter in succession, and you are highly discouraged from "skipping around." You should also work through the exercises that are presented at the end of each chapter before proceeding on to the next chapter.

Chapter 2, "Some Fundamentals," which covers some fundamental terminology about higher-level programming languages and the process of compiling programs, has been included to ensure that you understand the language used throughout the remainder of the text. From Chapter 3, "Compiling and Running Your First Program," on, you will be slowly introduced to the C language. By the time Chapter 16, "Input and Output Operations in C," rolls around, all the essential features of the language will have been covered. Chapter 16 goes into more depth about I/O operations in C. Chapter 17, "Miscellaneous and Advanced Features," includes those features of the language that are of a more advanced or esoteric nature.

Chapter 18, "Debugging Programs," shows how you can use the C preprocessor to help debug your programs. It also introduces you to interactive debugging. The popular debugger `gdb` was chosen to illustrate this debugging technique.

Over the last decade, the programming world has been abuzz with the notion of object-oriented programming, or OOP for short. C is not an OOP language; however, several other programming languages that are based on C are OOP languages. Chapter 19, "Object-Oriented Programming," gives a brief introduction to OOP and some of its terminology. It also gives a brief overview of three OOP languages that are based on C, namely C++, C#, and Objective-C.

Appendix A, "C Language Summary," provides a complete summary of the language and is provided for reference purposes.

Appendix B, "The Standard C Library," provides a summary of many of the standard library routines that you will find on all systems that support C.

Appendix C, "Compiling Programs with `gcc`," summarizes many of the commonly used options when compiling programs with GNU's C compiler `gcc`.

In Appendix D, "Common Programming Mistakes," you'll find a list of common programming mistakes.

Finally, Appendix E, "Resources," provides a list of resources you can turn to for more information about the C language and to further your studies.

Answers to the quizzes at the end of chapters can be found at `www.kochan-wood.com`.

This book makes no assumptions about a particular computer system or operating system on which the C language is implemented. The text makes brief mention of how to compile and execute programs using the popular GNU C compiler `gcc`.

I want to thank the following people for their help in the preparation of various versions of this text: Douglas McCormick, Jim Scharf, Henry Tabickman, Dick Fritz, Steve Levy, Tony Ianinno, and Ken Brown. I also want to thank Henry Mullish of New York University for teaching me so much about writing and for getting me started in the publishing business.

An earlier edition of this book was also dedicated to the memory of Maureen Connelly, a former production editor at Hayden Book Company, the publishers of the first edition of this book.

此为试读,需要完整PDF请访问: www.ertongbook.com