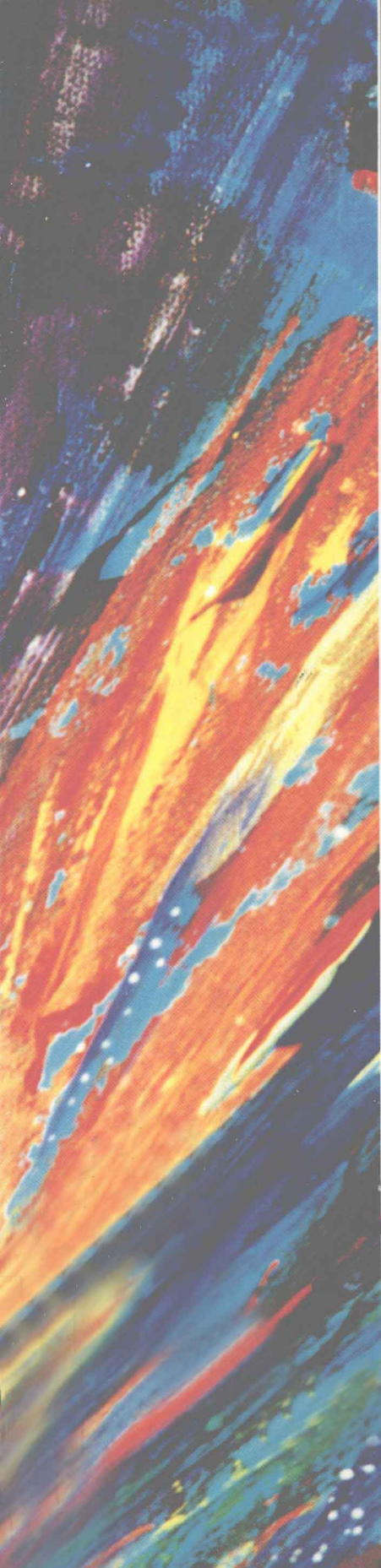




COMPUTER SCIENCE

*A Structured Programming
Approach Using C*

Behrouz A. Forouzan
Richard F. Gilberg

COMPUTER SCIENCE: A Structured Programming Approach Using C

Behrouz A. Forouzan

De Anza College

Richard F. Gilberg

De Anza College



PWS PUBLISHING COMPANY

I(T)P An International Thomson Publishing Company

Boston • Albany • Bonn • Cincinnati • Detroit • London • Madrid
Melbourne • Mexico City • New York • Pacific Grove • Paris
San Francisco • Singapore • Tokyo • Toronto • Washington



PWS Publishing Company
20 Park Plaza, Boston, MA 02116-4324

Reprinted in 1998 by PWS Publishing Company,
a division of International Thomson Publishing, Inc.
Original copyright © 1997 by West Publishing Company.

All rights reserved. No part of this book may be reproduced,
stored in a retrieval system, or transmitted by any means—
electronic, mechanical, photocopying, recording, or otherwise
—without prior written permission of PWS Publishing
Company.

ITP®

International Thomson Publishing
The trademark ITP is used under license.

Printed in the United States of America.
98 99 00 01 — 10 9 8 7 6 5 4

ISBN: 0-314-09573-X



*This book is printed on recycled,
acid-free paper.*

Sponsoring Editor: *Mike Sugarman*
Production Editor: *Andrea Goldman*
Manufacturing Coordinator: *Andrew Christensen*
Marketing Manager: *Nathan Wilbur*
Composition: *Carlisle Communications*
Text and Cover Printer: *West Publishing Company*

Library of Congress Cataloging-in-Publication-Data

Forouzan, Behrouz A.

Computer science: a structured programming
approach using C / Behrouz A. Forouzan, Richard F.
Gilberg

p. cm.

Includes index.

ISBN 0-314-09573-X (soft : alk. paper)

1. C (Computer program language) 2. Structured
programming. I. Gilberg, Richard II. Title.
QA76.73.C15F66 1997

005.13'3--dc20

97-31288

CIP

For more information, contact:
PWS Publishing Company
20 Park Plaza
Boston, MA 02116

International Thomson Publishing Europe
Berkshire House
168-173 High Holborn
London WC1V 7AA
England

Thomas Nelson Australia
102 Dodds Street
South Melbourne, 3205
Victoria, Australia

Nelson Canada
1120 Birchmont Road
Scarborough, Ontario
Canada M1K 5G4

International Thomson Editores
Campos Eliseos 385, Piso 7
Col. Polanco
11560 Mexico D.F., Mexico

International Thomson Publishing GmbH
Königswinterer Strasse 418
53227 Bonn, Germany

International Thomson Publishing Asia
221 Henderson Road
#05-10 Henderson Building
Singapore 0315

International Thomson Publishing Japan
Hirakawacho Kyowa Building, 31
2-2-1 Hirakawacho
Chiyoda-ku, Tokyo 102
Japan

COMPUTER SCIENCE:

A Structured Programming Approach Using C

TO OUR STUDENTS

who are the rationale for this text.

To my family

BAF

To Evelyn

RFG

PREFACE

A C LANGUAGE PERSPECTIVE

This text has two primary objectives. First, to teach the student the basic principles of programming as outlined in the ACM curriculum for a CS1 class. Second, to teach the basic constructs of the C Language. While accomplishing both of these objectives, this text also puts them in the context of good software engineering concepts that we have developed through over thirty years of experience in industry and academia.

While C is a complex and professional language, our experience of using an early version of this book in the classroom has shown that beginning students can easily understand it. We believe that if the language is put into a perspective that allows the student to understand its design it is not difficult.

There are two aspects of C that separate it from most other languages: expressions and pointers. The concept of expressions, as used in C, is unique. This is one of the reasons that we felt this book had to be built for C from the ground up, not based on a book that used Pascal. The first half of this text, therefore, builds a firm understanding of expressions, introducing pointers only to the extent necessary to cover pass-by-reference and arrays. Our experiences have shown that with a firm grasp of the expression concept, much of the mystery of C disappears.

Beginning slowly with Chapter 9, we develop the concept of pointers ending with a simple introduction to linked lists. While not all courses will have time to cover linked lists, those that do will give the students a head start into data structures. Good students will also find that they can study the linked lists material as a preview to data structures during their term break. The last chapter of the text covers C constructs that are generally beyond the scope of a CS1 class. This material is included to provide the students with references to these subjects that will be useful when they take other courses that require the complete C Language.

FEATURES OF THE BOOK

There are several features of this book that not only make it unique, but make it easier for beginning students to understand.

Structure and Style

One of our basic tenets is that good habits are formed early. The corollary to this tenet is that bad habits are hard to break. Therefore, we consistently emphasize the principles of structured programming and software engineering. Every complete program in the book uses a consistent style. As programs are analyzed, style and standards are further explained. We are not saying that there aren't other good styles, but our experience has shown that if students are exposed to a good style and implement it, they will be better able to adapt to other good styles. On the other hand, unlearning sloppy short-cut habits is very difficult.

Principle Before Practice

Whenever possible, we develop the principle of a subject before we introduce the language implementation. For example, in Chapter 5 we first introduce the concept of logical data and selection and then we introduce the if...else and switch statements. This approach gives the student an understanding of selection before introducing the nuances of the language.

Visual Approach

A brief scanning of the book will demonstrate that our approach is very visual. There are over 400 figures, plus over 70 tables and 180 program examples. While this tends to create a large book, the visual approach makes it much easier for students to follow the material.

Examples

While the programming examples vary in complexity, each of them uses a consistent style. Our experience working with productional programs that live for 10 to 20 years convinced us that readable and understandable programs are easier to work with than programs written in a terse, cryptic manner. For that reason, and to emphasize the structure of the language, we label the sections in a function with comments. We consistently follow a style that places only one declaration, definition, or statement on a line. These programs are available to allow students not only to run them, but to explore related topics through modification and experimentation.

Coding Techniques

Throughout the text we include coding techniques that make programs more readable and often more efficient. For example, on page 248 you will find the following discussion:

Where do we check for greater than? The answer is that we default the greater than condition to the nested else. When coding a two-way selection statement, try to code the most probable condition first; with nested selection statements, code the most probable first and the least probable last.

These techniques are drawn from our extensive industry experience and are not found in most texts.

Software Engineering

A discussion of software engineering principles concludes each chapter. Our intent is not to replace a separate course in software engineering. Rather, we firmly believe that by incorporating basic software engineering principles early in their studies, students will be better prepared for a formal treatment of the subject. Even more important, by writing well engineered programs from the beginning, they will not be forced to unlearn and relearn. They will also relate more to, and comprehend more of, software discussions in their subsequent classes.

While the Software Engineering Sections are found at the end of the chapter, they are most successfully taught by introducing them as the chapter unfolds. Then, a short review at the end of the chapter summarizes the principles that have been demonstrated during the lectures.

You will note that these sections are visually distinguishable from the rest of the chapter. They have been set apart for several reasons. First, they are in reality a small book in a book. While these sections contain important material, the rest of the book stands on its own without them. You may, therefore, decide to cover the Software Engineering Sections formally as a topics for lectures or informally while the chapter material is being covered. You may decide to leave them to the student as additional reading. Or, you may decide to exclude them entirely from the formal materials for your class.

In all but two chapters the Software Engineering Sections directly pertain to the chapter material. Of the other two chapters, the Software Engineering Section in Chapter 12, Strings, covers an important general topic, Software Quality. Then, in Chapter 15, we develop a case study that builds a structure chart.

Pedagogical End Material

The end material of each chapter contains two parts. The first part includes three sections that are intended to aid the student in reviewing the chapter. The second part, Practice Sets, includes three sections of questions that test the student's knowledge of the chapter material.

Tips and Common Programming Errors points out helpful hints and possible problem areas.

Key Terms provides a list of all of the boldface terms introduced in the chapter.

Summary contains a concise overview of all of the key points for students to understand in the chapter.

Practice Sets

Exercises are short questions covering the material in the chapter. The answers to the odd numbered questions are included in the back of the book.

Problems are short coding problems generally intended to be run on a computer. They can usually be developed in two to three hours.

Projects are longer, major assignments that may take the average student six to nine hours to develop.

The Instructor's Solution Manual contains a complete set of solutions to all exercises and problems.

APPENDIXES AND COVER MATERIAL

The appendixes are intended to provide quick reference material, such as the ANSI Character Set, or a review of material, such as numbering systems, usually covered in a general computer class. An appendix is also included that discusses the standards and guidelines used throughout the book. Inside the covers are two important figures that are used continually throughout the course: the Precedence Table and the Formatted I/O Codes.

ACKNOWLEDGMENTS

No text of this scope can be developed without the support of many people. This is especially true for this text, which was "field tested" for two years by our students at De Anza College. Our first acknowledgment, therefore, has to be to the hundreds of students who by using the text and commenting on it made a vital contribution. We thank four students for their tremendous assistance in correcting and clarifying the material: Madeline Damiano, Sophia Fegan, Shouli Tiechman and Tomo Nagai.

We would also like to acknowledge the support of the De Anza staff. Their encouragement helped us launch the project and their comments contributed to its success. To name them all is impossible, but we especially thank Beverly D'Urso, Anne Oney, and George Rice.

To anyone who has not been through the process, the value of peer reviews cannot be appreciated enough. Writing a text rapidly becomes a myopic process. The important guidance of reviewers who can stand back and review the text as a whole cannot be measured. To twist an old cliché, "They are not valuable, they are priceless." We would especially like to acknowledge the contributions of the following reviewers:

Stephen Allen, *Utah State University*
Ali Behforooz, *Towson State University*
George Berry, *Wentworth Institute of Technology*
Ernest Carey, *Utah Valley State College*
Constance Conner, *City College of San Francisco*
John S. DaPonte, *Southern Connecticut State University*
Maurice L. Eggen, *Trinity University*
Peter Gabrovsky, *CSU Northridge*
Robert Gann, *Hartwick College*
Rick Graziani, *Cabrillo College*
Jerzy Jaromczyk, *University of Kentucky*
John Kinio, *Humber College*
Roberta Klibaner, *College of Staten Island*
Joseph A. Konstan, *University of Minnesota*
Krishna Kulkarni, *Rust College*
John Lowther, *Michigan Technological University*
Mike Michaelson, *Palomar College*
Jo Ann Parikh, *Southern Connecticut State University*
Mark Parker, *Shoreline Community College*
Oskar Reiksts, *Kutztown University*
Jim Roberts, *Carnegie Mellon University*
Ali Salenia, *South Dakota State University*
Larry Sells, *Oklahoma City University*
Shashi Shekhar, *University of Minnesota*
Robert Signorile, *Boston College*
Brenda Sonderegger, *Montana State University*
Deborah Sturm, *College of Staten Island*
Venkat Subramanian, *University of Houston*
John B. Tappen, *University of Southern Colorado*
Marc Thomasm, *California State University, Bakersfield*
John Trono, *St. Michael's College*

Choosing a publisher is a difficult task. We chose West for several reasons, two of the most important ones being our editors, Rick Mixter and Keith Dodson. Our thanks to them for helping us to write the best book we could write. Paul O'Neill ably guided the book through production.

Last, and most obviously not the least, is the support of our families and friends. Many years ago an author described writing a text as a "locking yourself in a room" process. While the authors suffer through the writing process, families and friends suffer through their absence. We can only hope that as they view the final product, they feel that their sacrifices were worth it.

Behrouz A. Forouzan
Richard F. Gilberg
Cupertino, CA

CONTENTS

Chapter 1	INTRODUCTION TO COMPUTERS	1
1-1	COMPUTER SYSTEMS	2
1-2	COMPUTER HARDWARE	2
1-3	COMPUTER SOFTWARE	3
	System Software	3
	Application Software	4
1-4	COMPUTING ENVIRONMENTS	4
	Personal Computing Environment	4
	Time-Sharing Environment	5
	Client/Server Environment	5
1-5	COMPUTER LANGUAGES	7
	Machine Languages	7
	Symbolic Languages	8
	High-Level Languages	8
	Natural Languages	9
1-6	WRITING, EDITING, COMPILING, AND LINKING PROGRAMS	9
	Writing and Editing Programs	10
	Compiling Programs	10
	Linking Programs	11
1-7	PROGRAM EXECUTION	11
1-8	SYSTEM DEVELOPMENT	11
	System Development Life Cycle	12
	Program Development	13
	Understand the Problem	13
	Develop the Solution	13
	Write the Program	16
	Test the Program	16
	SOFTWARE ENGINEERING AND PROGRAMMING STYLE ...	19
	TIPS AND COMMON PROGRAMMING ERRORS	20
	KEY TERMS	21
	SUMMARY	21
	PRACTICE SETS	22
	Exercises	22
	Problems	22

Chapter 2	INTRODUCTION TO THE C LANGUAGE	23
2-1	BACKGROUND	24
2-2	C PROGRAMS	25
	Structure of a C Program	25
	A Simple Program	25
	Your First Program	26
	Comments	27
2-3	IDENTIFIERS	29
2-4	DATA TYPES	30
	void	31
	Integer	31
	char	32
	Floating Point	33
	Logical Data in C	34
2-5	VARIABLES	34
	Variable Declaration and Definition	34
	Variable Initialization	36
	<i>Print Sum</i>	36
2-6	CONSTANTS	38
	Integer Constants	38
	Float Constants	38
	Character Constants	38
	<i>ASCII Character Set</i>	39
	String Constants	39
2-7	CODING CONSTANTS	40
	Literal Constants	40
	Defined Constants	40
	Memory Constants	41
2-8	FORMATTED INPUT/OUTPUT	42
	Standard Files	42
	FORMATTED OUTPUT (printf)	42
	<i>Field Specification</i>	43
	<i>Format String Text</i>	45
	FORMATTED INPUT (scanf)	46
2-9	PROGRAMMING EXAMPLES	49
	Print Character Values	49
	Define Constants	50
	<i>Print a Report</i>	51
	SOFTWARE ENGINEERING AND PROGRAMMING STYLE	53
	Program Documentation	53
	<i>General Documentation</i>	53
	<i>Module Documentation</i>	53
	Data Names	53
	Data Hiding	55
	TIPS AND COMMON PROGRAMMING ERRORS	56
	KEY TERMS	57
	SUMMARY	57
	PRACTICE SETS	58

Exercises	58
Problems	60
Projects	60

Chapter 3	STRUCTURE OF A C PROGRAM	62
3-1	EXPRESSIONS	63
	Primary Expressions	64
	<i>Names</i>	64
	<i>Constants</i>	65
	<i>Parenthetical Expressions</i>	65
	Binary Expressions	65
	<i>Multiplicative Expressions</i>	65
	<i>Additive Expressions</i>	66
	Assignment Expressions	67
	<i>Simple Assignment</i>	67
	<i>Compound Assignment</i>	68
	Postfix Expressions	70
	<i>Function Call</i>	70
	<i>Postfix Increment/Decrement</i>	70
	Unary Expressions	71
	<i>Prefix Increment/Decrement</i>	71
	<i>sizeof</i>	72
	<i>Unary Plus/Minus</i>	73
3-2	PRECEDENCE AND ASSOCIATIVITY	73
	Precedence	73
	Associativity	74
	<i>Left Associativity</i>	74
	<i>Right Associativity</i>	75
3-3	SIDE EFFECTS	76
3-4	EVALUATING EXPRESSIONS	77
	Warning	80
3-5	MIXED TYPE EXPRESSIONS	81
	Implicit Type Conversion	81
	Explicit Type Conversion (Cast)	82
3-6	STATEMENTS	83
	Expression Statements	83
	Compound Statements	85
	Statements and Defined Constants	85
3-7	SAMPLE PROGRAMS	86
	Example: Calculate Average	86
	Example: Fahrenheit to Celsius	88
	Example: Calculate SalesTotal	89
	Example: Calculate Student Score	90
	SOFTWARE ENGINEERING AND PROGRAMMING STYLE	94
	KISS	94
	Parentheses	94

User Communication	95
TIPS AND COMMON PROGRAMMING ERRORS	95
KEY TERMS	96
SUMMARY	97
PRACTICE SETS	98
Exercises	98
Problems	98
Projects	99

Chapter 4 FUNCTIONS 101

4-1 DESIGNING STRUCTURED PROGRAMS	102
4-2 FUNCTIONS IN C	103
4-3 USER-DEFINED FUNCTIONS	106
Function Definition	107
<i>Function Header</i>	107
<i>Function Body</i>	107
<i>Formal Parameter List</i>	107
<i>Local Variables</i>	108
Prototype Declaration	108
The Function Call	109
<i>Void Functions with No Parameters</i>	111
<i>Void Functions with Parameters</i>	111
<i>Functions that Return Values</i>	112
Function Examples	113
<i>Print Least Significant Digit</i>	113
<i>Add Two Digits</i>	114
<i>Format Long Integer</i>	116
<i>Print Tuition for Strange College</i>	118
Parameter Passing	121
<i>Pass by Value</i>	121
<i>Pass by Reference</i>	121
4-4 STANDARD LIBRARY FUNCTIONS	126
Standard Functions for Mathematical Manipulation	126
<i>abs/fabs/labs</i>	126
<i>ceil</i>	127
<i>floor</i>	128
<i>pow</i>	128
<i>sqrt</i>	128
General Library Functions	129
<i>srand</i>	129
<i>rand</i>	129
4-5 SCOPE	131
General Rule	131
Global Scope	132
Local Scope	132
4-6 A PROGRAMMING EXAMPLE—CALCULATOR PROGRAM	133
SOFTWARE ENGINEERING AND PROGRAMMING STYLE	136
Structure Charts	136

Structure Chart Rules and Symbols	136
<i>Function Symbol</i>	136
<i>Reading Structure Charts</i>	137
Functional Cohesion	139
<i>Only One Thing</i>	139
<i>In One Place</i>	140
<i>Testability</i>	141
Top-Down Development	141
TIPS AND COMMON PROGRAMMING ERRORS	142
KEY TERMS	144
SUMMARY	144
PRACTICE SETS	145
Exercises	145
Problems	148
Projects	148

Chapter 5	SELECTION—MAKING DECISIONS	152
5-1	LOGICAL DATA AND OPERATORS	153
	Logical Data in C	153
	Logical Operators	153
	<i>not</i>	153
	<i>and</i>	153
	<i>or</i>	154
	Evaluating Logical Expressions	154
	Relational Operators	156
5-2	TWO-WAY SELECTION	158
	<i>if...else</i>	158
	<i>Null Else Statement</i>	161
	<i>Nested if Statements</i>	162
	<i>Dangling else Problem</i>	164
	<i>Simplifying if Statements</i>	165
	Conditional Expressions	166
	Two-Way Selection Example	167
5-3	MULTIWAY SELECTION	173
	<i>The switch Statement</i>	173
	<i>The else-if</i>	179
5-4	MORE STANDARD LIBRARY FUNCTIONS	181
	Standard Characters Functions	182
	<i>Classifying Functions</i>	182
	<i>Character Conversion Functions</i>	184
5-5	A MENU PROGRAM	184
	SOFTWARE ENGINEERING AND PROGRAMMING STYLE	190
	Dependent Statements	190
	Negative Logic	190
	Rules for Selection Statements	192
	Selection in Structure Charts	192
	TIPS AND COMMON PROGRAMMING ERRORS	193
	KEY TERMS	195

SUMMARY	195
PRACTICE SETS	196
Exercises	196
Problems	198
Projects	200

Chapter 6	REPETITION	204
6-1	CONCEPT OF A LOOP	205
6-2	PRETEST AND POSTTEST LOOPS	205
6-3	INITIALIZATION AND UPDATING	207
	Loop Initialization	207
	Loop Update	208
6-4	EVENT-CONTROLLED AND COUNTER-CONTROLLED LOOPS	208
	Event-Controlled Loops	208
	Counter-Controlled Loops	209
	Loop Comparison	210
6-5	LOOPS IN C	210
	The <i>while</i> Loop	211
	The <i>for</i> Loop	213
	The <i>do...while</i> Loop	216
	The Comma Expression	219
6-6	LOOP EXAMPLES	220
	<i>for</i> Loops	221
	<i>while</i> Loops	225
	<i>do...while</i> Loops	226
6-7	OTHER STATEMENTS RELATED TO LOOPING	229
	<i>break</i>	230
	<i>continue</i>	231
6-8	LOOPING APPLICATIONS	232
	Summation	232
	Product	233
	Smallest and Largest	234
	Inquiries	236
6-9	RECURSION	237
	Iterative Definition	238
	Recursive Definition	238
	Iterative Solution	239
	Recursive Solution	239
	Designing Recursive Functions	240
	Fibonacci Numbers	241
	Limitations of Recursion	244
	The Towers of Hanoi	244
	Recursive Solution of the Towers of Hanoi	245
6-10	A PROGRAMMING EXAMPLE—THE CALCULATOR PROGRAM	248
	SOFTWARE ENGINEERING AND PROGRAMMING STYLE	252
	Loops in Structure Charts	252
	Determining Algorithm Efficiency	253

Linear Loops	253
Logarithmic Loops	254
Nested Loops	255
<i>Linear Logarithmic</i>	255
<i>Dependent Quadratic</i>	255
<i>Quadratic</i>	256
Big-O Notation	256
Standard Measures of Efficiency	257
TIPS AND COMMON PROGRAMMING ERRORS	258
KEY TERMS	259
SUMMARY	259
PRACTICE SETS	260
Exercises	260
Problems	263
Projects	265

Chapter 7	TEXT FILES	271
7-1	CONCEPT OF A FILE	272
7-2	FILES AND STREAMS	273
	File Table	273
	Standard Files	274
	User Files	274
	File-Stream Association	275
7-3	STANDARD LIBRARY INPUT/OUTPUT FUNCTIONS	275
	File Open and Close	276
	<i>File open (fopen)</i>	276
	<i>File Mode</i>	278
	<i>File Close (fclose)</i>	279
	<i>Open and Close Errors</i>	280
7-4	FORMATTED INPUT/OUTPUT FUNCTIONS	281
	Format Strings	281
	<i>Whitespace</i>	281
	<i>Text</i>	281
	<i>Field Specification</i>	281
	<i>scanf</i> and <i>fscanf</i>	287
	<i>Input Data Formatting</i>	288
	<i>Side Effect and Value in Scan Functions</i>	288
	Two Common Mistakes	290
	<i>Invalid Address</i>	290
	<i>Data Type Conflict</i>	290
	<i>printf</i> and <i>fprintf</i>	292
	Side Effect and Value of <i>fprintf</i>	293
7-5	CHARACTER INPUT/OUTPUT FUNCTIONS	297
	<i>getchar</i>	298
	<i>putchar</i>	298
	<i>getc</i> and <i>fgetc</i>	298
	<i>putc</i> and <i>fputc</i>	299

	<i>ungetc</i>	299
7-6	CHARACTER INPUT/OUTPUT EXAMPLES	300
	Create Text File	300
	Copy Text File	300
	Count Characters and Lines	301
	Count Words in File	302
	SOFTWARE ENGINEERING AND PROGRAMMING STYLE	304
	Testing Files	304
	<i>Testing for Invalid User Input</i>	304
	<i>Value Errors</i>	305
	Data Terminology	306
	TIPS AND COMMON PROGRAMMING ERRORS	307
	KEY TERMS	308
	SUMMARY	309
	PRACTICE SETS	310
	Exercises	310
	Problems	311
	Projects	312

Chapter 8 **ARRAYS** **314**

8-1	CONCEPTS	315
8-2	USING ARRAYS IN C	317
	Declaration and Definition	318
	Accessing Elements in Arrays	318
	Storing Values in Arrays	319
	<i>Initialization</i>	319
	<i>Inputting Values</i>	319
	<i>Assigning Values</i>	320
	<i>Exchanging Values</i>	321
	<i>Outputting Values</i>	322
	Precedence of Index Operators	322
	Index Range Checking	323
8-3	ARRAYS AND FUNCTIONS	325
	Passing Individual Elements	325
	Passing the Whole Array	325
8-4	TWO ARRAY APPLICATIONS	327
	Frequency Arrays	327
	Histograms	329
8-5	SORTING	333
	Selection Sort	334
	Selection Sort Algorithm	335
	Bubble Sort	336
	Bubble Sort Algorithm	337
	Insertion Sort	339
	Insertion Sort Algorithm	339
	Sort Conclusions	341
8-6	SEARCHING	341
	Sequential Search	342