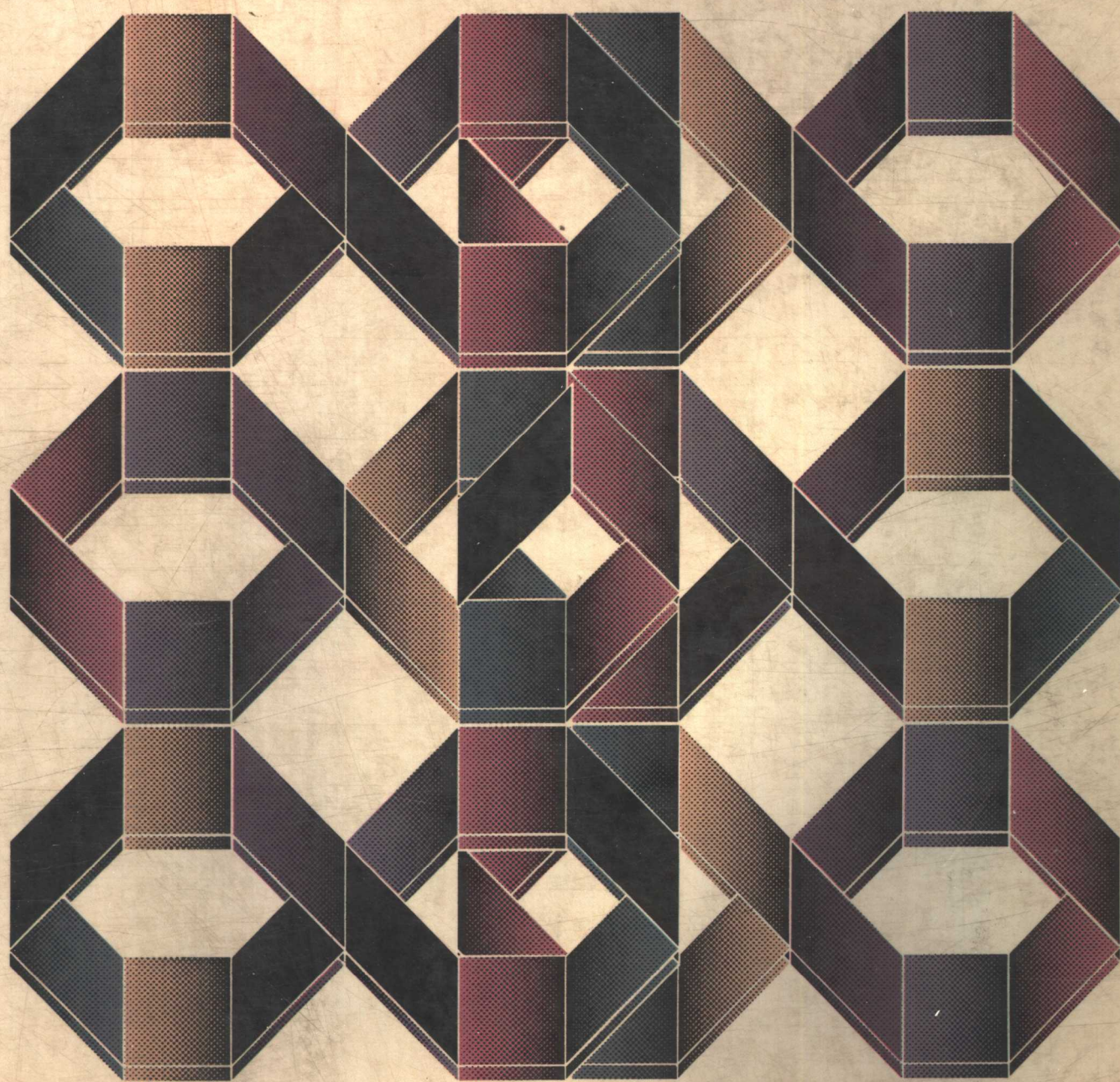


***Problem Solving with***

# **ANSI Structured BASIC**

***Rina Yarmish and Joshua Yarmish***





***Problem Solving with***

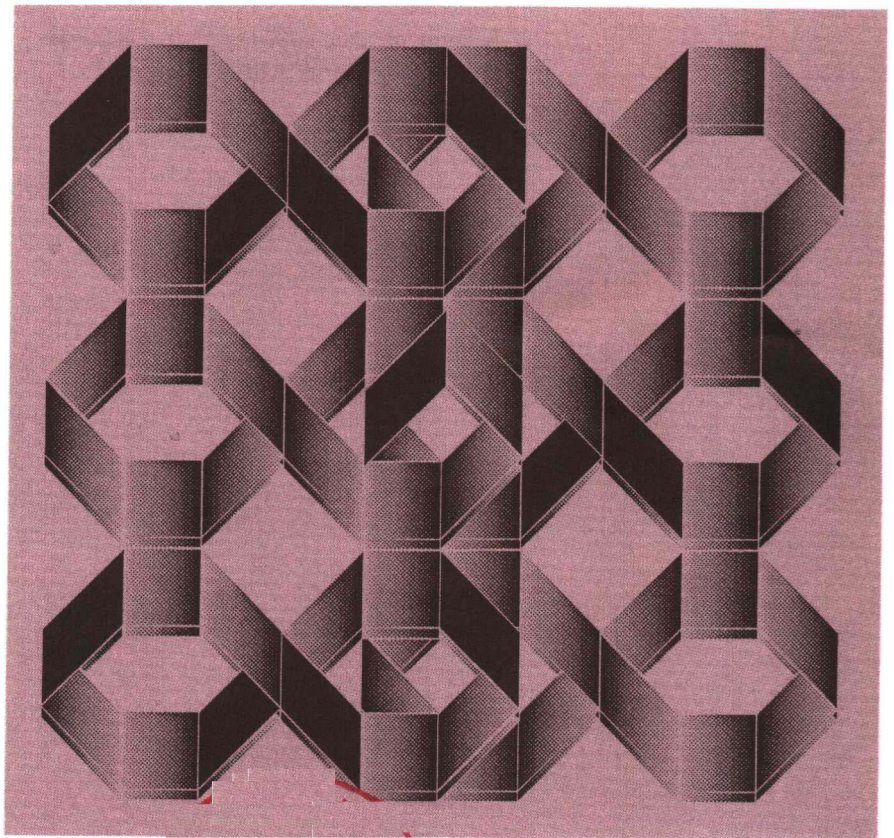
# **ANSI Structured BASIC**

***Rina Yarmish***

***Joshua Yarmish***

Kingsborough  
Community College

Pace  
University



**S R A**®

SCIENCE RESEARCH ASSOCIATES, INC.  
Chicago, Henley-on-Thames, Sydney, Toronto

An IBM Company

*To Lea Kobre, Max Kobre, Hannah Yarmish,  
and the memory of Louis Yarmish*

**Acknowledgment**

The corresponding software for this text was prepared by NKR Research, Inc., San Jose, California.

**Acquisitions Editor**

Michael Carrigg

**Project Editor**

Elizabeth Sugg

**Production Administrator**

Steve Leonardo

**Compositor**

Carlisle Publishers Services (A division of Carlisle Communications, Ltd.)

**Cover and Text Designer**

David Corona Design Associates

Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

**Library of Congress Cataloging-in-Publication Data**

Yarmish, Rina.

Problem solving with ANSI structured BASIC.

Includes index.

1. BASIC (Computer program language) 2. Structured programming.

I. Yarmish, Joshua. II. Title.

QA76.73.B3Y37 1987 005.13'3 87-9815  
ISBN 0-574-18680-8

Copyright © 1988 Science Research Associates, Inc. All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Science Research Associates, Inc.

# Introduction

*Problem Solving with ANSI Structured BASIC* introduces students to principles of program design and to computer programming using the BASIC language. No prior familiarity with computers or knowledge of programming is required.

## Approach

*Problem Solving with ANSI Structured BASIC* presents a systematic, disciplined approach to program development, beginning with an understanding and analysis of the problem to be solved and continuing throughout the design, coding, testing, and documentation phases of problem solving.

The BASIC language itself is also taught, along with presentation of a series of applications that may be processed using a computer. Moreover, since educators and computer professionals today widely accept the fact that structured design and structured programming form a solid foundation for good problem-solving methodology, this textbook is largely devoted to methods for implementing principles of good structured design and structured programming in BASIC.

In the first seven chapters, we introduce BASIC programming using an approach to problem solving that closely simulates human thinking. This enables the student to develop a "feel" for what programming is in a problem-solving environment that is not totally foreign. Beginning with chapter 8 and continuing throughout the remainder of the text, concepts of structured design and structured programming are introduced and are, from that point on, continually developed and practiced to enable structured programming to become as natural as possible to the reader. The goal is for the reader eventually to "think structure." It has been the authors' experience that this systematic introduction of structured concepts is most effective. That is, in order to develop a good understanding of what a well-structured program is, it is best for the beginner first to see problem solution in unstructured form, contrasted with structured solution to the same problem. (For those not of this opinion, we have suggested an alternate path through the text [see the paragraph on "Modularity," below] which will enable one to avoid exposure to unstructured programs.)

## Intended Audience

*Problem Solving with ANSI Structured BASIC* assumes no prior familiarity with computers. It begins with explanation and discussion of what a computer is, prior to the sections on problem solving and BASIC programming. However, while the text begins at a very elementary level, it enables the student to reach a level of proficiency at which one may create useful and interesting applications programs.

The text is written using clear, explanatory prose. The reading level is geared to that of the typical community college student or the senior college student in his or her first years of study. The textbook is intended for use in community or senior colleges as a main text for a one-semester course in BASIC programming or as an ancillary text for use in an introductory computer science or data processing course that includes BASIC programming as a major subtopic. Since explanations are complete and adequate examples and exercises are provided, the text is also suitable for self-study.

## Distinguishing Features

This book will distinguish itself from other BASIC books on the market in several important ways. First, the version of BASIC described is the latest ANSI BASIC standard, which is outstanding in that it provides instructions enabling one to write structured programs with instructions that are *truly suited* for that purpose. The ANSI BASIC instruction set allows one to incorporate structured constructs into BASIC programs so that the writing of BASIC programs may be done in a natural way and need not be "forced" through attempted simulation of those structures, as must be done with many other versions of BASIC. ANSI BASIC has brought together the simplicity of the original BASIC and important elements of modern programming languages. The resulting dialect is most suitable for business applications and educational environments.

Moreover, for the convenience of the instructor and user, and to provide absolute compatibility between text material and software, we have provided

- an ANSI BASIC interpreter that will enable users to run programs using the new standard. It is an implementation of the ANSI BASIC standard, developed by NKR Research, Inc., in San Jose, California. The interpreter includes elements of the ANSI core standard and extensions beyond the core standard, including relative (direct access) files.
- an IBM PC diskette with all the sample programs that have been assigned figure numbers in the text.

Another important feature of this book is its pedagogy, particularly its emphasis on the teaching of problem solving and design methodology. These ideas are introduced in chapter 2, while principles of structured design, stepwise refinement, and control structures are discussed comprehensively in chapter 8. Moreover, implementation of these ideas and methodologies continues throughout the text, both in discussion and in the many program examples that are provided. Discussion begins at an elementary level, with new concepts gradually and systematically appended.

Presentation of the BASIC language is clear, complete, and concise; many examples are provided to illustrate the uses of concepts discussed. The average college student may learn to program in BASIC using this text and a computer, without additional help.

Other distinguishing features include the following:

### **1. Realistic Problems and Applications**

*Problem Solving with ANSI Structured BASIC* uses a variety of realistic problems that the student may associate with the "real world." These problems will provide motivation for learning that contrived examples simply do not.

### **2. Introduction and Learning Objectives**

Each chapter begins with a motivating introduction, followed by a brief description of the main objectives of the chapter.

### **3. Instruction Summaries**

BASIC instructions described in the text are summarized in concise form. Summaries include general instruction formats, rules for usage, and examples. Each summary is boxed to distinguish it from the body of the text, to enable the reader to locate it quickly for easy reference.

### **4. Self-evaluation**

Questions for self-evaluation are provided after each main topic within a chapter. These allow students to test their understanding of current material before proceeding further. There are almost three dozen sets of self-evaluation exercises throughout the text. Answers to odd-numbered self-evaluation exercises are provided in appendix C.

### **5. "Common Errors"**

Each chapter that presents new BASIC instructions is accompanied by a section describing errors commonly committed by individuals to whom the instructions are "new." These sections will help students avoid the most common errors committed by novice programmers. They will thus help students to write their own programs and to correct program errors that are relevant to the programming techniques presented in the chapter. There are well over 100 such potential errors described in these sections.

### **6. "Interacting with the Computer"**

Each chapter that presents new BASIC instructions is accompanied by a section entitled "Interacting with the Computer." These sections use step-by-step directives to provide actual "hands-on" practice in applying the new statements and techniques in a variety of ways. Once a student has followed the steps prescribed in these sections, he or she will no longer approach the first programming implementation of these instructions "cold." The exercises are carefully designed to highlight both standard usages and eccentricities of the language features taught; they bring the instruction rules to life. There are almost 200 such exercises.

## **7. Questions, Problems, and Programming Exercises**

Each chapter is followed by carefully designed questions, problems, and—where appropriate—programming exercises. The questions and problems are designed to highlight important facts and concepts presented in the chapters. Applications chosen for programming exercises are generally of two types: (1) variations or extensions of problems in the main body of the chapter, and (2) problems that anticipate program examples of later chapters. The text contains more than 100 programming exercises and more than 300 other questions and problems.

## **8. Acting as a “Human Computer”**

Many chapter exercises request that the student determine the output of sample programs by acting as a “human computer”; that is, by “executing” the instructions by hand in the order and manner in which they would be executed by the computer itself. After determining what the output should be, students are asked to key in the programs and have the computer validate their results. These exercises force the student to visualize the progression of program logic; they thus contribute to enhanced understanding of that logic. There are about 130 such exercises throughout the text.

## **9. “Toward Deeper Understanding”**

Sections entitled “Toward Deeper Understanding” provide in-depth explanations of difficult concepts for the better student. These sections may be omitted without loss of continuity.

## **10. A Separate Introductory Chapter on Problem Solving**

Chapter 2 is fully devoted to a discussion of proper techniques for step-by-step development of algorithms. Solutions to two applications (a customer billing application and a checkbook balancing application) are then gradually developed through increasingly complex phases using appropriate methods. Students thus participate in the systematic development of increasingly complex algorithms.

## **11. Continuity**

Several programming applications are gradually developed, extended, and modified throughout the book as new techniques and methodologies are introduced. Thus, for example, algorithms for three versions of the checkbook balancing problem are developed in chapter 2; the problem is later coded in unstructured form. With the introduction of concepts of structured design in a subsequent chapter, a structured algorithm for the checkbook balancing problem is developed, followed by the corresponding structured programs in yet later chapters.

Presentation of the same or similar applications in different contexts using different techniques enables the reader to concentrate on the methodologies without becoming overly involved with the details of a particular application. This type of development—the provision of unstructured algorithms, unstructured program solutions, structured algorithms, and structured program solutions to the same problem—is used for a wide variety of applications. (As noted above, there are those who are of the opinion that avoiding exposure to unstructured programs is best; those individuals may follow a somewhat modified path through the text [see the paragraph on “Modularity,” below] which will enable one to avoid such exposure.)

## **12. Programming Aids**

Proper design and documentation of computer programs are enhanced by the use of traditional program flowcharts, and—for structured programming—structured flowcharts (also called Nassi-Schneiderman charts), which are particularly well suited for clear, concise illustration of control structures. Flowcharting techniques are discussed and proper flowchart form and usage are illustrated throughout the text in program examples. Pseudocode is also emphasized and used as a program design tool. The book contains more than 140 solved programs, more than 100 flowcharts, and some two dozen illustrations using pseudocode.

## **13. Up-to-Date**

The latest American National Standard BASIC has been adhered to throughout. This standard reflects

the industry's current emphasis on structured programming.

## **14. Accuracy**

All programs provided in the text have been tested and debugged. Actual computer output is provided for most numbered figures. Moreover, since diskettes containing all programs that have been included in numbered figures in the text are available to instructors, it is a simple matter to run these programs using the interpreter provided with the text and to observe the output.

## **15. Emphasis on Programming Style**

The text emphasizes an organized, disciplined approach to both the planning and coding of problem solutions. Proper indentation within programs has been used for visual clarity.

## **16. Structured Programming**

Particular attention has been paid to the design of programs using the control structures of structured programming. An entire chapter (chapter 8) is first devoted to a "language-free" discussion of the sequence, selection, and iteration structures and their implementation in structured programming. Chapters 9 and 10 present the BASIC statements to implement those structures. Structured style is standard in all subsequent chapters.

## **17. Coverage**

The text includes comprehensive coverage of some oft-neglected topics:

*Character Strings.* Basic material on character strings and string manipulation is provided in earlier chapters. However, chapter 14 is devoted to more comprehensive coverage of character constants, variables, and expressions; substrings; string subprograms, including those using built-in string functions and user-defined subprograms; and comparison of character expressions. Seventeen program examples in chapter 14 illustrate the uses of character strings.

*Matrices.* ANSI BASIC provides MAT statements for array handling, in addition to the treatment of

arrays with subscripted variables covered in chapter 13. Chapter 16 provides detailed coverage of the MAT statements, including input/output operations, built-in numeric array functions, and numerous operations with both numeric and string arrays.

*External Files and File Processing.* Working with files is crucial to many applications in industry; however, files and file processing are often difficult topics for beginning programmers. We believe that this difficulty is often related to weakness in understanding underlying concepts. For this reason, we have devoted an entire unit to the study of files. A separate chapter (chapter 17) provides a language-free presentation of file-related concepts: data organization, auxiliary storage, file organization and access mode, and file maintenance methodologies, including sequential and relative updating. This presentation is followed (in chapter 18) by a discussion of file implementation in ANSI BASIC, including comprehensive treatment of file updating with sequential and relative files. More than twenty program examples in chapter 18 illustrate various file-handling techniques.

*Debugging.* A separate appendix (appendix B) is devoted to discussion of debugging, (finding and correcting program errors). This appendix material may be covered at any point within the course sequence that the instructor feels is appropriate. However, as all programmers realize, debugging is an ongoing process, and the possibility of mishap should be kept in mind from the very beginning of the programming process. Therefore, each chapter that presents new instructions includes a section enumerating "common errors" to avoid; moreover, exercises requesting the student to "act as a human computer" are "trace" exercises that force awareness of the logical flow of programs throughout the learning process.

## **18. Modularity**

The instructor has great flexibility in choice and arrangement of topics, after coverage of fundamental "core" chapters. For those who wish to introduce students to structured concepts after they have obtained a "feel" for the language in the absence of structure, unit 1 ("The Computer and Problem Solving," chapters 1 and 2) and unit 2 ("BASIC



Language Fundamentals," chapters 3–7) should be covered first, followed by the first three chapters (chapters 8, 9, and 10) of unit 3 ("Structure and Style: Modular Program Design and Structured Programming"). Substantial flexibility is provided from that point on, as indicated below:

- (a) Chapters 1–10 are to be studied first.
- (b) The following may be covered *in any order* after chapter 10:
  - Chapter 11 ("Subprograms—Part I: Functions") and chapter 12 ("Subprograms—Part II: Subroutines").
  - Chapter 13 ("Arrays and Subscripted Variables"). Section G of this chapter should be deferred until after chapters 11 and 12.
  - Chapter 14 ("More on Character Strings"). Section E of this chapter should be deferred until after chapters 11 and 12.
  - Chapter 15 ("More on Input/Output").
  - Chapter 16 ("Matrices: The MAT Statements").
  - Chapter 17 ("Introduction to Files: A General Overview") and chapter 18 ("File Implementation in BASIC"). For students who are familiar with concepts of external files from their study of other languages, chapter 17 may be omitted without loss of continuity.

In other words, following chapter 10, the instructor may choose to cover any of the succeeding chapters of the text in any order—no one chapter depending on any other chapter—with the exception of chapter 12 (which should follow chapter 11) and the possible exception of chapter 18 (which should follow chapter 17 for students with no prior exposure to file concepts). This modularity affords great flexibility of sequencing and of topical selection.

Instructors who wish to avoid unstructured programs may choose the following path:

- (a) Chapters 1–5
- (b) In chapter 6, brief mention of material in section A, full coverage of section B, cursory treatment of section D, and full coverage of section F. (Omit sections C, E, and G.) Omit chapter 7.
- (c) Chapters 8, 9, and 10.

- (d) Any of chapters 11–12, 13, 14, 15, 16, or 17–18 may follow in any order, as described above.

Within the limitations described above, the instructor is free to determine the sequence of instruction and to emphasize the topics he or she feels are appropriate to the course. Similarly, the student is free to move to topics of interest without having to be concerned about missing prerequisite material. The course, in other words, need not be textbook-directed, but may be directed by the instructor or by the independent reader.

## 19. Textbook Supplements

This text is supplemented with

- the ANSI BASIC interpreter, discussed above. (The provision of this software presents a unique opportunity to teach the latest version of BASIC, one that enables the writing of structured BASIC programs in a dialect to which structure is "natural.")
- IBM PC diskettes with all programs that have been included in numbered figures in the text.
- an instructor's manual, including answers to text questions and exercises.
- transparency masters of illustrative text material, flowcharts, structured flowcharts, and program listings.
- computerized test bank.

We wish to express our sincere thanks and appreciation to all those who have contributed to the development of this book: to the reviewers (R. Ken Walter, Weber State College; Brian A. Rudolph, Bowling Green State University; Robert Norton, Mesa College; George Novacky, University of Pittsburgh; Ralph A. Szweda, Monroe Community College), whose constructive criticisms were very helpful in shaping and polishing the final manuscript; to our students, from whom we have always learned a great deal; and to the people at SRA who were so enthusiastic and who contributed so much time, energy, and talent to bringing this complex package to completion. We are particularly indebted to our editor, Michael Carrigg, who orchestrated this project



with enthusiasm and with vision; and to our developmental editor, Elizabeth Sugg, whose masterful coordination of activities was invaluable to the production of a superior work on a timely basis.

Comments and suggestions for improvement will be much appreciated and will be given careful consideration by the authors. Please direct all communication to either author.

We sincerely hope that you enjoy reading, studying, and learning from this text. If you do and you feel that you have gained in knowledge and in expertise through its use, the authors will be well rewarded.

Dr. Joshua Yarmish  
Pace University  
Pace Plaza  
New York, New York 10038

Dr. Rina Yarmish  
Kingsborough Community College  
2001 Oriental Boulevard  
Brooklyn, New York 11235

# Contents

Introduction    *xiii*

---

## Unit **1**    **The Computer and Problem Solving**    **1**

---

### **Chapter 1**

#### **Computers and Programming: An Overview    3**

*A Preliminary Note to the Reader    3*

*Introduction and Learning  
Objectives    4*

- A. What Is a Computer?    5
  - 1. Basic Components of a Computer    6
  - 2. The Central Processing Unit (CPU)    9
  - 3. Communicating with the Computer    9
  - 4. Peripheral Equipment    9
  - 5. How the Computer Executes a Program    10
  - 6. The Computer System    12
- B. What Is Programming?    13
- C. What Is a Programming Language?    14
  - 1. Machine Language    14
  - 2. Symbolic Language    15
  - 3. Programming Language Instructions    16

*Questions    17*

### **Chapter 2**

#### **Problem Solving    19**

*Introduction and Learning  
Objectives    19*

- A. What Is a "Problem" and What Is "Problem Solution"?    20
- B. Problem Solving: A Stepwise Approach    20
  - 1. Understanding the Problem: Describing the Problem Verbally    21
  - 2. Analyzing the Problem: Establishing the General Approach    21
  - 3. Developing Algorithms: Stepwise Procedures for Problem Solution    22
  - 4. Implementing the Algorithm    26
- C. Techniques and Tools for Describing Algorithms    26
  - 1. Flowcharts    27
  - 2. Examples    28
  - 3. Some Facts about Flowcharting    35
- D. Coding: Implementing the Algorithm Using the Computer    35

*Questions and Exercises    37*

---

## Unit **2**    **BASIC Language Fundamentals**    **41**

---

### **Chapter 3**

#### **Introduction to BASIC    43**

*Introduction and Learning  
Objectives    43*

- A. The BASIC Language    44
- B. Our First BASIC Program    45
- C. Statement Number and Statement Length    46

*Self-evaluation 1    47*

D. General Format of a BASIC Statement    47

E. The END Statement    48

- 1. Summary of Rules: The END Statement    48

F. Documentation within the Program: The REM Statement and Tail Comments    48

- 1. Summary of Rules: The REM Statement    49

*Self-evaluation 2    49*

G. Entering and Executing a BASIC Program	50
1. Getting Started	50
2. Creating and Editing Program Files	52
3. Useful DOS Commands	56
Questions and Exercises	58
Interacting with the Computer	59

## **Chapter 4**

### **Constants, Variables, and Arithmetic Expressions 61**

Introduction and Learning Objectives	61
A. Constants and Variables	62
1. Numeric Constants and Variables	62
2. Character Constants and Variables	64
Self-evaluation 1	65
B. Arithmetic Expressions	66
1. Arithmetic Operators	66
2. Hierarchy of Arithmetic Operators	67
3. Parentheses in Arithmetic Expressions	68
Self-evaluation 2	68
C. The Assignment Statement: The LET Statement	69
1. Simultaneous Assignment of a Value to a List of Variables	70
2. Summary of Rules: The LET Statement	71
Self-evaluation 3	71
Common Errors	72
Questions and Exercises	72
Interacting with the Computer	73

## **Chapter 5**

### **Input/Output: Reading Data into the Computer and Writing Information Out 75**

Introduction and Learning Objectives	75
A. The READ and DATA Statements	76

1. Summary of Rules: The DATA and READ Statements	79
Self-evaluation 1	80
B. The PRINT Statement	80
1. Printed Line Format	82
2. Using the Semicolon to Pack Print Zones	83
3. Using the Comma and Semicolon as Continuation Characters	84
4. Summary of Rules: The PRINT Statement	85
C. Program Examples	86
Self-evaluation 2	87
D. The INPUT Statement	88
1. Summary of Rules: The INPUT Statement	90
2. Prompting Program Input	90
Self-evaluation 3	91
Common Errors	92
Questions and Exercises	92
Interacting with the Computer	93
Programming Exercises	94

## **Chapter 6**

### **Branching and Looping: Controlling Program Logic Flow 97**

Introduction and Learning Objectives	97
A. The GOTO Statement	98
1. Summary of Rules: The GOTO Statement	100
Self-evaluation 1	100
B. Relational and Logical Expressions	101
1. Relational Expressions	101
2. Logical Expressions	101
C. The IF-THEN Statement	103
1. Summary of Rules: The IF-THEN Statement	105
Self-evaluation 2	106
D. Looping	107
1. Counters and Accumulators	107
2. Testing for the "Trailer Value"	108
3. Loop Structure	110



E. Program Examples	111
F. Testing for End of Data Using a READ Statement	116
<i>Self-evaluation 3</i>	117
G. The Computed GOTO Statement	117
1. Summary of Rules: The Computed GOTO Statement	119
2. Program Example	119
<i>Self-evaluation 4</i>	121
H. The STOP Statement	122
1. Summary of Rules: The STOP Statement	122
Common Errors	122
Questions and Exercises	123
Interacting with the Computer	124
Programming Exercises	126

## **Chapter 7**

### **BASIC Program Examples 127**

*Introduction and Learning Objectives* 127

Problem 1: Letter-Grade Assignment	127
Problem 2: Compound Interest	128
Problem 3: Payroll Computation	130
Problem 4: Customer Billing	132
Problem 5: Balancing Checking Accounts	135
Problem 6: Tax Computation	137
Problem 7: An Interactive Game	139
Problem 8: Menu-driven Program	140
Programming Exercises	143

---

# **Unit 3**

## **Structure and Style: Modular Program Design and Structured Programming 149**

---

### **Chapter 8**

#### **The Structured Approach to Problem Solving 151**

*Introduction and Learning Objectives* 151

A. Incentives for the Structured Approach	152
B. Top-down Design	152
C. Structured Programming	153
1. Control Structures	154
2. One Entry and One Exit	161
3. Combining Control Structures	161
4. Structured Programming and GOTO-less Programming	161
D. Structured Flowcharts	161
E. Pseudocode	163
F. Examples	166
G. Toward Deeper Understanding: Avoiding Exiting from the Middle of a Loop	170
Questions and Exercises	173

### **Chapter 9**

#### **Structured Programming Mechanisms—Part I: Mechanisms for Looping 177**

*Introduction and Learning Objectives* 177

A. FOR Loops	178
1. The EXIT FOR Statement	180
2. Nested FOR Loops	182
3. Summary of Rules: FOR Loops	183
4. Program Examples	184
<i>Self-evaluation 1</i>	185
B. DO Loops	186
1. Pretest DO Loops	187
2. Posttest DO Loops	189
3. Other Variations of DO Loops	191
4. The EXIT DO Statement	192
5. Nested DO Loops	192
6. Summary of Rules: DO Loops	194
7. Program Examples	195
<i>Self-evaluation 2</i>	195

- C. Toward Deeper Understanding:  
Expressing the FOR Loop in Terms of  
a DO Loop 199

<i>Common Errors</i>	199
<i>Questions and Exercises</i>	200
<i>Interacting with the Computer</i>	202
<i>Programming Exercises</i>	203

## **Chapter 10**

### **Structured Programming Mechanisms—Part II: Mechanisms for Decision Making 205**

<i>Introduction and Learning Objectives</i>	205
A. IF Blocks	205
1. Nested Combinations of Loop and Decision Structures	208
2. The ELSEIF Statement	209
3. Summary of Rules: IF Blocks	212
4. Program Examples	213
<i>Self-evaluation 1</i>	213
B. SELECT Blocks	216
1. Summary of Rules: SELECT Blocks	219
2. Program Examples	221
<i>Self-evaluation 2</i>	224
<i>Common Errors</i>	224
<i>Questions and Exercises</i>	225
<i>Interacting with the Computer</i>	227
<i>Programming Exercises</i>	228

## **Chapter 11**

### **Subprograms—Part I: Functions 229**

<i>Introduction and Learning Objectives</i>	229
A. Built-in Functions	231
1. Some Common Built-in Functions	231
2. Random Numbers: The RND Function	234

- 3. The RANDOMIZE Statement 236
- Self-evaluation 1* 238

B. User-defined Functions	239
1. One-Line Functions: The DEF Statement	239
<i>Self-evaluation 2</i>	242
2. Multiline Functions: The FUNCTION and END FUNCTION Statements	242
<i>Self-evaluation 3</i>	254

<i>Common Errors</i>	255
<i>Questions and Exercises</i>	256
<i>Interacting with the Computer</i>	259
<i>Programming Exercises</i>	260

## **Chapter 12**

### **Subprograms—Part II: Subroutines 263**

<i>Introduction and Learning Objectives</i>	263
A. Defining and Using Subroutines: The SUB, END SUB, and CALL Statements	264
1. The EXIT SUB Statement	266
2. Arguments and Parameters in Subroutines	266
3. External Subroutines, Program Units, and Data	268
4. Recursive Subroutines	269
5. Summary of Rules: Subroutines	270
6. Program Examples	271
<i>Self-evaluation 1</i>	272
B. Modular Programming	275
1. Modular Independence	275
2. Program Example	276
<i>Common Errors</i>	277
<i>Questions and Exercises</i>	278
<i>Interacting with the Computer</i>	279
<i>Programming Exercises</i>	280

---

## Unit **4** Additional Elements of ANSI BASIC 283

---

### **Chapter 13** **Arrays and Subscripted Variables 285**

*Introduction and Learning Objectives 285*

- A. Subscripted Variables 286
  - 1. Array Names 287
- B. Creating Arrays: The DIM Statement 288
  - 1. Subscript Bounds 288
  - 2. Summary of Rules: The DIM Statement 290
- C. Using Arrays 290
  - 1. Programming for One-dimensional Arrays 292
  - 2. Programming for Two-dimensional Arrays 292
- Self-evaluation 1 296*
- D. Program Examples 297
- E. Sorting Algorithms 304
  - 1. Bubble Sort 304
  - 2. Selection-with-Exchange Sort 306
  - 3. Efficiency of Bubble Sort v. Selection-with-Exchange Sort 308
- F. Searching Algorithms 310
  - 1. Linear or Sequential Search 310
  - 2. Binary Search 310
  - 3. Efficiency of Linear Search v. Binary Search 313
- G. Arrays in Subprograms 315
  - 1. Built-in Array Functions 315
  - 2. Program Examples 316

*Common Errors 318*  
*Questions and Exercises 318*  
*Interacting with the Computer 323*  
*Programming Exercises 324*

### **Chapter 14** **More on Character Strings 327**

*Introduction and Learning Objectives 327*

- A. The BASIC Character Set 328
- B. Character Constants and Variables 328
- C. Character Expressions 328
- D. Substrings 329
  - 1. Program Example 331
- Self-evaluation 1 332*
- E. String Subprograms 332
  - 1. Built-in String Functions 332
  - 2. User-defined Subprograms 336
- Self-evaluation 2 337*
- F. Comparing Character Expressions 337
- G. Program Examples 339
- Common Errors 348*  
*Questions and Exercises 349*  
*Interacting with the Computer 351*  
*Programming Exercises 352*

### **Chapter 15** **More on Input/Output 355**

*Introduction and Learning Objectives 355*

- A. The RESTORE Statement 356
  - 1. Summary of Rules: The RESTORE Statement 357
- B. The LINE INPUT Statement 358
  - 1. Summary of Rules: The LINE INPUT Statement 359
- C. Prompting Input within the INPUT and LINE INPUT Statements 359
  - 1. Summary of Rules: The INPUT and LINE INPUT Statements with the Optional Prompt Message 360
- Self-evaluation 1 360*
- D. The TAB Function 361
  - 1. Summary of Rules: The TAB Function 362
- E. The PRINT USING Statement 363
  - 1. Printing Formatted Numbers 364
  - 2. Printing Formatted Strings 369
  - 3. Summary of Rules: The PRINT USING Statement 370



Self-evaluation 2	372
Common Errors	373
Questions and Exercises	374
Interacting with the Computer	375
Programming Exercises	378

## **Chapter 16**

### **Matrices: The MAT Statements 379**

*Introduction and Learning Objectives 379*

A. Inputting Arrays	380
1. The Array Read Statement: The MAT READ Statement	380
2. Array Input Statements: The MAT INPUT and MAT INPUT PROMPT Statements	381
3. Array Line Input Statements: The MAT LINE INPUT and MAT LINE INPUT PROMPT Statements	382
4. Redimensioning Arrays	383
5. Input of Variable-Length, One-dimensional Arrays	384
6. Summary of Rules: Inputting Arrays	385

B. Printing Arrays	386
1. The Array Print Statement: The MAT PRINT Statement	386
2. The Array Formatted Print Statement: The MAT PRINT USING Statement	387
3. Summary of Rules: Printing Arrays	388

*Self-evaluation 1 389*

C. Numeric Array Operations	390
1. Numeric Array Assignment	390
2. Numeric Array Operators	391
3. Built-in Numeric Array Functions	394
4. Other Numeric Array MAT Statements	397
D. String Array Operations	398
1. String Array Assignment	398
2. The String Array Operator	399
3. The Null String Assignment Statement	400

*Self-evaluation 2 400*

E. Program Examples	401
---------------------	-----

*Common Errors 406*

*Questions and Exercises 408*

*Interacting with the Computer 410*

*Programming Exercises 412*

# **Unit 5**

## **Files and File Processing**

**415**

### **Chapter 17**

#### **Introduction to Files: A General Overview 417**

*Introduction and Learning Objectives 417*

A. Data Organization: Characters, Fields, Records, Files	418
B. Representation of Data	418
1. Characters and Numbers	420
C. Auxiliary Storage	421
1. Characteristics of Auxiliary Storage	421
2. Magnetic Tape	422
3. Magnetic Disk	422
D. File Organization and Modes of Access	423

1. Sequential File Organization	423
2. Direct or Relative File Organization	423

E. File Maintenance	424
---------------------	-----

1. Sequential Updating	425
------------------------	-----

2. Random Updating	425
--------------------	-----

F. Batch Processing v. Real-Time Processing	426
---	-----

*Questions 426*

### **Chapter 18**

#### **File Implementation in BASIC 429**

*Introduction and Learning Objectives 429*

<i>Overview</i>	430		
A. Opening and Closing Files: The OPEN and CLOSE Statements	432		
1. The OPEN Statement	432		
2. The CLOSE Statement	436		
B. Erasing Files: The ERASE Statement	436		
C. Inquiring about a File: The ASK Statement	437		
<i>Self-evaluation 1</i>	438		
D. Sequential Display Files	439		
1. Output and Input	439		
2. Program Examples	443		
3. Controlling the File Pointer: The SET POINTER Statement	447		
4. Program Example: Sequential File Updating	449		
<i>Self-evaluation 2</i>	452		
E. Sequential Internal and Stream Internal Files	452		
		1. Output and Input: The WRITE and READ Statements	453
		2. Program Examples	456
		<i>Self-evaluation 3</i>	459
		F. Relative Internal Files	460
		1. The SET RECORD Statement	460
		2. Output and Input	461
		3. Changing and Deleting a Record: The REWRITE and DELETE Statements	462
		4. Program Examples: Relative File Updating	464
		<i>Self-evaluation 4</i>	467
		<i>Common Errors</i>	470
		<i>Questions and Exercises</i>	471
		<i>Interacting with the Computer</i>	472
		<i>Programming Exercises</i>	474

---

## Unit 6 Appendixes

477

---

<i>Appendix A: The BASIC Character Set</i>	479
<i>Appendix B: Debugging Programs</i>	481
<i>Appendix C: Solutions to Odd Numbered Self-evaluation Exercises</i>	486

---

**Index 495**



# The Computer and Problem Solving

---

# 1