Preface xxi

1 Introduction to Computers and the Fortran Language 1

1.1 The Computer 2
  1.1.1 The CPU / 1.1.2 Main and Secondary Memory / 1.1.3 Input and Output Devices

1.2 Data Representation in a Computer 4
  1.2.1 The Binary Number System / 1.2.2 Octal and Hexadecimal Representations of Binary Numbers / 1.2.3 Types of Data Stored in Memory

1.3 Computer Languages 12

1.4 The History of the Fortran Language 13

1.5 The Evolution of Fortran 16

1.6 Summary 19
  1.6.1 Exercises

2 Basic Elements of Fortran 22

2.1 Introduction 22

2.2 The Fortran Character Set 23

2.3 The Structure of a Fortran Statement 24
  2.4.1 The Declaration Section / 2.4.2 The Execution Section / 2.4.3 The Termination Section / 2.4.4 Program Style /
  2.4.5 Compiling, Linking, and Executing the Fortran Program

2.4 The Structure of a Fortran Program 24

2.5 Constants and Variables 28
  2.5.1 Integer Constants and Variables / 2.5.2 Real Constants and Variables / 2.5.3 Character Constants and Variables / 2.5.4 Default and Explicit Variable Typing /
  2.5.5 Keeping Constants Consistent in a Program
2.6 Assignment Statements and Arithmetic Calculations 36
   2.6.1 Integer Arithmetic / 2.6.2 Real Arithmetic / 2.6.3 Hierarchy of Operations / 2.6.4 Mixed-Mode Arithmetic / 2.6.5 Mixed-Mode Arithmetic and Exponentiation

2.7 Intrinsic Functions 47

2.8 List-Directed Input and Output Statements 49

2.9 Initialization of Variables 55

2.10 The IMPLICIT NONE Statement 57

2.11 Program Examples 58

2.12 Debugging Fortran Programs 66

2.13 Summary 68
   2.13.1 Summary of Good Programming Practice / 2.13.2 Summary of Fortran Statements / 2.13.3 Exercises

3 Program Design and Branching Structures 82
   3.1 Introduction to Top-Down Design Techniques 83
   3.2 Use of Pseudocode and Flowcharts 87
   3.3 Logical Constants, Variables, and Operators 90
      3.3.1 Logical Constants and Variables / 3.3.2 Assignment Statements and Logical Calculations / 3.3.3 Relational Operators / 3.3.4 Combinational Logic Operators / 3.3.5 Logical Values in Input and Output Statements / 3.3.6 The Significance of Logical Variables and Expressions
   3.4 Control Constructs: Branches 96
      3.4.1 The Block IF Construct / 3.4.2 The ELSE and ELSE IF Clauses / 3.4.3 Examples Using Block IF Constructs / 3.4.4 Named Block IF Constructs / 3.4.5 Notes Concerning the Use of Block IF Constructs / 3.4.6 The Logical IF Statement / 3.4.7 The SELECT CASE Construct
   3.5 More on Debugging Fortran Programs 119
   3.6 Summary 120
      3.6.1 Summary of Good Programming Practice / 3.6.2 Summary of Fortran Statements and Constructs / 3.6.3 Exercises

4 Loops and Character Manipulation 128
   4.1 Control Constructs: Loops 128
      4.1.1 The While Loop / 4.1.2 The DO WHILE Loop / 4.1.3 The Iterative or Counting Loop / 4.1.4 The CYCLE and EXIT Statements / 4.1.5 Named Loops / 4.1.6 Nesting Loops and Block IF Constructs
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.2</td>
<td>Character Assignments and Character Manipulations</td>
<td>156</td>
</tr>
<tr>
<td></td>
<td>4.2.1 Character Assignments / 4.2.2 Substring Specifications</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.2.3 The Concatenation (//) Operator /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.2.4 Relational Operators with Character Data /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.2.5 Character Intrinsic Functions</td>
<td></td>
</tr>
<tr>
<td>4.3</td>
<td>Debugging Fortran Loops</td>
<td>170</td>
</tr>
<tr>
<td>4.4</td>
<td>Summary</td>
<td>171</td>
</tr>
<tr>
<td></td>
<td>4.4.1 Summary of Good Programming Practice / 4.4.2 Summary of Fortran</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Statements and Constructs / 4.4.3 Exercises</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Basic I/O Concepts</td>
<td>183</td>
</tr>
<tr>
<td>5.1</td>
<td>Formats and Formatted WRITE Statements</td>
<td>183</td>
</tr>
<tr>
<td>5.2</td>
<td>Output Devices</td>
<td>185</td>
</tr>
<tr>
<td>5.3</td>
<td>Format Descriptors</td>
<td>188</td>
</tr>
<tr>
<td></td>
<td>5.3.1 Integer Output—The I Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.2 Real Output—The F Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.3 Real Output—The E Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.4 True Scientific Notation—The ES Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.5 Logical Output—The L Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.6 Character Output—The A Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.7 Horizontal Positioning—The X and T Descriptors /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.8 Repeating Groups of Format Descriptors /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.9 Changing Output Lines—The Slash (I) Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.3.10 How Formats Are Used During WR ITEs</td>
<td></td>
</tr>
<tr>
<td>5.4</td>
<td>Formatted READ Statements</td>
<td>209</td>
</tr>
<tr>
<td></td>
<td>5.4.1 Integer Input—The I Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.4.2 Real Input—The F Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.4.3 Logical Input—The L Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.4.4 Character Input—The A Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.4.5 Horizontal Positioning—The X and T Descriptors /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.4.6 Vertical Positioning—The Slash (I) Descriptor /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.4.7 How Formats Are Used During READs</td>
<td></td>
</tr>
<tr>
<td>5.5</td>
<td>An Introduction to Files and File Processing</td>
<td>215</td>
</tr>
<tr>
<td></td>
<td>5.5.1 The OPEN Statement /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.5.2 The CLOSE Statement /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.5.3 READs and WR ITEs to Disk Files /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.5.4 The IOSTAT= and IOMSG= Clauses in the READ Statement /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.5.5 File Positioning</td>
<td></td>
</tr>
<tr>
<td>5.6</td>
<td>Summary</td>
<td>237</td>
</tr>
<tr>
<td></td>
<td>5.6.1 Summary of Good Programming Practice / 5.6.2 Summary of Fortran</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Statements and Structures /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5.6.3 Exercises</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Introduction to Arrays</td>
<td>251</td>
</tr>
<tr>
<td>6.1</td>
<td>Declaring Arrays</td>
<td>252</td>
</tr>
<tr>
<td>6.2</td>
<td>Using Array Elements in Fortran Statements</td>
<td>253</td>
</tr>
<tr>
<td></td>
<td>6.2.1 Array Elements Are Just Ordinary Variables /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.2.2 Initialization of Array Elements /</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6.2.3 Changing the</td>
<td></td>
</tr>
</tbody>
</table>
**TABLE OF CONTENTS**

**6. Subscript Range of an Array**
- 6.2.4 Out-of-Bounds Array Subscripts
- 6.2.5 The Use of Named Constants with Array Declarations

**6.3 Using Whole Arrays and Array Subsets in Fortran Statements**
- 6.3.1 Whole Array Operations
- 6.3.2 Array Subsets

**6.4 Input and Output**
- 6.4.1 Input and Output of Array Elements
- 6.4.2 The Implied DO Loop
- 6.4.3 Input and Output of Whole Arrays and Array Sections

**6.5 Example Problems**

**6.6 When Should You Use an Array?**

**6.7 Summary**
- 6.7.1 Summary of Good Programming Practice
- 6.7.2 Summary of Fortran Statements and Constructs
- 6.7.3 Exercises

**7. Introduction to Procedures**

**7.1 Subroutines**
- 7.1.1 Example Problem—Sorting
- 7.1.2 The INTENT Attribute
- 7.1.3 Variable Passing in Fortran: The Pass-By-Reference Scheme
- 7.1.4 Passing Arrays to Subroutines
- 7.1.5 Passing Character Variables to Subroutines
- 7.1.6 Error Handling in Subroutines
- 7.1.7 Examples

**7.2 Sharing Data Using Modules**

**7.3 Module Procedures**
- 7.3.1 Using Modules to Create Explicit Interfaces

**7.4 Fortran Functions**
- 7.4.1 Unintended Side Effects in Functions

**7.5 Passing Procedures as Arguments to Other Procedures**
- 7.5.1 Passing User-Defined Functions as Arguments
- 7.5.2 Passing Subroutines as Arguments

**7.6 Summary**
- 7.6.1 Summary of Good Programming Practice
- 7.6.2 Summary of Fortran Statements and Structures
- 7.6.3 Exercises

**8. Additional Features of Arrays**

**8.1 Two-Dimensional or Rank-2 Arrays**
- 8.1.1 Declaring Rank-2 Arrays
- 8.1.2 Rank-2 Array Storage
- 8.1.3 Initializing Rank-2 Arrays
- 8.1.4 Example Problem
- 8.1.5 Whole Array Operations and Array Subsets

**8.2 Multidimensional or Rank-n Arrays**
8.3 Using Fortran Intrinsic Functions with Arrays
   8.3.1 Elemental Intrinsic Functions / 8.3.2 Inquiry Intrinsic Functions / 8.3.3 Transformational Intrinsic Functions

8.4 Masked Array Assignment: The WHERE Construct
   8.4.1 The WHERE Construct / 8.4.2 The WHERE Statement

8.5 The FORALL Construct
   8.5.1 The Form of the FORALL Construct / 8.5.2 The Significance of the FORAL L Construct / 8.5.3 The FORAL L Statement

8.6 Allocatable Arrays

8.7 Summary
   8.7.1 Summary of Good Programming Practice / 8.7.2 Summary of Fortran Statements and Constructs / 8.7.3 Exercises

9 Additional Features of Procedures
   9.1 Passing Multidimensional Arrays to Subroutines and Functions
      9.1.1 Explicit-Shape Dummy Arrays / 9.1.2 Assumed-Shape Dummy Arrays / 9.1.3 Assumed-Size Dummy Arrays

   9.2 The SAVE Attribute and Statement

   9.3 Allocatable Arrays in Procedures

   9.4 Automatic Arrays in Procedures
      9.4.1 Comparing Automatic Arrays and Allocatable Arrays / 9.4.2 Example Program

   9.5 Allocatable Arrays in Fortran 2003 Procedures
      9.5.1 Allocatable Dummy Arguments / 9.5.2 Allocatable Functions

   9.6 Pure and Elemental Procedures
      9.6.1 Pure Procedures / 9.6.2 Elemental Procedures

   9.7 Internal Procedures

   9.8 Summary
      9.8.1 Summary of Good Programming Practice / 9.8.2 Summary of Fortran Statements and Structures / 9.8.3 Exercises

10 More about Character Variables
   10.1 Character Comparison Operations
      10.1.1 The Relational Operators with Character Data / 10.1.2 The Lexical Functions L LT, LLE, L GT, and L GE

   10.2 Intrinsic Character Functions

   10.3 Passing Character Variables to Subroutines and Functions
# Table of Contents

10.4 Variable-Length Character Functions 471
10.5 Internal Files 474
10.6 Example Problem 475
10.7 Summary
  10.7.1 Summary of Good Programming Practice /
  10.7.2 Summary of Fortran Statements and Structures /
  10.7.3 Exercises

11 Additional Intrinsic Data Types 487
11.1 Alternative Kinds of the REAL Data Type 487
  11.1.1 Kinds of REAL Constants and Variables /
  11.1.2 Determining the KIND of a Variable / 11.1.3 Selecting Precision in a Processor-Independent Manner /
  11.1.4 Mixed-Mode Arithmetic / 11.1.5 Higher Precision Intrinsic Functions / 11.1.6 When to Use High-Precision Real Values / 11.1.7 Solving Large Systems of Simultaneous Linear Equations
11.2 Alternative Lengths of the INTEGER Data Type 511
11.3 Alternative Kinds of the CHARACTER Data Type 513
11.4 The COMPLEX Data Type 514
  11.4.1 Complex Constants and Variables / 11.4.2 Initializing Complex Variables / 11.4.3 Mixed-Mode Arithmetic / 11.4.4 Using Complex Numbers with Relational Operators / 11.4.5 COMPLEX Intrinsic Functions
11.5 Summary
  11.5.1 Summary of Good Programming Practice /
  11.5.2 Summary of Fortran Statements and Structures /
  11.5.3 Exercises

12 Derived Data Types 530
12.1 Introduction to Derived Data Types 530
12.2 Working with Derived Data Types 532
12.3 Input and Output of Derived Data Types 533
12.4 Declaring Derived Data Types in Modules 534
12.5 Returning Derived Types from Functions 543
12.6 Dynamic Allocation of Derived Data Types (Fortran 2003 only) 547
12.7 Parameterized Derived Data Types (Fortran 2003 only) 548
12.8 Type Extension (Fortran 2003 only) 549
12.9 Type-Bound Procedures 550
12.10 The ASSOCIATE Construct (Fortran 2003 only) 555
12.11 Summary  
   12.11.1 Summary of Good Programming Practice  
   12.11.2 Summary of Fortran Statements and Structures  
   12.11.3 Exercises

13 Advanced Features of Procedures and Modules  
   13.1 Scope and Scoping Units  
   13.2 Recursive Procedures  
   13.3 Keyword Arguments and Optional Arguments  
   13.4 Procedure Interfaces and Interface Blocks  
      13.4.1 Creating Interface Blocks  
      13.4.2 Notes on the Use of Interface Blocks  
   13.5 Generic Procedures  
      13.5.1 User-Defined Generic Procedures  
      13.5.2 Generic Interfaces for Procedures in Modules  
      13.5.3 Generic Bound Procedures  
   13.6 Extending Fortran with User-Defined Operators and Assignments  
   13.7 Bound Assignments and Operators  
   13.8 Restricting Access to theContents of a Module  
   13.9 Advanced Options of the USE Statement  
   13.10 Intrinsic Modules  
   13.11 Access to Command Line Arguments and Environment Variables  
      13.11.1 Access to Command Line Arguments  
      13.11.2 Retrieving Environment Variables  
   13.12 The VOLATILE Attribute and Statement  
   13.13 Summary  
      13.13.1 Summary of Good Programming Practice  
      13.13.2 Summary of Fortran Statements and Structures  
      13.13.3 Exercises

14 Advanced I/O Concepts  
   14.1 Additional Format Descriptors  
      14.1.1 Additional Forms of the E and ES Format Descriptors  
      14.1.2 Engineering Notation—The EN Descriptor  
      14.1.3 Double-Precision Data—The D Descriptor  
      14.1.4 The Generalized (G) Format Descriptor  
      14.1.5 The Binary, Octal, and Hexadecimal (8, 0, and Z) Descriptors  
      14.1.6 The TAB Descriptors  
      14.1.7 The Colon (:) Descriptor  
      14.1.8 Scale Factors—The P Descriptor  
      14.1.9 The SIGN Descriptors  
      14.1.10 Blank Interpretation: The BN and BZ Descriptors  
      14.1.11 Rounding Control: The RU, RD, RZ, RN, RC, and RP Descriptors (Fortran 2003 only)
14.1.12 Decimal Specifier: The DC and DP Descriptors (Fortran 2003 only)

14.2 Defaulting Values in List-Directed Input

14.3 Detailed Description of Fortran I/O Statements

14.4 Namelist I/O

14.5 Unformatted Files

14.6 Direct Access Files

14.7 Stream Access Mode

14.8 Nondefault I/O for Derived Types (Fortran 2003 only)

14.9 Asynchronous I/O

14.10 Access to Processor-Specific I/O System Information

14.11 Summary

15 Pointers and Dynamic Data Structures

15.1 Pointers and Targets

15.2 Using Pointers in Assignment Statements

15.3 Using Pointers with Arrays

15.4 Dynamic Memory Allocation with Pointers

15.5 Using Pointers as Components of Derived Data Types

15.6 Arrays of Pointers

15.7 Using Pointers in Procedures

15.8 Procedure Pointers
## Table of Contents

### 15.9 Binary Tree Structures
- **15.9.1 The Significance of Binary Tree Structures** / 735
- **15.9.2 Building a Binary Tree Structure** / 735

### 15.10 Summary
- **15.10.1 Summary of Good Programming Practice** / 755
- **15.10.2 Summary of Fortran Statements and Structures** / 755
- **15.10.3 Exercises** / 755

### 16 Object-Oriented Programming In Fortran
- **16.1 An Introduction to Object-Oriented Programming** / 764
  - **16.1.1 Objects** / 764
  - **16.1.2 Messages** / 764
  - **16.1.3 Classes** / 764
  - **16.1.4 Class Hierarchy and Inheritance** / 764
  - **16.1.5 Object-Oriented Programming** / 764
- **16.2 The Structure of a Fortran Class** / 768
- **16.3 The CLASS Keyword** / 770
- **16.4 Implementing Classes and Objects In Fortran** / 772
  - **16.4.1 Declaring Fields (Instance Variables)** / 772
  - **16.4.2 Creating Methods** / 772
  - **16.4.3 Creating (Instantiating) Objects from a Class** / 772
- **16.5 First Example: A timer Class** / 775
  - **16.5.1 Implementing the timer Class** / 775
  - **16.5.2 Using the timer Class** / 775
  - **16.5.3 Comments on the timer Class** / 775
- **16.6 Categories of Methods** / 780
- **16.7 Controlling Access to Class Members** / 789
- **16.8 Finalizers** / 789
- **16.9 Inheritance and Polymorphism** / 793
  - **16.9.1 Superclasses and Subclasses** / 793
  - **16.9.2 Defining and Using Subclasses** / 793
  - **16.9.3 The Relationship between Superclass Objects and Subclass Objects** / 793
  - **16.9.4 Polymorphism** / 793
  - **16.9.5 The SELECT TYPE Construct** / 793
- **16.10 Preventing Methods from Being Overridden in Subclasses** / 808
- **16.11 Abstract Classes** / 809
- **16.12 Summary** / 829
  - **16.12.1 Summary of Good Programming Practice** / 829
  - **16.12.2 Summary of Fortran Statements and Structures** / 829
  - **16.12.3 Exercises** / 829

### 17 Redundant, Obsolescent, and Deleted Fortran Features
- **17.1 Pre-Fortran 90 Character Restrictions** / 836
- **17.2 Obsolescent Source Form** / 836
# Table of Contents

## 17.3 Redundant Data Type

## 17.4 Older, Obsolescent, and/or Undesirable Specification Statements

17.4.1 Pre-Fortran 90 Specification Statements / 17.4.2 The IMPLICIT Statement / 17.4.3 The DIMENSION Statement / 17.4.4 The DATA Statement / 17.4.5 The PARAMETER Statement

## 17.5 Sharing Memory Locations: COMMON and EQUIVALENCE

17.5.1 COMMON Blocks / 17.5.2 Initializing Data in COMMON Blocks: The BLOCK DATA Subprogram / 17.5.3 The Unlabeled COMMON Statement / 17.5.4 The EQUVALENCE Statement

## 17.6 Undesirable Subprogram Features

17.6.1 Alternate Subroutine Returns / 17.6.2 Alternate Entry Points / 17.6.3 The Statement Function / 17.6.4 Passing Intrinsic Functions as Arguments

## 17.7 Miscellaneous Execution Control Features

17.7.1 The PAUSE Statement / 17.7.2 Arguments Associated with the STOP Statement / 17.7.3 The END Statement

## 17.8 Obsolete Branching and Looping Structures

17.8.1 The Arithmetic IF Statement / 17.8.2 The Unconditional GO TO Statement / 17.8.3 The Computed GO TO Statement / 17.8.4 The Assigned GO TO Statement / 17.8.5 Older Forms of DO Loops

## 17.9 Redundant Features of I/O Statements

## 17.10 Summary

17.10.1 Summary of Good Programming Practice / 17.10.2 Summary of Fortran Statements and Structures

### Appendixes

A. ASCII AND EBCDIC Coding Systems

B. Fortran 95/2003 Intrinsic Procedures


C. Order of Statements in a Fortran 95/2003 Program

D. Glossary

E. Answers to Quizzes

Index