Contents

1 Overview . . . 1

2 Introduction to Computer Systems . . . 9

2.1 The core of a computer system . . . 10

2.1.1 Central processor unit — CPU . . . 10

2.1.2 Memory . . . 10

2.1.3 Bus . . . 10

2.2 Other components of a computer system . . . 11

2.2.1 Disks . . . 11

2.2.2 Others . . . 11

2.3 Software . . . 12

2.4 Problems . . . 13

2.5 Bibliography . . . 13

3 Introduction to Operating Systems . . . 15

3.1 History of operating systems . . . 16

3.1.1 The 1940s . . . 16

3.1.2 The 1950s . . . 16

3.1.3 The 1960s . . . 16

3.1.4 The 1960s and 1970s . . . 16

3.1.5 The 1970s, 1980s, and 1990s . . . 17

3.2 Networking . . . 17

3.3 Problems . . . 18

3.4 Bibliography . . . 18

4 Introduction to Using a Computer System . . . 19

4.1 Files . . . 20

4.2 Editors . . . 20

4.3 Single-user systems . . . 20

4.4 Networked systems . . . 20

4.5 Multiuser systems . . . 21

4.6 Other useful things to know . . . 21

4.7 Common methods of using computer systems to develop Fortran programs . . . 22
4.8 Bibliography . . . 23

5 Introduction to Problem Solving . . . 25
5.1 Natural language . . . 26
5.2 Artificial language . . . 27
5.2.1 Notations . . . 27
5.3 Resumé . . . 27
5.4 Algorithms . . . 28
5.4.1 Top-down . . . 28
5.4.2 Bottom-up . . . 28
5.4.3 Stepwise refinement. . . . 29
5.4.4 Modular programming . . . 29
5.4.5 Object oriented programming . . . 29
5.5 Systems analysis and design . . . 30
5.5.1 Problem definition . . . 30
5.5.2 Feasibility study and fact finding . . . 30
5.5.3 Analysis . . . 31
5.5.4 Design . . . 31
5.5.5 Detailed design . . . 31
5.5.6 Implementation . . . 31
5.5.7 Evaluation and testing . . . 31
5.5.8 Maintenance . . . 32
5.6 Conclusions . . . 32
5.7 Problems . . . 32
5.8 Bibliography . . . 33

6 Introduction to Programming Languages . . . 35
6.1 Some early theoretical work . . . 36
6.2 What is a programming language? . . . 36
6.3 Program language development and engineering . . . 36
6.4 The early days . . . 36
6.4.1 Fortran — The Early Days . . . 37
6.4.2 Fortran 77 . . . 37
6.4.3 Cobol . . . 37
6.4.4 Algol. . . . 38
6.5 Chomsky and program language development . . . 39
6.6 Lisp . . . 39
6.7 Snobol . . . 40
6.8 Second-generation languages . . . 40
6.8.1 PL/1 and Algol 68 . . . 40
6.8.2 Simula . . . 40
6.8.3 Pascal . . . 41

6.8.4 APL 41
6.8.5 Basic 41
6.8.6 C 41
6.9 Some other strands in language development 42
6.9.1 Abstraction, stepwise refinement and modules 42
6.9.2 Structured programming 42
6.9.3 Standardisation 42
6.10 Ada 43
6.11 Modula 43
6.12 Modula 2 44
6.13 Other language developments 44
6.13.1 Logo 44
6.13.2 Postscript, TeX and LaTeX 45
6.13.3 Prolog 45
6.13.4 SQL 45
6.13.5 ICON 46
6.14 Object orientated programming — OOP 46
6.14.1 Oberon and Oberon 2 46
6.14.2 Smalltalk 47
6.14.3 C++ 48
6.14.4 Java 48
6.14.5 Visual Basic 49
6.14.6 C# 49
6.15 Fortran 90 50
6.16 Fortran 1995 51
6.17 ISO technical reports TR15580 and TR15581 52
6.18 Fortran 2003 52
6.19 DTR 19767 enhanced module facilities. 53
6.20 Internet resources 54
6.20.1 Standards information 54
6.20.2 Fortran discussion lists 55
6.20.3 Other sources 55
6.21 Summary 56
6.22 Bibliography 56

7 Introduction to Programming 63
7.1 Language strengths and weaknesses 64
7.2 Elements of a programming language 64
7.2.1 Data description statements 65
7.2.2 Control structures 65
7.2.3 Data-processing statements 65
7.2.4 Input and output (I/O) statements 65
7.3 Variables — name, type and value 68

7.4 Notes . . . 70
7.5 Some more Fortran rules . . . 71
7.6 Fortran character set . . . 72
7.7 Good programming guidelines . . . 73
7.8 Compilers . . . 73
7.9 Program development . . . 74
7.10 Problems . . . 75

8 Arithmetic . . . 77
8.1 Rounding and truncation . . . 81
8.2 Time taken for light to travel from the Sun to Earth . . . 83
8.3 The PARAMETER statement . . . 84
8.4 Range, precision and size of numbers . . . 85
8.5 Health warning: optional reading, beginners are advised to leave until later . . . 88
8.5.1 Selecting different INTEGER kind types . . . 90
8.5.2 Selecting different REAL kind types . . . 91
8.5.3 Specifying kind types for literal integer and real constants . . . 91
8.5.4 Positional number systems . . . 92
8.5.5 Bit data type and representation model . . . 92
8.5.6 Integer data type and representation model . . . 93
8.5.7 Real data type and representation model . . . 93
8.5.8 IEEE 754 . . . 94
8.5.9 Testing the numerical representation of different kind types on a system . . . 94
8.5.10 Binary representation of different integer kind type numbers . . . 98
8.5.11 Binary representation of a real number . . . 100
8.5.12 Summary of how to select the appropriate kind type . . . 101
8.6 Variable status . . . 101
8.7 Summary . . . 101
8.8 Problems . . . 102
8.9 Bibliography . . . 105

9 Arrays 1: Some Fundamentals . . . 107
9.1 Tables of data . . . 108
9.1.1 Telephone directory . . . 108
9.1.2 Book catalogue . . . 108
9.1.3 Examination marks or results . . . 109
9.1.4 Monthly rainfall . . . 109
9.2 Arrays in Fortran . . . 110
9.3 The DIMENSION attribute . . . 110
9.4 An index . . . 111
9.5 Control structure . . . 111
9.6 Monthly rainfall . . . 111

9.6.1 Example 1: Rainfall 112
9.7 People's weights 113
9.7.1 Example 2: Setting array size with a parameter 114
9.8 Summary 115
9.9 Problems 116

10 Arrays 2: Further Examples 119
10.1 Varying the array size at run time 120
10.2 Higher-dimension arrays 121
10.2.1 A map 121
10.2.2 Example 3: Sensible tabular output 123
10.2.3 Example 4: Average of three sets of values 124
10.2.4 Example 5: Booking arrangements in a theatre or cinema 125
10.3 Additional forms of the DIMENSION attribute and DO loop statement 126
10.3.1 Example 6: Voltage from −20 to +20 volts 126
10.3.2 Example 7: Longitude from −180 to +180. 127
10.3.3 Notes 127
10.4 The DO loop and straight repetition. 127
10.4.1 Example 8: Table of temperatures 127
10.4.2 Example 9: Means and standard deviations 128
10.5 Summary 129
10.6 Problems 130

11 Whole Array and Additional Array Features. 133
11.1 Terminology. 134
11.1.1 Rank 134
11.1.2 Bounds 134
11.1.3 Extent. 134
11.1.4 Size. 134
11.1.5 Shape 134
11.1.6 Conformable 134
11.1.7 Array element ordering. 134
11.2 Whole array manipulation 135
11.2.1 Assignment 135
11.2.2 Expressions 135
11.3 Array sections 138
11.3.1 Rank 1 array example 138
11.3.2 Rank 2 array example 138
11.4 Array constructors 140
11.4.1 Rank 1 array example — explicit values 140
11.4.1.1 Rank 1 array example and implied DO loop. 141
11.4.1.2 Rank 1 array example and the DOT_PRODUCT intrinsic 141
11.4.2 Rank 1 example with step size of 2 in implied DO loop 143

11.4.3 Rank 1 array and the SUM intrinsic function 144
11.4.4 Rank 2 arrays and the SUM intrinsic function 145
11.5 Masked array assignment and the WHERE statement 146
11.5.1 Notes 147
11.6 The FORALL statement and FORALL construct 147
11.6.1 Syntax 147
11.6.2 Array element ordering and physical and virtual memory 148
11.7 Summary 148
11.8 Problems 149
11.9 Bibliography 149

12 Output of Results 151
12.1 Integers — I format or edit descriptor. 152
12.2 Reals — F format or edit descriptor. 155
12.2.1 Metric and imperial conversion 156
12.2.2 Overflow and underflow 156
12.3 Reals — E format or edit descriptor 158
12.3.1 Simple E format example 159
12.4 Spaces 160
12.5 Characters — A format or edit descriptor 160
12.5.1 Headings 161
12.6 Mixed type output in a FORMAT statement. 162
12.7 Common mistakes 162
12.8 OPEN (and CLOSE) 163
12.8.1 The OPEN statement 163
12.8.2 Writing 164
12.9 Repetition 165
12.10 Some more examples 167
12.11 Implied DO loops and array sections for array output 168
12.12 Formatting for a line printer 170
12.12.1 Mechanics of carriage control 171
12.12.2 Generating a new line on both line printers and terminals 172
12.13 Timing of writing formatted files 173
12.14 Timing of writing unformatted files. 174
12.15 Summary 176
12.16 Problems 176

13 Reading in Data 179
13.1 Reading from the terminal or keyboard versus reading from files 180
13.2 Fixed fields on input 180
13.2.1 Integers and the I format 180
13.2.2 Reals and the F format 181
13.2.3 Reals and the E Format 182

13.3 Blanks, nulls and zeros 185
13.4 Characters 186
13.5 Skipping spaces and lines 187
13.6 Reading 187
13.7 File manipulation again 188
13.8 Reading using array sections 189
13.9 Timing of reading formatted files 190
13.10 Timing of reading unformatted files. 191
13.11 Errors when reading 192
13.12 Summary 193
13.13 Problems 193

14 Files 195
14.1 Data files in Fortran 196
14.2 Summary of options on OPEN 198
14.3 More foolproof I/O 200
14.4 Summary 201
14.5 Problems 202

15 Functions 204
15.1 An introduction to predefined functions and their use 204
15.1.1 Example 1: Simple function usage 205
15.2 Generic functions 205
15.2.1 Example 2: The ABS generic function 206
15.3 Elemental functions 206
15.3.1 Example 3: Elemental function use 206
15.4 Transformational functions 206
15.4.1 Example 4: Simple transformational use 207
15.4.2 Example 5: Intrinsic DOT_PRODUCT use 207
15.5 Notes on function usage 207
15.6 Example 6: Easter 208
15.7 Complete list of predefined functions 210
15.7.1 Inquiry functions 210
15.7.2 Transfer and conversion functions 210
15.7.3 Computational functions 211
15.7.4 Array functions 211
15.7.5 Predefined subroutines 211
15.8 Supplying your own functions 212
15.8.1 Example 7: Simple user defined function 212
15.9 An introduction to the scope of variables and local variables. 214
15.10 Recursive functions 214
15.10.1 Example 8: Recursive factorial evaluation 215

15.11 Example 9: Recursive version of GCD 216
15.12 Example 10: After removing recursion 217
15.13 Pure functions 218
15.14 Elemental functions 218
15.15 Internal functions 218
15.15.1 Example 11: Stirling's approximation 218
15.16 Resumé 219
15.17 Function syntax 220
15.18 Rules and restrictions 220
15.19 Problems 220
15.20 Bibliography 221
15.20.1 Recursion and problem solving 222
16 Control Structures 223
16.1 Selection among courses of action 224
16.1.1 The BLOCK IF statement 225
16.1.2 Example 1: Quadratic roots. 227
16.1.3 Note 228
16.1.4 Example 2: Date calculation 228
16.1.5 The CASE statement. 229
16.1.6 Example 3: Simple calculator. 230
16.1.7 Example 4: Counting vowels, consonants, etc.. 231
16.2 The three forms of the DO statement 232
16.2.1 Example 5: Sentinel usage 232
16.2.2 CYCLE and EXIT 234
16.2.3 Example 6: e**x evaluation 234
16.2.4 Example 7: Wave breaking on an offshore reef 235
16.3 Summary 237
16.3.1 Control structure formal syntax 237
16.4 Problems 238
16.5 Bibliography 240
17 Characters 241
17.1 Character input 243
17.2 Character operators 244
17.3 Character substrings 245
17.4 Character functions 247
17.5 Collating sequence 248
17.6 Summary 250
17.7 Problems 251
18 Complex 253
18.1 Example 255
18.2 Complex and kind type 256
18.3 Summary 256

18.4 Problems . . . 256

19 Logical . . . 257

19.1 I/O . . . 261

19.2 Summary . . . 261

19.3 Problems . . . 262

20 User Defined Types . . . 263

20.1 Example 1: Dates . . . 264

20.2 Type definition . . . 264

20.3 Variable definition . . . 265

20.4 Example 2: Address lists . . . 265

20.5 Example 3: Nested user defined types. . . . 266

20.6 Problems . . . 268

20.7 Bibliography . . . 268

21 An Introduction to Pointers . . . 269

21.1 Some basic pointer concepts . . . 270

21.2 The ASSOCIATED intrinsic function . . . 272

21.2.1 CVF 6.6C . . . 272

21.2.2 Intel, Windows, 8.1 . . . 272

21.2.3 Lahey, Windows 5.70f . . . 272

21.2.4 NAG, Windows, 4.2 . . . 273

21.2.5 Salford 4.6.0 . . . 273

21.3 Referencing A and B before assignment. . . . 273

21.3.1 CVF . . . 274

21.3.2 Intel, Windows 8.1. . . . 274

21.3.3 Lahey, Windows 5.70f . . . 275

21.3.4 NAG, Windows 4.2 . . . 275

21.3.5 Salford 4.6.0 . . . 275

21.4 The NULL intrinsic . . . 275

21.5 Assignment via = . . . 276

21.6 Singly linked list. . . . 278

21.7 Reading in an arbitrary quantity of numeric data. . . . 280

21.8 Arrays of pointers . . . 283

21.9 Arrays of pointers and variable sized data sets — 1 . . . 284

21.10 Arrays of pointers and variable sized data sets — 2 . . . 285

21.11 Memory leak examples. . . . 285

21.12 Nonstandard pointer examples . . . 288

21.13 Problems . . . 293

22 Introduction to Subroutines . . . 295

22.1 Example 1. . . . 296

22.1.1 Defining a subroutine . . . 298

22.1.2 Referencing a subroutine . . . 299

22.1.3 Dummy arguments or parameters and actual arguments 299
22.1.4 Intent 299
22.1.5 Local variables 299
22.1.6 Local variables and the SAVE attribute 300
22.1.7 Scope of variables 300
22.1.8 Status of the action carried out in the subroutine. 300
22.2 Example 2. 300
22.3 Example 3 — Quadratic example with interface blocks 301
22.4 Example 4 — Quadratic example and the CONTAINS statement . . 304
22.5 Why bother? 306
22.6 Summary 307
22.7 Problems 307

23 Subroutines: 2 309

23.1 More on parameter passing. 310
23.1.1 Explicit-shape array 310
23.1.2 Assumed-shape array 310
23.1.3 Deferred-shape array. 310
23.1.4 Automatic arrays 310
23.1.5 Assumed-size array — Fortran 77 style 310
23.1.6 Adjustable arrays — Fortran 77 style 311
23.2 Common code example 311
23.3 Explicit-shape example. 311
23.4 Assumed-shape example 313
23.4.1 Notes 315
23.5 Characters arguments and assumed-length dummy arguments . . . 315
23.6 Rank 2 and higher arrays as parameters 316
23.6.1 Explicit-shape dummy arrays. 316
23.6.2 Assumed-shape dummy array arguments 319
23.6.3 Notes 320
23.6.4 Using the intrinsic functions MATMUL and TRANSPOSE 321
23.7 Automatic arrays and median calculation 322
23.7.1 Internal subroutines and scope 325
23.7.2 Timing the selection sort algorithm 325
23.7.2.1 Timing 326
23.8 Alternative median calculation algorithm 327
23.8.1 Timing 330
23.9 Recursive subroutines — Quicksort. 332
23.9.1 Note — Interface blocks. 336
23.9.2 Note — Recursive subroutine 337
23.9.3 Note — Flexible design 337
23.9.4 Note — Timing information 337
23.10 Summary 337

23.11 Problems 338
23.12 Bibliography 340
23.13 Commercial numerical and statistical subroutine libraries 340
24 An Introduction to Modules 341
24.1 Modules for global data 342
24.2 Modules for precision specification and constant definition. 343
24.2.1 Note 344
24.3 Modules for sharing arrays of data 345
24.4 Modules for derived data types 346
24.4.1 Person data type 347
24.5 Modules containing procedures — Quicksort example 349
24.6 Modules containing procedures — Statistics example 353
24.7 The solution of linear equations using Gaussian elimination 356
24.7.1 Notes 361
24.7.1.1 Module for kind type 361
24.7.1.2 Deferred-shape arrays 361
24.7.1.3 Intrinisic functions MAXVAL and MAXLOC. 361
24.8 Notes on module usage and compilation 361
24.9 Summary 362
24.10 Problems 362
24.11 Bibliography 363
25 Converting from Fortran 77 365
25.1 Deleted features 366
25.2 Obsolescent features 366
25.2.1 Arithmetic IF 366
25.2.2 Real and double precision DO control variables 366
25.2.3 Shared DO termination and non-ENDDO termination 366
25.2.4 Alternate RETURN 367
25.2.5 PAUSE statement 367
25.2.6 ASSIGN and assigned GOTO statements 367
25.2.7 Assigned FORMAT statements 367
25.2.8 H editing 367
25.3 Better alternatives 367
25.4 Example 1. 368
25.5 Example 2. 378
25.6 Commercial conversion tools 379
25.6.1 NAG 379
25.6.2 Polyhedron 395
25.6.3 Original Fortran 66. 407
25.6.4 Fortran 77 Version. 407
25.6.5 Fortran 90 Version. 408
25.7 Summary 409

25.8 Problems 409

26 Case Studies 411

26.1 Using linked lists for sparse matrix problems 412

26.1.1 Inner product of two sparse vectors 413

26.2 Solving a system of first-order ordinary differential equations using Runga–Kutta–Merson. 417

26.2.1 Note: Alternative form of the ALLOCATE statement 424

26.2.2 Note: Automatic arrays. 424

26.2.3 Note: Dummy procedure arguments. 425

26.2.4 Keyword and optional arguments 425

26.3 Generic procedures 427

26.4 A function that returns a variable length array 434

26.5 Operator and assignment overloading 436

26.6 A subroutine to extract the diagonal elements of a matrix 437

26.7 Perfectly balanced tree 439

26.8 Pure function example 442

26.8.1 Pure constraints 442

26.9 Elemental function example 443

26.9.1 Elemental constraints 444

26.10 Elemental subroutine example 445

26.11 Date class 446

26.12 Graphics example — dislin. 461

26.13 Problems 469

26.14 Bibliography 470

27 ISO TR 15580 — IEEE Arithmetic. 473

27.1 History 474

27.2 IEEE 754 Specifications 476

27.2.1 Single precision floating point format. 477

27.2.2 Double precision floating point format 479

27.2.3 Two classes of extended floating point formats 479

27.2.4 Accuracy requirements 479

27.2.5 Base conversion — Converting between decimal and binary floating point formats and vice versa 479

27.2.6 Exception handling 480

27.2.7 Rounding directions 480

27.2.8 Rounding precisions 480

27.3 Resumé 480

27.4 ISO TR 15580. 481

27.4.1 IEEE_FEATURES module. 481

27.4.2 IEEE_EXCEPTIONS module 481

27.4.3 IEEE_ARITHMETIC module 483

27.4.3.1 IEEE data type selection 484

27.4.3.2 General support enquiry functions 484
27.4.3.3 Rounding modes 485
27.4.3.4 Number classification 485
27.4.3.5 Arithmetic operations 487
27.5 Summary 488
27.6 Bibliography 488
27.6.1 Web-based sources 489
27.6.2 Hardware sources 490
27.6.3 Operating Systems 491
27.6.4 Java and IEEE 754. 491
27.6.5 C and IEEE 754 492

28 ISO TR 15581 Allocatable Enhancements 493
28.1 Allocatable dummy array example 494
28.2 Allocatable function result example 497
28.3 Allocatable structure component example 499
28.4 Summary 499
28.5 Problem 499

29 Fortran 2003 and the Enhanced Module Facility 501
29.1 Derived type enhancements 502
29.2 Object oriented programming support 502
29.3 Data manipulation enhancements 502
29.4 Input/output enhancements 503
29.5 Interoperability with the C programming language 503
29.6 Procedure pointers 504
29.7 Scoping enhancements 504
29.8 Support for IEC 60559 (IEEE 754) exceptions and arithmetic 504
29.9 Support for international usage: (ISO 10646) 504
29.10 Enhanced integration with the host operating system 505
29.11 The ASSOCIATE construct 505
29.12 Enhanced modules facility 505
29.13 Summary 506

30 Parallel Programming 507
30.1 MPI. 508
30.2 Co–array Fortran 508
30.3 Openmp 508
30.4 PVM 509
30.5 HPF. 509
30.6 Parallel programming and high-performance computing 509
30.6.1 Summary 510

31 Miscellaneous 511

31.1 Program development and software engineering 512
31.1.1 Modules. 513
31.1.2 Programming style — Programs should be easy to read 513
31.1.3 Programming style — Programs should behave well 514
31.2 Data structures. 514
31.3 Algorithms 514
31.4 Recursion 515
31.5 Structured programming and the GOTO statement 515
31.6 Efficiency, space-time trade-off. 516
31.7 Program testing 516
31.8 Simple debugging techniques 516
31.9 Software tools 517
31.9.1 Cross referencing 517
31.9.2 Pretty print 517
31.9.3 NAGWare f90 Tools 517
31.10 Numerical software sources 517
31.10.1 Numerical Algorithms Group 518
31.10.2 Visual Numerics 518
31.10.3 Netlib 518
31.11 Coda 518
31.12 Bibliography: All sources (bar one) taken from comp.software-eng. 518
31.12.1 Software engineering. 518
31.12.2 Programming style. 519
31.12.3 Software testing 519
31.12.4 Fun 519

A Glossary 520

B Sample Program Examples. 530

C ASCII Character Set. 534

D Intrinsic Functions and Procedures 535

E English and Latin Texts 568

F Coded Text Extract 569

G Formal syntax 570

H Compiler Options 575

Index 581