Contents

Preface ix

1 A modern approach to computing 1

1.1 An appraisal of the current situation 1
1.2 A way ahead 6

2 Specifications I 11

2.1 The nature of a specification 11
2.2 Pre and post conditions 13
2.3 Type constraints 13
2.4 Sequences of operations 16
2.5 More on types 18
2.5.1 *Primitive and constructed data types* 18
2.5.2 *Pairs* 18
2.5.3 *Triples* 21
2.5.4 *Tuples* 22
2.5.5 *Lists* 24
2.5.6 *Sets* 28
2.6 The characteristics of a specification 31
2.7 Refinement and transformation of specifications 32
2.8 States in specifications 32
2.9 States vs. Input/Output 37
2.10 Conclusion 37

3 Diagrams 39

3.1 Diagrams used in the program development process 39
3.2 An algebra of diagrams 46
3.3 Other diagramming systems 50
3.4 Graphs, networks and trees 55

4 Specifications II 61

4.1 Concise notation 61
4.2 Transformation and proof in specifications 63
4.3 What comes next? 70

5 PDL 72
5.1 Imperative and declarative languages 72
5.2 Why a PDL? 75
5.3 The PDL stage 76
5.4 The description of a PDL 78
5.4.1 *Function definition* 78
5.4.2 *Statement* 79
5.4.3 *Statements* 79
5.4.4 *Type* 80
5.4.5 *Variable* 81
5.4.6 *Expressions* 81
5.5 PDL data types – list and record 82
5.6 Representing specification data types in PDL 83
5.7 Examples 85
5.8 Other PDL issues 86
5.9 PDL summary 87

6 Code generation 88
6.1 Templates 88
6.2 Templates for Pascal 89
6.2.1 *Templates for control structures in Pascal* 90
6.2.2 *Templates for data structures in Pascal* 95
6.3 Templates for FORTRAN 99
6.3.1 *Templates for control structures in FORTRAN* 100
6.3.2 *Templates for data structures in FORTRAN* 106
6.4 Templates for COBOL 115
6.4.1 *Templates for control structures in COBOL* 115
6.4.2 *Templates for data structures in COBOL* 121
6.5 Templates for a minicomputer assembly language 126
6.5.1 *Templates for control structures* 129
6.5.2 *Templates for data structures* 139
6.6 Templates for a microprocessor assembly language 143
6.6.1 *Templates for control structures* 144
6.6.2 *Templates for data structures* 151

7 Verification 155
7.1 The implication operator 156
7.2 Control-flow diagrams and data-flow specification diagrams 166
7.3 Sequencing and alternation 171
7.4 Repetition 183
7.4.1 *Simple recursion* 184
7.4.2 *Quantifiers and induction* 195
7.4.3 *Iteration* 201
7.5 Conclusion 215

8 Examination of templates and target code 217

8.1 Assignment statements 218
8.2 Control statements 219
8.3 Parameter passing 224
8.4 Summary 226

9 Abstract data types 227

9.1 ADT example – a siding 228
9.2 ADT example – an In_Tray 233
9.3 ADT example – LR Lookup store 236
9.4 ADT example – a binary tree 237
9.4.1 *Recursive implementation of tree operations in PDL* 240
9.4.2 *Non-recursive implementation of tree operations* 242
9.5 On preserving ADT discipline 254
9.5.1 *What is ADT discipline?* 254
9.5.2 *Data Type Encapsulation* 258

10 The mathematical basis of abstract data types 262

10.1 Booleans 262
10.2 Lists 269
10.3 Some numeric types 271
10.4 Sets 282
10.5 Equations versus conditions 285

11 Utilisation of existing programs 289

11.1 Testing for good structure 290
11.2 Restructuring of unstructured programs 302
11.3 Analysis of programs 309

12 A small scale study – topological sorting 326

12.1 Problem formulation 326
12.2 Transformations 331
12.3 Towards PDL 333
12.4 Data structure considerations 336
12.5 PDL 340

Appendices

A Glossary of symbols 342
B Syntax of standard specifications 344
C The description of a PDL 348
D Transformations that remove recursion 353

References 365

Index 367