

Specifying Software

A HANDS-ON INTRODUCTION

R. D. Tennent

Queen's University, Kingston, Canada



CAMBRIDGE
UNIVERSITY PRESS

Contents

<i>Preface</i>	ix
<i>Introduction</i>	1
A Algorithms	11
<i>Introduction to Part A</i>	13
1 Specifying Algorithms	15
1.1 Case Study: Searching an Array	15
1.2 Declarative Interface for Array Search	18
1.3 Assertions	19
1.4 Completing the Specification for Array Search	25
1.5 Correctness of Code	27
1.6 Additional Reading	29
2 Verifying Algorithms: Basic Techniques	31
2.1 Some Programs for Array Searching	31
2.2 Correctness Statements	35
2.3 Simple Assignment Statements	36
2.4 Substitution into Assertions	38
2.5 Using Mathematical Facts	40
2.6 Formal Proofs and Proof Tableaux	42
2.7 Sequencing	43
2.8 If Statements	46
2.9 While Statements	49
2.10 Termination of Loops	53
2.11 Local Variables	55
2.12 Discussion	57
2.13 Additional Reading	58

3	<i>Verifying Algorithms: Some Examples</i>	61
3.1	Searching an Array	61
3.2	Minimal Entry of an Array	65
3.3	Powers	67
3.4	Division	69
3.5	Binary Search in a Sorted Array	71
3.6	The Number of Distinct Values in an Array	76
3.7	Additional Reading	79
4	<i>Additional Verification Techniques</i>	81
4.1	For Loops	81
4.2	Array-Component Assignment Statements	83
4.3	Do-While Loops	88
4.4	Sorting an Array	91
4.5	Combining Correctness Statements	99
4.6	Additional Exercises	102
4.7	Additional Reading	105
B	<i>Data Representations</i>	107
	<i>Introduction to Part B</i>	109
5	<i>Data Representation: A Case Study</i>	111
5.1	Informal Specification	111
5.2	Formal Specification	113
5.3	A Simple Implementation	114
5.4	Program Organization	116
5.5	Verification of a Data Representation	119
5.6	A Caching Implementation	122
5.7	Additional Reading	124
6	<i>Data Representation: Additional Examples</i>	125
6.1	Traversing a Sparse Set	125
6.2	Sparse Arrays	131
6.3	Sequences	136
6.4	Bit-String Representation of Sets	138
6.5	Reachability in a Directed Graph	143
6.6	Sorting a Partially Ordered Set	146
6.7	Additional Reading	152

C	Language Recognizers	155
	<i>Introduction to Part C</i>	157
7	Basic Concepts	159
7.1	Strings	159
7.2	Formal Languages	160
7.3	Basic Operations on Languages	162
7.4	Classes of Languages	167
7.5	Additional Reading	168
8	State-Transition Diagrams	169
8.1	Basic Definitions	169
8.2	Software Realization	172
8.3	Nondeterminism	176
8.4	State-Transition Diagrams with ϵ Transitions	183
8.5	Reactive Systems	184
8.6	Additional Reading	189
9	Regular Languages	191
9.1	From Regular Expressions to State Diagrams	191
9.2	From State Diagrams to Regular Expressions	196
9.3	Additional Operations on Regular Languages	199
9.4	Nonregular Languages	200
9.5	Other Formalisms	204
9.6	Additional Reading	205
10	Context-Free Languages	207
10.1	Backus-Naur Formalism	207
10.2	Derivations	210
10.3	Parse Trees	212
10.4	Ambiguity	212
10.5	Push-Down Automata	216
10.6	Non-Context-Free Languages	220
10.7	Other Formalisms	223
10.8	Additional Reading	224
11	Parsing	227
11.1	General Methods	227
11.2	Deterministic Push-Down Automata	228
11.3	Recursive Descent	228
11.4	Additional Reading	241

D Unimplementable Specifications 243

Introduction to Part D 245

12 A Taste of Computability Theory 247

12.1 The Halting Problem 247

12.2 The Church-Turing Thesis 249

12.3 Unsolvability by Reduction 250

12.4 Additional Reading 252

Appendices 255

A Programming Language Reference 257

A.1 Lexical Conventions 257

A.2 Basic Types and Constants 259

A.3 Strings 261

A.4 Variable Declarations 261

A.5 Enumerated and Structure Types, Defined Type Names 262

A.6 Expressions 262

A.7 Some Library Functions 264

A.8 Statements 266

A.9 Function Definitions 269

A.10 More Library Functions 270

A.11 Classes 273

A.12 Program Structure 273

A.13 Grammar 273

A.14 Additional Reading 278

B Hints for Selected Exercises 279

Index 285