

# Contents

<b>1</b>	<b>Introduction</b>	1
1.1	Invariance Examples	2
1.1.1	A Chess Board Problem	2
1.1.2	A Black and White Balls Game	4
1.2	The Way Ahead	5
<b>2</b>	<b>Background</b>	7
2.1	Predicates	8
2.1.1	Propositional Calculus	8
2.1.2	Predicate Calculus	10
2.1.3	Predicates Define Sets of States	10
2.1.4	Strong and Weak Predicates	11
2.2	Specifying Pre- and Postconditions	14
2.2.1	Hoare Triples as Specifications of Total Correctness	14
2.2.2	Weakest Preconditions and Semantics	16
2.3	Guarded Command Language	17
2.3.1	Empty Command	18
2.3.2	Diversion: Some Extreme Cases	18
2.3.3	Assignment	24
2.3.4	Composition	27
2.3.5	Selection	30
2.3.6	Repetition	32
2.4	Refinement Rules	34
2.4.1	Strengthen Postcondition Rule	35
2.4.2	Weaken Precondition Rule	36
2.4.3	Skip Rule	36
2.4.4	Sequences of Refinements	37
2.4.5	Refinement and Weakest Preconditions	37
2.4.6	Assignment Rule	37
2.4.7	Composition Rule	38
2.4.8	Following Assignment Rule	40

2.4.9	Selection Rule .....	41
2.4.10	Repetition Rule .....	42
2.4.11	Procedures and Procedure Calls .....	45
2.5	Object Orientation .....	45
2.6	Supplementary Notation .....	48
2.6.1	Morgan's Refinement Calculus .....	48
2.6.2	Arrays and Sequences .....	49
2.6.3	Additional GCL Commands .....	50
2.7	Revision Exercises .....	51
<b>3</b>	<b>Simple Examples .....</b>	<b>55</b>
3.1	Linear Search .....	56
3.1.1	Formulating the Problem .....	56
3.1.2	Choosing the Invariant .....	56
3.1.3	Establishing the Invariant .....	57
3.1.4	Refining to Create a Loop .....	58
3.1.5	Putting it All Together .....	60
3.2	Finding the Maximal Element .....	60
3.2.1	Formulating the Problem .....	60
3.2.2	Choosing the Invariant .....	61
3.2.3	Establishing the Invariant .....	62
3.2.4	Refining to Create a Loop .....	63
3.2.5	Putting it All Together .....	66
3.3	Binary Search .....	66
3.3.1	Formulating the Problem .....	66
3.3.2	Decomposing the Problem .....	67
3.3.3	Generating the Binary Search Code .....	68
3.3.4	After the Binary Search .....	72
3.3.5	Putting it All Together .....	73
3.4	Pattern Matching .....	74
3.4.1	Formulating the Problem .....	75
3.4.2	Developing the Loop .....	75
3.4.3	Putting it All Together .....	77
3.5	Exponentiation .....	77
3.5.1	Formulating the Problem .....	78
3.5.2	Establishing the Invariant .....	78
3.5.3	Refining to Create a Loop .....	79
3.5.4	Discussion .....	83
3.6	Integer Logarithm Approximation .....	84
3.6.1	Problem Statement and Invariant .....	84
3.6.2	Refinement Steps .....	85
3.6.3	Justifying the Assignment .....	85
3.6.4	Strengthening Predicates by Decreasing Ranges .....	86
3.6.5	Discussion .....	87
3.7	Revision Exercise .....	88

<b>4</b>	<b>Intermediary Examples</b>	95
4.1	Dutch National Flag	95
4.1.1	Formulating the Problem	96
4.1.2	Choosing the Invariant	98
4.1.3	Refining the Specification	99
4.1.4	Proving the Third Guard Command	100
4.1.5	Putting it All Together	102
4.1.6	Discussion	102
4.2	Longest Segment	103
4.2.1	Formulating the Problem	104
4.2.2	A First Attempt at Refinement	105
4.2.3	A Revised Attempt at Refinement	107
4.2.4	Putting it All Together	111
4.2.5	Discussion	112
4.3	Palindromes	112
4.3.1	The Outer Loop	113
4.3.2	Formulating the Problem	113
4.3.3	Refining the Specification	115
4.3.4	Putting it All Together	116
4.3.5	Discussion	117
4.4	Raster Lines	117
4.4.1	Formulating the Problem	118
4.4.2	Deriving the Loop	121
4.4.3	Developing the Loop's Body	122
4.4.4	Putting it All Together	125
4.4.5	Discussion	126
4.5	Raster Circle	127
4.5.1	Problem Statement	127
4.5.2	From Invariant to Loop	129
4.5.3	Refining the Loop's Body	129
4.5.4	Determining the Guards	132
4.5.5	Deriving the Guards	133
4.5.6	Putting it All Together	134
4.6	Majority Voting	136
4.6.1	Formulating the Problem	137
4.6.2	Arriving at an Invariant and Developing the Loop	138
4.6.3	Developing the Guards	139
4.6.4	Discussion	143
4.7	Computational Geometry	144
4.7.1	Background and Notation	144
4.7.2	The Approach to Solving the Problem	146
4.7.3	Deriving the Solution Constructively	147
4.7.4	Discussion	150
4.8	Revision Exercises	151

<b>5</b>	<b>Procedures and Recursion</b>	161
5.1	Introduction	161
5.2	Procedures	162
5.2.1	Parameterless Procedures	162
5.2.2	Pass by Value	164
5.2.3	Pass by Result	167
5.2.4	Pass by Value Result	168
5.2.5	Functions	169
5.3	Procedure Refinement Strategy	170
5.4	Recursive Procedures	171
5.5	Terminating Recursive Programs	173
5.6	Recursive Examples	177
5.6.1	Factorial	177
5.6.2	Searching a List	181
5.6.3	Evaluating an Expression Tree	185
5.6.4	MergeSort	191
5.7	Conclusion	194
<b>6</b>	<b>Case Study: Lattice Cover Graph Construction</b>	197
6.1	Introduction	197
6.2	Preliminaries	198
6.2.1	Lattices	198
6.2.2	Set Intersection-Closed Lattices	201
6.3	The Algorithm	205
6.3.1	The Basic Structure	206
6.3.2	Articulating and Attaining $inv1(i)$	207
6.3.3	Articulating and Attaining $inv2(i)$	208
6.3.4	Filling in $S_1$	210
6.3.5	Completing the Select Command	211
6.3.6	The Completed Algorithm	214
6.3.7	The Operational Implications	215
6.4	Refactorings	218
6.4.1	Efficiently Inserting $C_i \cap X$	218
6.4.2	Finding the Parent of $X$	219
6.4.3	Discussion	221
6.5	A Gentle Introduction to Formal Concept Analysis	222
<b>7</b>	<b>Case Study 2: Classifying MADFA Construction Algorithms</b>	227
7.1	Introduction	227
7.2	From DFAs to MADFAs	228
7.2.1	Deterministic Finite Automata—DFAs	228
7.2.2	Acyclic Deterministic Finite Automata—ADFAs	230
7.2.3	Minimum Acyclic Deterministic Finite Automata—MADFAs	231
7.2.4	Concepts for MADFA Construction Algorithms	232

7.3	An Abstract MADFA Construction Algorithm .....	237
7.3.1	Structural Invariant Instantiations .....	239
7.3.2	The Procedures to be Instantiated .....	241
7.3.3	The Importance of the Skeleton-Based Taxonomy .....	241
7.4	Trie Intermediate ADFA .....	242
7.4.1	Procedure <i>add_word<sub>T</sub></i> .....	242
7.4.2	Adding Only Prefix Words .....	244
7.4.3	Adding a Non-prefix Word in a Trie .....	244
7.4.4	Procedure <i>cleanup<sub>T</sub></i> .....	246
7.4.5	An Example .....	249
7.5	Arbitrary Intermediate ADFA .....	250
7.5.1	Procedure <i>add_word<sub>N</sub></i> .....	251
7.5.2	Procedure <i>cleanup<sub>N</sub></i> .....	255
7.5.3	Commentary .....	255
7.6	Word Adding Based on a Partial Order .....	255
<b>References</b> .....		259
<b>Index</b> .....		263



<http://www.springer.com/978-3-642-27918-8>

The Correctness-by-Construction Approach to  
Programming

Kourie, D.G.; Watson, B.W.

2012, XIV, 266 p., Hardcover

ISBN: 978-3-642-27918-8