

## Contents

<i>Preface</i>	<i>page</i> ix
<i>Acknowledgements</i>	xii
<b>1 Introduction</b>	<b>1</b>
1.1 The 440-Million-Dollar Bug	5
1.2 A New Perspective on the History of Programming	8
1.3 Program as a Mathematical Entity	10
1.4 The Hands-on Imperative	11
1.5 Software Development Lifecycles	13
1.6 A Proper Engineering Discipline	16
1.7 New Media for Thinking	18
1.8 The Past and the Present of Programming	20
1.9 Developing Software without Errors	21
1.10 Understanding Programs	23
1.11 Programming Education	25
1.12 How Cultures Meet and Clash	27
<b>2 Mathematisation of Programming</b>	<b>33</b>
2.1 Giant Electronic Brains	36
2.2 Sound Body of Knowledge	39
2.3 How Technology Became Language	41
2.4 Mathematical Science of Computing	45
2.5 Many Definitions of Algol	49
2.6 The Minority Report	54
2.7 Go to Statement Considered Harmful	56
2.8 Chief Programmer Teams	58
2.9 Program Proofs and Social Processes	60
2.10 Computing and Cultures of Proving	63
2.11 Fundamental Limits of Program Proofs	66
2.12 Proofs for Machines and Proofs for Humans	70
2.13 Mathematisation of Programming	71
<b>3 Interactive Programming</b>	<b>77</b>
3.1 Spacewar!	81
3.2 Transistorised Experimental Computer Zero	83

3.3	Symbol Manipulating Language	86
3.4	Augmenting Human Intellect	87
3.5	A Time Sharing Operator Program	91
3.6	A Step towards Man–Computer Symbiosis	92
3.7	There Should Be No Computer Art	95
3.8	Children, Computers and Powerful Ideas	97
3.9	The Mother of All Demos	100
3.10	Almost Anything Goes	102
3.11	Personal Dynamic Media	106
3.12	Articulate Languages for Communication	109
3.13	Homebrew Computer Club	112
3.14	Beginner’s All-purpose Symbolic Instruction Code	115
3.15	The Birth of an Industry	118
3.16	Show Us Your Screens!	122
3.17	Reinventions and Adaptations	125
<b>4</b>	<b>Software Engineering</b>	132
4.1	How Did Software Get So Reliable without Proof?	135
4.2	The Art of Electronic Computer Maintenance	137
4.3	On-line Debugging Techniques	144
4.4	The Debugging Epoch Opens	147
4.5	Orderly and Reliable Exception Handling	151
4.6	System Structure for Fault Tolerance	154
4.7	Professionalisation of Testing	156
4.8	A Slightly Ominous Note for Information Processing Management	159
4.9	Production of Large Computer Programs	161
4.10	Disciplinary Repertoire of Software Engineering	166
4.11	Software Aspects of Strategic Defense Systems	167
4.12	The Mythical Man-Month	171
4.13	Laws of Software Evolution	174
4.14	Paradigm Change in Software Engineering	175
4.15	The New New Approach	178
4.16	A Programming Environment for a Timeshared System	179
4.17	Extreme Programming	181
4.18	The Debugging Paradox	184
4.19	Different Ways of Trusting Software Systems	185
<b>5</b>	<b>Programming with Types</b>	191
5.1	Are Types Invented or Discovered?	195
5.2	Resolving Logical Paradoxes with Types	196
5.3	Evaluating Arithmetic Expressions in Two Modes	197
5.4	Structuring Scientific and Business Data	199
5.5	Towards a Universal Programming Language	202

5.6	The Next 700 Programming Languages	206
5.7	Types Are Not Sets	209
5.8	In Search of Types	212
5.9	The Definition of Standard Types	213
5.10	Types as a Lightweight Formal Method	217
5.11	Automating Mathematics Using Types	220
5.12	Programming in Type Theories	222
5.13	The Meaning of Types Is Their Use	226
5.14	Stalking the Elusive Type	229
5.15	Pluralism and Scientific Progress	230
<b>6</b>	<b>Object-Oriented Programming</b>	236
6.1	The Computer Revolution Hasn't Happened Yet	239
6.2	A Language for Describing Discrete Event Systems	241
6.3	Past Ghosts and Present Spectres	246
6.4	A New Medium for Communication	250
6.5	Let's (Not) Burn Our Disk Packs	254
6.6	What Is an Object-Oriented Programming Language?	255
6.7	Structured Programming of the 1980s	259
6.8	The Power of Simplicity	261
6.9	Making Programming Enjoyable for the Serious Programmer	263
6.10	The Enterprise Age of Visual Smalltalk	267
6.11	The Better Way Is Here Now	271
6.12	Managing the Object-Oriented Project	272
6.13	Object-Oriented Programming, Systems, Languages and Applications	277
<b>7</b>	<b>Conclusion: Cultures of Programming</b>	285
7.1	The Most Powerful Tool Available to Human Intellect	288
7.2	Abstracting the History of Programming	288
7.3	Theories of Knowledge	292
7.4	Aesthetics and Cultural Pointers	294
7.5	Exchanging Ideas in a Pluralistic Discipline	298
7.6	Collaborations and Struggles for Control	300
7.7	Why Cultures of Programming Matter	303
7.8	Many Things Go	305
	<i>References</i>	309
	<i>Index</i>	332