
Contents

1	Introduction	1
1.1	Historical Perspective	2
1.2	Models of Computation	6
1.2.1	The Imperative Model	7
1.2.2	The Functional Model	9
1.2.3	The Logic Model	10
1.3	The Origins of a Few Programming Languages	10
1.3.1	A Brief History of C and C++	11
1.3.2	A Brief History of Java	12
1.3.3	A Brief History of Python	14
1.3.4	A Brief History of Standard ML	14
1.3.5	A Brief History of Prolog	16
1.4	Language Implementation	18
1.4.1	Compilation	19
1.4.2	Interpretation	21
1.4.3	Virtual Machines	23
1.5	Types and Type Checking	25
1.6	Chapter Summary	27
1.7	Review Questions	28
1.8	Solutions to Practice Problems	29
2	Syntax	31
2.1	Terminology	31
2.2	Backus Naur Form (BNF)	33
2.2.1	BNF Examples	33
2.2.2	Extended BNF (EBNF)	34
2.3	Context-Free Grammars	34
2.3.1	The Infix Expression Grammar	35
2.4	Derivations	35
2.4.1	A Derivation	35
2.4.2	Types of Derivations	36
2.4.3	Prefix Expressions	36
2.4.4	The Prefix Expression Grammar	36

2.5	Parse Trees	37
2.6	Abstract Syntax Trees	38
2.7	Lexical Analysis	39
2.7.1	The Language of Regular Expressions	39
2.7.2	Finite State Machines	40
2.7.3	Lexer Generators	42
2.8	Parsing	42
2.9	Top-Down Parsers	43
2.9.1	An LL(1) Grammar	43
2.9.2	A Non-LL(1) Grammar	44
2.9.3	An LL(1) Infix Expression Grammar	45
2.10	Bottom-Up Parsers	45
2.10.1	Parsing an Infix Expression	46
2.11	Ambiguity in Grammars	49
2.12	Other Forms of Grammars	49
2.13	Limitations of Syntactic Definitions	50
2.14	Chapter Summary	51
2.15	Review Questions	51
2.16	Exercises	52
2.17	Solutions to Practice Problems	52
3	Assembly Language	57
3.1	Overview of the JCoCo VM	58
3.2	Getting Started	61
3.3	Input/Output	64
3.4	If-Then-Else Statements	66
3.4.1	If-Then Statements	69
3.5	While Loops	71
3.6	Exception Handling	73
3.7	List Constants	76
3.8	Calling a Method	77
3.9	Iterating Over a List	79
3.10	Range Objects and Lazy Evaluation	81
3.11	Functions and Closures	83
3.12	Recursion	87
3.13	Support for Classes and Objects	89
3.13.1	Inheritance	92
3.13.2	Dynamically Created Classes	94
3.14	Chapter Summary	98
3.15	Review Questions	98
3.16	Exercises	99
3.17	Solutions to Practice Problems	100

4	Object-Oriented Programming	111
4.1	The Java Environment	114
4.2	The C++ Environment	116
4.2.1	The Macro Processor	119
4.2.2	The Make Tool	120
4.3	Namespaces	121
4.4	Dynamic Linking	122
4.5	Defining the Main Function	123
4.6	I/O Streams	124
4.7	Garbage Collection	125
4.8	Threading	126
4.9	The PyToken Class	127
4.9.1	The C++ PyToken Class	128
4.10	Inheritance and Polymorphism	130
4.11	Interfaces and Adapters	134
4.12	Functions as Values	136
4.13	Anonymous Inner Classes	137
4.14	Type Casting and Generics	138
4.15	Auto-Boxing and Unboxing	141
4.16	Exception Handling in Java and C++	142
4.17	Signals	145
4.18	JCoCo in Depth	145
4.19	The Scanner	145
4.20	The Parser	148
4.21	The Assembler	151
4.22	ByteCode	152
4.23	JCoCo's Class and Interface Type Hierarchy	155
4.24	Code	157
4.25	Functions	158
4.26	Classes	160
4.27	Methods	160
4.28	JCoCo Exceptions and Tracebacks	163
4.29	Magic Methods	165
4.30	Dictionaries	168
4.30.1	Two New Classes	169
4.30.2	Two New Types	171
4.30.3	Two New Instructions	171
4.31	Chapter Summary	171
4.32	Review Questions	172
4.33	Exercises	173
4.34	Solutions to Practice Problems	175

5	Functional Programming	179
5.1	Imperative Versus Functional Programming	181
5.2	The Lambda Calculus	182
5.2.1	Normal Form	182
5.2.2	Problems with Applicative Order Reduction	184
5.3	Getting Started with Standard ML	184
5.4	Expressions, Types, Structures, and Functions	185
5.5	Recursive Functions	187
5.6	Characters, Strings, and Lists	190
5.7	Pattern Matching	193
5.8	Tuples	194
5.9	Let Expressions and Scope	194
5.10	Datatypes	197
5.11	Parameter Passing in Standard ML	200
5.12	Efficiency of Recursion	200
5.13	Tail Recursion	203
5.14	Currying	204
5.15	Anonymous Functions	206
5.16	Higher-Order Functions	207
5.16.1	Composition	207
5.16.2	Map	208
5.16.3	Reduce or Foldright	209
5.16.4	Filter	211
5.17	Continuation Passing Style	212
5.18	Input and Output	213
5.19	Programming with Side-effects	214
5.19.1	Variables in Standard ML	214
5.19.2	Sequential Execution	215
5.19.3	Iteration	216
5.20	Exception Handling	216
5.21	Encapsulation in ML	217
5.21.1	Signatures	217
5.21.2	Implementing a Signature	218
5.22	Type Inference	219
5.23	Building a Prefix Calculator Interpreter	220
5.23.1	The Prefix Calc Parser	222
5.23.2	The AST Evaluator	222
5.23.3	Imperative Programming Observations	224
5.24	Chapter Summary	224
5.25	Exercises	225
5.26	Solutions to Practice Problems	228

6	Compiling Standard ML	235
6.1	ML-lex	237
6.2	The Small AST Definition	241
6.3	Using ML-yacc	243
6.4	Compiling and Running the Compiler	248
6.5	Function Calls	252
6.6	Let Expressions	254
6.7	Unary Negation	257
6.8	If-Then-Else Expressions	259
6.9	Short-Circuit Logic	262
6.10	Defining Functions	265
6.10.1	Curried Functions	267
6.10.2	Mutually Recursive Functions	268
6.11	Reference Variables	269
6.12	Chapter Summary	272
6.13	Review Questions	273
6.14	Exercises	273
6.15	Solutions to Practice Problems	276
7	Logic Programming	277
7.1	Getting Started with Prolog	279
7.2	Fundamentals	280
7.3	The Prolog Program	281
7.4	Lists	283
7.5	The Accumulator Pattern	284
7.6	Built-In Predicates	285
7.7	Unification and Arithmetic	285
7.8	Input and Output	286
7.9	Structures	287
7.10	Parsing in Prolog	289
7.10.1	Difference Lists	292
7.11	Prolog Grammar Rules	293
7.12	Building an AST	294
7.13	Attribute Grammars	295
7.13.1	Synthesized Versus Inherited	298
7.14	Chapter Summary	298
7.15	Review Questions	299
7.16	Exercises	299
7.17	Solutions to Practice Problems	301
8	Standard ML Type Inference	305
8.1	Why Static Type Inference?	306
8.1.1	Exception Program	306
8.1.2	A Bad Function Call	307

8.2	Type Inference Rules	308
8.3	Using Prolog.	309
8.4	The Type Environment.	312
8.5	Integers, Strings, and Boolean Constants	313
8.6	List and Tuple Constants	314
8.7	Identifiers	315
8.8	Function Application	316
8.8.1	Instantiation.	319
8.9	Let Expressions	319
8.10	Patterns.	321
8.11	Matches	325
8.12	Anonymous Functions	326
8.13	Sequential Execution	327
8.14	If-Then and While-Do	327
8.15	Exception Handling	328
8.16	Chapter Summary.	329
8.17	Review Questions.	329
8.18	Exercises.	330
8.19	Solutions to Practice Problems	334
9	Appendix A: The JCoCo Virtual Machine Specification	337
9.1	Types	338
9.2	JCoCo Magic and Attr Methods.	339
9.3	Global Built-In Functions	340
9.4	Virtual Machine Instructions.	341
9.5	Arithmetic Instructions	342
9.6	Load and Store Instructions	342
9.7	List, Tuple, and Dictionary Instructions	344
9.8	Stack Manipulation Instructions	345
9.9	Conditional and Iterative Execution Instructions.	345
9.10	Function Execution Instructions	347
9.11	Special Instructions.	348
10	Appendix B: The Standard ML Basis Library	349
10.1	The Bool Structure	349
10.2	The Int Structure.	350
10.3	The Real Structure	352
10.4	The Char Structure	357
10.5	The String Structure	358
10.6	The List Structure.	361
10.7	The Array Structure	364
10.8	The TextIO Structure	365
	Bibliography	369

<http://www.springer.com/978-3-319-70789-1>

Foundations of Programming Languages

Lee, K.D.

2017, XIV, 370 p. 189 illus., 39 illus. in color., Softcover

ISBN: 978-3-319-70789-1