Contents

List of Figures **xiii**

Preface **xvii**

About the Authors **xxiii**

Acknowledgments **xxv**

1 Compilation **1**
1.1 Compilers . . . 1
1.1.1 Programming Languages . . . 1
1.1.2 Machine Languages . . . 2
1.2 Why Should We Study Compilers? . . . 3
1.3 How Does a Compiler Work? The Phases of Compilation . . . 4
1.3.1 Front End . . . 4
1.3.2 Back End . . . 5
1.3.3 "Middle End" . . . 6
1.3.4 Advantages to Decomposition . . . 6
1.3.5 Compiling to a Virtual Machine: New Boundaries . . . 7
1.3.6 Compiling JVM Code to a Register Architecture . . . 8
1.4 An Overview of the *j--* to JVM Compiler . . . 8
1.4.1 *j--* Compiler Organization . . . 9
1.4.2 Scanner . . . 10
1.4.3 Parser . . . 11
1.4.4 AST . . . 13
1.4.5 Types . . . 13
1.4.6 Symbol Table . . . 13
1.4.7 `preAnalyze()` and `analyze()` . . . 15
1.4.8 Stack Frames . . . 15
1.4.9 `codegen()` . . . 16
1.5 *j--* Compiler Source Tree . . . 18
1.6 Organization of This Book . . . 23
1.7 Further Readings . . . 24
1.8 Exercises . . . 24

2 Lexical Analysis **29**
2.1 Introduction . . . 29
2.2 Scanning Tokens . . . 30
2.3 Regular Expressions . . . 37
2.4 Finite State Automata . . . 39
2.5 Non-Deterministic Finite-State Automata (NFA) versus Deterministic Finite-State Automata (DFA) . . . 40

2.6 Regular Expressions to NFA . . . 41
2.7 NFA to DFA . . . 46
2.8 Minimal DFA . . . 48
2.9 JavaCC: Tool for Generating Scanners . . . 54
2.10 Further Readings . . . 56
2.11 Exercises . . . 57

3 Parsing **59**
3.1 Introduction . . . 59
3.2 Context-Free Grammars and Languages . . . 61
3.2.1 Backus–Naur Form (BNF) and Its Extensions . . . 61
3.2.2 Grammar and the Language It Describes . . . 63
3.2.3 Ambiguous Grammars and Unambiguous Grammars . . . 66
3.3 Top-Down Deterministic Parsing . . . 70
3.3.1 Parsing by Recursive Descent . . . 72
3.3.2 LL(1) Parsing . . . 76
3.4 Bottom-Up Deterministic Parsing . . . 90
3.4.1 Shift-Reduce Parsing Algorithm . . . 90
3.4.2 LR(1) Parsing . . . 92
3.4.3 LALR(1) Parsing . . . 110
3.4.4 LL or LR? . . . 116
3.5 Parser Generation Using JavaCC . . . 117
3.6 Further Readings . . . 122
3.7 Exercises . . . 123

4 Type Checking **127**
4.1 Introduction . . . 127
4.2 *j--* Types . . . 127
4.2.1 Introduction to *j--* Types . . . 127
4.2.2 Type Representation Problem . . . 128
4.2.3 Type Representation and Class Objects . . . 128
4.3 *j--* Symbol Tables . . . 129
4.3.1 Contexts and Idefns: Declaring and Looking Up Types and Local Variables . . . 129
4.3.2 Finding Method and Field Names in Type Objects . . . 133
4.4 Pre-Analysis of *j--* Programs . . . 134
4.4.1 An Introduction to Pre-Analysis . . . 134
4.4.2 `JCompilationUnit.preAnalyze()` . . . 135
4.4.3 `JClassDeclaration.preAnalyze()` . . . 136
4.4.4 `JMethodDeclaration.preAnalyze()` . . . 137
4.4.5 `JFieldDeclaration.preAnalyze()` . . . 139
4.4.6 Symbol Table Built by `preAnalyze()` . . . 139
4.5 Analysis of *j--* Programs . . . 140
4.5.1 Top of the AST . . . 141
4.5.2 Declaring Formal Parameters and Local Variables . . . 143
4.5.3 Simple Variables . . . 152
4.5.4 Field Selection and Message Expressions . . . 154
4.5.5 Typing Expressions and Enforcing the Type Rules . . . 158
4.5.6 Analyzing Cast Operations . . . 159
4.5.7 Java's Definite Assignment Rule . . . 161
4.6 Visitor Pattern and the AST Traversal Mechanism . . . 161

4.7 Programming Language Design and Symbol Table Structure 162
4.8 Attribute Grammars 163
4.8.1 Examples 163
4.8.2 Formal Definition 166
4.8.3 *j--* Examples 167
4.9 Further Readings 168
4.10 Exercises 168

5 JVM Code Generation 171
5.1 Introduction 171
5.2 Generating Code for Classes and Their Members 175
5.2.1 Class Declarations 176
5.2.2 Method Declarations 177
5.2.3 Constructor Declarations 177
5.2.4 Field Declarations 178
5.3 Generating Code for Control and Logical Expressions 178
5.3.1 Branching on Condition 178
5.3.2 Short-Circuited `&&` 180
5.3.3 Logical Not `!` 181
5.4 Generating Code for Message Expressions, Field Selection, and Array Access Expressions 181
5.4.1 Message Expressions 181
5.4.2 Field Selection 183
5.4.3 Array Access Expressions 184
5.5 Generating Code for Assignment and Similar Operations 184
5.5.1 Issues in Compiling Assignment 184
5.5.2 Comparing Left-Hand Sides and Operations 186
5.5.3 Factoring Assignment-Like Operations 188
5.6 Generating Code for String Concatenation 189
5.7 Generating Code for Casts 190
5.8 Further Readings 191
5.9 Exercises 191

6 Translating JVM Code to MIPS Code 205
6.1 Introduction 205
6.1.1 What Happens to JVM Code? 205
6.1.2 What We Will Do Here, and Why 206
6.1.3 Scope of Our Work 207
6.2 SPIM and the MIPS Architecture 209
6.2.1 MIPS Organization 209
6.2.2 Memory Organization 210
6.2.3 Registers 211
6.2.4 Routine Call and Return Convention 212
6.2.5 Input and Output 212
6.3 Our Translator 213
6.3.1 Organization of Our Translator 213
6.3.2 HIR Control-Flow Graph 214
6.3.3 Simple Optimizations on the HIR 221
6.3.4 Low-Level Intermediate Representation (LIR) 227
6.3.5 Simple Run-Time Environment 229
6.3.6 Generating SPIM Code 238

6.3.7 Peephole Optimization of the SPIM Code . . . 240
6.4 Further Readings . . . 241
6.5 Exercises . . . 241

7 Register Allocation 245
7.1 Introduction . . . 245
7.2 Naïve Register Allocation . . . 245
7.3 Local Register Allocation . . . 246
7.4 Global Register Allocation . . . 246
7.4.1 Computing Liveness Intervals . . . 246
7.4.2 Linear Scan Register Allocation . . . 255
7.4.3 Register Allocation by Graph Coloring . . . 268
7.5 Further Readings . . . 274
7.6 Exercises . . . 274

8 Celebrity Compilers 277
8.1 Introduction . . . 277
8.2 Java HotSpot Compiler . . . 277
8.3 Eclipse Compiler for Java (ECJ) . . . 280
8.4 GNU Java Compiler (GCJ) . . . 283
8.4.1 Overview . . . 283
8.4.2 GCJ in Detail . . . 284
8.5 Microsoft C# Compiler for .NET Framework . . . 285
8.5.1 Introduction to .NET Framework . . . 285
8.5.2 Microsoft C# Compiler . . . 288
8.5.3 Classic Just-in-Time Compilation in the CLR . . . 289
8.6 Further Readings . . . 292

Appendix A Setting Up and Running j-- 293
A.1 Introduction . . . 293
A.2 Obtaining *j--* . . . 293
A.3 What Is in the Distribution? . . . 293
A.3.1 Scripts . . . 295
A.3.2 Ant Targets . . . 295
A.4 Setting Up *j--* for Command-Line Execution . . . 296
A.5 Setting Up *j--* in Eclipse . . . 296
A.6 Running/Debugging the Compiler . . . 297
A.7 Testing Extensions to *j--* . . . 298
A.8 Further Readings . . . 298

Appendix B j-- Language 299
B.1 Introduction . . . 299
B.2 *j--* Program and Its Class Declarations . . . 299
B.3 *j--* Types . . . 301
B.4 *j--* Expressions and Operators . . . 302
B.5 *j--* Statements and Declarations . . . 302
B.6 Syntax . . . 302
B.6.1 Lexical Grammar . . . 303
B.6.2 Syntactic Grammar . . . 304
B.6.3 Relationship of *j--* to Java . . . 306

Appendix C Java Syntax **307**
C.1 Introduction . . . 307
C.2 Syntax . . . 307
C.2.1 Lexical Grammar . . . 307
C.2.2 Syntactic Grammar . . . 309
C.3 Further Readings . . . 313

Appendix D JVM, Class Files, and the `CLEmitter` **315**
D.1 Introduction . . . 315
D.2 Java Virtual Machine (JVM) . . . 315
D.2.1 `pc` Register . . . 316
D.2.2 JVM Stacks and Stack Frames . . . 316
D.2.3 Heap . . . 318
D.2.4 Method Area . . . 318
D.2.5 Run-Time Constant Pool . . . 318
D.2.6 Abrupt Method Invocation Completion . . . 319
D.3 Class File . . . 319
D.3.1 Structure of a Class File . . . 319
D.3.2 Names and Descriptors . . . 321
D.4 `CLEmitter` . . . 322
D.4.1 `CLEmitter` Operation . . . 322
D.4.2 `CLEmitter` Interface . . . 323
D.5 JVM Instruction Set . . . 327
D.5.1 Object Instructions . . . 328
D.5.2 Field Instructions . . . 328
D.5.3 Method Instructions . . . 329
D.5.4 Array Instructions . . . 330
D.5.5 Arithmetic Instructions . . . 331
D.5.6 Bit Instructions . . . 332
D.5.7 Comparison Instructions . . . 332
D.5.8 Conversion Instructions . . . 333
D.5.9 Flow Control Instructions . . . 333
D.5.10 Load Store Instructions . . . 335
D.5.11 Stack Instructions . . . 337
D.5.12 Other Instructions . . . 338
D.6 Further Readings . . . 339

Appendix E MIPS and the SPIM Simulator **341**
E.1 Introduction . . . 341
E.2 Obtaining and Running SPIM . . . 341
E.3 Compiling *j--* Programs to SPIM Code . . . 341
E.4 Extending the JVM-to-SPIM Translator . . . 343
E.5 Further Readings . . . 344

Bibliography **345**

Index **351**