Table of Contents

Preface xiii

Running the Example Programs xxix

1 Introduction to Programming Concepts 1

1.1 A calculator . . . 1
1.2 Variables . . . 2
1.3 Functions . . . 2
1.4 Lists . . . 4
1.5 Functions over lists . . . 7
1.6 Correctness . . . 9
1.7 Complexity . . . 10
1.8 Lazy evaluation . . . 11
1.9 Higher-order programming . . . 13
1.10 Concurrency . . . 14
1.11 Dataflow . . . 15
1.12 Explicit state . . . 16
1.13 Objects . . . 17
1.14 Classes . . . 18
1.15 Nondeterminism and time . . . 20
1.16 Atomicity . . . 21
1.17 Where do we go from here? . . . 22
1.18 Exercises . . . 23

I GENERAL COMPUTATION MODELS 27

2 Declarative Computation Model 29

2.1 Defining practical programming languages . . . 30
2.2 The single-assignment store . . . 42
2.3 Kernel language . . . 49
2.4 Kernel language semantics . . . 56
2.5 Memory management . . . 72
2.6 From kernel language to practical language . . . 79
2.7 Exceptions . . . 90

2.8 Advanced topics . . . 96
2.9 Exercises . . . 107

3 Declarative Programming Techniques **111**
3.1 What is declarativeness? . . . 114
3.2 Iterative computation . . . 118
3.3 Recursive computation . . . 124
3.4 Programming with recursion . . . 127
3.5 Time and space efficiency . . . 166
3.6 Higher-order programming . . . 177
3.7 Abstract data types . . . 195
3.8 Nondeclarative needs . . . 210
3.9 Program design in the small . . . 218
3.10 Exercises . . . 230

4 Declarative Concurrency **233**
4.1 The data-driven concurrent model . . . 235
4.2 Basic thread programming techniques . . . 246
4.3 Streams . . . 256
4.4 Using the declarative concurrent model directly . . . 272
4.5 Lazy execution . . . 278
4.6 Soft real-time programming . . . 304
4.7 The Haskell language . . . 308
4.8 Limitations and extensions of declarative programming . . . 313
4.9 Advanced topics . . . 326
4.10 Historical notes . . . 337
4.11 Exercises . . . 338

5 Message-Passing Concurrency **345**
5.1 The message-passing concurrent model . . . 347
5.2 Port objects . . . 350
5.3 Simple message protocols . . . 353
5.4 Program design for concurrency . . . 362
5.5 Lift control system . . . 365
5.6 Using the message-passing model directly . . . 377
5.7 The Erlang language . . . 386
5.8 Advanced topic . . . 394
5.9 Exercises . . . 399

6 Explicit State **405**
6.1 What is state? . . . 408
6.2 State and system building . . . 410
6.3 The declarative model with explicit state . . . 413
6.4 Data abstraction . . . 419
6.5 Stateful collections . . . 435

6.6 Reasoning with state . . . 440
6.7 Program design in the large . . . 450
6.8 Case studies . . . 463
6.9 Advanced topics . . . 479
6.10 Exercises . . . 482

7 Object-Oriented Programming 489
7.1 Inheritance . . . 491
7.2 Classes as complete data abstractions . . . 492
7.3 Classes as incremental data abstractions . . . 502
7.4 Programming with inheritance . . . 518
7.5 Relation to other computation models . . . 537
7.6 Implementing the object system . . . 545
7.7 The Java language (sequential part) . . . 551
7.8 Active objects . . . 556
7.9 Exercises . . . 567

8 Shared-State Concurrency 569
8.1 The shared-state concurrent model . . . 573
8.2 Programming with concurrency . . . 573
8.3 Locks . . . 582
8.4 Monitors . . . 592
8.5 Transactions . . . 600
8.6 The Java language (concurrent part) . . . 615
8.7 Exercises . . . 618

9 Relational Programming 621
9.1 The relational computation model . . . 623
9.2 Further examples . . . 627
9.3 Relation to logic programming . . . 631
9.4 Natural language parsing . . . 641
9.5 A grammar interpreter . . . 650
9.6 Databases . . . 654
9.7 The Prolog language . . . 660
9.8 Exercises . . . 671

II SPECIALIZED COMPUTATION MODELS 677

10 Graphical User Interface Programming 679
10.1 The declarative/procedural approach . . . 681
10.2 Using the declarative/procedural approach . . . 682
10.3 The Prototyper interactive learning tool . . . 689
10.4 Case studies . . . 690
10.5 Implementing the GUI tool . . . 703

10.6 Exercises 703

11 Distributed Programming **707**
11.1 Taxonomy of distributed systems 710
11.2 The distribution model 712
11.3 Distribution of declarative data 714
11.4 Distribution of state 720
11.5 Network awareness 723
11.6 Common distributed programming patterns 724
11.7 Distribution protocols 732
11.8 Partial failure 739
11.9 Security 743
11.10 Building applications 745
11.11 Exercises 746

12 Constraint Programming **749**
12.1 Propagate-and-search 750
12.2 Programming techniques 755
12.3 The constraint-based computation model 758
12.4 Defining and using computation spaces 762
12.5 Implementing the relational computation model 772
12.6 Exercises 774

III SEMANTICS **777**

13 Language Semantics **779**
13.1 The general computation model 780
13.2 Declarative concurrency 804
13.3 Eight computation models 806
13.4 Semantics of common abstractions 808
13.5 Historical notes 808
13.6 Exercises 809

IV APPENDIXES **813**

A Mozart System Development Environment **815**
A.1 Interactive interface 815
A.2 Command line interface 817

B Basic Data Types **819**
B.1 Numbers (integers, floats, and characters) 819
B.2 Literals (atoms and names) 824
B.3 Records and tuples 825

B.4 Chunks (limited records) 828
B.5 Lists 828
B.6 Strings 830
B.7 Virtual strings 831

C Language Syntax 833
C.1 Interactive statements 834
C.2 Statements and expressions 834
C.3 Nonterminals for statements and expressions 836
C.4 Operators 836
C.5 Keywords 839
C.6 Lexical syntax 839

D General Computation Model 843
D.1 Creative extension principle 844
D.2 Kernel language 845
D.3 Concepts 846
D.4 Different forms of state 849
D.5 Other concepts 850
D.6 Layered language design 850

References 853

Index 863