

# Introduction to Computing

*Explorations in  
Language, Logic, and Machines*

David Evans  
*University of Virginia*

For the latest version of this book and supplementary materials, visit:

<http://computingbook.org>

Version: August 19, 2011



Attribution-Noncommercial-Share Alike 3.0 United States License

# Contents

<b>1</b>	<b>Computing</b>	<b>1</b>
1.1	Processes, Procedures, and Computers . . . . .	2
1.2	Measuring Computing Power . . . . .	3
1.2.1	Information . . . . .	3
1.2.2	Representing Data . . . . .	8
1.2.3	Growth of Computing Power . . . . .	12
1.3	Science, Engineering, and the Liberal Arts . . . . .	13
1.4	Summary and Roadmap . . . . .	16
<b>Part I: Defining Procedures</b>		
<b>2</b>	<b>Language</b>	<b>19</b>
2.1	Surface Forms and Meanings . . . . .	19
2.2	Language Construction . . . . .	20
2.3	Recursive Transition Networks . . . . .	22
2.4	Replacement Grammars . . . . .	26
2.5	Summary . . . . .	32
<b>3</b>	<b>Programming</b>	<b>35</b>
3.1	Problems with Natural Languages . . . . .	36
3.2	Programming Languages . . . . .	37
3.3	Scheme . . . . .	39
3.4	Expressions . . . . .	40
3.4.1	Primitives . . . . .	40
3.4.2	Application Expressions . . . . .	41
3.5	Definitions . . . . .	44
3.6	Procedures . . . . .	45
3.6.1	Making Procedures . . . . .	45
3.6.2	Substitution Model of Evaluation . . . . .	46
3.7	Decisions . . . . .	48
3.8	Evaluation Rules . . . . .	50
3.9	Summary . . . . .	52
<b>4</b>	<b>Problems and Procedures</b>	<b>53</b>
4.1	Solving Problems . . . . .	53
4.2	Composing Procedures . . . . .	54
4.2.1	Procedures as Inputs and Outputs . . . . .	55
4.3	Recursive Problem Solving . . . . .	56
4.4	Evaluating Recursive Applications . . . . .	64
4.5	Developing Complex Programs . . . . .	67
4.5.1	Printing . . . . .	68

4.5.2	Tracing . . . . .	69
4.6	Summary . . . . .	73
<b>5</b>	<b>Data</b>	<b>75</b>
5.1	Types . . . . .	75
5.2	Pairs . . . . .	77
5.2.1	Making Pairs . . . . .	79
5.2.2	Triples to Octuples . . . . .	80
5.3	Lists . . . . .	81
5.4	List Procedures . . . . .	83
5.4.1	Procedures that Examine Lists . . . . .	83
5.4.2	Generic Accumulators . . . . .	84
5.4.3	Procedures that Construct Lists . . . . .	86
5.5	Lists of Lists . . . . .	90
5.6	Data Abstraction . . . . .	92
5.7	Summary of Part I . . . . .	102

## Part II: Analyzing Procedures

<b>6</b>	<b>Machines</b>	<b>105</b>
6.1	History of Computing Machines . . . . .	106
6.2	Mechanizing Logic . . . . .	108
6.2.1	Implementing Logic . . . . .	109
6.2.2	Composing Operations . . . . .	111
6.2.3	Arithmetic . . . . .	114
6.3	Modeling Computing . . . . .	116
6.3.1	Turing Machines . . . . .	118
6.4	Summary . . . . .	123
<b>7</b>	<b>Cost</b>	<b>125</b>
7.1	Empirical Measurements . . . . .	125
7.2	Orders of Growth . . . . .	129
7.2.1	Big $O$ . . . . .	130
7.2.2	Omega . . . . .	133
7.2.3	Theta . . . . .	134
7.3	Analyzing Procedures . . . . .	136
7.3.1	Input Size . . . . .	136
7.3.2	Running Time . . . . .	137
7.3.3	Worst Case Input . . . . .	138
7.4	Growth Rates . . . . .	139
7.4.1	No Growth: Constant Time . . . . .	139
7.4.2	Linear Growth . . . . .	140
7.4.3	Quadratic Growth . . . . .	145
7.4.4	Exponential Growth . . . . .	147
7.4.5	Faster than Exponential Growth . . . . .	149
7.4.6	Non-terminating Procedures . . . . .	149
7.5	Summary . . . . .	149
<b>8</b>	<b>Sorting and Searching</b>	<b>153</b>
8.1	Sorting . . . . .	153
8.1.1	Best-First Sort . . . . .	153
8.1.2	Insertion Sort . . . . .	157

8.1.3	Quicker Sorting . . . . .	158
8.1.4	Binary Trees . . . . .	161
8.1.5	Quicksort . . . . .	166
8.2	Searching . . . . .	167
8.2.1	Unstructured Search . . . . .	168
8.2.2	Binary Search . . . . .	168
8.2.3	Indexed Search . . . . .	169
8.3	Summary . . . . .	178

**Part III: Improving Expressiveness**

**9 Mutation 179**

9.1	Assignment . . . . .	179
9.2	Impact of Mutation . . . . .	181
9.2.1	Names, Places, Frames, and Environments . . . . .	182
9.2.2	Evaluation Rules with State . . . . .	183
9.3	Mutable Pairs and Lists . . . . .	186
9.4	Imperative Programming . . . . .	188
9.4.1	List Mutators . . . . .	188
9.4.2	Imperative Control Structures . . . . .	191
9.5	Summary . . . . .	193

**10 Objects 195**

10.1	Packaging Procedures and State . . . . .	196
10.1.1	Encapsulation . . . . .	196
10.1.2	Messages . . . . .	197
10.1.3	Object Terminology . . . . .	199
10.2	Inheritance . . . . .	200
10.2.1	Implementing Subclasses . . . . .	202
10.2.2	Overriding Methods . . . . .	204
10.3	Object-Oriented Programming . . . . .	207
10.4	Summary . . . . .	209

**11 Interpreters 211**

11.1	Python . . . . .	212
11.1.1	Python Programs . . . . .	213
11.1.2	Data Types . . . . .	216
11.1.3	Applications and Invocations . . . . .	219
11.1.4	Control Statements . . . . .	219
11.2	Parser . . . . .	221
11.3	Evaluator . . . . .	223
11.3.1	Primitives . . . . .	223
11.3.2	If Expressions . . . . .	225
11.3.3	Definitions and Names . . . . .	226
11.3.4	Procedures . . . . .	227
11.3.5	Application . . . . .	228
11.3.6	Finishing the Interpreter . . . . .	229
11.4	Lazy Evaluation . . . . .	229
11.4.1	Lazy Interpreter . . . . .	230
11.4.2	Lazy Programming . . . . .	232

[Click here to download full PDF material](#)