

# Contents

		<i>page</i>	
	<i>Preface</i>		<i>ix</i>
1	What is functional programming?		1
1.1	Functions and types		1
1.2	Functional composition		3
1.3	Example: common words		3
1.4	Example: numbers into words		7
1.5	The Haskell Platform		12
1.6	Exercises		14
1.7	Answers		17
1.8	Chapter notes		20
2	Expressions, types and values		22
2.1	A session with GHCi		22
2.2	Names and operators		25
2.3	Evaluation		27
2.4	Types and type classes		30
2.5	Printing values		33
2.6	Modules		35
2.7	Haskell layout		36
2.8	Exercises		37
2.9	Answers		42
2.10	Chapter notes		47
3	Numbers		49
3.1	The type class Num		49
3.2	Other numeric type classes		50
3.3	Computing floors		52
3.4	Natural numbers		56

	3.5	Exercises	59
	3.6	Answers	61
	3.7	Chapter notes	62
4		Lists	63
	4.1	List notation	63
	4.2	Enumerations	65
	4.3	List comprehensions	66
	4.4	Some basic operations	68
	4.5	Concatenation	69
	4.6	concat, map and filter	70
	4.7	zip and zipWith	73
	4.8	Common words, completed	75
	4.9	Exercises	77
	4.10	Answers	82
	4.11	Chapter notes	87
5		A simple Sudoku solver	89
	5.1	Specification	89
	5.2	Lawful program construction	95
	5.3	Pruning the matrix of choices	97
	5.4	Expanding a single cell	101
	5.5	Exercises	105
	5.6	Answers	107
	5.7	Chapter notes	109
6		Proofs	110
	6.1	Induction over natural numbers	110
	6.2	Induction over lists	113
	6.3	The function foldr	117
	6.4	The function foldl	122
	6.5	The function scanl	125
	6.6	The maximum segment sum	127
	6.7	Exercises	131
	6.8	Answers	135
	6.9	Chapter notes	144
7		Efficiency	145
	7.1	Lazy evaluation	145
	7.2	Controlling space	149
	7.3	Controlling time	154
	7.4	Analysing time	156
	7.5	Accumulating parameters	159

7.6	Tupling	164
7.7	Sorting	167
7.8	Exercises	172
7.9	Answers	175
7.10	Chapter notes	180
8	Pretty-printing	181
8.1	Setting the scene	181
8.2	Documents	183
8.3	A direct implementation	187
8.4	Examples	189
8.5	The best layout	191
8.6	A term representation	193
8.7	Exercises	199
8.8	Answers	203
8.9	Chapter notes	209
9	Infinite lists	210
9.1	Review	210
9.2	Cyclic lists	212
9.3	Infinite lists as limits	215
9.4	Paper–rock–scissors	221
9.5	Stream-based interaction	226
9.6	Doubly-linked lists	228
9.7	Exercises	231
9.8	Answers	234
9.9	Chapter notes	237
10	Imperative functional programming	239
10.1	The IO monad	239
10.2	More monads	244
10.3	The State monad	247
10.4	The ST monad	251
10.5	Mutable arrays	254
10.6	Immutable arrays	259
10.7	Exercises	263
10.8	Answers	267
10.9	Chapter notes	275
11	Parsing	276
11.1	Parsers as monads	276
11.2	Basic parsers	279
11.3	Choice and repetition	281

164	11.4	Grammars and expressions	Tupling	7.6	285
167	11.5	Showing expressions	Sorting	7.7	288
172	11.6	Exercises	Exercises	7.8	291
172	11.7	Answers	Answers	7.9	294
180	11.8	Chapter notes	Chapter notes	7.10	297
12		A simple equational calculator	Pretty-printing		298
181	12.1	Basic considerations	Setting the scene	8.1	298
183	12.2	Expressions	Documents	8.2	304
187	12.3	Laws	A direct implementation	8.3	310
189	12.4	Calculations	Examples	8.4	312
191	12.5	Rewrites	The best layout	8.5	315
193	12.6	Matchings	A term representation	8.6	317
199	12.7	Substitutions	Exercises	8.7	319
203	12.8	Testing the calculator	Answers	8.8	321
209	12.9	Exercises	Chapter notes	8.9	331
210	12.10	Answers	Infinite lists		333
210	12.11	Chapter notes	Review	9.1	337
212		<i>Index</i>	Cyclic lists	9.2	338
212			Infinite lists as limits	9.3	339
221			Paper-rock-scissors	9.4	341
226			Suzum-based interaction	9.5	341
228			Doubly-linked lists	9.6	341
231			Exercises	9.7	341
234			Answers	9.8	341
237			Chapter notes	9.9	341
239			Imperative functional programming		341
239			The IO monad	10.1	341
244			More monads	10.2	341
247			The State monad	10.3	341
251			The ST monad	10.4	341
254			Mutable arrays	10.5	341
259			Immutable arrays	10.6	341
263			Exercises	10.7	341
267			Answers	10.8	341
272			Chapter notes	10.9	341
276			Parsing		341
276			Parsers as monads	11.1	341
279			Basic parsers	11.2	341
281			Choice and repetition	11.3	341