

Inhaltsverzeichnis

0. Bevor wir anfangen ...	1
---------------------------	---

Teil I. Elementare funktionale Programmierung

1. Was die Mathematik uns bietet	11
1.1 Modelle und Darstellungen	11
1.2 Ein mathematisches Modell: Mengen und Funktionen	13
2. Funktionen als Programmiersprache	17
2.1 Definition einfacher Funktionen	17
2.1.1 Der Computer als „Taschenrechner“: Grundterme	18
2.1.2 Darstellung von Funktionen durch Terme	21
2.1.3 Die Funktionsdefinition	22
2.2 Definitions- und Wertebereiche: Typisierung	25
2.3 Einige Beispiele für Funktionen	27
2.4 Funktionen in ML und HASKELL	27
2.4.1 Funktionen in ML	28
2.4.2 Funktionen in GOFER und HASKELL	28
2.5 Kommentare	29
2.6 ASCII-Notationen für OPAL	31
3. Modularisierung	33
3.1 Strukturen	35
3.1.1 Schnittstellen (Signaturen)	35
3.1.2 Implementierungsteil	36
3.1.3 Importe	36
3.2 Elementare Strukturen	39
3.2.1 Zahlstrukturen	40
3.2.2 Zeichen und Texte	42
3.2.3 Die Wahrheitswerte	44
3.3 Modularisierung in ML und HASKELL	45
3.3.1 Modularisierung in ML	45
3.3.2 Modularisierung in HASKELL	46

4. Ausdrücke	49
4.1 Bedingte Ausdrücke	49
4.1.1 Alternative Ausdrücke	49
4.1.2 Bewachte Ausdrücke	50
4.2 Benennung von Teilausdrücken	54
4.3 Tupel von Ausdrücken	56
4.4 Ausdrücke in ML und HASKELL	57
4.4.1 Ausdrücke in ML	57
4.4.2 Ausdrücke in GOFER und HASKELL	57
5. Rekursion	59
5.1 Rekursive Funktionen	60
5.2 Beispiele für rekursive Funktionen	62
5.3 Klassifikation rekursiver Situationen	68
6. Ein bisschen syntaktischer Zucker	71
6.1 Wider die λ -Notation	71
6.1.1 Gleichungsartige Definitionen	71
6.1.2 Die „Wildcard“-Notation	72
6.2 Wider die Klammergebirge	73
6.3 Notationelle Eigenheiten von ML und HASKELL	75
6.3.1 Notationelle Eigenheiten von ML	75
6.3.2 Notationelle Eigenheiten von GOFER und HASKELL	77
7. Drei Beispiele aus der Numerik	79
7.1 Berechnung der Quadratwurzel	79
7.2 Numerisches Differenzieren	81
7.3 Numerisches Integrieren	83

Teil II. Weiterführende Aspekte funktionaler Programmierung

8. Funktionen höherer Ordnung	89
8.1 Funktionen als Parameter	90
8.2 Funktionen als Resultate	93
8.2.1 Die naive Sicht: Funktionale als notationelle Spielerei	93
8.2.2 Die tiefeschürfende Sicht: Funktionen, die Funktionen liefern	95
8.2.3 Wieder einmal: Notationen, Notationen	99
8.3 Eine Sammlung allgemeiner Funktionale	99
8.4 Noch einmal: Die Beispiele aus der Numerik	102
8.4.1 Berechnung der Quadratwurzel	104
8.4.2 Numerische Differenziation und Integration	105
8.5 Funktionale in ML und HASKELL	106
8.5.1 Ein Beispiel in ML	106

8.5.2	Ein Beispiel in HASKELL	106
9.	Formalismen 1: Zur Semantik von Funktionen	107
9.1	Termersetzung	107
9.2	Auswertung	110
9.3	Striktheit	111
9.4	Partielle Funktionen und Programmfehler	112
10.	Formalismen 2: Namen und wo sie gelten	115
10.1	Namen	115
10.2	Gültigkeitsbereich (<i>Scope</i>)	116
10.3	Überlagerung (<i>Overloading</i>)	120
10.4	Für Fortgeschrittene: Annotierte Namen	121
10.5	Bindung von Namen an Objekte	123
10.6	Namen in ML und HASKELL	124
10.6.1	Namen in ML	124
10.6.2	Namen in HASKELL	125
11.	Formalismen 3: Aufwand und Terminierung	127
11.1	Ein Beispiel	128
11.2	Die “O-Notation”	131
11.3	Von Programmen zu Kostenfunktionen	135
11.4	Terminierung	138
11.4.1	Über wohlfundierte Ordnungen	139
11.4.2	Wie beweist man Terminierung?	139

Teil III. Datenstrukturen

12.	Konstruktion von Datenstrukturen	145
12.1	Tupel (Produkt, Aggregation)	145
12.2	Varianten (Summe)	148
12.3	Aufzählungen	151
12.4	Rekursive Datenstrukturen	152
12.5	Anmerkungen zur Methodik	152
12.6	Datenstrukturen in ML und HASKELL	156
12.6.1	Datenstrukturen in ML	156
12.6.2	Datenstrukturen in HASKELL	158
13.	Mehr syntaktischer Zucker	161
13.1	Musterbasierte Funktionsdefinitionen	161
13.2	Musterbasierte Definitionen in ML und HASKELL	164
13.2.1	Musterbasierte Definitionen in ML	165
13.2.2	Musterbasierte Definitionen in HASKELL	166

14. Datenstrukturen und Modularisierung	167
14.1 Abstrakte Datentypen	167
14.2 Die TYPE-Deklaration	170
14.3 Generische abstrakte Datentypen – naiv betrachtet	171
14.3.1 Paare	172
14.3.2 <i>Maybe it's a good value</i>	173
14.4 Abstrakte Datentypen in ML und GOFER	174
14.4.1 Abstrakte Datentypen in ML	174
14.4.2 Abstrakte Datentypen in GOFER	175
15. Listen (Sequenzen)	177
15.1 Die Definition von Listentypen	177
15.2 Elementare Listenalgorithmen	179
15.3 Ein abstrakter Datentyp für Sequenzen	185
15.4 Listen in ML und HASKELL	187
15.4.1 Listen in ML	187
15.4.2 Listen in HASKELL	187
16. Funktionale auf Listen	189
16.1 Generierung von Listen	189
16.2 Map: <i>Jeder kommt dran</i>	191
16.3 Zip: <i>Das Reißverschluss-Prinzip</i>	192
16.4 Filter: <i>Die guten ins Töpfchen</i>	193
16.5 Reduce: <i>Alles wird eins</i>	193
16.6 Kombinationen von Funktionalen	194
16.7 Strukturen für Listenfunktionale	195
17. Beispiel: Numerische Interpolation	199
18. Bäume	205
18.1 Die Definition von Baumtypen	205
18.2 Elementare Baumalgorithmen	206
18.3 Ein abstrakter Datentyp für Bäume	209
18.4 Baumtraversierung	210
18.5 Funktionale auf Bäumen	210
18.6 Beispiele für die Verwendung von Bäumen	211
18.6.1 Codebäume	211
18.6.2 Ausdrücke als Bäume	216
19. Formalismen 4:	
Parametrisierung und Polymorphie	219
19.1 Warum Polymorphie?	220
19.2 Parametrisierte Strukturen	221
19.3 Uninstanzierter Import	224
19.4 Polymorphie in ML und HASKELL	225

19.4.1	Polymorphie in ML	225
19.4.2	Funktoren: Parametrisierte Strukturen in ML	227
19.4.3	Polymorphie in HASKELL	228
19.4.4	Typklassen in HASKELL	229
20.	Suchen und Sortieren	233
20.1	Suchen in Listen	233
20.2	Sortieren von Listen	234
20.2.1	<i>Selection sort</i>	236
20.2.2	<i>Quicksort</i>	237
20.2.3	<i>Insertion sort</i>	238
20.2.4	<i>Merge sort</i>	238
20.3	Suchen und Sortieren mit Bäumen	239
<hr/>		
Teil IV. Wo, bitte, geht's zur realen Welt?		
<hr/>		
21.	Ein-/Ausgabe: Konzeptuelle Sicht	245
21.1	Die reale Welt lässt sich nicht ganz ignorieren	246
21.1.1	Unaufhaltsam enteilet die Zeit! ⁴	247
21.1.2	Die Welt ist einzigartig	250
21.2	Ein kommandobasiertes Ein-/Ausgabe-Modell	252
22.	Ein-/Ausgabe: Die Programmierung	257
22.1	Ein-/Ausgabe-Programmierung naiv betrachtet	257
22.2	Kommandos	261
22.2.1	Verknüpfung von Kommandos	264
22.2.2	Abfangen von Fehlern	265
22.2.3	Funktionale für Ein-/Ausgabe	267
22.3	Was ist eigentlich ein Programm?	268
22.4	Zur Methodik der Ein-/Ausgabe-Programmierung	269
22.5	Anmerkung zu einem alternativen Modell: „Ströme“	271
22.6	Ein-/Ausgabe in ML und HASKELL	272
22.6.1	Ein-/Ausgabe in ML	272
22.6.2	Ein-/Ausgabe in HASKELL	273
22.6.3	Die do-Notation von HASKELL	274
22.6.4	Anmerkungen zu Monaden	275
23.	Compiler und Interpreter für OPAL, ML, HASKELL, GOFER	277
23.1	OPAL	277
23.1.1	OPAL-Interpreter	278
23.1.2	OPAL-Compiler	280
23.1.3	Bezug von OPAL	282
23.2	ML	283
23.2.1	ML interaktiv	283

XIV Inhaltsverzeichnis

23.2.2 Lesen von Programmen in Dateien	284
23.2.3 Bezug von ML	285
23.3 HASKELL/GOFER	286
23.3.1 GOFER	286
23.3.2 HASKELL	287
23.3.3 Bezug von HASKELL und GOFER	288
Literaturverzeichnis	289
Index	291

Hinweis: Eine Errata-Liste und weitere Hinweise zu diesem Buch werden ggf. über die Web-Adresse <http://uebb.cs.tu-berlin.de> zu erreichen sein.



<http://www.springer.com/978-3-540-43621-8>

Funktionale Programmierung
in OPAL, ML, HASKELL und GOFER
Pepper, P.

2003, XIV, 300 S. 49 Abb., Softcover
ISBN: 978-3-540-43621-8