

Auf einen Blick

1 Einleitung	15
2 Grundbegriffe der Programmiersprachen	29
3 Lexikalische Analyse	79
4 Syntaxanalyse	133
5 Abstrakter Syntaxbaum	225
6 Semantische Analyse	253
7 Variablenallokation	301
8 Codegenerierung	335
9 Optimierung	369
10 Ausblick	413

Inhalt

Materialien zum Buch	12
Vorwort	13
1 Einleitung	15
1.1 Compiler und Sprache	15
1.2 Aufbau dieses Buches	19
2 Grundbegriffe der Programmiersprachen	29
2.1 Paradigmen	30
2.1.1 Prozedurale Programmierung	31
2.1.2 Funktionale Programmierung	33
2.1.3 Objektorientierte Programmierung	34
2.1.4 Logikbasierte Sprachen	35
2.2 Konzepte der Programmiersprachen	37
2.2.1 Programm	37
2.2.2 Literale	38
2.2.3 Operatoren und Trennzeichen	39
2.2.4 Schlüsselwörter (Keywords)	40
2.2.5 Bezeichner (Identifier)	41
2.2.6 Gültigkeitsbereiche	43
2.2.7 Lebensdauer	46
2.2.8 Typen	47
2.2.9 Weitere Merkmale von Typsystemen	56
2.2.10 Typumwandlungen	56
2.2.11 Ausdrücke	59
2.2.12 Anweisungen	60
2.2.13 Unterprogramme	61
2.3 Die Beispielsprache SPL	63
2.3.1 Trennzeichen	65
2.3.2 Kommentare	66

2.3.3	Literale	66
2.3.4	Typen	67
2.3.5	Variablen	68
2.3.6	Ausdrücke	69
2.3.7	Prozeduren	71
2.3.8	Anweisungen	72
2.3.9	Das Programm	75
2.4	Zusammenfassung	76
2.5	Übungsaufgaben	77
2.5.1	Funktionales Paradigma	77
2.5.2	Logikorientiertes Paradigma	77
2.5.3	Prozedurales Paradigma	77
2.5.4	Gültigkeitsbereiche	78
2.5.5	SPL	78
3	Lexikalische Analyse	79
3.1	Einleitung	79
3.2	Lexikalische Elemente	80
3.3	Reguläre Ausdrücke	82
3.4	Endliche Automaten	90
3.4.1	Nichtdeterministische Automaten	93
3.4.2	Elimination von ϵ -Übergängen	99
3.4.3	Deterministische Automaten	103
3.4.4	Minimierung von DEAs	109
3.5	Scanner-Generatoren	114
3.5.1	Lex bzw. Flex	114
3.5.2	JFlex	124
3.6	Zusammenfassung	129
3.7	Übungen	129
3.7.1	Reguläre Ausdrücke	129
3.7.2	Reguläre Sprachen	130
3.7.3	Nichtdeterministische Automaten	130
3.7.4	Deterministische Automaten	130

3.7.5	Minimierung von endlichen Automaten	132
3.7.6	Vervollständigung des Codes	132
4	Syntaxanalyse	133
4.1	Einleitung	133
4.2	Grammatiken	135
4.3	Pumping-Lemma für reguläre Sprachen	143
4.4	Backus-Naur-Form	146
4.5	Ableitungsbäume	148
4.5.1	Ableitungsbäume	148
4.5.2	Mehrdeutigkeit	149
4.5.3	Präzedenzen	151
4.6	Top-Down-Parser	153
4.6.1	Rekursiver Abstiegs-Parser	155
4.6.2	Grammatiktransformationen	158
4.6.3	LL(1)-Parser	160
4.7	Bottom-Up-Parser	176
4.7.1	LR(0)-Parser	178
4.7.2	SLR(1)-Parser	190
4.7.3	LR(1)-Parser	194
4.7.4	LALR(1)-Parser	197
4.8	Fehlerbehandlung	200
4.9	Parsergeneratoren	201
4.9.1	Yacc/Bison	201
4.9.2	CUP	210
4.9.3	ANTLR	216
4.10	Zusammenfassung	220
4.11	Übungen	222
4.11.1	Grammatiken	222
4.11.2	First- und Follow-Mengen	223
4.11.3	LL(1)-Parser	223
4.11.4	SLR(1)-Parser	224
4.11.5	LR(1)-Parser	224

4.11.6	LALR(1)-Parser	224
4.11.7	Parsergeneratoren	224

5 Abstrakter Syntaxbaum 225

5.1	Einleitung	225
5.2	Attributierte Grammatiken	227
5.3	Erzeugung des AST für SPL	235
5.4	Zusammenfassung	250
5.5	Übungen	251
5.5.1	Erweiterungen	251
5.5.2	ANTLR	251

6 Semantische Analyse 253

6.1	Einleitung	253
6.2	Namensanalyse	255
6.2.1	Symboltabellen	256
6.2.2	Das Visitor-Pattern	265
6.2.3	Type Patterns in Switch-Anweisungen	271
6.2.4	Typdeklarationen	272
6.2.5	Variablendeklarationen	279
6.2.6	Prozedurdeklarationen	280
6.3	Typanalyse	283
6.3.1	Typanalyse für Ausdrücke	285
6.3.2	Typanalyse für Anweisungen	291
6.4	Semantische Analyse komplett	295
6.5	Vorgehen	296
6.6	Zusammenfassung	297
6.7	Übungen	299
6.7.1	Typen	299
6.7.2	Symboltabelle	300
6.7.3	Typanalyse	300

7	Variablenallokation	301
7.1	Einleitung	301
7.2	Aktivierungsrahmen	303
7.2.1	Aufrufargumente	309
7.2.2	Lokale Variablen	313
7.2.3	Sichern der Register	315
7.2.4	Beispiel für Speicherallokation	316
7.3	Umsetzung im SPL-Compiler	318
7.4	Dynamische Speicherverwaltung	320
7.4.1	Explizite Deallokation	322
7.4.2	Implizite Deallokation	323
7.5	Erweiterungen für andere Sprachen	326
7.5.1	Zugriff auf Variablen eines umgebenden Gültigkeitsbereichs	326
7.5.2	Funktionen	329
7.5.3	Weitere Datentypen	331
7.6	Zusammenfassung	331
7.7	Übungen	332
7.7.1	AllocatorVisitor	332
7.7.2	Aktivierungsrahmen	333
7.7.3	Implementierung	333
8	Codegenerierung	335
8.1	Einleitung	335
8.2	Ziel-Hardware	336
8.2.1	RISC versus CISC	337
8.3	ECO32	337
8.3.1	Unbedingte Sprungbefehle	339
8.3.2	Befehle zum Speicherzugriff	340
8.3.3	Rechenbefehle	341
8.3.4	Sprungmarken (Labels)	342
8.3.5	Bedingte Sprünge	342

8.4	Codemuster	344
8.4.1	Ausdrücke	345
8.4.2	Zuweisungen	351
8.4.3	If-Anweisung	353
8.4.4	While-Schleifen	355
8.4.5	Zusammengesetzte Anweisung	355
8.4.6	Prozeduren	356
8.4.7	Prozeduraufufe	357
8.4.8	Beispiel	358
8.4.9	Andere Anweisungstypen	361
8.4.10	Assembler-Direktiven	361
8.4.11	Post-Processing	362
8.5	Umsetzung im SPL-Compiler	363
8.6	Zusammenfassung	364
8.7	Übungen	366
8.7.1	Weitere Sprachkonstrukte	366
8.7.2	Auswertung von Ausdrücken	366
8.7.3	Codegenerierung für Anweisungen	367
8.7.4	Implementierung	367

9	Optimierung	369
9.1	Einleitung	369
9.2	Grundlagen für die Optimierung	372
9.3	Kontrollflussanalyse	374
9.4	Datenflussanalyse	383
9.5	Lokale und globale Optimierungen	389
9.6	Schleifenoptimierungen	392
9.7	Sonstige Optimierungen	396
9.7.1	Elimination von Endrekursion	397
9.7.2	Inlining	398
9.7.3	Leaf Routine Optimization	399
9.7.4	Registeroptimierung	401
9.8	Static-Single-Assignment	405

9.9	Zusammenfassung	409
9.10	Übungen	411
9.10.1	Kontrollflussanalyse	411
9.10.2	Datenflussanalyse	412
10 Ausblick		413
10.1	AOT und JIT	413
10.2	Forschungsfelder im Compilerbau	414
Literaturverzeichnis		417
Index		425