

Inhaltsverzeichnis

Band 1

Kapitel 1	Einführung in die Compillierung	1
1.1	Compiler	1
1.2	Analyse des Quellprogramms	6
1.3	Die Phasen eines Compilers	12
1.4	Die Umgebung eines Compilers	19
1.5	Zusammenfassen von Phasen	24
1.6	Compilerbau-Werkzeuge	27
	Literaturhinweise	29
Kapitel 2	Ein einfacher Ein-Pass-Compiler	31
2.1	Überblick	31
2.2	Definition der Syntax	32
2.3	Syntaxgerichtete Übersetzung	40
2.4	Syntaxanalyse	50
2.5	Ein Übersetzer für einfache Ausdrücke	59
2.6	Lexikalische Analyse	67
2.7	Einbau einer Symboltabelle	73
2.8	Abstrakte Stapelmaschinen	77
2.9	Zusammenwirken der Techniken	84
	Übungen	95
	Literaturhinweise	99
Kapitel 3	Lexikalische Analyse	101
3.1	Die Rolle des Scanners	102
3.2	Eingabepufferung	107
3.3	Spezifikation von Symbolen	111
3.4	Erkennen von Symbolen	119
3.5	Eine Sprache zur Spezifikation von Scannern	129

3.6	Endliche Automaten	137
3.7	Vom regulären Ausdruck zum NEA	147
3.8	Entwurf eines Scanner-Generators	156
3.9	Optimierung von DEA-basierten Mustererkennern	162
	Übungen	176
	Literaturhinweise	189
Kapitel 4	Syntaxanalyse	193
4.1	Die Rolle des Parsers	194
4.2	Kontextfreie Grammatiken	201
4.3	Erstellung von Grammatiken	209
4.4	Top-Down-Syntaxanalyse	220
4.5	Bottom-Up-Syntaxanalyse	237
4.6	Operator-Precedence-Syntaxanalyse	247
4.7	LR-Parser	262
4.8	Der Gebrauch mehrdeutiger Grammatiken	301
4.9	Parser-Generatoren	313
	Übungen	325
	Literaturhinweise	338
Kapitel 5	Syntaxgesteuerte Übersetzung	341
5.1	Syntaxgesteuerte Definitionen	342
5.2	Die Konstruktion von Syntaxbäumen	351
5.3	Bottom-up-Auswertung S-attributierter Definitionen	359
5.4	L-attributierte Definitionen	363
5.5	Top-Down-Übersetzung	370
5.6	Bottom-Up-Auswertung ererbter Attribute	378
5.7	Rekursive Auswerter	388
5.8	Speicherplatz für Attributwerte zur Übersetzungszeit	392
5.9	Zuweisung von Speicherplatz zur Konstruktionszeit	397
5.10	Die Analyse syntaxgesteuerter Definitionen	404
	Übungen	412
	Literaturhinweise	417
Kapitel 6	Typüberprüfung	421
6.1	Typsysteme	423
6.2	Die Spezifikation eines einfachen Typüberprüfers	428
6.3	Gleichheit von Typausdrücken	433
6.4	Typumwandlungen	440
6.5	Die Überladung von Funktionen und Operatoren	443
6.6	Polymorphe Funktionen	447
6.7	Ein Unifikationsalgorithmus	461
	Übungen	467
	Literaturhinweise	474

Kapitel 7	Laufzeit-Umgebungen	477
7.1	Bedingungen der Quellsprache	477
7.2	Speicherverwaltung	485
7.3	Speicherzuweisungsstrategien	491
7.4	Zugriff auf nichtlokale Namen	504
7.5	Parameterübergabe	520
7.6	Symboltabellen	526
7.7	Möglichkeiten der Sprache für die dynamische Speicherzuweisung	540
7.8	Dynamische Speicherzuweisungstechniken	543
7.9	Die Speicherzuweisung in Fortran	547
	Übungen	558
	Literaturhinweise	566
	Index	

Band 2

Kapitel 8	Zwischencode-Generierung	569
8.1	Zwischensprachen	570
8.2	Deklarationen	581
8.3	Zuweisungsanweisungen	587
8.4	Boolesche Ausdrücke	599
8.5	Case-Anweisungen	609
8.6	Backpatching (Nachträgliches Einsetzen von Sprungzielen)	613
8.7	Prozeduraufrufe	620
	Übungen	622
	Literaturhinweise	625
Kapitel 9	Code-Erzeugung	627
9.1	Entscheidungen beim Entwurf eines Code-Erzeugers	628
9.2	Die Zielmaschine	634
9.3	Laufzeitspeicher-Verwaltung	637
9.4	Grundblöcke und Flußgraphen	645
9.5	Informationen über die nächste Verwendung	652
9.6	Ein einfacher Code-Generator	654
9.7	Register-Vergabe und -Zuweisung	661
9.8	Die GAG-Darstellung von Grundblöcken	668
9.9	Guckloch-Optimierung	677
9.10	Code-Erzeugung aus einem GAG	681
9.11	Code-Erzeugung durch dynamisches Programmieren	694
9.12	Generatoren für Code-Erzeuger	699
	Übungen	708
	Literaturhinweise	712

Kapitel 10 Code-Optimierung	715
10.1 Einführung	716
10.2 Die prinzipiellen Optimierungsquellen	722
10.3 Optimierung von Grundblöcken	732
10.4 Schleifen in Flußgraphen	736
10.5 Einführung in die globale Datenfluß-Analyse	743
10.6 Iteratives Lösen von Datenfluß-Gleichungen	763
10.7 Code-verbessernde Transformationen	775
10.8 Behandlung von Decknamen	794
10.9 Datenfluß-Analyse in strukturierten Flußgraphen	808
10.10 Effiziente Datenfluß-Algorithmen	823
10.11 Ein Hilfsmittel für die Datenfluß-Analyse	833
10.12 Typabschätzungen	850
10.13 Symbolisches Debuggen von optimiertem Code	860
Übungen	870
Literaturhinweise	879
Kapitel 11 Wollen Sie einen Compiler schreiben?	885
11.1 Die Planung eines Compilers	885
11.2 Methoden der Compilerentwicklung	887
11.3 Die Compiler-Entwicklungsumgebung	892
11.4 Test und Wartung	895
Kapitel 12 Beispiele einiger Compiler	897
12.1 EQN	897
12.2 Compiler für Pascal	899
12.3 C-Compiler	900
12.4 Fortran H Compiler	902
12.5 BLISS/11 Compiler	906
12.6 Optimierender Modula-2 Compiler	908
Anhang Ein Programmierprojekt	911
A.1 Einleitung	911
A.2 Die Programmstruktur	911
A.3 Die Syntax einer Untermenge von Pascal	912
A.4 Lexikalische Vereinbarungen	915
A.5 Vorschläge für Übungen	915
A.6 Die Entwicklung des Interpreters	917
A.7 Erweiterungen	918
Bibliographie	919
Index	