

Inhaltsverzeichnis

1	Programmieren unter Linux	7
1.1	Das Unix-Betriebssystem	7
1.1.1	Die Unix-Familie	7
1.1.2	Besondere Eigenschaften von Unix	8
1.1.3	Die Werkzeug-Philosophie	9
1.2	Linux	10
1.2.1	Der Kernel	10
1.2.2	Distributionen	11
1.2.3	Kompatibilität und Portabilität	11
1.3	Kommerzielle und freie Software	12
1.3.1	Das GNU-Projekt	12
1.3.2	Die GNU General Public License	14
1.3.3	Der Erfolg freier Software	15
1.4	Programmentwicklung in Unix	15
1.4.1	Wichtige Begriffe	16
1.4.2	Systemdateien zur Entwicklung	18
1.5	Übungsfragen	20
2	Grundlagen der objektorientierten Programmierung in C++	21
2.1	Grundideen	22
2.1.1	Beherrschung der Komplexität	22
2.1.2	Rückblick auf strukturiertes Programmieren	23
2.1.3	Objekte	25
2.1.4	Klassen	27
2.1.5	Methoden und Prozessabstraktion	29
2.1.6	Datenabstraktion	30
2.1.7	Zusammenfassung	31
2.1.8	Übungsaufgaben	32
2.2	Die C++-Programmiersprache	32
2.2.1	Historisches	33
2.2.2	C++ und C	34
2.2.3	C++ und Linux	34
2.2.4	Das erste C++-Programm	35

	2.2.5	Datentypen und Typumwandlung	40
	2.2.6	Operatoren	45
	2.2.7	Ausdrücke	48
	2.2.8	Zusammenfassung	49
	2.2.9	Übungsaufgaben	50
2.3		Umgang mit dem GNU-C++-Compiler	51
	2.3.1	Installation	51
	2.3.2	Aufruf und Optionen	51
	2.3.3	Name für die ausführbare Datei	52
	2.3.4	Debug-Informationen	52
	2.3.5	Fehler und Warnungen	53
	2.3.6	Kompilierung zur Objektdatei	54
	2.3.7	Pfade zu Header-Dateien	55
	2.3.8	Bibliotheken	55
	2.3.9	Optimierung	56
	2.3.10	Info-Seiten	57
	2.3.11	Editieren mit <i>NEdit</i>	59
	2.3.12	Zusammenfassung	61
	2.3.13	Übungsaufgaben	61
2.4		Klassen und Objekte	62
	2.4.1	Klassendeklaration und -definition	62
	2.4.2	Objekte von Klassen	65
	2.4.3	Zugriffsbeschränkungen	66
	2.4.4	Freunde	68
	2.4.5	Zusammenfassung	69
	2.4.6	Übungsaufgaben	70
2.5		Funktionen und Methoden	70
	2.5.1	Funktionen in C++	70
	2.5.2	Der Prototyp	74
	2.5.3	Überladen von Funktionen	75
	2.5.4	Überladen von <code>main()</code>	77
	2.5.5	Vorgabewerte für Parameter	79
	2.5.6	Referenzen und Parameterübergabe	80
	2.5.7	Zugriffsroutinen	85
	2.5.8	Inline-Funktionen	87
	2.5.9	Zusammenfassung	91
	2.5.10	Übungsaufgaben	92
2.6		Konstruktoren und Destruktoren	94
	2.6.1	Überblick über Konstruktoren	94
	2.6.2	Standardkonstruktor	96
	2.6.3	Allgemeine Konstruktoren	98
	2.6.4	Initialisierung mit Listen	100
	2.6.5	Kopierkonstruktor	102
	2.6.6	Typumwandlungskonstruktor	105

2.6.7	Destruktoren	108
2.6.8	Beispiel: Benutzerinformationen	110
2.6.9	Zusammenfassung	115
2.6.10	Übungsaufgaben	115
2.7	Vererbung und Polymorphismus	116
2.7.1	Basisklassen und abgeleitete Klassen	116
2.7.2	Vererbung in C++	117
2.7.3	Erzeugung von Unterklassenobjekten	123
2.7.4	Zugriffsbeschränkungen	126
2.7.5	Mehrfachvererbung	129
2.7.6	Polymorphismus	131
2.7.7	Rein virtuelle Funktionen und abstrakte Klassen	135
2.7.8	Zusammenfassung	137
2.7.9	Übungsaufgaben	138
3	Programmieren mit C++	141
3.1	Basiselemente	142
3.1.1	Bedingungen	142
3.1.2	Mehrfache Auswahl	149
3.1.3	Schleifen	156
3.1.4	Zusammenfassung	169
3.1.5	Übungsaufgaben	170
3.2	Dateien und Ströme	170
3.2.1	Standardein- und -ausgabe	171
3.2.2	Ein- und Ausgabe mit Dateien	173
3.2.3	Positionierung des Dateizeigers	179
3.2.4	Ausgabeformatierung	180
3.2.5	Beispiel: Umrechnung DM – Euro	182
3.2.6	Zusammenfassung	189
3.2.7	Übungsaufgaben	190
3.3	Felder, Zeiger und dynamische Speicherverwaltung	190
3.3.1	Felder (Arrays)	191
3.3.2	Zeichenketten	193
3.3.3	Zeiger	195
3.3.4	Dynamische Speicherverwaltung	199
3.3.5	Konstruktoren und Destruktoren	206
3.3.6	Beispiel: CGI-Programmierung	209
3.3.7	Zusammenfassung	223
3.3.8	Übungsaufgaben	224
3.4	Die C-Bibliothek	227
3.4.1	Umfang der C-Bibliothek	227
3.4.2	Das <i>man</i> -Kommando	229
3.4.3	Mathematische Standardfunktionen (cmath)	231
3.4.4	Numerische Limits (climits und cfloat)	233

3.4.5	Auswertung von Fehlern bei Bibliotheksfunktionen (cerrno)	234
3.4.6	Behandlung von Signalen (csignal)	236
3.4.7	Allgemeine Utilities (cstdlib)	240
3.4.8	Ein- und Ausgabefunktionen (cstdio)	245
3.4.9	Zugriff auf und Manipulation von char-Strings (cstring)	245
3.4.10	Zusammenfassung	246
3.4.11	Übungsaufgaben	247
3.5	Tipps und Konventionen	248
3.5.1	Namenskonventionen	248
3.5.2	Projektorganisation	250
3.5.3	Programmierstil	251
3.5.4	Sicheres Programmieren	252
3.5.5	C++-Programmierstil	253
3.5.6	Zusammenfassung	255
4	Fortgeschrittenes C++	257
4.1	Namensräume	257
4.1.1	Definition	258
4.1.2	Zugriff auf Bezeichner in Namensräumen	260
4.1.3	Zusammenfassung mehrerer Namensräume	263
4.1.4	Verschachtelte Namensräume	263
4.1.5	Zusammenfassung	264
4.1.6	Übungsaufgaben	264
4.2	Templates	264
4.2.1	Funktionstemplates	265
4.2.2	Organisation des Quelltextes	268
4.2.3	Klassentemplates	269
4.2.4	Zusammenfassung	277
4.2.5	Übungsaufgaben	279
4.3	Die STL: die Containerklassen der C++-Standardbibliothek	280
4.3.1	Namenskonventionen	281
4.3.2	Strings	282
4.3.3	Container	284
4.3.4	Iteratoren	289
4.3.5	Algorithmen	290
4.3.6	Zusammenfassung	293
4.3.7	Übungsaufgaben	294
4.4	Operatoren zur Typumwandlung	295
4.4.1	Der static_cast-Operator	295
4.4.2	Der dynamic_cast-Operator	296
4.4.3	Der const_cast-Operator	298

4.4.4	Der reinterpret_cast-Operator	299
4.4.5	Zusammenfassung	301
4.4.6	Übungsaufgaben	302
4.5	Überladen von Operatoren	302
4.5.1	Operatorfunktionen und -methoden	303
4.5.2	Arten von Operatoren	305
4.5.3	Der Indexoperator	306
4.5.4	Der Inkrementoperator	308
4.5.5	Der Zuweisungsoperator	310
4.5.6	Vergleiche und mathematische Operatoren	313
4.5.7	Operatoren als Freunde	316
4.5.8	Ein- und Ausgabeoperator	317
4.5.9	Typumwandlungsoperator	318
4.5.10	Allgemeine Prinzipien	320
4.5.11	Zusammenfassung	322
4.5.12	Übungsaufgaben	323
4.6	Ausnahmebehandlung (Exceptions)	324
4.6.1	Behandlung von Fehlersituationen	325
4.6.2	Exception Handling	326
4.6.3	Allgemeine Syntax	327
4.6.4	Auffangen der Ausnahmen	328
4.6.5	Beispiel: Vektor	332
4.6.6	Exceptions und die Standardbibliothek	334
4.6.7	Tipps und Hinweise	335
4.6.8	Zusammenfassung	336
4.6.9	Übungsaufgaben	337
5	Editoren für die Programmierung	339
5.1	vi	340
5.1.1	Weitere Informationen	340
5.1.2	Starten und Beenden	340
5.1.3	Die Bearbeitungsmodi	342
5.1.4	Bewegen des Cursors	343
5.1.5	Text schreiben und löschen	344
5.1.6	Suchen und Ersetzen	345
5.1.7	Speichern und Laden	346
5.1.8	Kopieren und Verschieben	347
5.1.9	Weitere Befehle	348
5.1.10	Zusammenfassung	348
5.2	Der XEmacs-Editor	348
5.2.1	Grundlegende Befehle	349
5.2.2	Suchen und Ersetzen	356
5.2.3	Ausschneiden, Kopieren und Einfügen	358
5.2.4	Modi	359

	5.2.5	Fazit	361
5.3		Weitere Editoren	361
	5.3.1	gEdit	362
	5.3.2	KWrite	363
	5.3.3	XCoral	365
	5.3.4	Und was sonst?	369
6		Werkzeuge für die Softwareentwicklung	371
6.1		Steuerung der Übersetzung mit Make-Dateien	372
	6.1.1	Aufbau von Makefiles	373
	6.1.2	Arbeiten mit <i>make</i>	377
	6.1.3	Makros	378
	6.1.4	Eingebaute Regeln	379
	6.1.5	<i>make</i> für Fortgeschrittene	381
	6.1.6	Zusammenfassung	385
6.2		Fehlersuche mit dem Debugger	386
	6.2.1	Theoretische Fehlerquellen	387
	6.2.2	Statusausgaben im Code	390
	6.2.3	Ein fehlerhaftes Programm	391
	6.2.4	Fehlersuche mit <i>gdb</i>	393
	6.2.5	Der grafische Debugger <i>DDD</i>	406
	6.2.6	Fehlervermeidung durch die Überprüfung von Vorbedingungen	415
	6.2.7	Zusammenfassung	418
6.3		Versionskontrolle mit <i>RCS</i>	419
	6.3.1	Versionskontrolle mit Linux	419
	6.3.2	Vorgehensweise	420
	6.3.3	Registrieren	423
	6.3.4	Einchecken	423
	6.3.5	Auschecken	424
	6.3.6	Überblick über Änderungen	425
	6.3.7	Einheitliches Etikett	426
	6.3.8	Makros im Quelltext	427
	6.3.9	Zusammenfassung	429
7		Integrierte Entwicklungsumgebungen	431
7.1		XEmacs als IDE	432
	7.1.1	Der Editor	432
	7.1.2	Start des Compilers	433
	7.1.3	Start des Programms und des Debuggers	433
	7.1.4	Versionsverwaltung mit XEmacs	435
	7.1.5	Dateivergleich mit Ediff	435
	7.1.6	Zusammenfassung	437
7.2		Source Code Engineering mit SNIFF+	437

7.2.1	Überblick über die SNIFF-Umgebung	438
7.2.2	Projekte in SNIFF+	439
7.2.3	Intelligente Browser	444
7.2.4	Editieren, Kompilieren und Debuggen	448
7.2.5	Entwickeln im Team	456
7.2.6	Zusammenfassung	460
7.3	Weitere Entwicklungsumgebungen	461
7.3.1	Source Navigator	462
7.3.2	CodeWarrior	463
7.3.3	KDevelop	465
7.3.4	Zusammenfassung	467
	Literaturverzeichnis	469
	Verzeichnis der Abkürzungen	472
	Index	475