

Inhaltsverzeichnis

1	Programmieren unter Linux	1
1.1	Das Unix-Betriebssystem	1
1.1.1	Die Unix-Familie	1
1.1.2	Besondere Eigenschaften von Unix	2
1.1.3	Die Werkzeug-Philosophie	3
1.2	Linux	4
1.3	Kommerzielle und freie Software	5
1.3.1	Das GNU-Projekt	6
1.3.2	Die GNU General Public License	7
1.3.3	Andere Open-Source-Ansätze	9
1.3.4	Vorteile von Open Source	10
1.3.5	Motivation für Open Source	11
1.3.6	Fazit	13
1.4	Programmentwicklung in Unix	14
1.4.1	Wichtige Begriffe	15
1.4.2	Systemdateien zur Entwicklung	17
1.5	Übungsfragen	19
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	26
2.1.4	Klassen	28
2.1.5	Methoden und Prozessabstraktion	30
2.1.6	Datenabstraktion	31
2.1.7	Zusammenfassung	32
2.1.8	Übungsaufgaben	33
2.2	Die C++-Programmiersprache	34
2.2.1	Historisches	34

2.2.2	C++ und C	35
2.2.3	C++ und Linux	36
2.2.4	Das erste C++-Programm	38
2.2.5	Datentypen und Typumwandlung	43
2.2.6	Operatoren	48
2.2.7	Ausdrücke	51
2.2.8	Zusammenfassung	52
2.2.9	Übungsaufgaben	53
2.3	Umgang mit dem GNU-C++-Compiler	54
2.3.1	Installation	55
2.3.2	Aufruf und Optionen	55
2.3.3	Name für die ausführbare Datei	56
2.3.4	Debug-Informationen	56
2.3.5	Fehler und Warnungen	57
2.3.6	Kompilierung zur Objektdatei	58
2.3.7	Pfade zu Header-Dateien	59
2.3.8	Vorkompilierte Header-Dateien	59
2.3.9	Bibliotheken	60
2.3.10	Optimierung	61
2.3.11	Info-Seiten und Texteditoren	63
2.3.12	Zusammenfassung	65
2.3.13	Übungsaufgaben	66
2.4	Klassen und Objekte	67
2.4.1	Klassendeklaration und -definition	67
2.4.2	Objekte von Klassen	70
2.4.3	Zugriffsbeschränkungen	71
2.4.4	Freunde	73
2.4.5	Zusammenfassung	75
2.4.6	Übungsaufgaben	75
2.5	Namensräume	77
2.5.1	Definition	78
2.5.2	Zugriff auf Bezeichner in Namensräumen	80
2.5.3	Zusammenfassung mehrerer Namensräume	81
2.5.4	Verschachtelte Namensräume	82
2.5.5	Zusammenfassung	83
2.5.6	Übungsaufgaben	83
2.6	Funktionen und Methoden	83
2.6.1	Funktionen in C++	83
2.6.2	Der Prototyp	87
2.6.3	Überladen von Funktionen	89
2.6.4	Überladen von main()	91

2.6.5	Vorgabewerte für Parameter	93
2.6.6	Referenzen und Parameterübergabe	94
2.6.7	Zugriffsroutinen	100
2.6.8	Inline-Funktionen	101
2.6.9	Zusammenfassung	106
2.6.10	Übungsaufgaben	107
2.7	Konstruktoren und Destruktoren	110
2.7.1	Überblick über Konstruktoren	110
2.7.2	Standardkonstruktor	112
2.7.3	Allgemeine Konstruktoren	114
2.7.4	Initialisierung mit Listen	117
2.7.5	Kopierkonstruktor	119
2.7.6	Typumwandlungskonstruktor	122
2.7.7	Destruktoren	125
2.7.8	Beispiel: Benutzerinformationen	127
2.7.9	Zusammenfassung	132
2.7.10	Übungsaufgaben	132
2.8	Vererbung und Polymorphismus	134
2.8.1	Basisklassen und abgeleitete Klassen	134
2.8.2	Vererbung in C++	136
2.8.3	Erzeugung von Unterklassenobjekten	142
2.8.4	Zugriffsbeschränkungen	144
2.8.5	Mehrfachvererbung	149
2.8.6	Polymorphismus	151
2.8.7	Rein virtuelle Funktionen und abstrakte Klassen	156
2.8.8	Zusammenfassung	158
2.8.9	Übungsaufgaben	159
3	Programmieren mit C++	163
3.1	Basiselemente	164
3.1.1	Bedingungen	164
3.1.2	Mehrfache Auswahl	171
3.1.3	Schleifen	178
3.1.4	Zusammenfassung	185
3.1.5	Übungsaufgaben	186
3.2	Dateien und Ströme	188
3.2.1	Standardein- und -ausgabe	188
3.2.2	Ein- und Ausgabe mit Dateien	190
3.2.3	Positionierung des Dateizeigers	197
3.2.4	Ausgabeformatierung	198
3.2.5	Beispiel: Umrechnung Dollar – Euro	200

3.2.6	Zusammenfassung	208
3.2.7	Übungsaufgaben	209
3.3	Felder, Zeiger und dynamische Speicherverwaltung	210
3.3.1	Felder (Arrays)	210
3.3.2	Zeichenketten	214
3.3.3	Zeiger	215
3.3.4	Dynamische Speicherverwaltung	220
3.3.5	Konstruktoren und Destruktoren	227
3.3.6	Beispiel: Verkettete Listen	230
3.3.7	Zusammenfassung	235
3.3.8	Übungsaufgaben	236
3.4	Die C-Bibliothek	239
3.4.1	Umfang der C-Bibliothek	240
3.4.2	Das <i>man</i> -Kommando	242
3.4.3	Mathematische Standardfunktionen (cmath)	244
3.4.4	Numerische Limits (climits und cfloat)	246
3.4.5	Auswertung von Fehlern bei Bibliotheksfunktionen (cerrno)	246
3.4.6	Behandlung von Signalen (csignal)	249
3.4.7	Allgemeine Utilities (cstdlib)	253
3.4.8	Ein- und Ausgabefunktionen (cstdio)	258
3.4.9	Zugriff auf und Manipulation von char-Strings (cstring)	258
3.4.10	Zusammenfassung	260
3.4.11	Übungsaufgaben	260
3.5	Eigene Bibliotheken	262
3.5.1	Statische Bibliotheken	264
3.5.2	Beispiel: Zugriff auf Verzeichnisse	268
3.5.3	Dynamische Bibliotheken (shared libraries)	272
3.5.4	Zusammenfassung	278
3.5.5	Übungsaufgaben	279
3.6	Tipps und Konventionen	280
3.6.1	Namenskonventionen	281
3.6.2	Projektorganisation	282
3.6.3	Programmierstil	284
3.6.4	Sicheres Programmieren	285
3.6.5	C++-Programmierstil	286
3.6.6	Zusammenfassung	288

4	Fortgeschrittenes C++	289
4.1	Templates	289
4.1.1	Funktionstemplates	290
4.1.2	Organisation des Quelltextes	293
4.1.3	Klassentemplates	294
4.1.4	Zusammenfassung	304
4.1.5	Übungsaufgaben	305
4.2	Die STL: die Containerklassen der C++-Standardbibliothek	306
4.2.1	Namenskonventionen	307
4.2.2	Strings	308
4.2.3	Container	310
4.2.4	Iteratoren	315
4.2.5	Algorithmen	317
4.2.6	Zusammenfassung	320
4.2.7	Übungsaufgaben	321
4.3	Operatoren zur Typumwandlung	322
4.3.1	Der static_cast-Operator	323
4.3.2	Der dynamic_cast-Operator	323
4.3.3	Der const_cast-Operator	325
4.3.4	Der reinterpret_cast-Operator	327
4.3.5	Zusammenfassung	328
4.3.6	Übungsaufgaben	329
4.4	Überladen von Operatoren	330
4.4.1	Operatorfunktionen und -methoden	330
4.4.2	Arten von Operatoren	332
4.4.3	Der Indexoperator	333
4.4.4	Der Inkrementoperator	336
4.4.5	Der Zuweisungsoperator	337
4.4.6	Vergleiche und mathematische Operatoren	341
4.4.7	Operatoren als Freunde	343
4.4.8	Ein- und Ausgabeoperator	344
4.4.9	Typumwandlungsoperator	345
4.4.10	Allgemeine Prinzipien	347
4.4.11	Zusammenfassung	350
4.4.12	Übungsaufgaben	351
4.5	Ausnahmebehandlung (Exceptions)	353
4.5.1	Behandlung von Fehlersituationen	353
4.5.2	Exception Handling	355
4.5.3	Allgemeine Syntax	355
4.5.4	Auffangen der Ausnahmen	357

4.5.5	Beispiel: Vektor	361
4.5.6	Exceptions und die Standardbibliothek	363
4.5.7	Tipps und Hinweise	365
4.5.8	Zusammenfassung	365
4.5.9	Übungsaufgaben	366
5	Editoren für die Programmierung	369
5.1	vi improved	370
5.1.1	Starten und Beenden	370
5.1.2	Die Bearbeitungsmodi	372
5.1.3	Bewegen des Cursors	373
5.1.4	Text schreiben und löschen	374
5.1.5	Suchen und Ersetzen	375
5.1.6	Speichern und Laden	376
5.1.7	Kopieren und Verschieben	377
5.1.8	Weitere Befehle	378
5.1.9	Zusammenfassung	378
5.2	Der XEmacs-Editor	379
5.2.1	Grundlegende Befehle	380
5.2.2	Suchen und Ersetzen	387
5.2.3	Ausschneiden, Kopieren und Einfügen	389
5.2.4	Modi	390
5.2.5	Fazit	392
5.3	Weitere Editoren	393
6	Werkzeuge für die Softwareentwicklung	395
6.1	Steuerung der Übersetzung mit Make-Dateien	396
6.1.1	Aufbau von Makefiles	397
6.1.2	Arbeiten mit <i>make</i>	401
6.1.3	Makros	402
6.1.4	Eingebaute Regeln	403
6.1.5	<i>make</i> für Fortgeschrittene	406
6.1.6	Zusammenfassung	410
6.2	Fehlersuche mit dem Debugger	411
6.2.1	Theoretische Fehlerquellen	412
6.2.2	Statusausgaben im Code	415
6.2.3	Ein fehlerhaftes Programm	416
6.2.4	Fehlersuche mit gdb	419
6.2.5	Fehlervermeidung durch die Überprüfung von Vorbedin- gungen	431

6.2.6	Zusammenfassung	434
6.3	Versionskontrolle mit CVS	435
6.3.1	Versionskontrolle mit Linux	436
6.3.2	Vorgehensweise	437
6.3.3	Versionsverwaltung mit CVS	437
6.3.4	Zusammenfassung	458
7	Integrierte Entwicklungsumgebungen	461
7.1	XEmacs als IDE	462
7.1.1	Der Editor	462
7.1.2	Start des Compilers	463
7.1.3	Start des Programms und des Debuggers	463
7.1.4	Versionsverwaltung mit XEmacs	464
7.1.5	Dateivergleich mit Ediff	466
7.1.6	Zusammenfassung	468
7.2	KDevelop	468
7.2.1	Überblick	468
7.2.2	Die KDevelop-Entwicklungsumgebung	469
7.2.3	Mit Konsolenanwendungen arbeiten	474
7.2.4	KDE-Anwendungen mit KDevelop entwickeln	478
7.2.5	Zusammenfassung	493
7.2.6	Übungsaufgaben	494
7.3	Eclipse	495
7.3.1	Die Idee von Eclipse	495
7.3.2	Die Eclipse-Plattform unter der Haube	496
7.3.3	Eclipse als IDE	497
7.3.4	Ein Beispielprogramm	500
7.3.5	Anbindung von CVS	505
7.3.6	Zusammenfassung	507
	Literaturverzeichnis	509
	Index	513