

<b>Vorwort .....</b>	<b>23</b>
----------------------	-----------

<b>Teil I: Einführung in C++ .....</b>	<b>25</b>
--	-----------

<b>1 Es geht los! .....</b>	<b>27</b>
1.1 Historisches .....	27
1.2 Arten der Programmierung .....	28
1.3 Werkzeuge zum Programmieren .....	29
1.4 Das erste Programm .....	30
1.5 Integrierte Entwicklungsumgebung .....	36
1.6 Einfache Datentypen und Operatoren .....	39
1.6.1 Ausdruck .....	39
1.6.2 Regeln für Namen .....	39
1.6.3 Ganze Zahlen .....	40
1.6.4 Reelle Zahlen .....	48
1.6.5 Konstanten .....	52
1.6.6 Zeichen .....	54
1.6.7 Logischer Datentyp bool .....	57
1.6.8 Regeln zum Bilden von Ausdrücken .....	58
1.6.9 Standard-Typumwandlungen .....	59
1.7 Gültigkeitsbereich und Sichtbarkeit .....	61
1.7.1 Namespace std .....	63
1.8 Kontrollstrukturen .....	64

1.8.1	Anweisungen .....	64
1.8.2	Sequenz (Reihung) .....	65
1.8.3	Auswahl (Selektion, Verzweigung) .....	66
1.8.4	Fallunterscheidungen mit switch .....	71
1.8.5	Wiederholungen.....	74
1.8.6	Kontrolle mit break und continue .....	82
1.8.7	goto .....	84
1.9	Selbst definierte und zusammengesetzte Datentypen .....	85
1.9.1	Aufzählungstypen .....	85
1.9.2	Strukturen.....	88
1.9.3	Der C++-Standardtyp vector .....	89
1.9.4	Der C++-Standardtyp array.....	95
1.9.5	Zeichenketten: der C++-Standardtyp string.....	95
1.9.6	Container und Schleifen .....	98
1.9.7	Typermittlung mit auto .....	100
1.9.8	Deklaration einer strukturierten Bindung mit auto.....	102
1.9.9	Bitfeld und Union .....	103
1.10	Einfache Ein- und Ausgabe .....	104
1.10.1	Standardein- und -ausgabe.....	104
1.10.2	Ein- und Ausgabe mit Dateien .....	108
1.11	Guter Programmierstil.....	113
<b>2</b>	<b>Programmstrukturierung .....</b>	<b>115</b>
2.1	Funktionen.....	116
2.1.1	Aufbau und Prototypen .....	116
2.1.2	nodiscard.....	118
2.1.3	Gültigkeitsbereiche und Sichtbarkeit in Funktionen .....	119
2.1.4	Lokale static-Variable: Funktion mit Gedächtnis .....	120
2.2	Schnittstellen zum Datentransfer .....	121
2.2.1	Übergabe per Wert .....	122
2.2.2	Übergabe per Referenz .....	124
2.2.3	Gefahren bei der Rückgabe von Referenzen.....	126
2.2.4	Vorgegebene Parameterwerte und unterschiedliche Parameterzahl .....	127
2.2.5	Überladen von Funktionen .....	128
2.2.6	Funktion main() .....	130
2.2.7	Beispiel Taschenrechnersimulation .....	130
2.2.8	Spezifikation von Funktionen.....	136

2.2.9	Reihenfolge der Auswertung von Argumenten .....	136
2.3	Präprozessordirektiven .....	136
2.3.1	#include .....	137
2.3.2	#define, #if, #ifdef, #ifndef, #elif, #else, #endif, #elifdef, #elifndef.....	137
2.3.3	Vermeiden mehrfacher Inkludierung .....	138
2.3.4	__has_include .....	140
2.3.5	Textersetzung mit #define .....	140
2.3.6	Umwandlung von Parametern in Zeichenketten.....	142
2.3.7	Verifizieren logischer Annahmen zur Laufzeit .....	143
2.3.8	Verifizieren logischer Annahmen zur Compilationszeit .....	143
2.3.9	Fehler- und Warnmeldungen.....	144
2.3.10	Fehler ohne Programmabbruch lokalisieren.....	144
2.4	Modulare Programmgestaltung .....	145
2.4.1	Projekt: Mehrere cpp-Dateien bilden ein Programm .....	145
2.4.2	Projekt in der IDE anlegen.....	147
2.4.3	Übersetzungseinheit, Deklaration, Definition .....	148
2.4.4	Dateiübergreifende Gültigkeit und Sichtbarkeit.....	150
2.5	Namensräume .....	151
2.5.1	Gültigkeitsbereich auf Datei beschränken .....	154
2.6	inline-Funktionen und -Variablen.....	155
2.7	constexpr-Funktionen.....	156
2.7.1	Berechnung zur Compilationszeit mit consteval.....	158
2.8	Rückgabotyp auto.....	159
2.9	Funktions-Templates .....	161
2.9.1	Spezialisierung von Templates .....	163
2.9.2	Einbinden von Templates .....	164
2.10	C++-Header.....	167
2.11	Module .....	169
<b>3</b>	<b>Objektorientierung 1 .....</b>	<b>173</b>
3.1	Datentyp und Objekt .....	175
3.2	Abstrakter Datentyp .....	175
3.3	Klassen .....	177
3.3.1	const-Objekte und Methoden.....	180
3.3.2	inline-Elementfunktionen.....	181
3.4	Initialisierung und Konstruktoren .....	182
3.4.1	Standardkonstruktor.....	182

3.4.2	Direkte Initialisierung der Attribute .....	184
3.4.3	Allgemeine Konstruktoren .....	184
3.4.4	Kopierkonstruktor .....	187
3.4.5	Typumwandlungskonstruktor .....	190
3.4.6	Konstruktor und mehr vorgeben oder verbieten .....	191
3.4.7	Einheitliche Initialisierung und Sequenzkonstruktor .....	192
3.4.8	Delegierender Konstruktor .....	194
3.4.9	constexpr-Konstruktor und -Methoden .....	195
3.5	Beispiel Rationale Zahlen .....	199
3.5.1	Aufgabenstellung .....	199
3.5.2	Entwurf .....	200
3.5.3	Implementation .....	203
3.6	Destruktoren .....	208
3.7	Wie kommt man zu Klassen und Objekten? Ein Beispiel .....	210
3.8	Gegenseitige Abhängigkeit von Klassen .....	215
<b>4</b>	<b>Zeiger .....</b>	<b>217</b>
4.1	Zeiger und Adressen .....	218
4.2	C-Arrays .....	222
4.2.1	C-Array, std::size() und sizeof .....	223
4.2.2	Initialisierung von C-Arrays .....	224
4.2.3	Zeigerarithmetik .....	224
4.2.4	Indexoperator bei C-Arrays .....	225
4.2.5	C-Array durchlaufen .....	226
4.3	C-Zeichenketten .....	227
4.3.1	Schleifen und C-Strings .....	230
4.4	Dynamische Datenobjekte .....	233
4.4.1	Freigeben dynamischer Objekte .....	236
4.5	Zeiger und Funktionen .....	239
4.5.1	Parameterübergabe mit Zeigern .....	239
4.5.2	C-Array als Funktionsparameter .....	240
4.5.3	const und Zeiger-Parameter .....	242
4.5.4	Parameter des main-Programms .....	243
4.5.5	Gefahren bei der Rückgabe von Zeigern .....	243
4.6	this-Zeiger .....	244
4.7	Mehrdimensionale C-Arrays .....	246
4.7.1	Statische mehrdimensionale C-Arrays .....	246

4.7.2	Mehrdimensionales C-Array als Funktionsparameter .....	248
4.8	Dynamisches 2D-Array .....	251
4.9	Binäre Ein-/Ausgabe .....	257
4.10	Zeiger auf Funktionen.....	260
4.11	Typumwandlungen für Zeiger.....	264
4.12	Zeiger auf Elementfunktionen und -daten .....	265
4.13	Komplexe Deklarationen lesen .....	267
4.13.1	Lesbarkeit mit typedef und using verbessern .....	267
4.14	Alternative zu rohen Zeigern, new und delete.....	269
<b>5</b>	<b>Objektorientierung 2 .....</b>	<b>273</b>
5.1	Eine String-Klasse .....	273
5.1.1	friend-Funktionen .....	279
5.2	String-Ansicht (View) .....	280
5.3	Typbestimmung mit decltype und declval.....	283
5.4	Klassenspezifische Daten und Funktionen .....	286
5.4.1	Klassenspezifische Konstante.....	290
5.5	Klassen-Templates .....	293
5.5.1	Ein Stack-Template .....	293
5.5.2	Stack mit statisch festgelegter Größe.....	295
5.5.3	Stack auf Basis verschiedener Container .....	297
5.6	Code Bloat bei der Instanziierung von Templates vermeiden.....	298
5.6.1	extern-Template .....	299
<b>6</b>	<b>Vererbung .....</b>	<b>301</b>
6.1	Vererbung und Initialisierung .....	306
6.2	Zugriffsschutz .....	307
6.3	Typbeziehung zwischen Ober- und Unterklasse .....	310
6.4	Oberklassen-Schnittstelle verwenden .....	311
6.4.1	Konstruktor erben.....	312
6.5	Überschreiben von Funktionen in abgeleiteten Klassen .....	314
6.5.1	Virtuelle Funktionen.....	316
6.5.2	Abstrakte Klassen .....	319
6.5.3	Virtueller Destruktor.....	324
6.5.4	Vererbung verbieten .....	327
6.5.5	Private virtuelle Funktionen.....	328
6.6	Probleme der Modellierung mit Vererbung.....	330
6.7	Mehrfachvererbung.....	333

6.8	Typumwandlung bei Vererbung .....	340
6.9	Typinformationen zur Laufzeit.....	342
6.10	Private-/Protected-Vererbung .....	343
<b>7</b>	<b>Fehlerbehandlung .....</b>	<b>347</b>
7.1	Ausnahmebehandlung .....	349
7.1.1	Exception-Spezifikation in Deklarationen .....	352
7.1.2	Exception-Hierarchie .....	353
7.1.3	Besondere Fehlerbehandlungsfunktionen.....	355
7.1.4	Arithmetische Fehler/Division durch 0.....	356
7.2	Speicherbeschaffung mit new .....	358
7.3	Exception-Sicherheit .....	359
7.4	Fehlerbehandlung mit optional und expected .....	360
7.4.1	Fehlerbehandlung mit optional .....	361
7.4.2	Fehlerbehandlung mit expected.....	363
7.4.3	Monadische Operationen.....	364
<b>8</b>	<b>Überladen von Operatoren .....</b>	<b>367</b>
8.1	Rationale Zahlen – noch einmal .....	369
8.1.1	Arithmetische Operatoren.....	369
8.1.2	Ausgabeoperator << .....	371
8.1.3	Gleichheitsoperator .....	372
8.2	Eine Klasse für Vektoren .....	374
8.2.1	Indexoperator [ ] .....	377
8.2.2	Zuweisungsoperator = .....	379
8.2.3	Mathematische Vektoren .....	382
8.2.4	Multiplikationsoperator .....	383
8.3	Inkrement-Operator ++ .....	385
8.4	Typumwandlungsoperator .....	389
8.5	Smart Pointer: Operatoren -> und * .....	390
8.5.1	Smart Pointer und die C++-Standardbibliothek .....	395
8.6	Objekt als Funktion .....	396
8.7	Spaceship-Operator <=> .....	398
8.7.1	Ordnungen in C++ .....	399
8.7.2	Automatische Erzeugung der Vergleichsoperatoren.....	401
8.7.3	Klassenspezifische Sortierung .....	402
8.7.4	Freie Funktionen statt Elementfunktionen .....	403
8.8	new und delete überladen .....	404

8.8.1	Unterscheidung zwischen Heap- und Stack-Objekten .....	408
8.8.2	Empfehlungen im Umgang mit new und delete .....	410
8.9	Operatoren für Literale .....	410
8.9.1	Stringliterale .....	411
8.9.2	Benutzerdefinierte Literale .....	412
8.10	Indexoperator für Matrizen .....	414
8.10.1	Zweidimensionale Matrix als Vektor von Vektoren .....	415
8.10.2	Zweidimensionale Matrix mit zusammenhängendem Speicher .....	416
8.11	Zuweisung, Kopie und Vergleich bei Vererbung .....	419
8.11.1	Polymorpher Vergleich .....	420
8.11.2	Kopie mit clone()-Methode erzeugen .....	421
<b>9</b>	<b>Dateien und Ströme .....</b>	<b>423</b>
9.1	Eingabe .....	425
9.2	Ausgabe .....	427
9.3	Formatierung mit std::format .....	429
9.3.1	Syntax für Platzhalter .....	429
9.3.2	Formatierung eigener Datentypen .....	433
9.4	Formatierung mit Flags .....	433
9.5	Formatierung mit Manipulatoren .....	437
9.6	Fehlerbehandlung .....	443
9.7	Typumwandlung von Dateiobjekten nach bool .....	445
9.8	Arbeit mit Dateien .....	446
9.8.1	Positionierung in Dateien .....	446
9.8.2	Lesen und Schreiben in derselben Datei .....	447
9.9	Umleitung auf Strings .....	448
9.10	Formatierte Daten lesen .....	449
9.10.1	Eingabe benutzerdefinierter Typen .....	449
9.11	Blockweise lesen und schreiben .....	451
9.11.1	vector-Objekt binär lesen und schreiben .....	451
9.11.2	array-Objekt binär lesen und schreiben .....	453
9.11.3	Matrix binär lesen und schreiben .....	454
<b>10</b>	<b>Die Standard Template Library (STL) .....</b>	<b>457</b>
10.1	Container, Iteratoren, Algorithmen .....	458
10.2	Iteratoren im Detail .....	463
10.3	Beispiel verkettete Liste .....	465
10.4	Ranges und Views .....	469

<b>Teil II: Fortgeschrittene Themen .....</b>	<b>473</b>
<b>11 Performance, Wert- und Referenzsemantik .....</b>	<b>475</b>
11.1 Performanceproblem Wertsemantik .....	477
11.1.1 Auslassen der Kopie .....	477
11.1.2 Temporäre Objekte bei der Zuweisung .....	478
11.2 Referenzsemantik für R-Werte .....	479
11.2.1 Kategorien von Ausdrücken .....	479
11.2.2 Referenzen auf R- und L-Werte .....	480
11.2.3 Auswertung von Referenzen auf R-Werte .....	481
11.2.4 Referenz-Qualifizierer .....	482
11.3 Optimierung durch Referenzsemantik für R-Werte.....	484
11.3.1 Bewegungskonstruktor .....	487
11.3.2 Bewegender Zuweisungsoperator .....	488
11.4 Die move()-Funktion .....	489
11.4.1 move() und Initialisierung der Attribute .....	490
11.5 Referenzen auf R-Werte und Template-Parameter .....	491
11.5.1 Auswertung von Template-Parametern – ein Überblick.....	493
11.6 Ein effizienter Plusoperator .....	493
11.6.1 Eliminieren auch des Bewegungskonstruktors .....	494
11.6.2 Kopien temporärer Objekte eliminieren .....	495
11.7 Rule of three/five/zero .....	496
<b>12 Lambda-Funktionen.....</b>	<b>501</b>
12.1 Eigenschaften.....	502
12.1.1 Äquivalenz zum Funktionszeiger .....	503
12.1.2 Lambda-Funktion und Klasse .....	504
12.2 Generische Lambda-Funktionen .....	505
12.3 Parametererfassung mit [] .....	507
<b>13 Metaprogrammierung mit Templates .....</b>	<b>509</b>
13.1 Grundlagen .....	510
13.2 Variadic Templates: Templates mit variabler Parameterzahl .....	512
13.2.1 Ablauf der Auswertung durch den Compiler.....	513
13.2.2 Anzahl der Parameter .....	514
13.2.3 Parameterexpansion .....	515
13.3 Fold-Expressions.....	516
13.3.1 Weitere Varianten .....	518



13.3.2	Fold-Expression mit Kommaoperator .....	519
13.4	Klassen-Template mit variabler Stelligkeit .....	520
13.5	Type Traits .....	521
13.5.1	Wie funktionieren Type Traits? – ein Beispiel .....	522
13.5.2	Abfrage von Eigenschaften .....	525
13.5.3	Abfrage numerischer Eigenschaften .....	527
13.5.4	Typumwandlungen .....	527
13.5.5	Auswahl weiterer Traits .....	528
13.6	Concepts .....	530
<b>14</b>	<b>Reguläre Ausdrücke .....</b>	<b>535</b>
14.1	Elemente regulärer Ausdrücke .....	536
14.1.1	Greedy oder lazy? .....	538
14.2	Interaktive Auswertung .....	540
14.3	Auszug der regex-Schnittstelle .....	543
14.4	Verarbeitung von \n .....	544
14.5	Anwendungen .....	546
<b>15</b>	<b>Threads und Coroutinen .....</b>	<b>547</b>
15.1	Zeit und Dauer .....	548
15.2	Threads .....	549
15.2.1	Automatisch join() .....	553
15.3	Die Klasse jthread .....	554
15.3.1	Übergabe eines Funktors .....	556
15.3.2	Thread-Group .....	558
15.4	Synchronisation kritischer Abschnitte .....	559
15.4.1	Data Race erkennen .....	562
15.5	Thread-Steuerung: Pausieren, Fortsetzen, Beenden .....	562
15.6	Warten auf Ereignisse .....	566
15.7	Atomare Veränderung von Variablen .....	572
15.8	Asynchrone verteilte Bearbeitung einer Aufgabe .....	575
15.9	Thread-Sicherheit .....	578
15.10	Coroutinen .....	579
<b>16</b>	<b>Grafische Benutzungsschnittstellen .....</b>	<b>585</b>
16.1	Ereignisgesteuerte Programmierung .....	586
16.2	GUI-Programmierung mit Qt .....	587
16.2.1	Meta-Objektsystem .....	587

16.2.2	Der Programmablauf .....	588
16.2.3	Ereignis abfragen .....	589
16.3	Signale, Slots und Widgets .....	590
16.4	Dialog .....	599
16.5	Qt oder Standard-C++? .....	602
16.5.1	Threads .....	602
16.5.2	Verzeichnisbaum durchwandern .....	604
<b>17</b>	<b>Internet-Anbindung .....</b>	<b>607</b>
17.1	Protokolle .....	608
17.2	Adressen .....	608
17.3	Socket .....	611
17.3.1	Bidirektionale Kommunikation .....	614
17.3.2	UDP-Sockets .....	616
17.3.3	Atomuhr mit UDP abfragen .....	618
17.4	HTTP .....	620
17.4.1	Verbindung mit GET .....	621
17.4.2	Verbindung mit POST .....	626
17.5	Mini-Webserver .....	627
17.6	OpenAI-Schnittstellen zu ChatGPT und DALL-E 2 .....	635
17.6.1	ChatGPT .....	636
17.6.2	DALL-E 2 .....	638
<b>18</b>	<b>Datenbankanbindung .....</b>	<b>639</b>
18.1	C++-Interface .....	640
18.2	Anwendungsbeispiel .....	643
 <b>Teil III: Ausgewählte Methoden und Werkzeuge der Softwareentwicklung .....</b>		<b>649</b>
<b>19</b>	<b>Effiziente Programmerzeugung mit make .....</b>	<b>651</b>
19.1	Wirkungsweise .....	652
19.2	Variablen und Muster .....	654
19.3	Universelles Makefile für einfache Projekte .....	656
19.4	Automatische Ermittlung von Abhängigkeiten .....	657
19.4.1	Makefiles für verschiedene Betriebssysteme und Compiler .....	659
19.4.2	Getrennte Verzeichnisse: src, obj, bin .....	660
19.5	Makefile für Verzeichnisbäume .....	661

---

19.5.1	Nur ein Makefile auf Projektebene .....	663
19.5.2	Rekursive Make-Aufrufe .....	664
19.6	Erzeugen von Bibliotheken .....	666
19.6.1	Statische Bibliotheksmodule.....	666
19.6.2	Dynamische Bibliotheksmodule.....	668
19.7	Weitere Build-Tools.....	671
<b>20</b>	<b>Unit-Test.....</b>	<b>673</b>
20.1	Werkzeuge .....	674
20.2	Boost Unit Test Framework.....	675
20.2.1	Fixture .....	677
20.2.2	Testprotokoll und Log-Level.....	677
20.2.3	Prüf-Makros .....	678
20.2.4	Kommandozeilen-Optionen.....	683
20.3	Test Driven Development .....	684
 <b>Teil IV: Das C++-Rezeptbuch:</b>		
	<b>Tipps und Lösungen für typische Aufgaben .....</b>	<b>685</b>
<b>21</b>	<b>Sichere Programmentwicklung.....</b>	<b>687</b>
21.1	Regeln zum Design von Methoden .....	688
21.2	Defensive Programmierung.....	689
21.2.1	double- und float-Werte richtig vergleichen .....	690
21.2.2	const und constexpr verwenden .....	691
21.2.3	Anweisungen nach for/if/while einklammern .....	691
21.2.4	int und unsigned/size_t nicht mischen .....	692
21.2.5	size_t oder auto statt unsigned int verwenden.....	692
21.2.6	Postfix++ mit Präfix++ implementieren .....	692
21.2.7	Ein Destruktor darf keine Exception werfen .....	693
21.2.8	explicit-Typumwandlungsoperator bevorzugen .....	693
21.2.9	explicit-Konstruktor für eine Typumwandlung bevorzugen .....	693
21.2.10	Leere Standardkonstruktoren vermeiden .....	693
21.2.11	Mit override Schreibfehler reduzieren.....	694
21.2.12	Kopieren und Zuweisung verbieten .....	694
21.2.13	Vererbung verbieten .....	695
21.2.14	Überschreiben einer virtuellen Methode verhindern .....	695
21.2.15	»Rule of zero« beachten .....	695

21.2.16	One Definition Rule.....	695
21.2.17	Defensiv Objekte löschen .....	696
21.2.18	Hängende Referenzen vermeiden .....	696
21.2.19	Speicherbeschaffung und -freigabe kapseln.....	696
21.2.20	Programmierrichtlinien einhalten .....	696
21.3	Exception-sichere Beschaffung von Ressourcen.....	696
21.3.1	Sichere Verwendung von <code>unique_ptr</code> und <code>shared_ptr</code> .....	697
21.3.2	So vermeiden Sie <code>new</code> und <code>delete</code> !.....	697
21.3.3	<code>shared_ptr</code> für C-Arrays korrekt verwenden.....	698
21.3.4	<code>unique_ptr</code> für C-Arrays korrekt verwenden .....	699
21.3.5	Exception-sichere Funktion .....	700
21.3.6	Exception-sicherer Konstruktor .....	701
21.3.7	Exception-sichere Zuweisung .....	701
21.4	Empfehlungen zur Thread-Programmierung.....	702
21.4.1	Warten auf die Freigabe von Ressourcen .....	702
21.4.2	Deadlock-Vermeidung.....	702
21.4.3	<code>notify_all</code> oder <code>notify_one</code> ?.....	703
21.4.4	Performance mit Threads verbessern?.....	704
<b>22</b>	<b>Von der UML nach C++ .....</b>	<b>705</b>
22.1	Vererbung .....	706
22.2	Interface anbieten und nutzen .....	706
22.3	Assoziation .....	708
22.3.1	Aggregation.....	712
22.3.2	Komposition .....	712
<b>23</b>	<b>Algorithmen für verschiedene Aufgaben .....</b>	<b>713</b>
23.1	Algorithmen mit Strings .....	714
23.1.1	String splitten .....	714
23.1.2	String in Zahl umwandeln.....	715
23.1.3	Zahl in String umwandeln.....	718
23.1.4	Strings sprachlich richtig sortieren .....	718
23.1.5	Umwandlung in Klein- bzw. Großschreibung.....	720
23.1.6	Strings sprachlich richtig vergleichen.....	722
23.1.7	Von der Groß-/Kleinschreibung unabhängiger Zeichenvergleich .....	722
23.1.8	Von der Groß-/Kleinschreibung unabhängige Suche .....	723
23.2	Textverarbeitung .....	724
23.2.1	Datei durchsuchen .....	724

23.2.2	Ersetzungen in einer Datei.....	726
23.2.3	Lines of Code (LOC) ermitteln .....	728
23.2.4	Zeilen, Wörter und Zeichen einer Datei zählen .....	729
23.2.5	CSV-Datei lesen .....	730
23.2.6	Kreuzreferenzliste .....	731
23.3	Operationen auf Folgen .....	733
23.3.1	Vereinfachungen .....	733
23.3.2	Folge mit gleichen Werten initialisieren .....	735
23.3.3	Folge mit Werten eines Generators initialisieren .....	736
23.3.4	Folge mit fortlaufenden Werten initialisieren .....	737
23.3.5	Summe und Produkt.....	737
23.3.6	Mittelwert und Standardabweichung.....	738
23.3.7	Skalarprodukt.....	739
23.3.8	Folge der Teilsummen oder -produkte .....	740
23.3.9	Folge der Differenzen.....	742
23.3.10	Kleinstes und größtes Element finden.....	743
23.3.11	Elemente rotieren .....	744
23.3.12	Elemente verwürfeln.....	746
23.3.13	Dubletten entfernen .....	746
23.3.14	Reihenfolge umdrehen .....	749
23.3.15	Stichprobe.....	749
23.3.16	Anzahl der Elemente, die einer Bedingung genügen.....	750
23.3.17	Gilt ein Prädikat für alle, kein oder wenigstens ein Element einer Folge? ..	751
23.3.18	Permutationen .....	752
23.3.19	Lexikografischer Vergleich.....	755
23.4	Sortieren und Verwandtes .....	757
23.4.1	Partitionieren .....	757
23.4.2	Sortieren.....	760
23.4.3	Stabiles Sortieren .....	761
23.4.4	Partielles Sortieren .....	762
23.4.5	Das n.-größte oder n.-kleinste Element finden .....	764
23.4.6	Verschmelzen (merge) .....	765
23.5	Suchen und Finden .....	767
23.5.1	Element finden .....	767
23.5.2	Element einer Menge in der Folge finden .....	768
23.5.3	Teilfolge finden.....	770
23.5.4	Teilfolge mit speziellem Algorithmus finden.....	771

23.5.5	Bestimmte benachbarte Elemente finden .....	772
23.5.6	Bestimmte aufeinanderfolgende Werte finden .....	773
23.5.7	Binäre Suche.....	774
23.6	Mengenoperationen auf sortierten Strukturen .....	777
23.6.1	Teilmengenrelation .....	778
23.6.2	Vereinigung .....	779
23.6.3	Schnittmenge .....	779
23.6.4	Differenz .....	780
23.6.5	Symmetrische Differenz.....	781
23.7	Heap-Algorithmen .....	782
23.8	Vergleich von Containern auch ungleichen Typs.....	786
23.8.1	Unterschiedliche Elemente finden .....	786
23.8.2	Prüfung auf gleiche Inhalte .....	788
23.9	Rechnen mit komplexen Zahlen: Der C++-Standardtyp complex.....	789
23.10	Vermischtes .....	791
23.10.1	Erkennung eines Datums.....	791
23.10.2	Erkennung einer IPv4-Adresse .....	793
23.10.3	Erzeugen von Zufallszahlen .....	794
23.10.4	for_each – auf jedem Element eine Funktion ausführen.....	799
23.10.5	Verschiedene Möglichkeiten, Container-Bereiche zu kopieren.....	800
23.10.6	Vertauschen von Elementen, Bereichen und Containern .....	803
23.10.7	Elemente transformieren .....	804
23.10.8	Ersetzen und Varianten .....	806
23.10.9	Elemente herausfiltern .....	807
23.10.10	Grenzwerte von Zahltypen .....	809
23.10.11	Minimum und Maximum .....	810
23.10.12	Wert begrenzen.....	812
23.10.13	ggT, kgV und Mitte .....	813
23.11	Parallelisierbare Algorithmen .....	814
<b>24</b>	<b>Datei- und Verzeichnisoperationen.....</b>	<b>815</b>
24.1	Übersicht .....	816
24.2	Pfadoperationen.....	817
24.3	Datei oder Verzeichnis löschen.....	818
24.4	Datei oder Verzeichnis kopieren .....	820
24.5	Verzeichnis anlegen .....	821
24.6	Datei oder Verzeichnis umbenennen .....	822

24.7 Verzeichnis anzeigen .....	823
24.8 Verzeichnisbaum anzeigen .....	824

## **Teil V: Die C++-Standardbibliothek .....825**

### **25 Aufbau und Übersicht .....827**

25.1 Auslassungen .....	830
25.2 Beispiele des Buchs und die C++-Standardbibliothek .....	831

### **26 Hilfsfunktionen und -klassen .....833**

26.1 Unterstützung der Referenzsemantik für R-Werte .....	833
26.2 Paare .....	835
26.3 Tupel .....	837
26.4 bitset .....	839
26.5 Indexfolgen .....	842
26.6 variant statt union .....	843
26.7 Funktionsobjekte .....	844
26.7.1 Arithmetische, vergleichende und logische Operationen .....	844
26.7.2 Binden von Argumentwerten .....	845
26.7.3 Funktionen in Objekte umwandeln .....	846
26.8 Templates für rationale Zahlen .....	848
26.9 Hüllklasse für Referenzen .....	850

### **27 Container ..... 851**

27.1 Gemeinsame Eigenschaften .....	853
27.1.1 Reversible Container .....	855
27.1.2 Initialisierungsliste (initializer_list) .....	856
27.1.3 Konstruktion an Ort und Stelle .....	857
27.2 Sequenzen .....	858
27.2.1 vector .....	859
27.2.2 vector<bool> .....	860
27.2.3 array .....	861
27.2.4 list .....	864
27.2.5 deque .....	867
27.3 Container-Adapter .....	868
27.3.1 stack .....	868
27.3.2 queue .....	870
27.3.3 priority_queue .....	871

27.4	Assoziative Container .....	873
27.4.1	Sortierte assoziative Container .....	875
27.4.2	Hash-Container .....	882
27.5	Sicht auf Container (span) .....	889
<b>28</b>	<b>Iteratoren .....</b>	<b>891</b>
28.1	Iterator-Kategorien .....	892
28.1.1	Anwendung von Traits .....	894
28.2	Abstand und Bewegen .....	897
28.3	Zugriff auf Anfang und Ende .....	898
28.3.1	Reverse-Iteratoren .....	899
28.4	Insert-Iteratoren .....	900
28.5	Stream-Iteratoren .....	902
<b>29</b>	<b>Algorithmen .....</b>	<b>905</b>
29.1	Algorithmen mit Prädikat .....	906
29.2	Übersicht .....	907
<b>30</b>	<b>Nationale Besonderheiten .....</b>	<b>911</b>
30.1	Sprachumgebung festlegen und ändern .....	912
30.1.1	Die locale-Funktionen .....	914
30.2	Zeichensätze und -codierung .....	915
30.3	Zeichenklassifizierung und -umwandlung .....	919
30.4	Kategorien .....	920
30.4.1	collate .....	920
30.4.2	ctype .....	921
30.4.3	numeric .....	923
30.4.4	monetary .....	924
30.4.5	messages .....	927
30.5	Konstruktion eigener Facetten .....	928
<b>31</b>	<b>String .....</b>	<b>931</b>
31.1	string_view für String-Literale .....	940
<b>32</b>	<b>Speichermanagement .....</b>	<b>943</b>
32.1	unique_ptr .....	943
32.2	shared_ptr .....	946
32.3	weak_ptr .....	948
32.4	new mit Speicherortangabe .....	949



<b>33</b>	<b>Ausgewählte C-Header .....</b>	<b>951</b>
33.1	<cassert> .....	951
33.2	<cctype> .....	952
33.3	<cmath> .....	953
33.4	<cstdlib> .....	954
33.5	<stdlib> .....	954
33.6	<cstring> .....	955
33.7	<ctime> .....	957
<b>A</b>	<b>Anhang.....</b>	<b>959</b>
A.1	ASCII-Tabelle .....	959
A.2	C++-Schlüsselwörter .....	961
A.3	Compilerbefehle .....	962
A.3.1	Optimierung .....	963
A.4	Rangfolge der Operatoren .....	963
A.5	C++-Attribute für den Compiler .....	965
A.6	Lösungen zu den Übungsaufgaben .....	966
A.7	Änderungen in der 7. Auflage .....	976
	<b>Glossar .....</b>	<b>977</b>
	<b>Literaturverzeichnis .....</b>	<b>987</b>
	<b>Register.....</b>	<b>991</b>