Auf einen Blick

TEIL I
Erste Schritte 23

TEIL II
Die Elemente von C++ 87

TEIL III
Datenstrukturen 223

TEIL IV
Fortgeschrittene Themen 365

Inhalt

Geleitwort des Fachgutachters 18
Vorwort 20

TEIL I Erste Schritte

1 Über dieses Buch 25

1.1 Der C++-Standard 26
1.2 Verwendete Formatierungen 27

2 Vom Problem zum Programm 29

2.1 Was ist Programmieren? 30
2.2 Softwareentwicklungsmethoden 30
2.3 Entwurfsmuster 32
2.4 Algorithmen 33
2.5 Ressourcen 34

3 Programmieren in C++ 36

3.1 Übersetzen 37
3.2 Aktuelle Compiler 37
3.2.1 Gnu C++ 39
3.2.2 Clang++ der LLVM 39
3.2.3 Microsoft Visual Studio 39
3.3 Entwicklungsumgebungen 39
3.4 Die Kommandozeile unter Ubuntu 41
3.4.1 Ein Programm erstellen 42
3.4.2 Automatisieren mit Makefile 44
3.5 Die IDE »Microsoft Visual Studio Express« unter Windows 44

3.6 Schneller ... 47
3.7 Aufgaben ... 47

4 Ein ganz schneller Überblick 49

4.1 Kommentare ... 50
4.2 Die »include«-Direktive ... 50
4.3 Die Standardbibliothek ... 51
4.4 Die Funktion »main()« ... 51
4.5 Typen ... 51
4.6 Variablen ... 52
4.7 Initialisierung ... 52
4.8 Ausgabe auf der Konsole ... 53
4.9 Anweisungen ... 53
4.10 Aufgaben ... 54

5 Ohne Eile erklärt 56

5.1 Leerräume, Bezeichner und Token ... 58
5.2 Kommentare ... 59
5.3 Funktionen und Argumente ... 60
5.4 Seiteneffekt-Operatoren ... 61
5.5 Die »main«-Funktion ... 62
5.6 Anweisungen ... 64
5.7 Ausdrücke ... 66
5.8 Zuweisungen ... 68
5.9 Typen ... 69
5.9.1 Der Typ »int« ... 71
5.9.2 Der Typ »bool« ... 72
5.9.3 Die Typen »const char*« und »std::string« ... 72
5.9.4 Parametrisierte Typen ... 73
5.10 Variablen – Deklaration, Definition und Initialisierung ... 74
5.11 Details zur »include«-Direktive ... 75

5.12 Eingabe und Ausgabe ... 76
5.13 Der Namensraum »std« ... 77
5.14 Aufgaben ... 79

6 Programmiertechnik, 1. Dan: Lesbarer Code 81

6.1 Kommentare ... 81
6.2 Dokumentation ... 82
6.3 Einrückungen und Zeilenlänge ... 83
6.4 Zeilen pro Funktion und Datei ... 84
6.5 Klammern und Leerzeichen ... 84
6.6 Namen ... 86

TEIL II Die Elemente von C++

7 Operatoren 89

7.1 Operatoren und Operanden ... 90
7.2 Überblick über Operatoren ... 90
7.3 Arithmetische Operatoren ... 91
7.4 Bitweise Arithmetik ... 92
7.5 Zuweisungsoperatoren ... 93
7.6 Post- und Präinkrement sowie Post- und Prädekrement ... 94
7.7 Relationale Operatoren ... 95
7.8 Logische Operatoren ... 95
7.8.1 Kurzschluss-Auswertung ... 95
7.8.2 Alternative Token ... 96
7.9 Pointer- und Dereferenzierungsoperatoren ... 97
7.10 Besondere Operatoren ... 97
7.11 Funktionsähnliche Operatoren ... 99
7.12 Operatorreihenfolge ... 100
7.13 Aufgaben ... 101

8 Eingebaute Typen 103

8.1 Eingebaute Datentypen ... 105

8.2 Eingebaute Datentypen initialisieren ... 105

8.3 Ein schneller Überblick ... 106

8.4 Ganzzahlen ... 107

8.4.1 Operationen auf Ganzzahlen ... 108

8.4.2 Arithmetische Typumwandlung ... 109

8.4.3 Ganzzahl-Überlauf ... 110

8.4.4 Ganzzahl-Literale ... 111

8.4.5 Alias-Zahlentypen ... 112

8.4.6 Fließkomma-Literale ... 115

8.4.7 Die Fließkomma-Besonderheiten ... 116

8.4.8 Fließkomma-Interna ... 117

8.5 Wahrheitswerte ... 118

8.6 Zeichentypen ... 119

8.6.1 Internationale Zeichen ... 121

8.6.2 Unicode in C++ ... 122

8.7 Aufgaben ... 122

9 Strings und Streams 124

9.1 Der Zeichenkettentyp »string« ... 124

9.1.1 Initialisierung ... 126

9.1.2 Funktionen und Methoden ... 127

9.1.3 Andere Stringtypen ... 127

9.2 Streams ... 129

9.3 Eingabe- und Ausgabeoperatoren ... 130

9.3.1 »getline« ... 131

9.3.2 Dateien für die Ein- und Ausgabe ... 131

9.3.3 Manipulatoren ... 132

9.3.4 Der Manipulator »endl« ... 133

9.4 Aufgaben ... 134

10 Behälter und Zeiger 135

10.1 Parametrisierte Typen ... 136

10.2 Die einfachen Sequenzcontainer ... 136

10.2.1 »array« ... 137

10.2.2 »vector« ... 139

10.3 Weitere Container ... 141

10.3.1 Sequenzbasierte Container ... 141

10.3.2 Assoziative Container ... 142

10.4 Container-Gemeinsamkeiten ... 144

10.5 Algorithmen ... 145

10.6 Zeiger und C-Arrays ... 146

10.6.1 Zeigertypen ... 146

10.6.2 C-Arrays ... 146

10.7 Aufgaben ... 147

11 Funktionen 148

11.1 Deklaration und Definition einer Funktion ... 149

11.2 Funktionstyp ... 150

11.3 Funktionen verwenden ... 150

11.4 Eine Funktion definieren ... 151

11.5 Mehr zu Parametern ... 152

11.5.1 Call-by-Value ... 153

11.5.2 Call-by-Reference ... 153

11.5.3 Konstante Referenzen ... 154

11.5.4 Aufruf als Wert, Referenz oder konstante Referenz? ... 155

11.6 Funktionskörper ... 156

11.7 Parameter umwandeln ... 158

11.8 Funktionen überladen ... 160

11.9 Default-Parameter ... 162

11.10 Beliebig viele Argumente ... 163
11.11 Alternative Schreibweise zur Funktionsdeklaration ... 164
11.12 Spezialitäten ... 165
11.12.1 noexcept ... 165
11.12.2 constexpr ... 165
11.12.3 Gelöschte Funktionen ... 166
11.12.4 Spezialitäten bei Klassenmethoden ... 166
11.13 Aufgaben ... 167

12 Anweisungen im Detail 169

12.1 Anweisungsblock ... 171
12.1.1 Freistehende Blöcke und Gültigkeit von Variablen ... 172
12.2 Die leere Anweisung ... 174
12.3 Deklarationsanweisung ... 175
12.4 Ausdrucksanweisung ... 176
12.5 Die if-Anweisung ... 176
12.6 »while«-Schleife ... 179
12.7 »do-while«-Schleife ... 180
12.8 »for«-Schleife ... 181
12.9 Die bereichsbasierte »for«-Schleife ... 182
12.10 Die »switch«-Verzweigung ... 183
12.11 »break«-Anweisung ... 187
12.12 Die »continue«-Anweisung ... 188
12.13 Die »return«-Anweisung ... 189
12.14 Die »goto«-Anweisung ... 190
12.15 »try-catch«-Block und »throw« ... 192
12.16 Zusammenfassung ... 193
12.17 Aufgaben ... 193

13 Ausdrücke im Detail 196

13.1 Berechnungen und Seiteneffekte ... 197
13.2 Arten von Ausdrücken ... 198
13.3 Literale ... 199
13.4 Bezeichner ... 200
13.5 Klammern ... 201
13.6 Funktionsaufruf und Index-Zugriff ... 201
13.7 Zuweisung ... 201
13.8 Typumwandlung ... 203
13.9 Aufgaben ... 204

14 Fehlerbehandlung 205

14.1 Fehlerbehandlung mit Rückgabewerten ... 207
14.2 Was ist eine Ausnahme? ... 210
14.2.1 Ausnahmen auslösen und behandeln ... 211
14.2.2 Aufrufstapel abwickeln ... 212
14.3 Kleinere Fehlerbehandlungen ... 213
14.4 Weiterwerfen – »rethrow« ... 213
14.5 Die Reihenfolge im »catch« ... 214
14.5.1 Kein »finally« ... 215
14.5.2 Exceptions der Standardbibliothek ... 215
14.6 Typen für Exceptions ... 216
14.7 Wenn eine Exception aus »main« herausfällt ... 217
14.8 Aufgaben ... 217

15 Programmiertechnik, 2. Dan: Modularisierung 219

15.1 Programm, Bibliothek, Objektdatei ... 219
15.2 Bausteine ... 220
15.3 Trennen der Funktionalitäten ... 221

TEIL III Datenstrukturen

16 Erste eigene Datentypen 225

16.1 Initialisierung ... 226

16.2 Rückgabe eigener Typen ... 227

16.3 Methoden statt Funktionen ... 228

16.4 Das bessere »drucke« ... 231

16.5 Eine Ausgabe wie jede andere ... 232

16.6 Methoden inline definieren ... 233

16.7 Implementierung und Definition trennen ... 234

16.8 Initialisierung per Konstruktor ... 235

16.8.1 Member-Defaultwerte in der Deklaration ... 238

16.8.2 Konstruktor-Delegation ... 238

16.8.3 Default-Werte für die Konstruktor-Parameter ... 239

16.8.4 »init«-Methode nicht im Konstruktor aufrufen ... 240

16.8.5 Exceptions im Konstruktor ... 241

16.9 Struktur oder Klasse? ... 241

16.9.1 Kapselung ... 242

16.9.2 Public und Private, Struktur und Klasse ... 242

16.9.3 Daten mit »struct«, Verhalten mit »class« ... 243

16.9.4 Initialisierung von Typen mit privaten Daten ... 243

16.10 Zusammenfassung ... 245

16.11 Aufgaben ... 245

17 Verwendung eigener Datentypen 248

17.1 Klassen als Werte verwenden ... 251

17.1.1 Wert-Parameter ... 251

17.1.2 Rückgaben ... 252

17.1.3 Performance beim Zurückgeben ... 253

17.2 Konstruktoren nutzen ... 253

17.3 Typumwandlungen ... 254

17.4 Kapseln und entkapseln ... 256
17.4.1 Entkapseln ... 256
17.4.2 Setzen Sie Konvertierungsmethoden sparsam ein ... 257
17.4.3 Totale Kapselung ... 258
17.4.4 Flüssiges Programmieren ... 259
17.4.5 Methoden mit »const« markieren ... 260
17.5 Typen lokal einen Namen geben ... 260
17.6 Typdeduktion mit »auto« ... 263
17.7 Eigene Klassen in Standardcontainern ... 266
17.8 Aufgaben ... 268

18 Namespace und Static 270

18.1 Der Namensraum »std« ... 270
18.2 Anonymer Namensraum ... 274
18.3 »static« macht lokal ... 275
18.4 »static« teilt gern ... 276
18.5 »static« macht dauerhaft ... 279
18.6 Zusammenfassung ... 281
18.7 Aufgaben ... 281

19 Const 284

19.1 Const-Parameter ... 285
19.2 Const-Methoden ... 286
19.3 Const-Variablen ... 288
19.4 Const-Rückgaben ... 289
19.4.1 Const zusammen mit static ... 292
19.4.2 Noch konstanter mit »constexpr« ... 293
19.4.3 »constexpr« als Rückgabe ... 295
19.5 Const-Korrektheit ... 296
19.6 Zusammenfassung ... 297
19.7 Aufgaben ... 298

20 Vererbung 300

20.1 Beziehungen 301
20.1.1 Hat-ein-Komposition 301
20.1.2 Hat-ein-Aggregation 301
20.1.3 Ist-ein-Vererbung 302
20.1.4 Nicht: ist eine instanz-von 302

20.2 Vererbung in C++ 303

20.3 Hat-ein versus ist-ein 304

20.4 Gemeinsamkeiten finden 304

20.5 Abgeleitete Typen erweitern 307

20.6 Methoden überschreiben 308

20.7 Wie Methoden funktionieren 309

20.8 Virtuelle Methoden 310

20.9 Konstruktoren in Klassenhierarchien 312

20.10 Typumwandlung in Klassenhierarchien 314
20.10.1 Die Vererbungshierarchie aufwärts unwandeln 314
20.10.2 Die Vererbungshierarchie abwärts umwandeln 314
20.10.3 Referenzen behalten auch die Typinformation 315

20.11 Wann virtuell? 315

20.12 Andere Designs zur Erweiterbarkeit 317

20.13 Aufgaben 318

21 Der Lebenszyklus von Klassen 321

21.1 Erzeugung und Zerstörung 322

21.2 Temporary: Kurzlebige Werte 324

21.3 Der Destruktor zum Konstruktor 325
21.3.1 Kein Destruktor nötig 327
21.3.2 Ressourcen im Destruktor 328

21.4 Yoda-Bedingung 330

21.5 Konstruktion, Destruktion und Exceptions 331

21.6 Kopieren 333

21.7 Zuweisungsoperator 335

21.8 Streichen von Methoden ... 339
21.9 Verschiebeoperationen ... 340
21.10 Operatoren ... 344
21.11 Eigene Operatoren in einem Datentyp ... 348
21.12 Besondere Klassenformen ... 353
21.12.1 Abstrakte Klassen und Methoden ... 353
21.12.2 Aufzählungsklassen ... 355
21.13 Aufgaben ... 356

22 Programmiertechnik, 3. Dan: Die Nuller-Regel 359

22.1 Die großen Fünf ... 359
22.2 Hilfskonstrukt per Verbot ... 360
22.3 Die Nullerregel und ihr Einsatz ... 361
22.4 Ausnahmen von der Nullerregel ... 362

TEIL IV Fortgeschrittene Themen

23 Zeiger 367

23.1 Adressen ... 368
23.2 Zeiger ... 369
23.3 Heapspeicher und Stapelspeicher ... 372
23.3.1 Der Stapel ... 372
23.3.2 Der Heap ... 374
23.4 Smarte Pointer ... 376
23.4.1 »unique_ptr« ... 378
23.4.2 »shared_ptr« ... 382
23.5 Rohe Zeiger ... 385
23.6 C-Arrays ... 390
23.6.1 Rechnen mit Zeigern ... 391
23.6.2 Verfall von C-Arrays ... 392
23.6.3 Dynamische C-Arrays ... 393
23.6.4 Zeichenkettenliterale ... 394

23.7 Iteratoren ... 396
23.7.1 Mehr Funktionalität mit Iteratoren ... 398
23.7.2 Zeiger als Iteratoren ... 399
23.8 Zeiger im Container ... 400
23.9 Die Ausnahme: Wann das Wegräumen nicht nötig ist ... 400
23.10 Aufgaben ... 402

24 Makros 405

24.1 Der Präprozessor ... 406
24.2 Vorsicht vor fehlenden Klammern ... 410
24.3 Vorsicht vor Mehrfachausführung ... 410
24.4 Typvariabilität von Makros ... 411
24.5 Zusammenfassung ... 413
24.6 Aufgaben ... 414

25 Schnittstelle zu C 416

25.1 Mit Bibliotheken arbeiten ... 417
25.2 C-Header ... 418
25.3 C-Ressourcen ... 421
25.4 »void«-Pointer ... 422
25.5 Daten lesen ... 423
25.6 Das Hauptprogramm ... 424
25.7 Zusammenfassung ... 425
25.8 Aufgaben ... 425

26 Template-Funktionen 427

26.1 Überladung ... 428
26.2 Ein Typ als Parameter ... 429
26.3 Funktionskörper einer Template-Funktion ... 429
26.4 Zahlen als Template-Parameter ... 431
26.5 Viele Funktionen ... 432
26.6 Parameter mit Extras ... 432
26.7 Template-Methoden sind auch nur Funktionen ... 435
26.8 Template-Funktionen in der Standardbibliothek ... 436
26.9 Iteratoren statt Container als Template-Parameter ... 437
26.10 Beispiel: Informationen über Zahlen ... 439
26.11 Aufgaben ... 440

27 Eine Klasse als Funktion 442

27.1 Werte für einen »function«-Parameter ... 443
27.2 C-Funktionspointer ... 444
27.3 Die etwas andere Funktion ... 445
27.4 Praktische Funktoren ... 448
27.5 Algorithmen mit Funktoren ... 450
27.6 Anonyme Funktionen alias Lambda-Ausdrücke ... 451
27.6.1 Lambdas mit Zugriff nach außen ... 452
27.6.2 Für Ihre Bequemlichkeit ... 454
27.7 Aufgaben ... 455

Anhang 457

A C++11-Besonderheiten ... 458
B Operator-Präzedenzen ... 470
C Lösungen ... 472

Index ... 515