

Inhalt

1 Einstieg in die Welt von C++	19
1.1 Der C++-Standard	19
1.2 Die nötigen Werkzeuge für C++	20
1.3 Übersetzen mit g++ und clang++	24
1.4 Übersetzen mit einer Entwicklungsumgebung	26
1.5 Listings zum Buch	30
1.6 Kontrollfragen und Aufgaben im Buch	30
1.7 Aufgabe	30
1.8 Aufgabe zu C++ 23	31
2 Erste Schritte in C++	33
2.1 Das erste Programm in C++	33
2.2 Anweisungen und Ausdrücke	35
2.3 Die Standardeingabe- und -ausgabestreams	37
2.3.1 Die Streams von C++	37
2.3.2 Ausgabe mit »std::cout«	37
2.3.3 Ausgabe mit »std::print«	38
2.3.4 Eingabe mit »std::cin«	39
2.3.5 Ausgabe mit »std::cerr«	40
2.4 Einige Begriffe zu C++	43
2.4.1 Bezeichner	43
2.4.2 Literale	43
2.4.3 Kommentare	44
2.5 Kontrollfragen und Aufgaben	45

Inhalt

3	Die eingebauten C++-Basisdatentypen	46
3.1	Variablen	46
3.2	Definition und Deklaration von Variablen	47
3.3	Initialisierung und Zuweisung von Werten	48
3.4	Ganzzahltypen	51
3.4.1	Literale von Ganzzahltypen	56
3.5	Ganzzahldatentyp für Zeichen	58
3.5.1	Weitere Datentypen für Zeichen	64
3.5.2	Die Unicode-Typen »char8_t«, »char16_t« und »char32_t«	66
3.6	Fließkommazahlentypen	68
3.6.1	Literale von Fließkommazahlen	70
3.7	Der »auto«-Typ	71
3.8	Konstanten	72
3.9	Die Byte-Größe mit dem »sizeof«-Operator	73
3.10	Limits für die Basisdatentypen	74
3.11	Kontrollfragen und Aufgaben	77
4	Arbeiten mit den eingebauten Typen	78
4.1	Arithmetische Operatoren	78
4.1.1	Kurzschreibweise arithmetischer Operatoren	83
4.1.2	Inkrement- und Dekrementoperator	83
4.2	Ungenaue Fließkommazahlen	86
4.3	Typumwandlung	88
4.3.1	Implizite Umwandlung durch den Compiler	88

4.3.2	Die automatische Typumwandlung beschränken	91
4.3.3	Explizite Typumwandlung	92
4.4	Formatierte Ausgabe von Werten	94
4.5	Kontrollfragen und Aufgaben	95
5	Kontrollstrukturen	97
5.1	Der eingebaute Datentyp »bool«	97
5.2	Vergleichsoperatoren	98
5.3	Bedingte Anweisung mit »if«	101
5.4	Anweisungsblock für Kontrollstrukturen	102
5.5	Alternative »else«-Verzweigung	103
5.6	Bedingte Anweisung mit Initialisierung	105
5.7	Mehrfache Verzweigung	105
5.8	Der Bedingungsoperator »?:«	108
5.9	Logische Operatoren	109
5.10	Die Fallunterscheidung – »switch«	111
5.11	Die kopfgesteuerte »while«-Schleife	114
5.12	Die fußgesteuerte »do-while«-Schleife	116
5.13	Die Zählschleife »for«	117
5.14	Kontrollierte Sprunganweisungen	121
5.14.1	Die »break«-Anweisung	121
5.14.2	Die »continue«-Anweisung	122
5.15	Kontrollfragen und Aufgaben	123

Inhalt

6	Arrays und Strings	125
6.1	Arrays	125
6.1.1	Der C++-Container »std::vector«	127
6.1.2	Der C++-Container »std::array«	135
6.1.3	C-Arrays	136
6.2	Strings in C++	139
6.2.1	Der C++-Container »std::string«	140
6.2.2	Unterstützung von Unicode	141
6.2.3	C-Zeichenketten	142
6.2.4	Zeichenkettenliterale	144
6.3	Kontrollfragen und Aufgaben	145
7	Referenzen und Zeiger	146
7.1	Referenzen	146
7.2	Zeiger	149
7.2.1	Die Syntax von Zeigern	149
7.2.2	Zeiger dereferenzieren	152
7.2.3	Der Zeiger »nullptr«	153
7.2.4	Zeiger prüfen	154
7.2.5	Die Adresse einer Referenz	155
7.2.6	Referenzen vs. Zeiger: Wann verwenden Sie was?	156
7.2.7	Verwendung von Zeigern und Alternativen	157
7.3	Kontrollfragen und Aufgaben	158

8	Funktionen	160
8.1	Grundlegendes zu Funktionen	160
8.1.1	Funktionen definieren und aufrufen	160
8.1.2	Funktionen deklarieren	162
8.1.3	Funktionsparameter (Call-by-Value)	164
8.1.4	Konstante Funktionsparameter	166
8.1.5	Standardparameter	167
8.1.6	Rückgabe aus Funktionen	169
8.1.7	Funktionen überladen	172
8.1.8	Gültigkeitsbereich und Sichtbarkeit von Variablen	175
8.1.9	Die »main()«-Funktion	178
8.1.10	Aufruf eines Programms mit Parametern	178
8.1.11	Das Programmende	179
8.2	Referenzen als Parameter und Rückgabe	180
8.2.1	Referenzen als Parameter	182
8.2.2	Konstante Funktionsparameter	183
8.2.3	Referenzen als Rückgabe	185
8.3	Zeiger als Parameter und Rückgabewert	187
8.3.1	Referenzen vs. Zeiger als Parameter	188
8.4	Übergabe großer Elemente als Funktionsparameter	188
8.5	C-Arrays oder C-Strings als Funktionsparameter	190
8.6	Kontrollfragen und Aufgaben	191
9	Modularisierung und Präprozessor	193
9.1	Präprozessor-Direktiven	193
9.1.1	Die »#include«-Direktive	194
9.1.2	Module und »import std;«	195

Inhalt

9.1.3	Die »#define«-Direktive	196
9.1.4	Bedingte Kompilierung	197
9.2	Modulare Programmierung	199
9.2.1	Aufteilung	199
9.2.2	Die öffentliche Schnittstelle (Headerdatei)	200
9.2.3	Die nicht öffentliche(n) Datei(en)	202
9.2.4	Die Client-Datei	203
9.2.5	Aufgabe	203
9.2.6	Wenn nur Objektcode oder eine Bibliothek vorhanden ist	204
9.3	Namensräume	205
9.3.1	Einen Namensraum deklarieren und verwenden	206
9.3.2	Namensräume verschachteln	209
9.3.3	Ein Namensraum ist ein eigener Gültigkeitsbereich	209
9.3.4	Einen Namensraum mit »using« importieren	212
9.3.5	Einzelne Bezeichner mit »using« importieren	213
9.3.6	Alias für Namensräume	214
9.3.7	Anonymer Namensraum	214
9.3.8	Der Namensraum »std«	215
9.4	Spezifizierer und Qualifikatoren	217
9.4.1	Das Schlüsselwort »static«	217
9.4.2	Das Schlüsselwort »extern«	219
9.4.3	Das Schlüsselwort »constexpr«	220
9.4.4	Das Schlüsselwort »const«	221
9.4.5	Das Schlüsselwort »inline«	222
9.5	Kontrollfragen und Aufgaben	223

10 Strukturen, Aufzählungen und dynamische Speicherobjekte	226
10.1 Erste eigene Datentypen mit Strukturen	226
10.1.1 Strukturen definieren, Elemente erzeugen und initialisieren	227
10.1.2 Zugriff auf die Strukturelemente	230
10.1.3 Zugriff auf die Elemente in einer Funktion	231
10.1.4 Strukturen in einem Vektor oder Array	232
10.1.5 Methoden statt Funktionen	233
10.1.6 Strukturen vergleichen	235
10.2 Der Aufzählungstyp »enum«	235
10.3 Eigene Namen mit »using«	237
10.4 Dynamische Speicherobjekte	238
10.4.1 Dynamisch Objekte mit »new« anlegen und mit »delete« freigeben	240
10.4.2 Dynamisch Arrays mit »new[]« anlegen und mit »delete[]« freigeben	243
10.4.3 Der smarte »unique_ptr«-Pointer	246
10.5 Kontrollfragen und Aufgaben	249
11 Klassen	250
11.1 Klassen	250
11.1.1 Klassendefinition	250
11.1.2 Zugriffskontrolle mit »public« und »private«	252
11.1.3 Methoden definieren	256
11.1.4 Objekte erzeugen und benutzen	259
11.2 Konstruktoren	264
11.2.1 Konstruktoren deklarieren	266

Inhalt

11.2.2 Konstruktoren definieren	266
11.2.3 Konstruktoren delegieren	269
11.2.4 Der Standardkonstruktor (Default-Konstruktor)	272
11.2.5 Implizite Konvertierungen und wie Sie sie verhindern – »explicit«	273
11.2.6 Der Kopierkonstruktor (Copy-Konstruktor)	276
11.2.7 Der Verschiebekonstruktor (Move-Konstruktor)	278
11.3 Destruktoren	281
11.3.1 Die Lebensdauer eines Objekts	281
11.3.2 Wann ist ein Destruktor erforderlich?	282
11.3.3 Einen Destruktor deklarieren	283
11.3.4 Destruktor definieren	284
11.4 Weitere Formen von Methoden	286
11.4.1 »inline«-Methoden	287
11.4.2 Konstante Methoden (»nur-lesend«)	290
11.4.3 Konstante Methoden explizit ausschließen	292
11.4.4 »this«-Zeiger	293
11.5 Kontrollfragen und Aufgaben	295
12 Objekte und Klassenelemente	297
<hr/>	
12.1 Objekt als Parameter	297
12.1.1 Objekte an eine Funktion übergeben	297
12.1.2 Objekte an eine Methode übergeben	300
12.2 Freundfunktionen (»friend«)	302
12.3 Objekte einer Klasse als Rückgabewerte	304
12.3.1 Referenzen auf eine Klasse als Rückgabewerte	307
12.4 Arrays von Objekten	310
12.5 Dynamische Objekte	311

12.6 Klassenobjekte als Klassenattribute	313
12.7 Eine Containerklasse als Klassenattribut	318
12.8 Smart Pointer als Klassenattribut	321
12.9 Statische und konstante Klassenelemente	325
12.9.1 Statische Klassenelemente	326
12.9.2 Konstante Klassenelemente	330
12.9.3 Rohe Zeiger als Klassenelemente oder direkt die Nullregel	332
12.10 Die Nullregel (Rule of Zero)	333
12.10.1 Die großen Fünf	334
12.11 Kontrollfragen und Aufgaben	336
13 Operatoren überladen	338
<hr/>	
13.1 Das Schlüsselwort »operator«	340
13.2 Zweistellige (arithmetische) Operatoren überladen	341
13.2.1 Operatorüberladung als Methode einer Klasse	343
13.2.2 Operatorüberladung als globale Hilfsfunktion	346
13.3 Einstellige Operatoren überladen	348
13.4 Den Zuweisungsoperator überladen	352
13.4.1 Die Zuweisung von Operatoren unterbinden	354
13.4.2 Die Verschiebezuweisung	355
13.5 Ausgabe- und Eingabeoperatoren überladen	357
13.5.1 Den Ausgabeoperator << überladen	358
13.5.2 Den Eingabeoperator >> überladen	359
13.6 Vergleichsoperatoren	361
13.6.1 Der Drei-Wege-Vergleichsoperator	362
13.7 Weitere Operatorüberladungen	364

Inhalt

13.8 Konvertierungsoperatoren	364
13.8.1 Der Konvertierungskonstruktor	365
13.8.2 Die Konvertierungsfunktion	366
13.9 Kontrollfragen und Aufgaben	368
14 Vererbung (Abgeleitete Klassen)	370
<hr/>	
14.1 Die Vorbereitung	371
14.2 Das Ableiten einer Klasse	373
14.2.1 Erben und erweitern	374
14.2.2 »public«-Zugriffsrechte einer abgeleiteten Klasse	375
14.2.3 Methoden überschreiben	377
14.2.4 Konstruktoren	379
14.2.5 Programmbeispiel	380
14.2.6 Konstruktoren vererben	381
14.2.7 Destruktor	383
14.2.8 Mehrfachvererbung	383
14.2.9 Die Zugriffsspezifikation »protected«	384
14.2.10 Implizite Typumwandlung abgeleiteter Klassen	385
14.2.11 Überschreiben mit virtuellen Methoden	387
14.2.12 Vererbung und Überschreiben verhindern	390
14.2.13 Abstrakte Klassen und rein virtuelle Methoden	391
14.3 Kontrollfragen und Aufgaben	393
15 Templates	395
<hr/>	
15.1 Funktionstemplates	395
15.1.1 Funktionstemplates definieren	396
15.1.2 Funktionstemplates über mehrere Module	399

15.1.3	Ein Funktionstemplate spezialisieren	399
15.1.4	Templates mit verschiedenen Parametern	401
15.1.5	Explizite Template-Argumente	402
15.1.6	Einschränkungen mit Concepts	403
15.1.7	Wiederverwendbare Templates	404
15.2	Klassentemplates	405
15.2.1	Klassentemplate definieren	406
15.2.2	Template-Methoden definieren	406
15.2.3	Template-Methoden spezialisieren	407
15.2.4	Klassentemplate instanziieren	408
15.2.5	Klassentemplates mit mehreren formalen Parametern	410
15.3	Kontrollfragen und Aufgaben	411
16	Ausnahmebehandlung (Fehlerbehandlung)	413
16.1	Grundlagen der Fehlerbehandlung	413
16.2	Eine Ausnahme auslösen	415
16.3	Eine Ausnahme auffangen und behandeln	416
16.3.1	Ausnahmen aus der Standardbibliothek	418
16.3.2	Alternatives »catch (...)«	421
16.3.3	Eine Ausnahme mit »throw« weiterwerfen	422
16.3.4	Das Schlüsselwort »noexcept«	424
16.3.5	Stack-Abwicklung	425
16.4	Standardausnahmen (Fehlerklassen)	426
16.4.1	Die Ausnahme »std::invalid_argument«	427
16.4.2	Die Ausnahme »std::out_of_range«	427
16.4.3	Die Ausnahme »ios_base::failure«	428
16.4.4	Systemausnahmen	430

Inhalt

16.5 Die Alternative »std::expected«	431
16.6 Vermeidung von Fehlern mit [[nodiscard]]	433
16.7 Fehlerdiagnose mit »std::source_location«	434
16.8 Kontrollfragen	435

17 Ein-/Ausgabestreams für Dateien	436
---	-----

17.1 Der Umgang mit Dateien in C++	436
17.2 Verschiedene Streams für Dateien	436
17.3 Eine Datei öffnen und schließen	437
17.3.1 Prüfung auf Fehler	438
17.3.2 Schließen von Filestreams	439
17.3.3 Verschiedene Modi zum Öffnen von Dateien	440
17.3.4 Byteweise lesen und schreiben	443
17.3.5 Zeilenweise lesen und schreiben	444
17.3.6 Blockweise lesen und schreiben	446
17.3.7 Die Lese- oder Schreibposition ändern	447
17.4 Mit »std::filesystem« arbeiten	449
17.5 Kontrollfragen und Aufgaben	450

18 Die Standardbibliothek und weitere Sprachelemente	452
---	-----

18.1 Die Container der Standardbibliothek	452
18.1.1 Sequence Container und Iteratoren	453
18.1.2 Assoziative Container	459
18.1.3 Zusammenfassende Übersicht der Container	461

18.2 Algorithmen der Standardbibliothek	465
18.3 Fortgeschrittene Sprachelemente	468
18.3.1 Verbundtypen der Standardbibliothek	468
18.3.2 Mit »std::ranges« Container als Ganzes behandeln	472
18.4 Fortgeschrittene Funktionskonzepte	473
18.4.1 Typermittlung mit »decltype«	474
18.4.2 Rückgabesyntax mit nachlaufendem Rückgabetyp	475
18.4.3 Automatische Ermittlung des Rückgabetyps	476
18.4.4 Funktionsobjekte (Funktor)	478
18.4.5 Lambda-Funktionen	479
18.5 Smart Pointer	483
18.5.1 »unique_ptr«	484
18.5.2 »shared_ptr«	487
18.5.3 »weak_ptr«	489
18.5.4 Die Move-Semantik	492
18.6 Textansichten und Formatierung	494
18.6.1 Effiziente Textansichten mit »std::string_view«	494
18.6.2 Mehr zu »std::format«	495
18.6.3 Eigene Datentypen formatieren mit »std::formatter«	497
18.7 Die Zeitbibliothek	501
18.7.1 Zeitdauer (Duration)	502
18.7.2 Vorhandene Zeitgeber	503
18.7.3 Die »ratio«-Bibliothek	506
18.8 Ausblick auf das Multithreading	507

Inhalt

Anhang	509
<hr/>	
A Lösungen der Übungsaufgaben	509
A.1 Lösungen zu Kapitel 2	509
A.2 Lösungen zu Kapitel 3	510
A.3 Lösungen zu Kapitel 4	510
A.4 Lösungen zu Kapitel 5	512
A.5 Lösungen zu Kapitel 6	513
A.6 Lösungen zu Kapitel 7	514
A.7 Lösungen zu Kapitel 8	515
A.8 Lösungen zu Kapitel 9	517
A.9 Lösungen zu Kapitel 10	519
A.10 Lösungen zu Kapitel 11	522
A.11 Lösungen zu Kapitel 12	524
A.12 Lösungen zu Kapitel 13	525
A.13 Lösungen zu Kapitel 14	527
A.14 Lösungen zu Kapitel 15	529
A.15 Lösungen zu Kapitel 16	530
A.16 Lösungen zu Kapitel 17	530
Index	533