

Übersicht

Vorwort	19
1 Grundlagen	21
2 Anweisungsarten	51
3 Funktionen	77
4 Arrays	95
5 Strings	111
6 Klassen – Einführung	121
7 Dynamische Speicherverwaltung	143
8 Klassen – Vertiefung I	165
9 Klassen – Vertiefung II	197
10 Operatoren überladen	217
11 Namensbereiche	243
12 Templates	251
13 STL	257
14 Praxis	277
15 Vererbung – Die Technik	293
16 Vererbung – Das Design	327
17 Ausnahmen	351
18 Praxis	377
19 Vererbung – Vertiefung	401
20 MaoMao	409
21 Schach	461
Literaturverzeichnis	513
Index	515

Inhalt

Vorwort	19
1 Grundlagen	21
1.1 Am Anfang war das Problem	21
1.1.1 Vom Problem zur Lösung	22
1.1.2 Von der Lösung zum Algorithmus	22
1.1.3 Vom Algorithmus zum Programm	23
1.2 Das kleinste C++-Programm	24
1.3 Die Ausgabe	24
1.3.1 cout	25
1.3.2 include	26
1.3.3 Namensbereiche	28
1.4 Variablen	29
1.4.1 Initialisierung	31
1.4.2 Konstanten	31
1.5 Eingabe	32
1.6 Grundrechenarten	33
1.6.1 Vorsicht bei unterschiedlichen Datentypen	34
1.6.2 Typchen wechsele dich	36
1.6.3 Kombinierte Zuweisungsoperatoren	37
1.6.4 Inkrement und Dekrement	37
1.7 Kommentare	38
1.8 Ans Eingemachte	40
1.8.1 Die elementaren Datentypen	40
1.8.2 Literale	42
1.8.3 Bitweise Operatoren	45
1.8.4 Speicherdauer	48
2 Anweisungsarten	51
2.1 Zusammengesetzte Anweisungen	51
2.2 Bedingungen	52
2.2.1 Relationale Operatoren und Vergleichsoperatoren	52

Inhalt

2.3	Verzweigungen	54
2.3.1	else	55
2.3.2	Logische Operatoren	56
2.4	Der ?:-Operator	59
2.5	Die Fallunterscheidung	60
2.6	Schleifen	63
2.6.1	while	63
2.6.2	do-while	64
2.6.3	for	65
2.7	Kontrollbefehle	68
2.7.1	break	68
2.7.2	continue	69
2.8	Ans Eingemachte	70
2.8.1	Kurzschlusseigenschaft	70
2.8.2	Deklaration und Definition	71
2.8.3	static_assert	72
2.8.4	assert	73
3	Funktionen	77
3.1	Funktionsdefinition	77
3.2	return	79
3.3	Standardwerte	80
3.4	Alternative Funktionssyntax	81
3.4.1	auto	82
3.5	Funktionsdeklaration	82
3.6	Module	83
3.7	Funktionen überladen	84
3.7.1	Unterscheidung in der Parameteranzahl	84
3.7.2	inline	85
3.7.3	Unterscheidung im Parametertyp	86
3.8	Funktions-Templates	86
3.8.1	Mehrere variable Datentypen	87
3.8.2	Variabler Rückgabetyyp	88
3.9	Ans Eingemachte	89
3.9.1	Die Präprozessor-Direktiven	89
3.9.2	Die Wertebereiche der elementaren Datentypen	93

4	Arrays	95
4.1	Arrays definieren	95
4.1.1	sizeof	96
4.2	Arbeiten mit Arrays	96
4.2.1	Initialisierung	96
4.3	Arrays als Funktionsparameter I	97
4.4	Zeiger	98
4.4.1	Sedezimalsystem	98
4.4.2	Der Adressoperator	99
4.4.3	Definition eines Zeigers	100
4.4.4	Dereferenzierung	100
4.4.5	Zeiger als Funktionsparameter	101
4.4.6	Zeiger auf Zeiger	103
4.4.7	Arrays als Funktionsparameter II	104
4.4.8	Zeigerarithmetik	105
4.5	Referenzen	106
4.6	Notation und Bezeichner-Namen	108
4.7	Ans Eingemachte	109
4.7.1	cmath	109
5	Strings	111
5.1	C-Strings	111
5.2	Strings	113
5.2.1	Tastatureingabe von Strings	113
5.3	Ans Eingemachte	115
5.3.1	cstddef	115
5.3.2	cctype	115
5.3.3	cstring	117
6	Klassen – Einführung	121
6.1	Objektorientierte Programmierung	121
6.1.1	Objekte als abgrenzbare Einheiten	121
6.1.2	Nicht objektorientierte Objekte	122
6.2	Klassen als Bauplan	123
6.2.1	Definition	123
6.3	Zugriffsrechte	124

Inhalt

6.4	Konstrukturen	125
6.4.1	Standardkonstruktor	126
6.4.2	Einheitliche Initialisierung	127
6.4.3	Der Destruktor	127
6.5	Methoden	128
6.5.1	Zugriffsmethoden	128
6.5.2	Konstanz wahrende Methoden	129
6.6	Externe Definition	130
6.7	Mehrfachdefinition	132
6.8	Objekte als Funktionsparameter	133
6.8.1	Referenzen auf Objekte	134
6.8.2	Zeiger auf Objekte	134
6.8.3	Objekte als Methodenparameter	136
6.9	Ans Eingemachte	137
6.9.1	Standardwerte für Attribute	137
6.9.2	Verschachtelte Klassendefinitionen	137
6.9.3	Typ-Definitionen	139
6.9.4	cv-Qualifizierung	142
7	Dynamische Speicherverwaltung	143
7.1	Zeiger	143
7.1.1	Zeiger und Konstanten	144
7.1.2	Zeiger auf Funktionen	145
7.1.3	Zeiger auf Klassenelemente	146
7.2	Referenzen	148
7.3	»new« und »delete«	149
7.3.1	Ein Beispiel	150
7.4	Smart-Pointer	152
7.4.1	Unique Pointer	152
7.4.2	Shared Pointer	154
7.4.3	Weak Pointer	156
7.4.4	Auto-Pointer	159
7.4.5	Auto-Pointer und Arrays	160
7.5	Ans Eingemachte	160
7.5.1	Rohspeicher	160
7.5.2	Allokatoren	161
7.5.3	Probleme mit »new«	162

8	Klassen – Vertiefung I	165
8.1	Reihenfolge der Zugriffsrechte	165
8.2	Konstruktoren	167
8.2.1	Standardkonstruktor	169
8.2.2	Kopierkonstruktor	169
8.2.3	Element-Initialisierungsliste	173
8.2.4	Verschiebekonstruktor	176
8.2.5	Implizite Typumwandlung	179
8.2.6	Triviale Konstruktoren	179
8.2.7	Konstruktor-Delegation	180
8.3	Destruktoren	181
8.3.1	Triviale Destruktoren	181
8.4	Konstante Objekte und Elemente	181
8.4.1	Implizite Objektparameter	181
8.4.2	const-resistente Variablen mit »mutable«	184
8.5	Ans Eingemachte	190
8.5.1	Implizite Klasselemente	190
8.5.2	Funktionen als »deleted« oder »default« definieren	194
9	Klassen – Vertiefung II	197
9.1	Statische Elemente	197
9.1.1	Statische Methoden	197
9.1.2	Statische Attribute	199
9.1.3	Statische Variablen	204
9.2	Konstruktoren und ihre Anwendung	205
9.2.1	Funktionsaufruf aus Konstruktoren heraus	205
9.2.2	Unvollendet konstruierte Objekte	207
9.3	Der »this«-Zeiger	210
9.4	Ans Eingemachte	211
9.4.1	Einzige Objekte	211
9.4.2	Abbau von Singleton-Objekten	213
10	Operatoren überladen	217
10.1	Zuweisungsoperatoren	217
10.1.1	Kopierzweisungsoperator	217
10.1.2	Verschiebezweisungsoperator	219
10.1.3	Kombinierte Zuweisungsoperatoren	219

Inhalt

10.2	Rechenoperatoren	220
10.2.1	Operation als Methode	221
10.2.2	Operation als Funktion	221
10.2.3	Methode oder Funktion?	223
10.2.4	Operatoren mit Verschiebe-Semantik	224
10.2.5	Standardverhalten nachbilden	227
10.3	Vergleichsoperatoren	229
10.3.1	Operator-Templates	230
10.4	Die Operatoren << und >>	231
10.4.1	operator<<	231
10.4.2	operator>>	231
10.5	Der Operator []	232
10.6	Der Operator ()	233
10.7	Die Operatoren -> und *	234
10.8	Umwandlungsoperatoren	237
10.9	Die Operatoren ++ und --	237
10.9.1	Prä-Operatoren	239
10.9.2	Post-Operatoren	240
10.9.3	Weitere Operatoren	241
10.10	Ans Eingemachte	241
10.10.1	Probleme mit Operatoren	241
11	Namensbereiche	243
11.1	Deklarative Bereiche, potenzielle und tatsächliche Bezugsrahmen	243
11.2	Namensbereiche definieren	246
11.3	Die Using-Direktive	247
11.4	Ein Alias für Namensbereiche	248
11.5	Unbenannte Namensbereiche	248
11.6	Die Using-Deklaration	249
12	Templates	251
12.1	Klassen-Templates	251
12.2	Funktions-Templates	253
12.3	Template-Parameter	254
12.3.1	Standardargumente	254
12.4	Template-Spezialisierung	255
12.5	typename	256

13 STL	257
13.1 Die Komponenten der STL	257
13.1.1 Container	258
13.1.2 Iteratoren	258
13.1.3 Iteratoren erzeugen	263
13.1.4 Algorithmen	263
13.2 Die STL im Einsatz	267
13.2.1 Element suchen	267
13.2.2 Element suchen mit eigener Bedingung	269
13.2.3 Elemente löschen	269
13.2.4 Elemente kopieren	270
13.2.5 Elemente sortieren	271
14 Praxis	277
14.1 Ein Perl-Array	277
14.1.1 Konstruktoren	278
14.1.2 Zuweisungsoperatoren und size	278
14.1.3 Indexoperator	279
14.1.4 at	279
14.2 Ein nichtsensitiver String	280
14.2.1 Klassendefinition	280
14.2.2 »get_org« und »get_low«	281
14.2.3 »set«	281
14.2.4 Konstruktoren	281
14.2.5 »to_lower«	282
14.2.6 Verbesserung (1. Schritt)	282
14.2.7 Kopierzuweisungsoperator	284
14.2.8 Verbesserung (2. Schritt)	284
14.2.9 Additionsoperatoren	285
14.2.10 Verbesserung (3. Schritt)	285
14.2.11 Vergleichsoperatoren	286
14.2.12 Ausgabe-Operator	286
14.2.13 Indexoperator	287
14.2.14 Die Klasse »CChar«	287
14.2.15 Verbesserung (4. Schritt)	289
14.2.16 Indexoperator für konstante Objekte	289
14.2.17 Verbesserung (5. Schritt)	290

15 Vererbung – Die Technik	293
15.1 Das Klassendiagramm der UML	293
15.2 Vererbung in C++	295
15.3 Die Vererbungssyntax	297
15.4 Geschützte Elemente	301
15.4.1 Zugriff auf Basisklassen-Elemente	303
15.5 Polymorphie	304
15.6 Verdecken von Methoden	305
15.7 Überschreiben von Methoden	307
15.8 Virtuelle Methoden	309
15.8.1 Virtuelle Methoden und Konstruktoren	311
15.8.2 Downcasts	313
15.8.3 Virtuelle Destruktoren	314
15.9 Rein virtuelle Methoden	316
15.9.1 Rein virtuelle Methoden mit Implementierung	316
15.9.2 Rein virtuelle Destruktoren	317
15.10 Vererbung und Arrays	318
15.11 Vererbung und Standardwerte	319
15.12 Vererbung und überladene Operatoren	320
15.13 Versiegelte Elemente	321
15.13.1 Versiegelte Klasse	321
15.13.2 Versiegelte Methode	322
15.13.3 Warum Elemente versiegeln?	322
15.14 Geerbte Konstruktoren verwenden	323
15.15 Überschreibungshilfe	324
16 Vererbung – Das Design	327
16.1 Beziehungen	327
16.1.1 »ist ein«	327
16.1.2 »ist implementiert mit«	332
16.1.3 »hat ein«	335
16.2 Was wird vererbt?	336
16.2.1 Schnittstelle mit verbindlicher Implementierung	336
16.2.2 Schnittstelle mit überschreibbarer Implementierung	338
16.2.3 Schnittstelle	339
16.2.4 Implementierung	340
16.3 Das Offen-Geschlossen-Prinzip	341
16.4 Operationen oben, Daten unten	345

16.5	Das Umkehrung-der-Abhängigkeit-Prinzip	346
16.6	Das Einzelne-Verantwortung-Prinzip	349
17	Ausnahmen	351
17.1	Warum Ausnahmen?	351
17.2	Ausnahmen in C++	354
17.3	Vordefinierte Ausnahmen	356
17.3.1	Der Header »exception«	356
17.3.2	Der Header »typeinfo«	357
17.3.3	Der Header »memory«	357
17.3.4	Der Header »new«	358
17.3.5	Der Header »stdexcept«	358
17.4	Ausnahmen im Detail	358
17.4.1	terminate	359
17.4.2	Das Verlassen eines Try-Blocks	359
17.4.3	uncaught_exception	360
17.4.4	Das Werfen einer Ausnahme	361
17.4.5	Das Fangen einer Ausnahme	362
17.5	Ausnahme-Spezifikationen	364
17.5.1	Ausnahme-Spezifikationen und Zeiger	365
17.5.2	Virtuelle Methoden mit Ausnahme-Spezifikation	366
17.5.3	unexpected	366
17.5.4	Ausnahme-Spezifikationen in der Praxis	368
17.6	Ausnahmen und Konstruktoren	369
17.7	Ausnahmen und Destruktoren	371
17.8	Ausnahmen und dynamische Speicherverwaltung	371
17.9	Ressourcen-Erwerb ist Initialisierung	373
17.10	Funktions-Try-Blöcke	374
17.11	Ausnahmensicherheit	376
18	Praxis	377
18.1	Ein ausnahmensicherer insensitiver String	377
18.1.1	Der Standardkonstruktor	378
18.1.2	Der Kopierkonstruktor	378
18.1.3	Der Konstruktor für Strings	379
18.1.4	Der Kopierzuweisungsoperator	379
18.1.5	Der Additionszuweisungsoperator	381
18.1.6	Der Additionsoperator	382
18.1.7	Verschiebe-Semantik	382

Inhalt

18.2	Eine verbreitete Ringpuffer-Implementierung	383
18.2.1	Die Methode »vergroessern«	388
18.2.2	Der Kopierkonstruktor	389
18.2.3	Der Kopierzuweisungsoperator	389
18.2.4	»out«	390
18.2.5	Ausnahmensicherheit	390
18.2.6	Anforderungen an den verwalteten Typ	390
18.2.7	Granularität	391
18.3	Ein besserer Ringpuffer	391
18.3.1	Grundgerüst	392
18.3.2	Expliziter Konstruktor	392
18.3.3	Destruktor	393
18.3.4	Ringpuffer	395
18.3.5	Standardkonstruktor	395
18.3.6	Kopierkonstruktor	395
18.3.7	Kopierzuweisungsoperator	396
18.3.8	front	397
18.3.9	back	397
18.3.10	out_front	397
18.3.11	out_back	398
18.3.12	in_back	398
18.3.13	Ausnahmensicherheit	399
18.3.14	Anforderungen an den verwalteten Typ	399
18.3.15	Granularität	399
18.3.16	Fazit	399
19	Vererbung – Vertiefung	401
19.1	Gemeinsame Basisklassen	401
19.2	Virtuelle Basisklassen	404
19.3	Einsatz von Mehrfachvererbung	407
20	MaoMao	409
20.1	Die Fenster	410
20.1.1	Die Klasse »ITxtFenster«	410
20.1.2	Die Klasse »DosFenster«	411
20.1.3	Die Klasse »NetFenster«	412
20.2	Die Spielkarte	413
20.2.1	Die Klasse »Farbe«	415
20.2.2	Die Klassen »FarbeDe« und »FarbeEn«	421

20.3	Das Kartenspiel.	423
20.3.1	Die Schnittstelle »IKartenspiel«	423
20.3.2	Die Klasse »Kartenspiel«	424
20.3.3	Die abstrakte Fabrik	428
20.3.4	Die Schnittstelle »IKartenfabrik«	429
20.3.5	Die Klassen »KartenfabrikDe« und »KartenfabrikEn« ...	430
20.3.6	Die Klasse »KartenspielMM«	430
20.4	Die Spieler.	432
20.4.1	Die abstrakte Textfabrik	433
20.4.2	Die Klasse »MenschSpielerMM«	436
20.4.3	Die Klasse »ComputerSpielerMM«	442
20.5	Das MaoMao-Spiel.	443
20.5.1	Der Spielablauf prozedural.	447
20.5.2	Das Zustand-Muster	450
20.5.3	Der Spielablauf objektorientiert.	451
21	Schach.	461
21.1	Anforderungen	461
21.2	Die Farben.	462
21.2.1	Die Klasse »IFarbe«	462
21.2.2	Die Klasse »FarbeWeiss«	462
21.2.3	Die Klasse »Farbverwalter«	464
21.2.4	Die Klasse »ITeamschema«	467
21.2.5	Die Klasse »Teamschema«	468
21.2.6	Die Klasse »Teamverwalter«	469
21.3	Die Spielbretter.	472
21.3.1	Die Klasse »ISchachbrett«	473
21.3.2	Die Klasse »Schachbrett«	474
21.3.3	Die Klasse »SchachbrettRechteckig«	479
21.4	Die Figuren	480
21.4.1	Die Klasse »IFigur«	481
21.4.2	Die Klasse »Koordinaten«	482
21.4.3	Die Klasse »Figur«	483
21.4.4	Die Klasse »FigurSpringer«	488
21.4.5	Die Klasse »FigurDame«	490
21.5	Die Problemlösungen.	493
21.5.1	Ein Brett mit Ausgabe.	493
21.5.2	Das Springer-Problem	501
21.5.3	Das Dame-Problem.	503

Inhalt

21.6	Solitaire	505
21.6.1	Die Klasse »Solitairebrett«	506
21.6.2	Die Klasse »FigurSolitaire«	508
21.6.3	Die Lösung des Spiels	510
	Literaturverzeichnis	513
	Index	515