

Inhaltsverzeichnis

1	Vorbemerkungen	1
1.1	namespace std	1
1.2	Header-Dateien	2
1.3	Eigenschaften	2
1.4	Kanonische Klassen	4
1.5	Komplexität und Aufwand.....	5
2	Konzeption und Entwicklung der STL	7
2.1	Eine Feldklasse	7
2.2	Eine Listenklasse	8
2.3	Folgerungen.....	9
2.4	Standardisierung der Suchfunktion.....	9
2.5	Die Schnittstelle eines Iterators.....	11
2.6	Ein Iterator für die Feldklasse	12
2.7	Ein Iterator für die Listenklasse.....	13
2.8	Zusammenfassung	13
2.9	const-Korrektheit.....	14
2.10	Flexible Suche mit Funktionsobjekten	16
2.11	Kleinste Bausteine	18
2.12	Programmübersetzungs- und -laufzeit	20
2.13	Der Rückgabetyt für eine Zählfunktion	22
2.14	Fehlerquellen	25
2.15	Ist die STL objektorientiert?	27
2.16	Einsatz der Standardbibliothek	29
2.17	Aufgaben.....	30
3	Funktionsobjekte	33
3.1	Basisklassen für Funktionsobjekte	33
3.2	Arithmetische, logische und Vergleichsoperationen	35
3.3	Projektionen	38
3.3.1	binder1st	38
3.3.2	bind1st	39
3.3.3	binder2nd	39
3.3.4	bind2nd	40
3.3.5	Beispiel.....	40
3.4	Negativierer.....	41
3.4.1	unary_negate	41
3.4.2	not1	42
3.4.3	binary_negate	42
3.4.4	not2	42
3.5	Adapter für Funktionszeiger	43
3.5.1	pointer_to_unary_function	44

3.5.2	ptr_fun für einstellige Funktionen.....	44
3.5.3	pointer_to_binary_function.....	44
3.5.4	ptr_fun für zweistellige Funktionen	45
3.6	Adapter für Zeiger auf Elementfunktionen.....	45
3.6.1	mem_fun_ref_t und const_mem_fun_ref_t	47
3.6.2	mem_fun_ref für Elementfunktionen ohne Argument.....	47
3.6.3	mem_fun1_ref_t und const_mem_fun1_ref_t.....	48
3.6.4	mem_fun_ref für Elementfunktionen mit Argument.....	49
3.6.5	mem_fun_t und const_mem_fun_t.....	49
3.6.6	mem_fun für Elementfunktionen ohne Argument	50
3.6.7	mem_fun1_t und const_mem_fun1_t	50
3.6.8	mem_fun für Elementfunktionen mit Argument.....	51
3.7	Funktionen zusammensetzen	51
3.8	Aufgaben	52
4	Hilfsmittel	55
4.1	Vergleichsoperatoren.....	55
4.2	pair	56
4.3	Aufgaben	57
5	Container	59
5.1	vector	60
5.2	Allgemeine Anforderungen an Container.....	62
5.3	Anforderungen an reversible Container	68
5.4	Anforderungen an sequenzielle Container	68
5.5	Optionale Anforderungen an sequenzielle Container.....	73
5.6	Weitere Funktionen.....	76
5.7	deque	79
5.8	list	84
5.9	Auswahl nach Aufwand.....	90
5.10	Aufgaben	91
6	Containeradapter	93
6.1	stack.....	93
6.2	queue	95
6.3	priority_queue	96
6.4	Aufgaben	99
7	Assoziative Container	101
7.1	map.....	102
7.2	Anforderungen an assoziative Container	104
7.3	Der Indexoperator der Klasse map	109
7.4	multimap	110
7.5	set.....	113
7.6	multiset	116
7.7	Elemente in set und multiset modifizieren.....	118
7.8	Übersicht der Container.....	120
7.9	Aufgaben	120

8	Iteratoren	123
8.1	Iteratoranforderungen	123
8.2	Input-Iteratoren	125
8.3	Output-Iteratoren	126
8.4	Forward-Iteratoren	127
8.5	Bidirectional-Iteratoren	129
8.6	Random-Access-Iteratoren	129
8.7	Übersicht über die Iteratorkategorien	131
8.8	Hilfsklassen und -funktionen für Iteratoren	132
8.8.1	iterator_traits	132
8.8.2	Die Basisklasse iterator	134
8.8.3	Iteratorfunktionen	135
8.8.3.1	advance	135
8.8.3.2	distance	136
8.9	Reverse-Iteratoren	137
8.10	Insert-Iteratoren	140
8.10.1	back_insert_iterator	141
8.10.2	front_insert_iterator	142
8.10.3	insert_iterator	143
8.11	Stream-Iteratoren	144
8.11.1	istream_iterator	145
8.11.2	ostream_iterator	147
8.12	Aufgaben	148
9	Algorithmen	151
9.1	Übersicht	153
9.2	Nichtmodifizierende Algorithmen	157
9.2.1	for_each	157
9.2.2	find und find_if	158
9.2.3	find_end	160
9.2.4	find_first_of	161
9.2.5	adjacent_find	163
9.2.6	count und count_if	164
9.2.7	mismatch	165
9.2.8	equal	167
9.2.9	search	168
9.2.10	search_n	169
9.3	Modifizierende Algorithmen	170
9.3.1	copy	170
9.3.2	copy_backward	172
9.3.3	swap	173
9.3.4	iter_swap	173
9.3.5	swap_ranges	174
9.3.6	transform	175
9.3.7	replace und replace_if	177
9.3.8	replace_copy und replace_copy_if	178
9.3.9	fill und fill_n	179

9.3.10	generate und generate_n	181
9.3.11	remove_copy und remove_copy_if.....	182
9.3.12	remove und remove_if.....	183
9.3.13	unique_copy	185
9.3.14	unique.....	187
9.3.15	reverse	188
9.3.16	reverse_copy	189
9.3.17	rotate.....	190
9.3.18	rotate_copy	191
9.3.19	random_shuffle	192
9.3.20	partition und stable_partition	194
9.4	Sortieren und ähnliche Operationen	195
9.4.1	sort.....	195
9.4.2	stable_sort.....	197
9.4.3	partial_sort.....	198
9.4.4	partial_sort_copy	199
9.4.5	nth_element	200
9.5	Binäre Suchalgorithmen	202
9.5.1	lower_bound	202
9.5.2	upper_bound.....	203
9.5.3	equal_range.....	204
9.5.4	binary_search.....	206
9.5.5	Schlüsselsuche mit Funktionsobjekt.....	207
9.6	Mischalgorithmen.....	208
9.6.1	merge.....	208
9.6.2	inplace_merge.....	209
9.7	Mengenalgorithmen für sortierte Bereiche	210
9.7.1	includes.....	211
9.7.2	set_union	212
9.7.3	set_intersection	213
9.7.4	set_difference	215
9.7.5	set_symmetric_difference.....	216
9.7.6	Mengenalgorithmen für multiset-Objekte	217
9.8	Heap-Algorithmen	218
9.8.1	make_heap	219
9.8.2	pop_heap.....	219
9.8.3	push_heap	219
9.8.4	sort_heap	220
9.8.5	Beispielprogramm	220
9.9	Minimum und Maximum	221
9.9.1	min	221
9.9.2	max.....	222
9.9.3	min_element.....	223
9.9.4	max_element	224
9.10	Permutationen.....	225
9.10.1	lexicographical_compare	225
9.10.2	next_permutation.....	226

9.10.3	prev_permutation	227
9.11	Numerische Algorithmen.....	229
9.11.1	accumulate.....	229
9.11.2	inner_product.....	230
9.11.3	adjacent_difference	231
9.11.4	partial_sum	231
9.12	Erweitern der Bibliothek mit eigenen Algorithmen.....	232
9.13	Präfix- versus Postfixoperatoren.....	234
9.14	Aufgaben.....	235
10	Allokatoren	239
10.1	Der Standardallokator	239
10.2	allocator<void>	243
10.3	Aufgaben.....	243
11	Strings	245
11.1	Containereigenschaften	246
11.2	basic_string.....	248
11.3	Implementierungsdetails.....	262
11.4	Sortieren von Strings	263
11.5	Aufgaben.....	264
12	Streams	267
12.1	Überblick	267
12.2	ios_base	270
12.2.1	Formatierung	272
12.2.2	Streamstatus.....	277
12.2.3	Initialisierung	278
12.2.4	Nationale Einstellungen.....	278
12.2.5	Synchronisation	280
12.3	basic_ios	280
12.3.1	Statusfunktionen	282
12.3.2	Ausnahmen	283
12.4	basic_ostream.....	284
12.4.1	sentry	285
12.4.2	Formatierte Ausgaben.....	287
12.4.3	Unformatierte Ausgaben	288
12.5	basic_istream.....	288
12.5.1	sentry	290
12.5.2	Formatierte Eingaben	290
12.5.3	Unformatierte Eingaben	291
12.6	basic_iostream.....	294
12.7	Ein- und Ausgabe von Objekten benutzerdefinierter Klassen	295
12.8	Namensdeklarationen.....	296
12.9	Manipulatoren.....	297
12.9.1	Manipulatoren ohne Parameter.....	297
12.9.2	Manipulatoren mit einem Parameter	300
12.10	Positionieren von Streams.....	301

12.11	Streams für Dateien	303
12.11.1	Modi zum Öffnen von Dateien	303
12.11.2	Die Header-Datei <fstream>	304
12.12	Streams für Strings	307
12.13	Aufgaben	309
13	Weitere Komponenten der C++-Standardbibliothek	311
13.1	auto_ptr.....	311
13.2	bitset.....	317
13.3	vector<bool>	323
13.4	complex	325
13.5	numeric_limits.....	331
13.6	valarray.....	337
13.6.1	slice und slice_array	343
13.6.2	gslice und gslice_array.....	346
13.6.3	mask_array	348
13.6.4	indirect_array.....	350
13.7	Aufgaben	351
14	Zeiger in Containern verwalten	353
14.1	Beispielklassen	353
14.2	Ein set-Objekt verwaltet Zeiger	356
14.3	Smart-Pointer	357
14.4	Ein set-Objekt verwaltet Smart-Pointer.....	358
14.5	Ein set-Objekt verwaltet Zeiger mittels Funktionsobjekten.....	359
14.6	Ein map-Objekt verwaltet Zeiger.....	361
14.7	Aufgaben	363
15	Lösungen	365
Anhang		399
A	Die Containerklasse slist	399
B	Die Klasse LogAllocator	413
C	Literatur.....	415
Index		416

Die C++-Standardbibliothek

Einführung und Nachschlagewerk

Kuhlins, S.; Schader, M.

2005, XIV, 424 S. 77 Abb., Softcover

ISBN: 978-3-540-25693-9