

目录

庖丁解牛（侯捷自序）	i
目录	v
前言	xvii
本书定位	xvii
合适的读者	xviii
最佳阅读方式	xviii
我所选择的剖析对象	xix
各章主题	xx
编译工具	xx
中英术语的运用风格	xxi
英文术语采用原则	xxii
版面字形风格	xxiii
源码形式与下载	xxiv
在线服务	xxvi
推荐读物	xxvi
第 1 章 STL 概论与版本简介	001
1.1 STL 概论	001
1.1.1 STL 的历史	003
1.1.2 STL 与 C++ 标准程序库	003

1.2 STL 六大组件 — 功能与运用	004
1.3 GNU 源码开放精神	007
1.4 HP STL 实现版本	009
1.5 P.J. Plauger STL 实现版本	010
1.6 Rouge Wave STL 实现版本	011
1.7 STLport 实现版本	012
1.8 SGI STL 实现版本 总览	013
1.8.1 GNU C++ header 文件分布	014
1.8.2 SGI STL 文件分布与简介	016
STL 标准头文件 (无扩展名)	017
C++ 标准规格定案前, HP 规范的 STL 头文件 (扩展名 .h)	017
SGI STL 内部文件 (SGI STL 真正实现于此)	018
1.8.3 SGI STL 的组态设定 (configuration)	019
1.9 可能令你困惑的 C++ 语法	026
1.9.1 <code>stl_config.h</code> 中的各种组态	027
组态 3 : static template member	027
组态 5 : class template partial specialization	028
组态 6 : function template partial order	028
组态 7 : explicit function template arguments	029
组态 8 : member templates	029
组态 10 : default template argument depend on previous template parameters	030
组态 11 : non-type template parameters	031
组态 : bound friend template function	032
组态 : class template explicit specialization	034
1.9.2 临时对象的产生与运用	036
1.9.3 静态常数整数成员在 class 内部直接初始化 in-class static const <i>integral</i> data member initialization	037

1.9.4 increment/decrement/dereference 运算符	037
1.9.5 “前闭后开”区间表示法 [)	039
1.9.6 function call 运算符 (operator())	040
第 2 章 空间配置器 (allocator)	043
2.1 空间配置器的标准接口	043
2.1.1 设计一个简单的空间配置器, JJ::allocator	044
2.2 具备次配置力 (sub-allocation) 的 SGI 空间配置器	047
2.2.1 SGI 标准的空间配置器, std::allocator	047
2.2.2 SGI 特殊的空间配置器, std::alloc	049
2.2.3 构造和析构基本工具: construct() 和 destroy()	051
2.2.4 空间的配置与释放, std::alloc	053
2.2.5 第一级配置器 __malloc_alloc_template 剖析	056
2.2.6 第二级配置器 __default_alloc_template 剖析	059
2.2.7 空间配置函数 allocate()	062
2.2.8 空间释放函数 deallocate()	064
2.2.9 重新充填 free-lists	065
2.2.10 内存池 (memory pool)	066
2.3 内存基本处理工具	070
2.3.1 uninitialized_copy	070
2.3.2 uninitialized_fill	071
2.3.3 uninitialized_fill_n	071
第 3 章 迭代器 (iterators) 概念与 traits 编程技法	079
3.1 迭代器设计思维 — STL 关键所在	079
3.2 迭代器是一种 smart pointer	080
3.3 迭代器相应型别 (associated types)	084
3.4 Traits 编程技法 — STL 源码门钥	085

Partial Specialization (偏特化) 的意义	086
3.4.1 迭代器相应型别之一 value_type	090
3.4.2 迭代器相应型别之二 difference_type	090
3.4.3 迭代器相应型别之三 pointer_type	091
3.4.4 迭代器相应型别之四 reference_type	091
3.4.5 迭代器相应型别之五 iterator_category	092
以 advanced() 为例	093
消除 “ 单纯传递调用函数 ”	097
以 distance() 为例	098
3.5 std::iterator class 的保证	099
3.6 iterator 相关源码完整重列	101
3.7 SGI STL 的私房菜 : __type_traits	103
第 4 章 序列式容器 (sequence containers)	113
4.1 容器概观与分类	113
4.1.1 序列式容器 (sequence containers)	114
4.2 vector	115
4.2.1 vector 概述	115
4.2.2 vector 定义摘要	115
4.2.3 vector 的迭代器	117
4.2.4 vector 的数据结构	118
4.2.5 vector 的构造与内存管理 : constructor, push_back	119
4.2.6 vector 的元素操作 : pop_back, erase, clear, insert	123
4.3 list	128
4.3.1 list 概述	128
4.3.2 list 的节点 (node)	129
4.3.3 list 的迭代器	129
4.3.4 list 的数据结构	131

4.3.5 <code>list</code> 的构造与内存管理 : <code>constructor</code> , <code>push_back</code> , <code>insert</code>	132
4.3.6 <code>list</code> 的元素操作 : <code>push_front</code> , <code>push_back</code> , <code>erase</code> , <code>pop_front</code> , <code>pop_back</code> , <code>clear</code> , <code>remove</code> , <code>unique</code> , <code>splice</code> , <code>merge</code> , <code>reverse</code> , <code>sort</code>	136
4.4 <code>deque</code>	143
4.4.1 <code>deque</code> 概述	143
4.4.2 <code>deque</code> 的中控器	144
4.4.3 <code>deque</code> 的迭代器	146
4.4.4 <code>deque</code> 的数据结构	150
4.4.5 <code>deque</code> 的构造与内存管理 : <code>ctor</code> , <code>push_back</code> , <code>push_front</code>	152
4.4.6 <code>deque</code> 的元素操作 : <code>pop_back</code> , <code>pop_front</code> , <code>clear</code> , <code>erase</code> , <code>insert</code>	161
4.5 <code>stack</code>	167
4.5.1 <code>stack</code> 概述	167
4.5.2 <code>stack</code> 定义式完整列表	167
4.5.3 <code>stack</code> 没有迭代器	168
4.5.4 以 <code>list</code> 做为 <code>stack</code> 的底层容器	168
4.6 <code>queue</code>	169
4.6.1 <code>queue</code> 概述	169
4.6.2 <code>queue</code> 定义式完整列表	170
4.6.3 <code>queue</code> 没有迭代器	171
4.6.4 以 <code>list</code> 做为 <code>queue</code> 的底层容器	171
4.7 <code>heap</code> (隐式表述 , <code>implicit representation</code>)	172
4.7.1 <code>heap</code> 概述	172
4.7.2 <code>heap</code> 算法	174
<code>push_heap</code>	174
<code>pop_heap</code>	176
<code>sort_heap</code>	178
<code>make_heap</code>	180

4.7.3 heap 没有迭代器	181
4.7.4 heap 测试实例	181
4.8 priority-queue	183
4.8.1 priority-queue 概述	183
4.8.2 priority-queue 定义式完整列表	183
4.8.3 priority-queue 没有迭代器	185
4.8.4 priority-queue 测试实例	185
4.9 slist	186
4.9.1 slist 概述	186
4.9.2 slist 的节点	186
4.9.3 slist 的迭代器	188
4.9.4 slist 的数据结构	190
4.9.5 slist 的元素操作	191
第 5 章 关联式容器 (associated containers)	197
5.1 树的导览	199
5.1.1 二元搜寻树 (binary search tree)	200
5.1.2 平衡二元搜寻树 (balanced binary search tree)	203
5.1.3 AVL tree (Adelson-Velskii-Landis tree)	203
5.1.4 单旋转 (Single Rotation)	205
5.1.5 双旋转 (Double Rotation)	206
5.2 RB-tree (红黑树)	208
5.2.1 插入节点	209
5.2.2 一个由上而下的程序	212
5.2.3 RB-tree 的节点设计	213
5.2.4 RB-tree 的迭代器	214
5.2.5 RB-tree 的数据结构	218
5.2.6 RB-tree 的构造与内存管理	221

5.2.7 RB- tree 的元素操作	223
元素插入动作 <code>insert_equal</code>	223
元素插入动作 <code>insert_unique</code>	224
真正的插入执行程序 <code>__insert</code>	224
调整 RB- tree (旋转及改变颜色)	225
元素的搜寻 <code>find</code>	229
5.3 set	233
5.4 map	237
5.5 multiset	245
5.6 multimap	246
5.7 hashtable	247
5.7.1 hashtable 概述	247
线性探测 (linear probing)	249
二次探测 (quadratic probing)	251
分离链 (separate chaining)	253
5.7.2 hashtable 的桶子 (buckets) 与节点 (nodes)	253
5.7.3 hashtable 的迭代器	254
5.7.4 hashtable 的数据结构	256
5.7.5 hashtable 的构造与内存管理	258
插入动作 (insert) 与表格重整 (resize)	259
判知元素的落脚处 (bkt_num)	262
复制 (copy_from) 和整体删除 (clear)	263
5.7.6 hashtable 运用实例 (find, count)	264
5.7.7 hash functions	268
5.8 hash_set	270
5.9 hash_map	275
5.10 hash_multiset	279

5.11 hash_multi_map	282
第 6 章 算法 (algorithms)	285
6.1 算法概观	285
6.1.1 算法分析与复杂度表示 $O(\)$	286
6.1.2 STL 算法总览	288
6.1.3 mutating algorithms — 会改变操作对象之值	291
6.1.4 nonmutating algorithms — 不改变操作对象之值	292
6.1.5 STL 算法的一般形式	292
6.2 算法的泛化过程	294
6.3 数值算法 <stl_numeric.h>	298
6.3.1 运用实例	298
6.3.2 accumulate	299
6.3.3 adjacent_difference	300
6.3.4 inner_product	301
6.3.5 partial_sum	303
6.3.6 power	304
6.3.7 itoa	305
6.4 基本算法 <stl_algobase.h>	305
6.4.1 运用实例	305
6.4.2 equal	307
fill	308
fill_n	308
iter_swap	309
lexicographical_compare	310
max, min	312
mismatch	313
swap	314
6.4.3 copy, 强化效率无所不用其极	314
6.4.4 copy_backward	326
6.5 Set 相关算法 (应用于有序区间)	328

6.5.1	set_union	331
6.5.2	set_intersection	333
6.5.3	set_difference	334
6.5.4	set_symmetric_difference	336
6.6	heap 算法 : make_heap, pop_heap, push_heap, sort_heap	338
6.7	其它算法	338
6.7.1	单纯的数据处理	338
	adjacent_find	343
	count	344
	count_if	344
	find	345
	find_if	345
	find_end	345
	find_first_of	348
	for_each	348
	generate	349
	generate_n	349
	includes (应用于有序区间)	349
	max_element	352
	merge (应用于有序区间)	352
	min_element	354
	partition	354
	remove	357
	remove_copy	357
	remove_if	357
	remove_copy_if	358
	replace	359
	replace_copy	359
	replace_if	359
	replace_copy_if	360
	reverse	360
	reverse_copy	361
	rotate	361
	rotate_copy	365
	search	365
	search_n	366
	swap_ranges	369
	transform	369
	unique	370
	unique_copy	371

6.7.2 lower_bound (应用于有序区间)	375
6.7.3 upper_bound (应用于有序区间)	377
6.7.4 binary_search (应用于有序区间)	379
6.7.5 next_permutation	380
6.7.6 prev_permutation	382
6.7.7 random_shuffle	383
6.7.8 partial_sort / partial_sort_copy	386
6.7.9 sort	389
6.7.10 equal_range (应用于有序区间)	400
6.7.11 inplace_merge (应用于有序区间)	403
6.7.12 nth_element	409
6.7.13 merge sort	411
第 7 章 仿函数 (functor, 另名 函数对象 function objects)	413
7.1 仿函数 (functor) 概观	413
7.2 可配接 (adaptable) 的关键	415
7.2.1 unary_function	416
7.2.2 binary_function	417
7.3 算术类 (Arithmetic) 仿函数	418
plus, minus, multiplies, divides, modulus, negate, identity_element	
7.4 关系类 (Relational) 仿函数	420
equal_to, not_equal_to, greater, greater_equal, less, less_equal	
7.5 逻辑运算类 (Logical) 仿函数	422
logical_and, logical_or, logical_not	
7.6 证同 (identity)、选择 (select)、投射 (project)	423
identity, select1st, select2nd, project1st, project2nd	
第 8 章 配接器 (adapter)	425
8.1 配接器之概观与分类	425
8.1.1 应用于容器, container adapters	425

8.1.2 应用于迭代器, iterator adapters	425
运用实例	427
8.1.3 应用于仿函数, functor adapters	428
运用实例	429
8.2 container adapters	434
8.2.1 stack	434
8.2.2 queue	434
8.3 iterator adapters	435
8.3.1 insert iterators	435
8.3.2 reverse iterators	437
8.3.3 stream iterators (istream_iterator, ostream_iterator)	442
8.4 function adapters	448
8.4.1 对传回值进行逻辑否定: not1, not2	450
8.4.2 对参数进行系结(绑定): bind1st, bind2nd	451
8.4.3 用于函数合成: compose1, compose2(未纳入标准)	453
8.4.4 用于函数指针: ptr_fun	454
8.4.5 用于成员函数指针: mem_fun, mem_fun_ref	456
附录 A 参考资料与推荐读物(Bibliography)	461
附录 B 侯捷网站简介	471
附录 C STLport 的移植经验(by 孟岩)	473
Borland C++Builder 5	474
Microsoft Visual C++ 6.0	477
索引	481

