

Longxiaojun30天读书笔记分享计划  
Day 29: C++

EC条款50: 了解new和delete的合理替换时机

- 用来检测运用上的错误
- 为了强化效能
- 为了收集使用上的统计数据
- 为了增加分配和归还的速度
- 为了降低缺省内存管理器带来的空间额外开销
- 为了弥补缺省分配器中的非最佳齐位
- 为了将相关对象成簇集中
- 为了获得非传统的行为

C++ STL

- make\_heap
  - void make\_heap( RandomIt first, RandomIt last )
  - void make\_heap( RandomIt first, RandomIt last, Compare comp )
  - 在范围 [first, last) 中构造最大堆
  - 函数第一版本用 operator< 比较元素, 第二版本用给定的比较函数 comp
- sort\_heap
  - void sort\_heap( RandomIt first, RandomIt last )
  - void sort\_heap( RandomIt first, RandomIt last, Compare comp )
  - 转换最大堆 (first, last) 为以升序排序的范围, 产生的范围不再拥有堆属性
  - 函数的第一版本用 operator< 比较元素, 第二版本用给定的比较函数 comp 比较

EC条款51: 编写new和delete时需固守常规

C++ 规定, 即使客户要求0byte, operator new也得返回一个合法指针, 其中的伎俩是把0bytes申请量视为1byte申请量

内存被成功分配

- operator new内含一个无穷循环, 退出循环的唯一办法是

new-handling函数中实现了

- 让更多内容可用
- 安装另一个new-handler
- 卸载new-handler
- 抛出bad\_alloc异常 (或其派生物)
- 或是承认失败而直接return

Class专属版本还应该处理 "比正确大小更大的错误的申请"

operator delete应该在受到null指针时不做什么事, Class专属版本则还应该处理 "比正确大小更大的错误的申请"

C++ STL

- pop\_heap
  - void pop\_heap( RandomIt first, RandomIt last )
  - void pop\_heap( RandomIt first, RandomIt last, Compare comp )
  - 交换在位置 first 的值和在位置 last-1 的值, 并令子范围 [first, last-1) 变为最大堆。这拥有从范围 [first, last) 所定义的堆移除首个 (最大) 元素的效应
  - 函数的首个版本使用 operator< 比较元素, 第二版本使用给定的比较函数 comp
- push\_heap
  - void push\_heap( RandomIt first, RandomIt last )
  - void push\_heap( RandomIt first, RandomIt last, Compare comp )
  - 插入位于位置 last-1 的元素到范围 [first, last-1) 所定义的最大堆中
  - 函数的第一版本用 operator< 比较元素, 第二版本用给定的比较函数 comp