LongXiaJun30天读书笔记分享计划
Day 12: C++

- MEC条款20：协助完成返回值优化RVO： Return Value Optimization
  - 通过返回ctor以取代对象
  - 返回对象，一个有效率而且正确的做法
- STL C++
  - swap
    - void swap(T& a, T& b)
    - void swap(T2 (&a)[N], T2 (&b)[N]) noexcept
      - *交换给定值*
        1. 交换 a 与 b
        2. 交换 a 与 b 数组。等效于调用 std::swap_ranges(a, a+N, b)
    - ForwardIt2 swap_ranges(ForwardIt1 first1, ForwardIt1 last1, ForwardIt2 first2)
      - 在范围 [first1, last1) 和始于 first2 的另一范围间交换元素
    - void iter_swap(ForwardIt1 a, ForwardIt2 b)
      - 交换给定的迭代器所指向的元素的值
    - **利用swap实现选择排序，精简漂亮**

```
 template<typename ForwardIt>
void selection_sort(ForwardIt begin, ForwardIt end)
 {
  for (ForwardIt i = begin; i != end; ++i)
   std::iter_swap(i, std::min_element(i, end));
 }
```

- MEC条款21： 利用重载技术避免隐式类型转换
  - 隐式类型转换会产生临时对象，产生额外的内存和计算开销
  - 可以利用操作符重载禁止临时对象产生
  - C++中每个重载的操作符必须获得至少一个用户定制类型的自变量，不然就改变了原先定义的操作符定义
  - 不能大量使用80-20法则
- EC条款14：在资源管理类中小心copying行为
  - 条款12中RAII对象被复制时的处理方式
    - 禁止复制，条款6中让类继承自另一个Uncopyable类
    - 对底层资源使用引用计数法 Reference Counting
    - 复制底部资源，深度拷贝
    - 转移底部资源的拥有权auto_ptr
  - 复制RAII对象必须一并复制它所管理的资源，所以资源的copying行为决定RAII对象的copying行为

2018.12.18 Tuesday. By KuKuXia@github.com