第四章

\*头文件的代码一般不应该使用using声明

左值：当对象被用作左值的时候，用的是对象的身份（在内存中的位置）。

右值：当对象被用作右值的时候，用的是对象的值（内容）。

\*赋值运算符需要一个（非常量）左值做的其左侧运算对象，得到的结果也仍然是一个左值。

\*取地址符作用于一个左值运算对象，返回一个指向该运算对象的指针，这个指针是右值。

\*内置解引用运算符，下标运算符，迭代器解引用运算符，string和vector的下标运算符的求值结果都是左值。

\*内置类型和迭代器的递增减运算符作用于左值运算对象，其前置版本所得的结果也是左值。

第七章

Mutable 关键字

Explicit 关键字

\*聚合类（aggregate class） 老版本的struct 类

第九章

\*顺序容器： forward\_list(单向链表), array(数组)

选择容器的基本原则

\*除非你由很好的理由选择其他容器，否则应使用vector

\*如果你的程序有很多小的元素，且空间的额外开销很重要，则不要使用list或forward\_list

\*随机访问元素， 使用vector或deque

\*需要在头尾位置插入或删除元素，但不会在中间位置进行插入或删除，则使用deque

\*读取输入时才需要在容器中间插入元素，随后需要随机访问元素，则：

---vector追加数据，然后调用sort函数

---如果必须在中间插入元素，考虑在输入阶段使用list， 输入完成后list中的内容拷贝到一个vector

第十章

\*lambda不能有默认的参数

\*bind函数，普通函数

第12章

\*Shared\_ptr: 允许多个指针指向同一个对象。

\*Unique\_Ptr: “独占”所指向的对象。

\*Weak\_ptr: 伴随类，弱引用，指向 shared\_ptr所管理的对象。

\*Shared\_ptr 独有的操作:

make\_shared, shared\_ptr, p = q, p.unique(), p.use\_count()

\*使用智能指针语法 shared\_ptr<A> a1(new A());

\*某个时刻只能有一个unique\_ptr指向一个给定对象。

\*unique\_ptr被摧毁时，它所指向的对象也被摧毁。

\*weak\_ptr指向由一个shared\_ptr管理的对象。weak\_ptr绑定到一个shared\_ptr不会改变shared\_ptr的引用计数。一旦最后一个指向对象的shared\_ptr被摧毁，对象就会被释放。

第十三章

\*拷贝构造函数不应该是explicit的

\*三/五法则：

拷贝构造好书，拷贝赋值运算符，析构函数，移动构造函数，移动赋值运算符

\*如果需要一个析构函数，我们几乎肯定也需要一个拷贝构造函数和一个拷贝赋值运算符

\*Sales\_data() = default;

\*Sales\_data() = delete; 阻止拷贝

\*不能delete 析构函数；~Sales\_data() = delete 错误

\*swap 函数 自定义

\*拷贝并交换 (copy and swap) ????

\*move函数

\*右值引用，&& 来获取右值引用

\*一个左值的表达式表示的是一个对象的身份，而一个右值表达式表示的是对象的值

\*左值持久： 右值短暂

\*右值要么是字面常量，要么是在表达式求值过程中创建的临时对象。

\*移动构造函数不分配任何新内存

\*五个拷贝控制成员应该看成一个整体，定义任何一个拷贝操作，则应该定义所有五个操作。

第十四章

\*函数对象

\*lambda表达式 auto mod = [](int i, int j) { return i %j; } ;

\*function 关键字 可以放入 函数指针，函数对象，lambda表达式

Function<int(int,int)> f1 = add; //函数指针

Function<int(int,int)> f2 = divide(); //函数对象

Function<int(int,int)> f3 = [] (int i, int j) { return i%j; }; //lambda

第十五章

\*final 关键字

第十六章

\*std::move

其他 略

第十七章

\*tuple 类型

希望将一些数据组合成单一对象，但又不想定义一个新的结构来表示，tuple是非常有用的。

\*bitset

\*正则表达式： RE库

\*随机数

第十八章

\*内联命名空间 inline namespace FifthEd {}

\*虚继承

第十九章

略