序

面向对象语言编译器提供了很多服务，如构造函数，析构函数，虚函数，继承，多态..等

这本书解释编译器对我们写的C++代码做了什么手脚

目前所有编译器对虚函数的实现都是使用各个class专属的virtual table，大小固定，在程序执行前就构造好了。

第一章：关于对象

类的对象占用的存储空间：只有非静态的成员数据加上数据在内存中对齐而填补的空间加上指向虚函数表的虚指针占用存储空间。

C++未选用的两种对象模型：简单对象模型，表格驱动对象模型。

C++对象模型：每一个对象内部都有非静态数据成员。静态数据成员，静态成员函数和非静态成员函数不在每一个对象内部，只在外部存在一份。虚函数分两步，第一步每一个类的每一个虚函数都有一个指向它的指针，所有虚函数的指针放在一张表格中，虚表（vtbl），第二步每一个类的对象安插一个指针，指向相关的虚表，虚指针（vptr），这时每个类的对象的模型中存着它的非静态数据成员和一个vptr，vptr指向vtbl，vtbl中的第一个槽存储的是类的type\_info

C++对象模型加上继承：1、bptr，也就是base class table模型，但是添加了间接性，继承了几层，就要间接几层。2、在vtbl中加入base类的指针。

C++三种编程范式：程序模型，抽象数据模型，面向对象模型。

C++支持多态：基类指针结合虚函数实现多态；dynamic\_cast实现基类指针转为派生类指针。

指针类型：所有指针都是一样，大小一样，而指针类型决定了指向的地址怎么处理，如int的指针，int占4个字节，那么指针指向的地址开始连续四个字节属于这个int对象。而void指针只有地址没有空间，因为void类型没有空间。

Cstruct的C++合理用途：传递一个复杂的class object的全部或者部分到某个c函数去时，struct可以将数据封装起来，并保证拥有和c兼容的空间布局。

加上多态：一个指针或者引用之所以支持多态，是因为它们不引发内存中任何“与类型有关的内存委托操作”，会受到改变的，只有他们所指向内存的大小和内容解释方式而已。

第二章：构造语义函数学