C 标准编程：虚函数与内联计算机等级考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/645/2021_2022_C___E6_A0_87_E5_87_86_E7_c97_645788.htm 我们曾经在讨论C 的时候，经常会问到：“ 虚函数能被声明为内联吗？ ”现在，我们几乎听不到这个问题了。现在听到的是：“ 你不应该使print成为内联的。声明一个虚函数为内联是错误的！ ”这种说法的两个主要的原因是（1）虚函数是在运行期决议而内联是一个编译期动作，所以，我们将虚函数声明为内联并得不到什么效果；（2）声明一个虚函数为内联导致了函数的多分拷贝，而且我们为一个不应该在任何时候内联的函数白白花费了存储空间。这样做很没脑子。 不过，事实并不是这样。我们先来看看第一个：许多情况下，虚拟函数都被静态地决议了比如在派生类虚拟函数中调用基类的虚拟函数的时候。为什么这样做呢？封装。一个比较明显的例子就是派生类析构函数调用链。所有的虚析构函数，除了最初触发这个析构链的虚析构函数，都被静态的决议了。如果不将基类的虚析构函数内联，我们无法从中获利[a]。这和不内联一个虚拟析构函数有什么不同吗？如果继承体系层次比较深并且有许多这样的类的实例要被销毁的话，答案是肯定的。 再来看另外一个不用析构函数的例子，想象一下设计一个图书馆类。我们将MaterialLocation作为抽象类LibraryMaterial的一个成员。将它的print成员函数声明为一个纯虚函数，并且提供函数定义：它输出MaterialLocation。 class LibraryMaterial { private: MaterialLocation _loc. // shared data // ... public: // declares pure virtual function inline virtual void print(ostream 100Test 下载频道