



程序设计实习

郭炜 微博 <http://weibo.com/guoweiofpku>

<http://blog.sina.com.cn/u/3266490431>

刘家瑛 微博 <http://weibo.com/pkuliujiaying>



多态的实现原理

思考

“多态”的关键在于通过基类指针或引用调用一个虚函数时，编译时不确定到底调用的是基类还是派生类的函数，运行时才确定 ----- 这叫“**动态联编**”。“**动态联编**”底是怎么实现的呢？

提示：请看下面例子程序：

```
class Base {  
    public:  
    int i;  
    virtual void Print() { cout << "Base:Print" ; }  
};  
class Derived : public Base{  
    public:  
    int n;  
    virtual void Print() { cout <<"Drived:Print" << endl; }  
};  
int main() {  
    Derived d;  
    cout << sizeof( Base) << ","<< sizeof( Derived ) ;  
    return 0;  
}
```

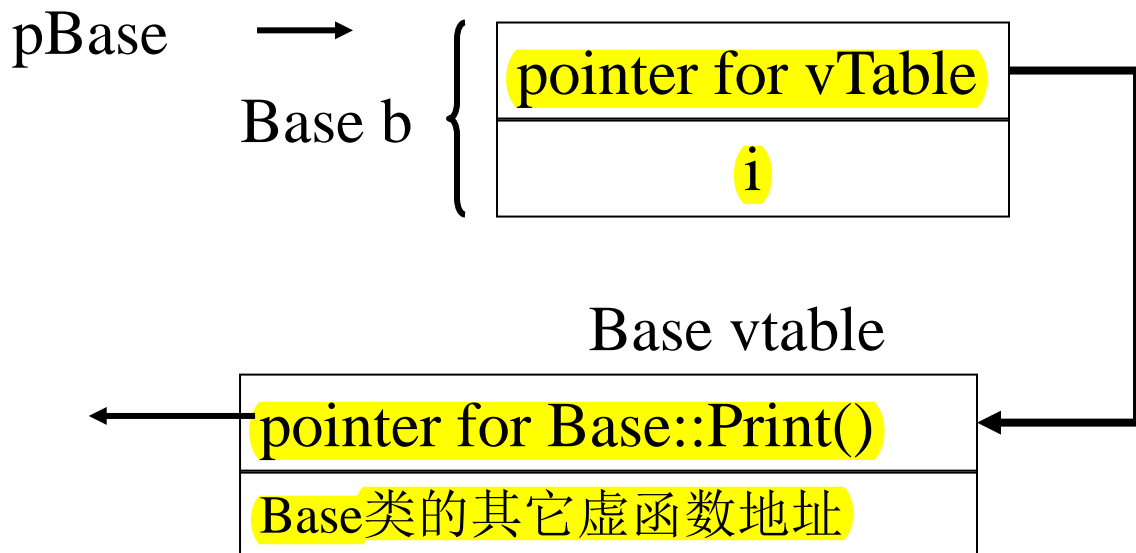
程序运行输出结果： 8, 12



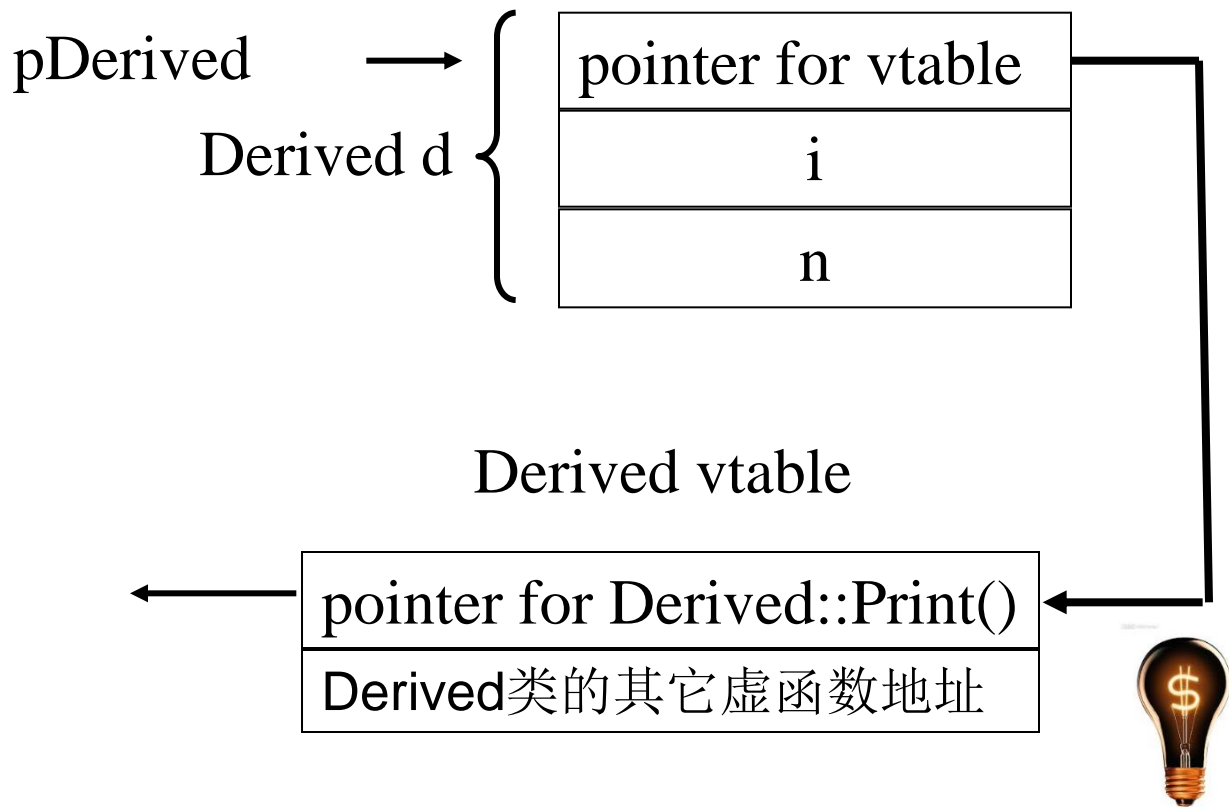
为什么都多了4个字节？

多态实现的关键 —— 虚函数表

每一个有虚函数的类（或有虚函数的类的派生类）都有一个虚函数表，该类的任何对象中都放着虚函数表的指针。虚函数表中列出了该类的虚函数地址。多出来的4个字节就是用来放虚函数表的地址的。



多态实现的关键 —— 虚函数表



```
pBase = pDerived;
pBase->Print();
```

➤ 多态的函数调用语句被编译成一系列根据基类指针所指向的（或基类引用所引用的）对象中存放的虚函数表的地址，在虚函数表中查找虚函数地址，并调用虚函数的指令。