电子信息与工程学院

姓名	学号	专业	联系方式
黄炜恺	17311023	微电子科学与工程	13725951349

代码

```cpp
#include <iostream>
using namespace std;

class A
{
protected:
    int x;
public:
    A(){x = 1000;}
    virtual void p()
    {
        cout << "x= " << x << '\n';
        p2();
    }
    virtual void p2()
    {
        cout << "A::p2()" << endl;
    }
};

class C:public A
{
    int z;
public:
    C(){z = 3000;}
    void p()
    {
        cout << "z= " << z << '\n';
        p2();
    }
    virtual void p2()
    {
        cout << "C::p2()" << endl;
    }
};

int main()
{
    C c;
    A a,*pa = &a;
    pa->p();
    pa = &c;
    pa->p();
    return 0;
}
```

分析

1、一旦把基类的成员函数定义为虚函数，由基类所派生出的所有派生类中，该函数均保持虚函数的特性。

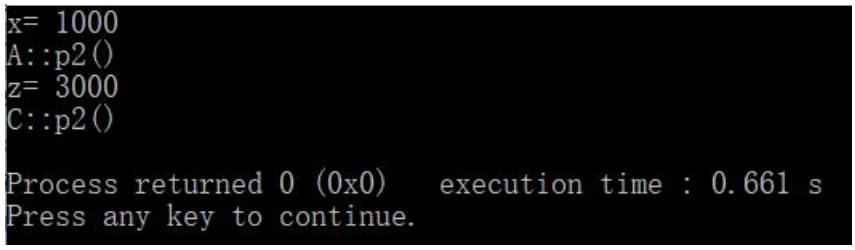
2、一旦一个函数被声明为虚函数，那么他从该点之后的继承层次结构中都是虚函数，不管它在有没有再次声明是不是虚函数，有些程序员为了提高程序的清晰度，在继承结构中喜欢再次明确的声明这些虚函数。在派生类中声明的虚函数性质并不能向上延伸到基类。例如，将基类 A 中的对 p 的 virtual 虚函数声明搬到派生类 C 对 p 的声明时，将不能出现理想的结果（只能调用基类 A 中的 p（））。

3、在派生类中重新定义基类的虚函数时，可以不用关键字 virtual 来修饰这个成员函数。

4、代码中 pa = &c;pa->p();是实现多态的例子：用父类指针指向子类空间并在类外通过指针调用子类的成员函数。但输出结果中的 C::p2 却并不是把 p2 声明为虚函数的功劳，无论在 p2 前加不加 virtual 声明结果都相同。原因是在派生类中新声明的成员函数 p2 覆盖了基类中的成员函数 p2。

5、虚函数不能是内联函数，即使在声明虚函数前加上 inline，编译器也会忽略掉这一声明。

运行结果截图


x= 1000
A::p2()
z= 3000
C::p2()
Process returned 0 (0x0)   execution time : 0.661 s
Press any key to continue.