抽象基类和纯虚函数

纯虚函数“

所谓的纯虚函数与我们之间学习的虚函数的区别在于：  
虚函数： virtual void speak() {}

纯虚函数： virtual void speak() = 0;

纯虚函数在虚函数的后面加上了 = 0

抽象基类：

基类中添加了至少一个纯虚函数的基类称为抽象类。

从“抽象“这两个字可以看出，这是一个范围很大的类，抽象即不是具体的，有公共使用的意思。由于抽象基类中有纯虚函数，而纯虚函数都是没有具体实现的，只是在后面加上 = 0，则抽象基类是无法实例化的，因为，纯虚函数都是没有具体实现的。

当继承一个抽象类时，子类必须要将父类中的纯虚函数给具体实现，不然继承下来的子类依然时抽象类，是无法实例化的。

纯虚函数 virtual void fun() = 0其实就是告诉编译器在抽象类的vftable中为该纯虚函数保留一个位置，但是该位置并没有存放相应的地址。是为了等待子类继承并具体实现。

总结：

抽象类是为了建立一个公共使用的类，将子类的公共操作抽象出来，然后子类们通过继承这个抽象基类，可以在它的基础上完成更加具体的实现。因此，这个抽象基类是不需要实现的，即纯虚函数 = 0 的形式就可以理解了。

实例：

抽象基类是泡饮品的抽象操作

子类 泡咖啡是 泡饮品的具体操作

子类 泡茶是 泡饮品的具体操作

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<iostream>

using namespace std;

//抽象基类，制作饮品

class AbstractDrinking

{

public:

//烧水

virtual void Boil() = 0;

//泡

virtual void Brow() = 0;

//倒入杯子

virtual void PourInCup() = 0;

//辅料

virtual void PutSomething() = 0;

//规定流程

void MakeDrink()

{

Boil();

Brow();

PourInCup();

PutSomething();

}

};

//子类制作咖啡

class Coffee : public AbstractDrinking

{

public:

//烧水

virtual void Boil()

{

cout << "煮山泉" << endl;

}

//泡

virtual void Brow()

{

cout << "泡咖啡" << endl;

}

//倒入杯子

virtual void PourInCup()

{

cout << "咖啡倒入杯子" << endl;

}

//辅料

virtual void PutSomething()

{

cout << "加牛奶" << endl;

}

};

//子类 泡茶

class Tea : public AbstractDrinking

{

public:

//烧水

virtual void Boil()

{

cout << "煮白开水" << endl;

}

//泡

virtual void Brow()

{

cout << "泡茶" << endl;

}

//倒入杯子

virtual void PourInCup()

{

cout << "茶倒入杯子" << endl;

}

//辅料

virtual void PutSomething()

{

cout << "加盐" << endl;

}

};

//业务函数

void DoBussiness(AbstractDrinking\* drink)

{

drink->MakeDrink();

delete drink;

}

void test()

{

DoBussiness(new Coffee);

cout << endl;

DoBussiness(new Tea);

}

int main()

{

test();

return 0;

}

