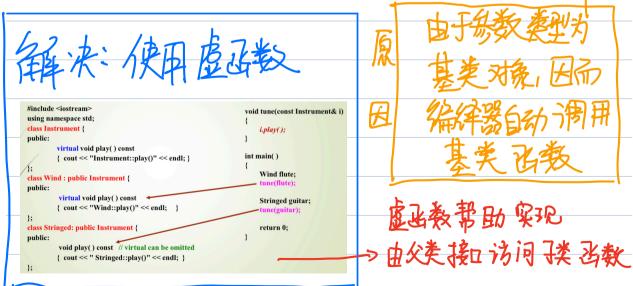
## 第十五章 多态性

```
CASE 1: Why use virtual function?
#include <iostream>
using namespace std;
class Instrument {
public:
         void play() const
         { cout << "Instrument::play()" << endl; }
                                                                 int main()
class Wind : public Instrument {
                                                                      Wind flute;
public:
                                                                      tune(flute);
         void play() const
         { cout << "Wind::play()" << endl; }
                                                                      Stringed guitar;
                                                                      tune(guitar);
class Stringed: public Instrument {
public:
                                                                     return 0;
         void play() const
         { cout << " Stringed::play()" << endl; }
```



虚出数使用的级关键字virtual使得派函数变成了一个接口,编绎器格根据传入的对象实现不同派生类的近数操作

义当基类中的战数被足沙克或数时, 派生类中的同型运数数与自动超步发力 虚光数,Virtual 光键字可缺省。

※基类中定义即虚函数和关中的虚函数格计多处 一模一样 →> ①返回类型

了设数名

※<u>传入虚记数</u>/调用虚设数的务必为对象的 ?|用或指针→>才能动态绑定

```
#include <iostream>
                                                                  class Stringed: public Instrument
using namespace std;
                                                                  public:
class Instrument {
                                                                      void play() const
public:
         void tune() { play();
                                                                         cout << " Stringed ::play()";
         void play() const
         { cout << "Instrument::play()" << endl;
                                                                  };
                         virtual void play() const;
                                                                  int main() {
                                                                       Wind flute;
class Wind: public Instrument
                                                                       flute.tune();
public:
                                                                       Stringed guitar;
         void play() const
                                                                       guitar.tune();
         { cout << "Wind::play()" << endl; }
                                                                       return 0;
};
```

2、析构的数度的数化)

# 般将基类的折构的数定效虚的数

```
#include <iostream>
                                                                  int main()
using namespace std;
                                                                      Item * p;
class Item {
                                                                      p = new Item();
public:
                                                                      delete p;
         Item() { id = 0; }
         virtual ~Item() { cout <<"Item deleted"<<endl; }
                                                                       p = new BookItem();
private: int id;
                                                                       delete p;
                                                                       return 0;
class BookItem : public Item {
    BookItem() { title = new char [50]; }
  ~BookItem()
                                                                    III Microsoft Visual Studio 调试控制台
   {
                                                                    Item deleted
        cout << "BookItem deleted" << endl;
                                                                   BookItem deleted
                          delete[] title;
       if (title != nullptr)
                                                                    Item deleted
private:
            char * title;
```

#### 此时不要求函数名一样

3、虚函数的实现原理 -> 动态舒定

新路教的多数邻定类型 切为静态绑定, 即函数的调用与参数的数据类型相关 一个人,不是有多态性

与之机时,虚战数别实现利用了动态绑定,即 强数的调用与程序当前 发入的引用或指针 所动态指定的对象有关

舒虚函数的类件被编绎器的门隐含的描述一个指针,指向包含液类所有虚函数的函数 表 [作用域为淡类],供动态绑定时使用

义 7类中庭出数的这回类型可以与基类中的虚比数不同,但必须为父类虚当教 返回类型的子类型 4、抽象类和练虚函数 一个纯虚函数是定义为不被实现的虚函数,该虚 无函数体效 函数从'=0'为标明标记,是派类连接其类虚函数。 的接口 virtual void Function ( ) =0 j 当一个类中定义了经虚函数,则在其一类中应当对该接口 功能进行实现, 到其类也继承派纯应函数 **什为抽象**类 于由免类:PD 成员已数中包含有纯度已数的类。 这种类 将作为父类存在 , 表示 . 种 抽象概 危,因而无法足少实例 戏 义 特别的,对于纯虚析构出数 ,允许其在类外给予定义 以满足对强销段时调用折构出数的形式完本

#### 15.6 Pure virtual destructor In common sense, we don't give the source code for pure virtual function. But in the special, it's possible to provide a definition for a pure virtual function in the base class. There may be a common piece of code that we want some or all of the derived class definitions to call rather than duplicating that code in every function. #include <iostream> using namespace std; class Pet { // Upcase public: virtual ~Pet() = 0; Pet \*p = new Dog(); // Don't implement in the class // Virtual destructor call Pet::~Pet() { cout << "~Pet()" << endl; } delete p; return 0; class Dog: public Pet { public: ~Dog() { cout << "~Dog()" << endl; }

5.	父类向子类的访问(自上而下) downcasting

### 

②不同的子类之间是平级且无联级的,不能互相 轻化。

