C++ 中的虚函数

C++中的虚函数，结合动态绑定特性，是实现C++中多态的根本。
C++中的虚函数定义如下：

```
struct Base{
    virtual void func(){
        // do something
    }

};
struct Derived: Base{
    virtual void func(){
        // ...
    }
};

void foo(const Base & obj){// void foo(const Base *obj)
    obj.func();
}
Base b;
Derived d;
foo(b);// 调用Base中的func()
foo(d);// 调用Derived中的func()
```

即加上`virtual`字段，并且C++中所有的虚函数必须要定义。虚函数的概念，虚就虚在所谓的“动态绑定”上，也就是`obj`的`dynamic type`可以是`Base`，也可以是`Derived`。该动态类型到运行时才可以被确定。

纯虚函数 (pure virtual function)

纯虚函数是在基类中声明的虚函数，其在基类中没有定义，要求派生类都要定义自己的是实现方法。声明方法如下：

```
virtual void func()=0;
```

引入原因

1. 为了方便使用多态性，我们需要在基类中定义虚函数
2. 在多数情况下，基类本身不需要生成对象。例如，动物类作为一个基类，可以派生出猫类，狗类等，但动物类本身生成对象则不是合理的。

为了解决这个问题，便引入了纯虚函数的概念，将函数定义为纯虚函数，则编译器要求在派生类中必须予以重写以实现多态性。同时将含有纯虚函数的类称为**抽象类**，它不可以生成对象。

声明了纯虚函数的类是一个抽象类，它不可以创建类的实例，只可以创建它派生类的实列。纯虚函数最显著的特征是：他们必须在继承类中重新声明函数，而且在抽象类中没有往往没有定义（可以在类外定义纯虚函数）。定义纯虚函数的目的在于，使派生类仅仅继承函数的接口。