多态性与虚函数

一、什么是多态性

多态性：**向不同的对象发送同一个消息，不同的对象在接收时会产生不同的行为。也就是说，每个对象可以用自己的方式去响应共同的消息。所谓消息，就是调用函数，不同的行为就是指不同的实现，即执行不同的函数。**

例如：学生和老师听到下课铃（消息）后，学生出去玩，老师收拾东西下课离开教室。

例如：使用运算符+时，使两个数值相加，就是发送一个消息，它要调用operator + 函数。实际上，整型、单精度型的加法操作是互不相同的，是由不同内容的函数实现的。显然，它们以不同的而行为或方法来响应同一消息。

二、利用虚函数实现动态多态性

虚函数，就是在基类声明函数是虚拟的，不是实际存在的函数，在派生类中才正式定义此函数。在程序运行期间，用指针指向某一派生类对象，这样就能调用指针指向的派生类对象中的函数，而不会调用其他派生类中的函数。

1、虚函数的作用

虚函数的作用：**允许在派生类中重新定义与基类同名的函数，并且可以通过基类指针或引用来访问基类和派生类中的同名函数。**（指向基类的指针可以指向派生类）

```
#include <iostream>
#include <string>

using namespace std;
class A   //声明基类A
{
public:
    virtual void display()//虚函数，在派生类中可以重新定义
    {
        a=1;
        b=100;
        cout<<a<<endl<<b<<endl;//输出a和b
    }
protected:
    int a;
    int b;
};

class B:public A//派生类B，父类为A
{
public:
    void display()//与基类A中虚函数同名，重新定义此函数，执行时执行此条函数
    {
        a=1;
        b=100;
        c=222;
        cout<<a<<endl<<b<<endl<<c<<endl;
```

```
    }
    //若在基类中不加virtual关键词，则执行的是基类的display()。
private:
    int c;
};

int main()
{
    A a1;
    B b1;
    A *p1=&a1;//指向基类的指针
    p1->display();
    p1=&b1;//指向派生类
    p1->display();//调用派生类中的display
    return 0;
}
```

运行结果：

```
1
100
1
100
222

Process returned 0 (0x0)   execution time : 0.015 s
Press any key to continue.
```

※**总结：虚函数的使用方法**：

①在基类中用virtual声明成员函数为虚函数，在类外定义虚函数时，不必再加virtual

②在派生类中重新定义此函数，函数名

2、静态关联与动态关联

3、声明虚函数的情况

使用虚函数时，有两点要注意：

①只能用virtual声明类的**成员函数**，把它作为虚函数，而不能将类外的**普通函数**声明为虚函数。因为虚函数的作用是允许在派生类中对基类的虚函数重新定义。

②一个成员函数被声明为虚函数后，在同一类族中的类就不能再定义一个非virtual的但与该虚函数具有相同的参数（包括个数和类型）和函数返回值类型的同名函数。

4、虚析构函数

如果用new运算符建立了一个临时对象，若基类中有析构函数，并且定义了一个指向该基类的指针变量。在程序用带指针参数的delete运算符撤销对象时，会发生一个情况：系统只执行基类的析构函数，不执行派生类的。

虚析构函数的定义

```
virtual ~Point()
{
    cout<<"executing Point destructor"<<endl;
}
```

三、纯虚函数与抽象类

1、纯虚函数

基类中不想定义虚函数的内容，就把这个虚函数定义为纯虚函数。

```
virtual float area() const=0;   //纯虚函数
```

声明的一般形式:

virtual 函数类型 函数名 (参数列表) = 0;

注意：①纯虚函数没有函数体 ②最后的 “=0”并不表示函数返回值为0，这是一个形式，告诉编译器这是纯虚函数。 ③后面应有分号。

2、抽象类

抽象类定义的目的是：作为基类去建立派生类。

3、应用