

第9章 统一接口 不同实现—多态性

1、多态性的基本概念

2、派生类对象替换基类对象

3、虚函数的定义

4、抽象类的定义

5、宠物类的设计

6、运算符重载

7、日期类对象判断大小

8、分数类对象运算符重载

抽象类

□ 类是对象的集合，类是从相似对象中抽取共性而得到的抽象数据类型

□ 将不用来声明对象（实例化）的类称为**抽象类**。只供继承

□ **纯虚函数**的定义如下：

`virtual 返回类型 函数名（参数表）=0`

具体实现只能在派生类中完成

□ **抽象类**又可以定义成：

至少包含一个纯虚函数的类

抽象类的使用要求

👉 抽象类不能实例化，即不声明对象

👉 抽象类只作为基类被继承

👉 可以定义抽象类的指针或引用

抽象类的设计举例

- 平面上的几何图形可以抽象定义为类，如矩形类、圆类、三角形类等
 - 将所有几何图形再加以抽象，定义为形状类
 - 将形状类定义为抽象类shape
 - 由于几何图形类中都包含求面积函数和求周长函数
 - 在shape类中，将函数area()和circumference()声明为纯虚函数
 - 再用shape类派生出矩形类和圆类，在派生类中具体定义相应的求面积与周长的函数
 - 通过抽象类的对象指针访问派生类对象，实现动态绑定
-

基类定义为抽象类

```
#include<iostream>  
#include<cmath>  
using namespace std;  
#define PI 3.1415926  
class Shape  
{  
public:  
    virtual double area()=0;  
    virtual double circumference()=0;  
};
```

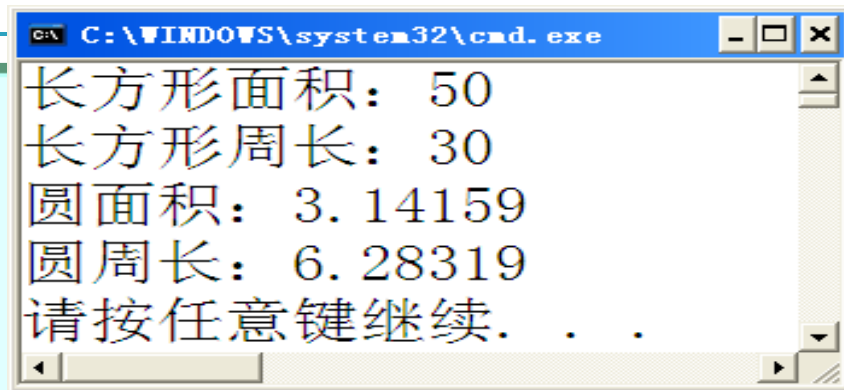
两个派生类中定义重载虚函数

```
class Rectangle:public Shape
{
    int x,y;
    int width,height;
public:
    Rectangle(int x,int y,int w,int h)
    {
        this->x=x;this->y=y;           width=w;height=h; }
    virtual double area()
    {
        return width*height;         }
    virtual double circumference()
    {
        return 2.0*(width+height);   }
};

class Circle:public Shape
{
    int x,y;
    int r;
public:
    Circle(int x,int y,int r)
    {
        this->x=x;this->y=y;this->r=r;   }
    virtual double area()
    {
        return PI*r*r;                 }
    virtual double circumference()
    {
        return 2.0*PI*r;               }
};
```

抽象类测试主函数

```
void main()  
{  
    Rectangle r1(10,10,10,5);  
    Circle c1(1,2,1);  
    Shape *p1=&r1,&p2=c1;  
    cout<<"长方形面积: " <<p1->area()<<endl;  
    cout<<"长方形周长: " <<p1->circumference()<<endl;  
    cout<<"圆面积: " <<p2.area()<<endl;  
    cout<<"圆周长: " <<p2.circumference()<<endl;  
}
```



A screenshot of a Windows command prompt window titled "C:\WINDOWS\system32\cmd.exe". The window displays the output of the program: "长方形面积: 50", "长方形周长: 30", "圆面积: 3.14159", "圆周长: 6.28319", and "请按任意键继续. . .".

将形状类shape进行调整

- 将派生类的点坐标移入基类定义
 - 意义：所有派生类图形的中心坐标点
 - 由于点坐标是私有成员，需要增加下列函数
 - 构造函数Shape()
 - 输出点坐标函数print()
 - 得到点坐标值函数getx(), gety()
-

基类增加坐标点成员，派生类不变

```
class Shape  
{  
  private:  
    int x,y;  
  public:  
    Shape(int xx,int yy):x(xx),y(yy){}  
    int getx(){return x;}  
    int gety(){return y;}  
    void print(){cout<< '['<<x<<','<<y<<']'<<endl;}  
    virtual double area()=0;  
    virtual double circumference()=0;  
};
```

感谢收看！