中国计量大学实验报告

实验课程：	面向对象程序设计	实验名称：	Virtual Functions and Polymorphism
班　　级：	15 计算机	学　　号：	1500303111
姓　　名：	**陈嘉豪**	实验日期：	5.11
教　　师：	**周永霞**	成　　绩：	

一.　实验题目：

1．设计抽象基类 Shape，要求有如下虚函数（请自己决定哪些作为纯虚函数）：
　1）分别计算周长和面积的 2 个函数 Circumference 和 Area
　2）用于输出形状名称的函数 ShapeName
　3）用于判断一个点是否在当前 Shape 里的函数 PointIsIn
2. 再设计如下派生类（这些派生类的基类请自己合理设计）：
　1）点 Point、圆 Circle、三角形 Triangle、矩形 Rectangle
　2）按需实现这些派生类的虚函数，其中 ShapeName 应该输出图形的名称+编号，比如 2 个圆对象的名称应该是 Circle1 和 Circle2。
　3）重载输入输出运算符，使之能够输出到文件，也能够从文件中读取数据
3. 生成若干个图形对象
　1）用 vector 创建各种图形对象的数组
　2）图形对象的部分数据用随机函数生成
　3）把图形对象的保存到文件中
　4）从文件中读取图形数据
4. 要用到标准 IO 流、文件 IO 流和字符串 IO 流

二.　　题目要求的实现情况

模块或功能	**子功能**	**是否实现**	**说明**
1 抽象基类 Shape	计算周长和面积的 2 个函数 Circumference 和 Area	√	
	ShapeName	√	
	PointIsIn	√	
2 派生类	点 Point、圆 Circle、三角形 Triangle、矩形 Rectangle	√	
	派生类的虚函数，其中 ShapeName 应该输出图形的名称+编号	√	
	重载输入输出运算符，使之能够输出到文件，也能够从文件中读取数据	√	
3. 生成若干个图形对象	用 vector 创建各种图形对象的数组	√	
	图形对象的部分数据用随机函数生成	√	
	把图形对象的保存到文件中	√	
	从文件中读取图形数据	√	
4. 要用到标准 IO 流、文件 IO 流和字符串 IO 流		√	

三.　　测试结果

输入	通过随机函数生成

输出	Point1:(4,2)
	Point2:(1,4.5)
	Point3:(0.5,1)
	
	Circle1:圆心:(3.5,2),半径:1.5
	Circumference=9.42
	Area=7.065
	Circle2:圆心:(3,1),半径:4
	Circumference=25.12
	Area=50.24
	Circle3:圆心:(3.5,2.5),半径:4.5
	Circumference=28.26
	Area=63.585
	
	Triangle1: 三个顶点坐标分别为:(7.33333,5.33333),(6,8),(2.66667,6.66667)

	Circumference=11.4249
	Area=5.33333
	Triangle2: 三个顶点坐标分别为:(5.33333,0),(8.66667,1.33333),(0.666667,5.33333)
	Circumference=19.6211
	Area=12
	Triangle3: 三个顶点坐标分别为:(4.66667,4),(4.66667,8.66667),(9.33333,9.33333)
	Circumference=16.4675
	Area=10.8889
	
	Rectangle1:对角线上的两点坐标:(1.5,5.5),(3.5,5)
	Circumference=5
	Area=1
	Rectangle2:对角线上的两点坐标:(4,5),(3.5,3.5)
	Circumference=4
	Area=0.75
	Rectangle3:对角线上的两点坐标:(2,2.5),(5,5)
	Circumference=11
	Area=7.5

是否正确	正确
错误原因	

四.　分析与探讨

1. 测试结果分析:

多态性只能用于引用和指针。

先构造再析构，先构造的后析构，后构造的先析构。

2. 探讨:

三角形的面积、如何判断点与三角形位置等的算法不清楚，通过百度以及询问同学后解决。

3. 问题:

暂无。

五.　附录：源代码

```
header.h:
#include<iostream>
#include<string>
#include<time.h>
#include<vector>
#include<math.h>
#include<cstdlib>
#include<fstream>

using namespace std;

class Shape       //定义基类 Shape
{public:
     virtual float Circumference() const {return 0.0;}        //虚函数
     virtual float Area() const {return 0.0;}        //虚函数
     virtual void ShapeName() const{}       //纯虚函数
     virtual void PointIsIn() const{}       //纯虚函数
};

class Point:public Shape       //定义继承类 Point，继承于 Point
{public:
     void setPoint();     //随机构造函数
     virtual void ShapeName() const {cout<<"Point:";}        //对虚函数再定义
     friend ostream & operator << (ostream &,const Point&);       //重载运算符<<

 protected:
     float x,y;
};

class Circle:public Point      //定义继承类 Circle，继承于 Point
{public:
     void setCircle();        //随机构造函数
     virtual float Circumference() const;
     virtual float Area() const;
     virtual void ShapeName() const {cout<<"Circle:";}        //对虚函数再定义
     friend ostream & operator << (ostream &,const Circle&);      //重载运算符<<
     virtual void PointIsIn() const;

protected:
    float r;
};
```

```
class Triangle:public Point         //定义继承类 Triangle，继承于 Point
{public:
     void setTriangle();       //随机构造函数
     virtual float Circumference() const;
     virtual float Area() const;
     virtual void ShapeName() const {cout<<"Triangle:";}      //对虚函数再定义
     friend ostream & operator << (ostream &,const Triangle&);        //重载运算符<<
     virtual void PointIsIn() const;

 protected:
     float x1,x2,x3,y1,y2,y3;
};

class Rectangle:public Point        //定义继承类 Rectangle，继承于 Shape
{public:
     void setRectangle();      //随机构造函数
     virtual float Circumference() const;
     virtual float Area() const;
     virtual void ShapeName() const {cout<<"Rectangle:";}     //对虚函数再定义
     friend ostream & operator << (ostream &,const Rectangle&);       //重载运算符<<
     virtual void PointIsIn() const;

 protected:
    float x1,x2,y1,y2;
};

define.cpp:
#include"header.h"

using namespace std;


void Point::setPoint()     //随机生成点坐标
{
    x=(rand()%10+1)/2.0;
    y=(rand()%10+1)/2.0;
}
ostream &operator<<(ostream &output,const Point &b)
{
    output<<"("<<b.x<<","<<b.y<<")"<<endl;
    return output;
}

void Circle::setCircle()       //随机生成圆心坐标与半径
```

```
{
    x=(rand()%10+1)/2.0;
    y=(rand()%10+1)/2.0;
    r=(rand()%10+1)/2.0;
}
float Circle::Circumference() const     //计算周长
{
    return 2*3.14*r;
}
float Circle::Area() const     //计算面积
{
    return 3.14*r*r;
}
void Circle::PointIsIn() const     //判断点位置
{
    float a,b;
    cout<<"请输入点的坐标:"<<endl;
    cout<<"x=";
    cin>>a;
    cout<<"y=";
    cin>>b;

    if(sqrt((a-x)*(a-x)+(b-y)*(b-y))<r)
        cout<<"点在圆内"<<endl;
    else if(sqrt((a-x)*(a-x)+(b-y)*(b-y))==r)
        cout<<"点在圆上"<<endl;
    else
        cout<<"点在圆外"<<endl;
    cout<<endl;
}
ostream &operator<<(ostream &output,const Circle &b)
{
    output<<"圆心:("<<b.x<<","<<b.y<<"),半径:"<<b.r<<endl;
    return output;
}

void Triangle::setTriangle()       //随机生成三角形各顶点坐标
{
    x1=rand()%15/1.5;
    y1=rand()%15/1.5;
    x2=rand()%15/1.5;
    y2=rand()%15/1.5;
    x3=rand()%15/1.5;
    y3=rand()%15/1.5;
```

```
}
float Triangle::Circumference() const       //计算周长
{
    float m1,m2,m3;
    m1=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2));
    m2=sqrt((x1-x3)*(x1-x3)+(y1-y3)*(y1-y3));
    m3=sqrt((x2-x3)*(x2-x3)+(y2-y3)*(y2-y3));
    return m1+m2+m3;
}
float Triangle::Area() const       //计算面积
{
    float m;
    float m1,m2,m3;
    m1=sqrt((x1-x2)*(x1-x2)+(y1-y2)*(y1-y2));
    m2=sqrt((x1-x3)*(x1-x3)+(y1-y3)*(y1-y3));
    m3=sqrt((x2-x3)*(x2-x3)+(y2-y3)*(y2-y3));
    m=(m1+m2+m3)/2.0;
    return sqrt(m*(m-m1)*(m-m2)*(m-m3));
}
void Triangle::PointIsIn() const       //判断点位置
{
    float a,b;
    cout<<"请输入点的坐标:"<<endl;
    cout<<"x=";
    cin>>a;
    cout<<"y=";
    cin>>b;

    float s1,s2,s3,s;
    s1=(1/2)*(a*y2+x2*y3+x3*b-a*y3-x2*b-x3*y2);
    s2=(1/2)*(x1*b+a*y3+x3*y1-x1*y3-a*y1-x3*b);
    s3=(1/2)*(x1*y2+x2*b+a*y1-x1*b-x2*y1-a*y2);
    s=(1/2)*(x1*y2+x2*y3+x3*y1-x1*y3-x2*y1-x3*y2);

    if(s1+s2+s3 == s)
        cout<<"点在三角形内"<<endl;
    else
        cout<<"点不在三角形内"<<endl;
    cout<<endl;
}
ostream &operator<<(ostream &output,const Triangle &b)
{
    output<<"三个顶点坐标分别
为:"<<"("<<b.x1<<","<<b.y1<<"),"<<"("<<b.x2<<","<<b.y2<<"),"<<"("<<b.x3<<","<<b.y3<<")"
```

```
<<endl;
    return output;
}

void Rectangle::setRectangle()     //随机生成矩形对角线上两点坐标
{
    x1=(rand()%10+2)/2.0;
    y1=(rand()%10+2)/2.0;
    x2=(rand()%10+2)/2.0;
    y2=(rand()%10+2)/2.0;
}
float Rectangle::Circumference() const      //计算周长
{
    return 2*(abs(x2-x1)+abs(y2-y1));
}
float Rectangle::Area() const      //计算面积
{
    return (abs(x2-x1))*(abs(y2-y1));
}
void Rectangle::PointIsIn() const      //判断点位置
{
    float a,b;
    cout<<"请输入点的坐标:"<<endl;
    cout<<"x=";
    cin>>a;
    cout<<"y=";
    cin>>b;

    if((a>x1||a>x2)&&(a<x1||a<x2)&&(b>y1||b>y2)&&(b<y1||b<y2))
    {
        cout << "点在矩形内" << endl;
    }
    else
    {
        cout << "点不在矩形内" << endl;
    }
    cout<<endl;
}
ostream &operator<<(ostream &output,const Rectangle &b)
{
    output<<"对角线上的两点坐
标:"<<"("<<b.x1<<","<<b.y1<<"),"<<"("<<b.x2<<","<<b.y2<<")"<<endl;
    return output;
}
```

```
main.cpp:
#include"header.h"

using namespace std;

int main()
{
    cout << "1500303111     15 计算机 1 班      陈嘉豪" << endl;
    int i;
    srand(time(0));

    cout << 
"***************************************************************************" << endl;
    vector<Point>p(3);          //用 vector 生成三组数据
    for(i=0;i<3;i++)
    {
        Shape *pt;
        pt=&p[i];
        p[i].ShapeName();       //静态关联
        p[i].setPoint();        //静态关联
        cout<<i+1<<":";
        cout<<p[i];
    }
    cout << 
"***************************************************************************" << endl;
    vector<Circle>s(3);         //用 vector 生成三组数据
    vector<float>Circle_Area(3);        //用 vector 生成三组数据
    vector<float>Circle_Circumference(3);       //用 vector 生成三组数据
    for(i=0;i<3;i++)
    {
        Shape *pt;
        pt=&s[i];
        s[i].ShapeName();       //静态关联
        s[i].setCircle();       //静态关联
        cout<<i+1<<":"<<s[i];
        cout<<"周长:"<<pt->Circumference()<<endl;        //用指针建立动态关联
        cout<<"面积:"<<pt->Area()<<endl;         //用指针建立动态关联
        Circle_Area[i]=pt->Area();      //用指针建立动态关联
        Circle_Circumference[i]=pt->Circumference();     //用指针建立动态关联
        pt->PointIsIn();        //用指针建立动态关联
    }
    cout << 
"***************************************************************************" << endl;
```

```
    vector<Triangle>t(3);      //用 vector 生成三组数据
    vector<float>Triangle_Area(3);      //用 vector 生成三组数据
    vector<float>Triangle_Circumference(3);      //用 vector 生成三组数据
    for (i=0;i<3;i++)
    {
        Shape *pt;
        pt = &t[i];
        t[i].ShapeName();      //静态关联
        t[i].setTriangle();        //静态关联
        cout<<i+1<<":"<<t[i];
        cout<<"周长:"<<pt->Circumference()<<endl;        //用指针建立动态关联
        cout<<"面积:"<<pt->Area()<<endl;        //用指针建立动态关联
        Triangle_Area[i]=pt->Area();        //用指针建立动态关联
        Triangle_Circumference[i]=pt->Circumference();        //用指针建立动态关联
        pt->PointIsIn();       //用指针建立动态关联
    }
    cout <<
"****************************************************************************" << endl;
    vector<Rectangle>q(3);     //用 vector 生成三组数据
    vector<float>Rectangle_Area(3);          //用 vector 生成三组数据
    vector<float>Rectangle_Circumference(3);         //用 vector 生成三组数据
    for(i=0;i<3;i++)
    {
        Shape *pt;
        pt=&q[i];
        q[i].ShapeName();      //静态关联
        q[i].setRectangle();       //静态关联
        cout<<i+1<<":"<<q[i];
        cout<<"周长:"<<pt->Circumference()<<endl;        //用指针建立动态关联
        cout<<"面积:"<<pt->Area()<<endl;        //用指针建立动态关联
        Rectangle_Area[i]=pt->Area();        //用指针建立动态关联
        Rectangle_Circumference[i] = pt->Circumference();         //用指针建立动态关联
        pt->PointIsIn();       //用指针建立动态关联
    }
    cout <<
"****************************************************************************" << endl;

    fstream outfile;
    outfile.open("1500303111.txt", ios::out);
    if(!outfile.is_open())
    {
        cout<<"open error"<<endl;
        return 0;
    }
    for(i=0;i<3;i++)
    {
        outfile<<"Point"<<i+1<<":"<<p[i]<<endl;
    }
    outfile<<endl<<endl;
    for(i=0;i<3;i++)
    {

    outfile<<"Circle"<<i+1<<":"<<s[i]<<"Circumference="<<Circle_Circumference[i]<<endl<
<"Area="<<Circle_Area[i]<<endl;
    }
    outfile << endl << endl;
    for (i = 0; i < 3; i++)
    {

    outfile<<"Triangle"<<i+1<<":"<<t[i]<<"Circumference="<<Triangle_Circumference[i]<<e
ndl<<"Area="<<Triangle_Area[i]<<endl;
    }
    outfile<<endl<<endl;
    for(i=0;i<3;i++)
    {

    outfile<<"Rectangle"<<i+1<<":"<<q[i]<<"Circumference="<<Rectangle_Circumference[i]<
<endl<<"Area="<<Rectangle_Area[i]<<endl;
    }
    outfile.close();
    cout<<endl<<"从文件中读取信息"<<endl;
    fstream fin("1500303111.txt",ios::in);
    vector<string>str(3*14);
    for (i=0;i<(3*14);i++)
    {
        fin>>str[i];
        cout<<str[i]<<endl;
    }

    system("pause");      //用于解决运行窗口闪退
    return 0;
}
```