

面向对象方法程序实例

– 对象成员

【例2-23】设计一个Point类，再定义一个Circle类，Circle类对象的圆心属性是Point类对象。要求：

- ①根据需要，合理地设计描述Point类和Circle类的属性和方法。
- ②合理地设计类成员的访问控制方式。
- ③考虑如何初始化Point类对象和Circle类对象。
- ④考虑如何输出Circle类对象的属性信息。
- ⑤用主函数测试类。
- ⑥要求用多文件结构实现程序。

类名	Point	
	含义	C++描述
属性	X坐标 Y坐标	private: int m_nX; private: int m_nY;
方法	构造函数 析构函数	public: Point(int nX, int nY); public: ~Point();
友元	Circle类	friend class Circle;

类名	Circle	
	含义	C++描述
属性	圆心 半径	private: Point m_pointCenter; private: int m_nR;
方法	构造函数 析构函数 显示属性信息	public: Circle(int nX, int nY, int nR); public: ~Circle(); public: void displayInfo();


```
// Point.h
class Point
{
private:
    int m_nX;
    int m_nY;
public:
    Point(int nX, int nY);
    ~Point();
    friend class Circle;
};
```

```
// Point.cpp
#include "Point.h"
#include <iostream>
using namespace std;

Point::Point(int nX, int nY)
{
    m_nX = nX;
    m_nY = nY;
    cout<<"Point类构造函数被调用！"<<endl;
}

Point::~~Point()
{
    cout<<"Point类析构函数被调用！"<<endl;
}
```

```
// Circle.h
#include "Point.h"
class Circle
{
private:
    Point m_pointCenter;
    int m_nR;
public:
    Circle(int nX, int nY, int nR);
    ~Circle();
    void displayInfo();
};
```



```
// Circle.cpp
#include "Circle.h"
#include <iostream>
using namespace std;
Circle::Circle(int nX, int nY, int nR) : m_pointCenter(nX, nY)
{
    m_nR = nR;
    cout<<"Circle类构造函数被调用！"<<endl;
}
Circle::~Circle()
{
    cout<<"Circle类析构函数被调用！"<<endl;
}
void Circle::displayInfo()
{
    cout<<"圆心为：("<<m_pointCenter.m_nX<<","<<m_pointCenter.m_nY<<")"<<endl
    <<"半径为："<<m_nR<<endl;
}
```

```
// testCircle.cpp  
#include "Circle.h"
```

```
int main()  
{  
    Circle c(10, 15, 5);  
    c.displayInfo();  
    return 0;  
}
```

运行结果：

Point类构造函数被调用！

Circle类构造函数被调用！

圆心为：(10,15)

半径为：5

Circle类析构函数被调用！

Point类析构函数被调用！