c++类的对象作为参数传递时，有三种传递方式

```
#include <iostream>

using namespace std;

//求圆的面积
class Mycircle{

public:
    double r;
    double s;
public:
    double getR(){
        return r;
    }
    void setR(double a){
        r = a;

    }
    double getS(){
        s = 3.14*r*r;
        return s;
    }


};
//类做函数参数有三种方法
/*第一种，最常用（这种适用于普通类型变量传值，当对象作为函数参数时候用引用形式，即第二种，

void printCircle(Mycircle mc){
    cout<<"半径是"<<mc.getR()<<endl;
}

*/

//第二种，引用方式（与第一种的调用方法一样，直接传参数，这是当传递的参数是一个对象时使用引
void printCircle(Mycircle &mc){
    cout<<"半径是"<<mc.getR()<<endl;
}
//第三种，指针（需要传入一个地址）
void printCircle(Mycircle *mc){
    cout<<"半径是"<<mc->getR()<<endl;
    /*不能使用mc.getR()或mc.r,与结构体不一样（结构体既可以A.B,也可以A->B,等价的），但
    A.B,对象指针调用方法是A->B。
    简单来说，“->”的前面一定是一个“指向结构体的指针”或“对象指针”,后面是结构或对象的一个
    如有:A->B，则可以肯定A是一个结构体指针或类的对象指针，而B是A中的一个成员。
    类封装了变量和方法，更丰富
    */
}
```

```cpp
int main()
{
    Mycircle c;
    c.setR(2.4);
    cout << "面积是"<<c.getS() << endl;
    printCircle(&c);//指针方式
    printCircle(c);//引用方式
    //printCircle(c);//常用方式,这里注释掉，因为测试会与引用方式重载冲突
    return 0;
}
```