**1.**

#include <iostream>

using namespace std;

static const double PI = 3.14159265359;

class Circle

{

public:

struct POINT

{

double x;

double y;

};

Circle(double \_x, double \_y, double \_radium);

Circle(Circle& c) : centre(c.centre), radium(c.radium) {}

void Set(double \_New\_x, double \_New\_y, double \_New\_radium);

void Print();

POINT Centre() { return centre; }

double Radium() { return radium; }

double Area() { return PI \* radium \* radium; }

~Circle() { cout << "Destructor called!" << endl; }

private:

POINT centre;

double radium;

};

Circle::Circle(double \_x, double \_y, double \_radium)

{

centre.x = \_x;

centre.y = \_y;

radium = \_radium;

}

void Circle::Set(double \_New\_x, double \_New\_y, double \_New\_radium)

{

centre.x = \_New\_x;

centre.y = \_New\_y;

radium = \_New\_radium;

}

void Circle::Print()

{

cout << "Centre: (" << centre.x << ", " << centre.y << ") Radium: " << radium << endl;

}

int main()

{

Circle c1(2.0, 2.0, 4.0);

Circle c2(c1);

c1.Print();

c2.Print();

cout << c1.Area() << " " << c2.Area() << endl;

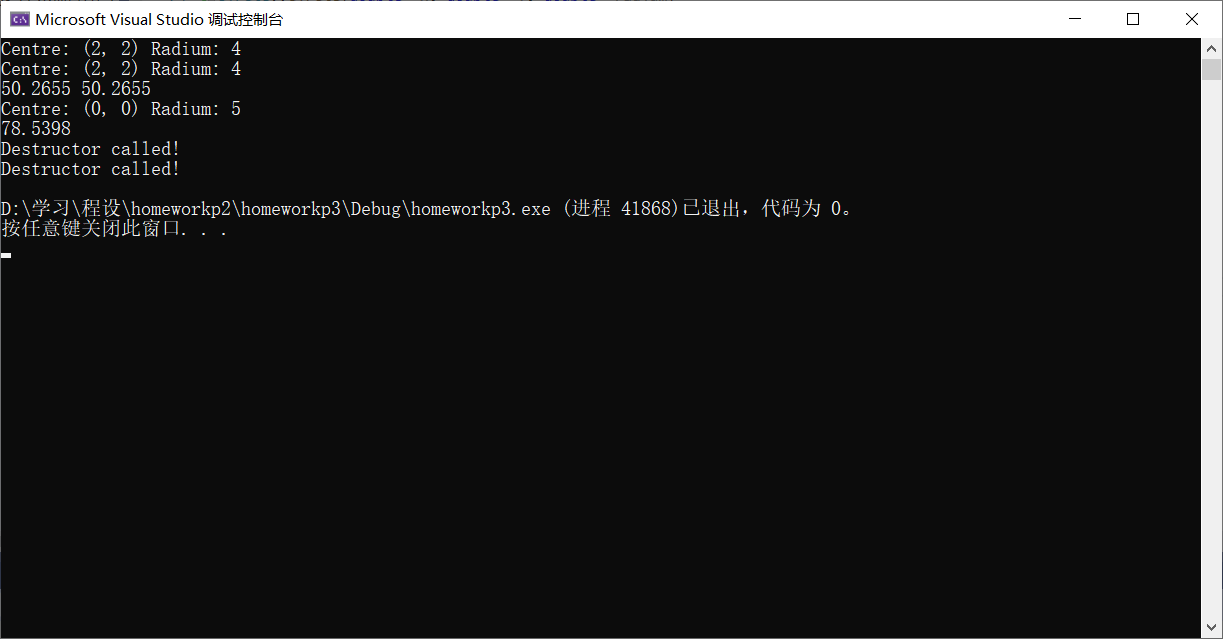
c2.Set(0.0, 0.0, 5.0);

c2.Print();

cout << c2.Area() << endl;

return 0;

}



**2.**

#include <iostream>

using namespace std;

inline double Abs(double x) { return x < 0 ? -x : x; }

class POINT

{

public:

double x;

double y;

POINT(double \_x, double \_y) { x = \_x; y = \_y; }

POINT() { x = y = 0; }

POINT(POINT& newpoint) { x = newpoint.x; y = newpoint.y; }

};

class Rectangle

{

private:

POINT leftbottom;

POINT righttop;

public:

Rectangle() {}

Rectangle(double left, double bottom, double right, double top) : leftbottom(left, bottom), righttop(right, top) {}

Rectangle(Rectangle& newrec) : leftbottom(newrec.leftbottom), righttop(newrec.righttop) {}

void Set(double left, double bottom, double right, double top)

{

leftbottom.x = left; leftbottom.y = bottom;

righttop.x = right; righttop.y = top;

}

void Print()

{

cout << "The left bottom corner is (" << leftbottom.x << ", " << leftbottom.y << ")";

cout << " and the right top corner is (" << righttop.x << ", " << righttop.y << ")." << endl;

}

double Area() { return Abs(righttop.x - leftbottom.x) \* Abs(righttop.y - leftbottom.y); }

double left() { return leftbottom.x; }

double bottom() { return leftbottom.y; }

double right() { return righttop.x; }

double top() { return righttop.y; }

~Rectangle() { cout << "Destructor called!" << endl; }

};

int main()

{

Rectangle r1;

r1.Set(0, 0, 4, 5);

r1.Print();

cout << r1.Area() << endl;

Rectangle r2(r1);

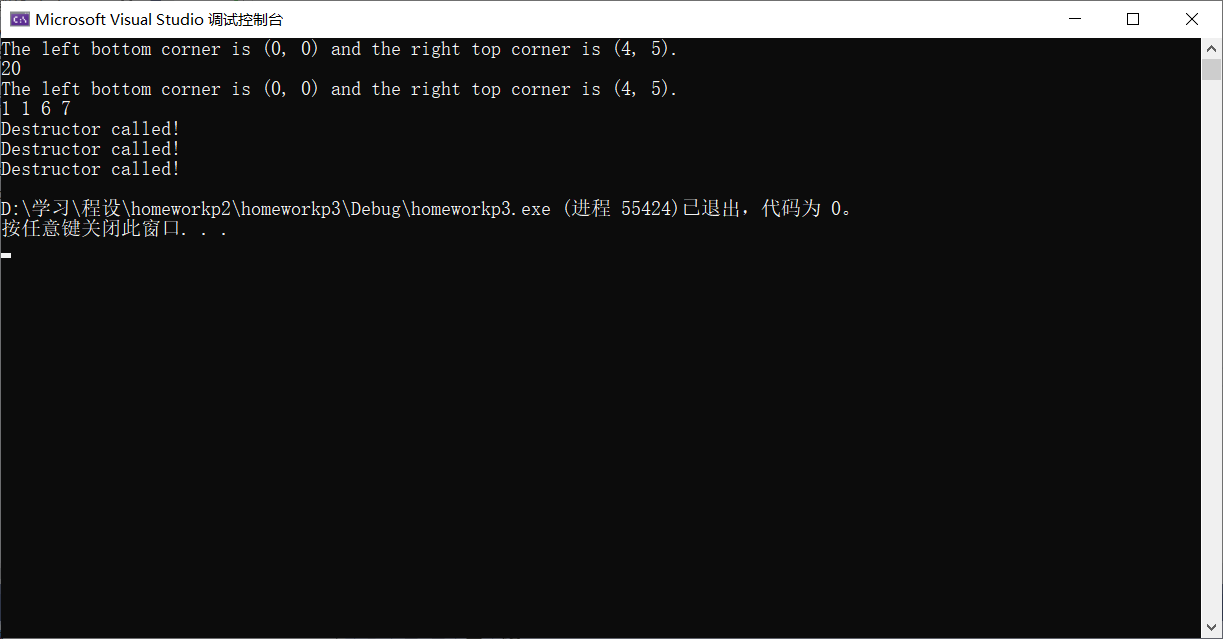
r2.Print();

Rectangle r3(1, 1, 6, 7);

cout << r3.left() << " " << r3.bottom() << " " << r3.right() << " " << r3.top() << endl;

return 0;

}



**3.（探究题）**

#include <iostream>

using namespace std;

class Car;

class Boat

{

private:

double weight;

public:

Boat() { weight = 0; }

Boat(double \_weight) { weight = \_weight; }

void Set(double w) { weight = w > 0 ? w : 0; }//Weight must be non-negative!

void Print() { cout << "Weight: " << weight << endl; }

double Weight() { return weight; }

friend double TotalWeight(Boat B, Car C);

};

class Car

{

private:

double weight;

public:

Car() { weight = 0; }

Car(double \_weight) { weight = \_weight; }

void Set(double w) { weight = w > 0 ? w : 0; }//Weight must be non-negative!

void Print() { cout << "Weight: " << weight << endl; }

double Weight() { return weight; }

friend double TotalWeight(Boat B, Car C);

};

double TotalWeight(Boat B, Car C)

{

return B.weight + C.weight;

}

int main()

{

Boat b;

Car c(7.0);

cout << b.Weight() << " " << c.Weight() << endl;

b.Set(6.0);

cout << TotalWeight(b, c) << endl;

b.Print();

c.Print();

return 0;

}

