**计算机程序设计基础（C++)**

**实验报告**

专业班级：软工2204班

学 号：8209220420

姓 名：赵洪锐

**实验报告成绩：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验** | **实验一** | **实验二** | **实验三** | **实验四** | **实验五** | **总评** |
| **成绩** |  |  |  |  |  |  |

**批阅教师：**

**实验五**

**类与对象**

**【实验目的与要求】**

1、掌握声明类的方法，类和类的成员的概念以及定义对象的方法。

2、初步掌握用类和对象编制基于对象的程序。

3、学习检查和调试基于对象的程序。

**【实验内容】**

1、有以下程序：

#include<iostream>

using namespace std；

class Time

// 定义 Time 类

{

public：

// 数据成员为公用的

int hour;

int minute;

int sec ;

};

int main()

{

Time tl;

//定义 t1 为 Time 类对象

cin>>t1.hour;

//输入设定的时间

cin>>t1.minute;

cin>>t1.sec;

cout<<t1.hourl<<”：”<<t1.minute<<”：”<<t1.sec<<endl ;

return 0;

}

改写程序，要求：

(1)将数据成员改为私有的；

(2)将输入和输出的功能改为由成员函数实现；

(3)在类体内定义成员函数。

然后编译和运行程序。请分析什么成员应指定为公用的?什么成员应指定为私有的? 什么函

数最好放在类中定义? 什么函数最好在类外定义?

2、分别给出如下的 3 个文件：

(1)含类定义的头文件 student.h，

//student.h

(这是头文件，在此文件中进行类的声明)

class Student

//类声明

{

public:

//公用成员函数原型声明

void display();

private:

int num;

char name[20];

char sex;

};

(2)包含成员函数定义的源文件 student.cpp

//student.cpp

在此文件中进行函数的定义

#include <iostream>

#include”student.h”

//不要漏写此行，否则编译通不过

void Student::display()

//在类外定义 display 类函数

{

cout<<”num：”<<num<<endl;

cout<<”name：”<<name<<endl;

cout<<”sex：”<<sex<<endl;

}

(3)包含主函数的源文件 main.cpp。

为了组成一个完整的源程序，应当有包括主函数的源文件：

//main.cpp

主函数模块

#include <iostream>

//将类声明头文件包含进来

#include “student.h”

int main()

{

Student stud;

//定义对象

Student stud1(007,”tcg”,’m’);

stud.display();

//执行 stud 对象的 display 函数

return 0;

}

请完善该程序，在类中增加一个对数据成员赋初值的成员函数 set\_value。上机调试并运行。

3、需要求 3 个长方柱的体积，请编一个基于对象的程序。数据成员包括 length(长)、width(宽)、

height(高)。要求用成员函数实现以下功能：

(1)由键盘分别输入 3 个长方柱的长、宽、高；

(2)计算长方柱的体积；

(3)输出 3 个长方柱的体积。

请编程序，上机调试并运行。

4、编写程序：定义抽象基类 Shape，由它派生出五个派生类：Circle（圆形）、Square（正

方形）、Rectangle（ 长方形）、Trapezoid （梯形）和 Triangle （三角形），用虚函数分别计算

各种图形的面积，并求出它们的和。要求用基类指针数组。使它的每一个元素指向一个派生类的

对象。

注：主函数中定义如下对象

Circle circle(10.2);

Square square(3);

Rectangle rectangle(3,4);

Trapezoid trapezoid(2.0,4.5,3);

Triangle triangle(4,5,6);5、设计一个函数，返回两个向量的交集，函数头如下：

template<typename T>

vector<T> intersection(const vector<T>& v1, const vector<T>& v2)

两个向量的交际为它们的共同包含的元素，编写一个测试程序，提示用户输入两个向量，每

个包含 5 个字符串，打印输出它们的交集。

1. 程序设计

1.

#include<iostream>

using namespace std;

class Time // 定义 Time 类

{

private:// 数据成员为公用的

int hour;

int minute;

int sec;

public:

void setHour()

{

cout << "请输入hour：";

cin >> this->hour;

}

void setMinute()

{

cout << "请输入minute：";

cin >> this->minute;

}

void setSec()

{

cout << "请输入second：";

cin>>this->sec;

}

void showTime()

{

cout << this->hour << ":" << this->minute << ":" << this->sec << endl;

}

};

int main()

{

Time t1; //定义 t1 为 Time 类对象

//cin >> t1.hour; //输入设定的时间

//cin >> t1.minute;

//cin >> t1.sec;

//cout << t1.hourl << ":" << t1.minute <<":" << t1.sec << endl;

t1.setHour();

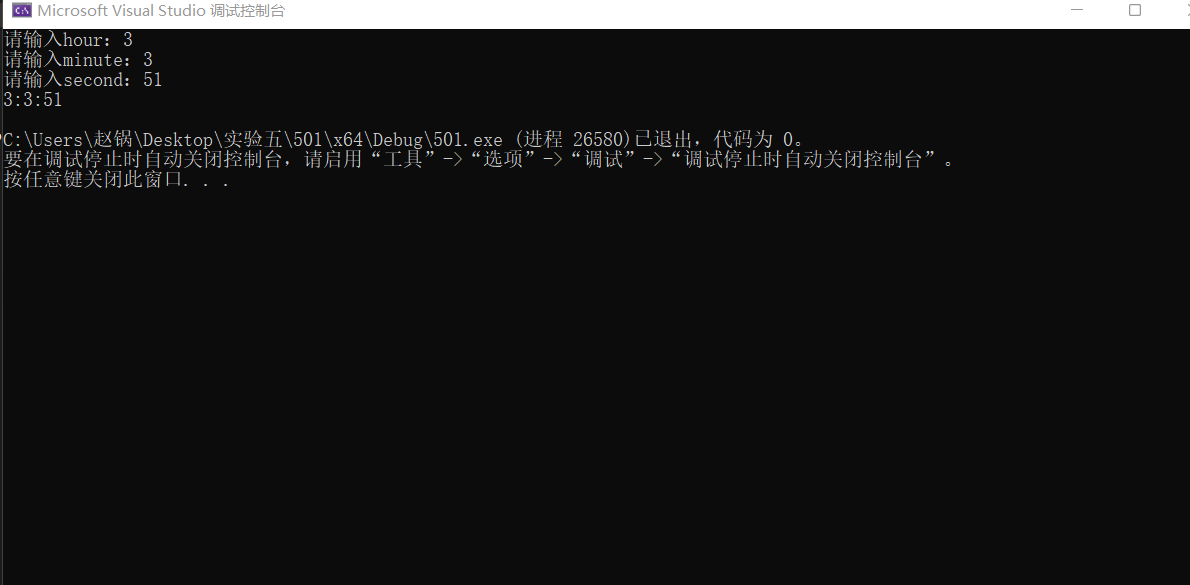
t1.setMinute();

t1.setSec();

t1.showTime();

return 0;

结果：



2.

//student.cpp

#include <iostream>

#include"student.h" //不要漏写此行，否则编译通不过

using namespace std;

void Student::display() //在类外定义 display 类函数

{

cout <<"num:" << num << endl;

cout << "name" << name << endl;

cout << "sex" << sex << endl;

}

void Student::set\_value(int num, string name, char sex)

{

this->num = num;

this->sex = sex;

this->name = name;

}

Student::Student(int num, const string name, char sex) {

this->num = num;

this->sex = sex;

this->name = name;

this->name = name;

}

Student::Student(){}

#pragma once

//student.h

using namespace std;

//student.h (这是头文件，在此文件中进行类的声明)

class Student //类声明

{

public: //公用成员函数原型声明

void display();

void set\_value(int num, string name, char sex);

Student(int num, string name, char sex);

Student();

private:

int num;

string name;

char sex;

};

//main.cpp 主函数模块

#include <iostream> //将类声明头文件包含进来

#include "student.h"

int main()

{

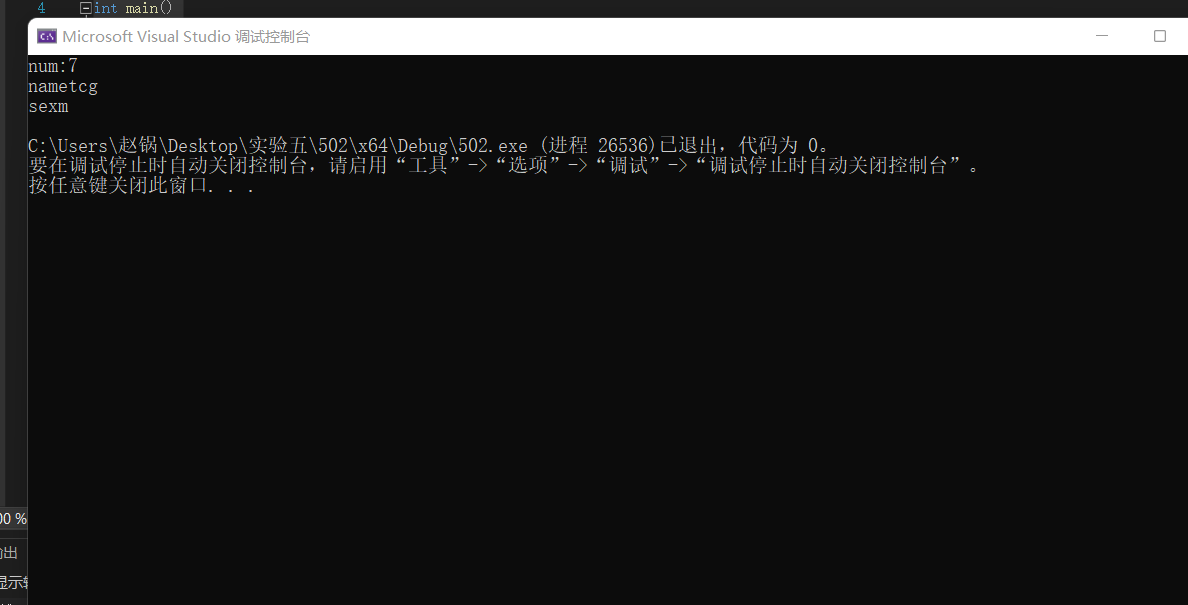
Student stud; //定义对象

Student stud1(007, "tcg", 'm');

stud1.display(); //执行 stud 对象的 display 函数

return 0;

}

结果：

3.

#include<iostream>

using namespace std;

class Square

{

private:

double length;

double width;

double height;

public:

void set\_length()

{

cout << "请输入长方体的长：";

cin >> this->length;

}

void set\_width()

{

cout << "请输入长方体的宽：";

cin >> this->width;

}

void set\_height()

{

cout << "请输入长方体的高：";

cin >> this->height;

}

void get\_area()

{

cout << "该长方体的体积为：";

cout << this->height \* this->length \* this->width << endl;;

}

};

int main()

{

Square square1;

Square square2;

Square square3;

cout << "下面请输入square1的数据：" << endl;

square1.set\_length();

square1.set\_height();

square1.set\_width();

square1.get\_area();

cout << "下面请输入square2的数据：" << endl;

square2.set\_length();

square2.set\_height();

square2.set\_width();

square2.get\_area();

cout << "下面请输入square3的数据：" << endl;

square3.set\_length();

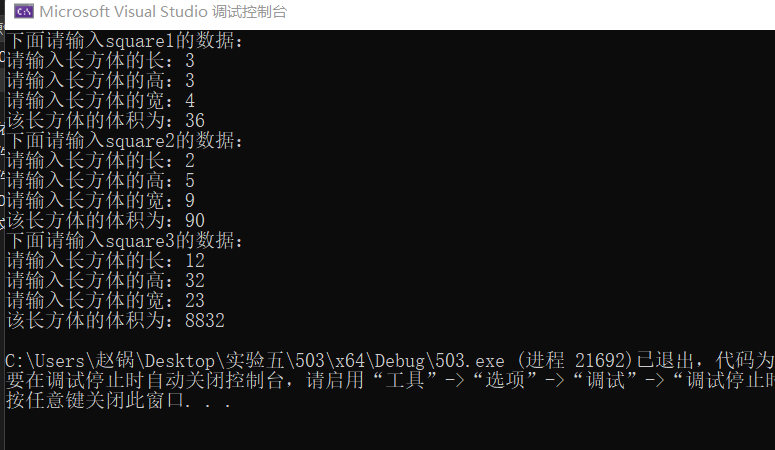
square3.set\_height();

square3.set\_width();

square3.get\_area();

}

结果：



4.

#include<iostream>

#include<string>

#include<cmath>

using namespace std;

class Shape

{

public:

virtual double caculate\_area()=0;

};

class Circle:public Shape

{

private:

double m\_r;

public:

Circle(double r )

{

m\_r = r;

}

double caculate\_area()

{

return 3.14 \* m\_r \* m\_r;

};

};

class Square:public Shape

{

private:

double a;

public:

Square(double a)

{

this->a = a;

}

double caculate\_area()

{

return a \* a;

}

};

class Rectangle:public Shape

{

private:

double a;

double b;

public:

Rectangle(double a, double b)

{

this->a = a;

this->b = b;

}

double caculate\_area()

{

return a\*b;

}

};

class Trapezoid : public Shape

{

private:

double a;

double b;

double h;

public:

Trapezoid(double a, double b, double h)

{

this->a = a;

this->b = b;

this->h = h;

}

double caculate\_area()

{

return (a + b) \* h / 2;

}

};

class Triangle:public Shape

{

private:

double a;

double b;

double c;

public:

Triangle(double a,double b,double c)

{

this->a = a;

this->b = b;

this->c = c;

}

double caculate\_area()

{

double p;

p = (a + b + c) / 2;

double S = sqrt(p \* (p - a) \* (p - b) \* (p - c));

return S;

}

};

int main()

{

Shape\* p[5];

Circle circle(10.2);

Square square(3);

Rectangle rectangle(3, 4);

Trapezoid trapezoid(2.0, 4.5, 3);

Triangle triangle(4, 5, 6);

p[0] = &circle;

p[1] = &square;

p[2] = &rectangle;

p[3] = &trapezoid;

p[4] = &triangle;

cout << "circle(10.2):" << p[0]->caculate\_area() << endl;

cout << "square(3):" << p[1]->caculate\_area() << endl;

cout << "rectangle(3,4)" << p[2]->caculate\_area() << endl;

cout << "trapezoid(2.0, 4.5, 3):" << p[3]->caculate\_area() << endl;

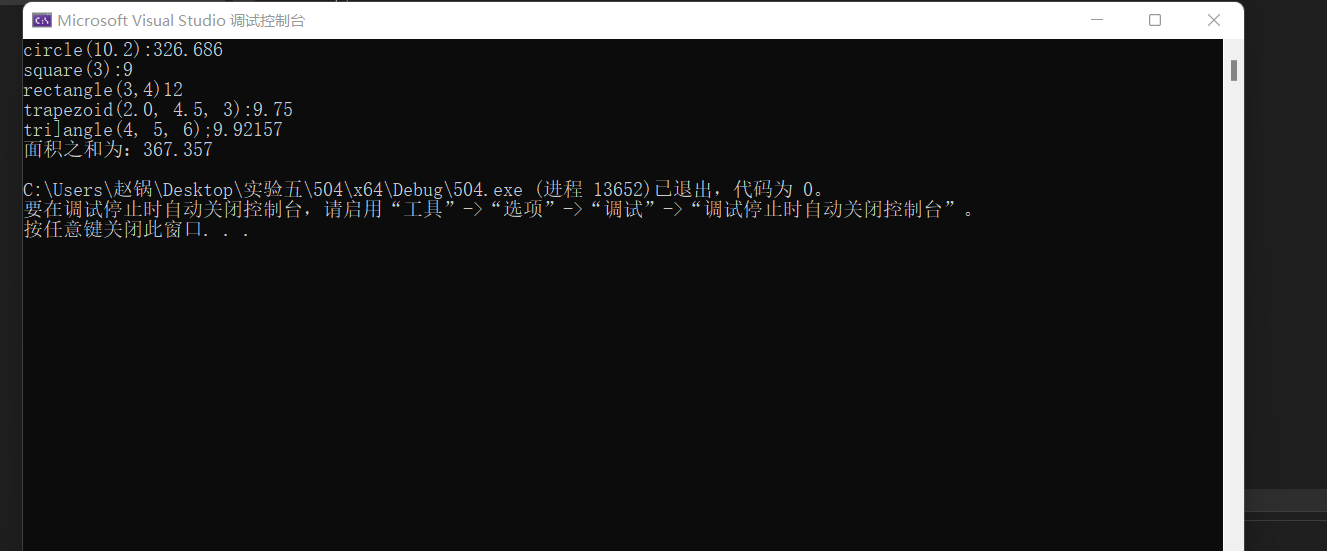
cout << "tri]angle(4, 5, 6);" << p[4]->caculate\_area() << endl;

cout << "面积之和为：" << circle.caculate\_area() + square.caculate\_area() + rectangle.caculate\_area()

+ trapezoid.caculate\_area() +triangle.caculate\_area() << endl;

}

结果：



5.

#include<iostream>

#include<vector>

#include<algorithm>

#include<string>

using namespace std;

template<class T>

vector<T>intersection( const vector<T>& v1, const vector<T>& v2)

{

vector<T>temp;

for (class vector<T>::const\_iterator it1 = v1.begin(); it1 != v1.end(); it1++)

{

for (class vector<T>::const\_iterator it2 = v2.begin(); it2 != v2.end(); it2++)

{

if (\*it1 == \*it2)

{

temp.push\_back(\*it1);

}

}

}

return temp;

}

void test()

{

vector<string>v1(5);

vector<string>v2(5);

cout << "请输入两个向量，每个向量有5个字符串" << endl;

cout << "第一个向量：" << endl;

for (vector<string>::iterator it = v1.begin(); it != v1.end(); it++)

{

cin >> \*it;

}

cout << "第二个向量：" << endl;

for (vector<string>::iterator it = v2.begin(); it != v2.end(); it++)

{

cin >> \*it;

}

cout << "这两个向量的交集是：";

vector<string>a= intersection(v1, v2);

for (vector<string>::iterator it = a.begin(); it != a.end(); it++)

{

cout << \*it << " ";

}

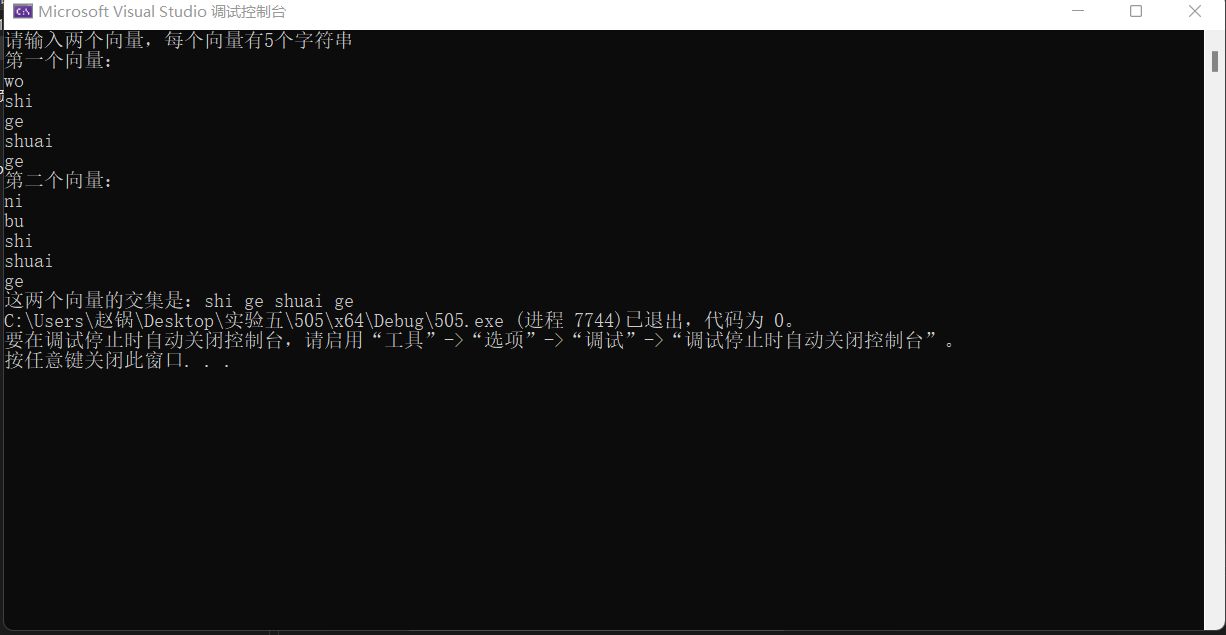
}

int main()

{

test();

}

结果：

四、遇到的问题与解决办法

编写最后一个程序时，忘记了vector容器的一些用法，花了些时间复习后才能开始编写；在遍历两个容器时，我把vector的迭代器的定义写在了for循环的外面，v2只能遍历一轮，导致了一系列错误，之后我断点一步一步溯源错误才找到错误并改正

五、体会

本次的实验设计对我而言是有一定难度的，花的时间大部分集中在寻找代码的错误上，我认为这是我的代码不规范造成的，今后一定要学习规范编写代码，以防出现一些很隐蔽的错误，使溯源错误所需的成本变小