**计算机程序设计基础（C++)**

**实验报告**

专业班级：软工2204班

学 号：8209220409

姓 名：陈思汗

**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;14

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。

**三、实验步骤、算法与结果分析**

**1.**

#include<iostream>

using namespace std;

class Time

{

private:

int hour;

int minute;

int sec;

public:

void settime()

{

int h, m, s;

cin >> h >> m >> s;

hour = h;

minute = m;

sec = s;

}

void showtime()

{

cout << hour << ":" << minute << ':' << sec << endl;

}

};

int main()

{

Time t1;

t1.settime();

t1.showtime();

}

//分析：数据成员应当被设为私有以控制属性不被随意更改；函数成员应被设置为公有来使用属性；

// 程序段较短的函数最好放在类中定义；程序段较长的函数最好放在类外定义；

**2.**

**Main.cpp:**

#include<iostream>

#include"student.h"

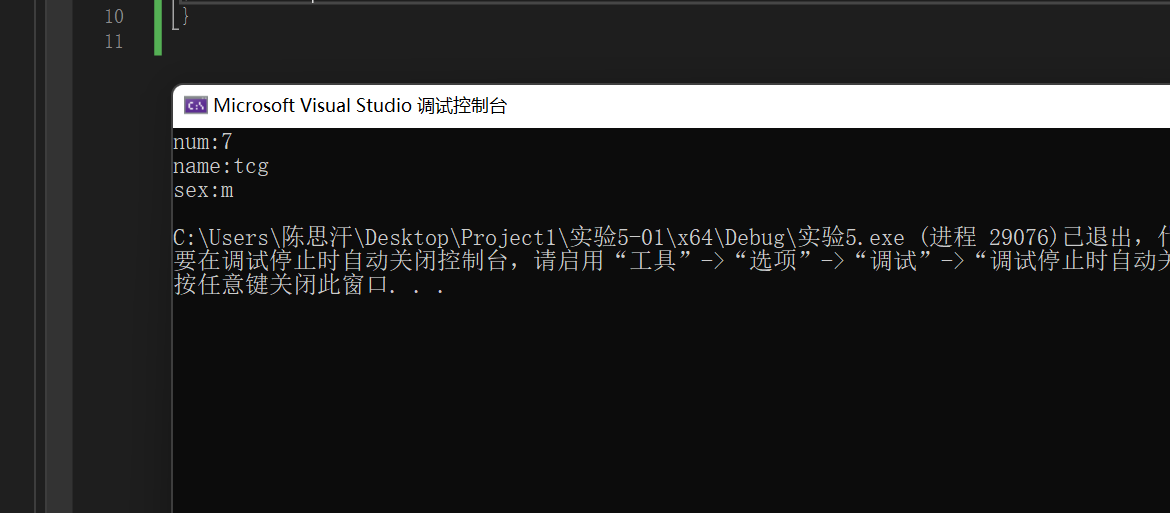
using namespace std;

int main()

{

Student stud;

stud.setvalue(007,"tcg",'m');

 stud.display();

return 0;

}

**Student.cpp:**

//student.cpp

#include<iostream>

#include"student.h"

using namespace std;

void Student::display()

{

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

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

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

}

void Student::setvalue(int n,const char N[],char s)

{

num = n;

for (int i = 0; i < 20; i++)

{

name[i] = N[i];

}

sex = s;

}

**Student.h:**

#pragma once

class Student

{

private:

int num;

char name[20];

char sex;

public:

void display();

void setvalue(int n,const char name[], char s);

};

**3.**

#include<iostream>

using namespace std;

class countvolume

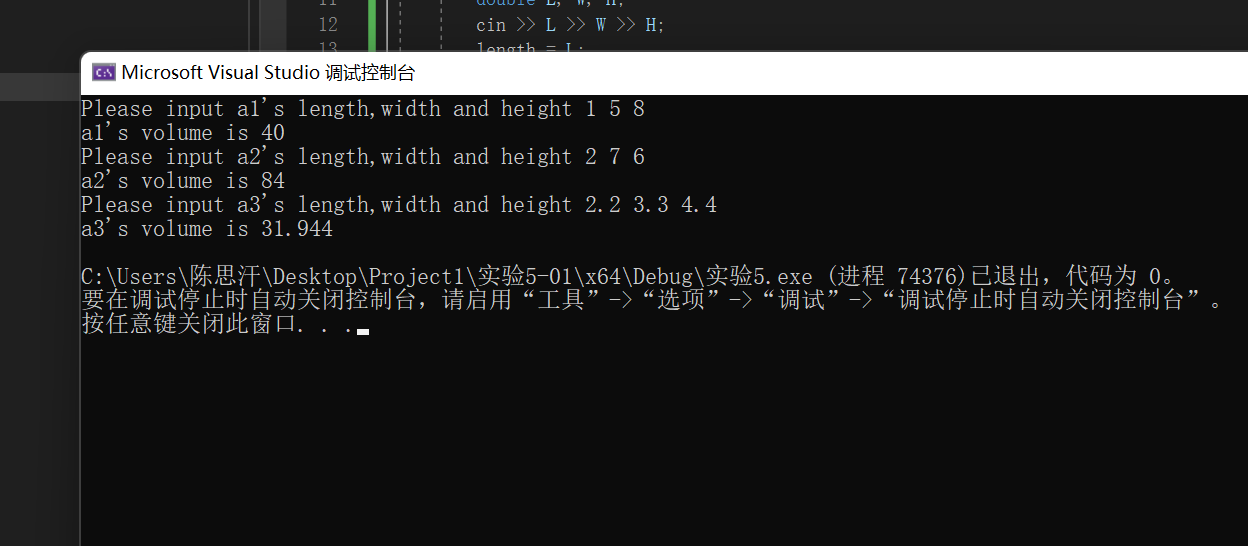
{

private:

double length, width, height;

double volume;

public:

 void setdata()

{

double L, W, H;

cin >> L >> W >> H;

length = L;

width = W;

height = H;

}

void count()

{

volume=length\*width\*height;

}

void showvolume()

{

cout << volume << endl;

}

};

int main()

{

countvolume a1, a2, a3;

cout << "Please input a1's length,width and height" << ' ';

a1.setdata();

a1.count();

cout << "a1's volume is" << ' ';

a1.showvolume();

cout << "Please input a2's length,width and height" << ' ';

a2.setdata();

a2.count();

cout << "a2's volume is" << ' ';

a2.showvolume();

cout << "Please input a3's length,width and height" << ' ';

a3.setdata();

a3.count();

cout << "a3's volume is" << ' ';

a3.showvolume();

return 0;

}

4.

**开方.h:**

#pragma once

#include<iostream>

using namespace std;

double squa(double a = 0)

{

double x = a;

double y = (x + a / x) / 2;

while ((y - x) > 1e-10 || (y - x) < -1e-10)

{

x = y;

y = (x + a / x) / 2;

}

return y;

}

**函数实现：**

#include<iostream>

#include"开方.h"

using namespace std;

const double PI = 3.14;

class Shape

{

public:

virtual double area() const = 0;

virtual void showarea() const = 0;

};

class Circle :public Shape

{

private:

double radius;

public:

Circle(double r = 0)

{

this->radius = r;

}

double area() const

{

return PI \* radius \* radius;

}

void showarea() const

{

cout << "此圆的面积为：" << ' ' << area() << endl;

}

};

class Square :public Shape

{

private:

double length;

public:

Square(double a = 0)

{

this->length = a;

}

double area() const

{

return length \* length;

}

void showarea() const

{

cout << "此正方形的面积为：" << ' ' << area() << endl;

}

};

class Retangle :public Shape

{

private:

double length;

double width;

public:

Retangle(double a = 0, double b = 0)

{

this->length = a;

this->width = b;

}

double area() const

{

return length \* width;

}

void showarea() const

{

cout << "此长方形的面积为：" << ' ' << area() << endl;

}

};

class Trapezoid :public Shape

{

private:

double upper;

double lower;

double height;

public:

Trapezoid(double a = 0, double b = 0, double h = 0)

{

this->upper = a;

this->lower = b;

this->height = h;

}

double area() const

{

return (upper + lower) \* height / 2;

}

void showarea() const

{

cout << "此梯形的面积为：" << ' ' << area() << endl;

}

};

class Triangle :public Shape

{

private:

double a, b, c;

public:

Triangle(double a = 0, double b = 0, double c = 0)

{

if ((a + b) < c || (a + c) < b || (b + c) < a)

{

cout << "这无法构成三角形" << endl;

}

else

{

this->a = a;

this->b = b;

this->c = c;

}

}

double area() const

{

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

return squa(p \* (p - a) \* (p - b) \* (p - c));

}

void showarea() const

{

cout << "此三角形的面积为：" << ' ' << area() << endl;

}

};

int main()

{

Circle circle(10.2);

Square square(3);

Retangle retangle(3, 4);

Trapezoid trapezoid(2.0, 4.5, 3);

Triangle triangle(4, 5, 6);

Shape\* shape[5];

shape[0] = new Circle(circle);

shape[1] = new Square(square);

shape[2] = new Retangle(retangle);

shape[3] = new Trapezoid(trapezoid);

shape[4] = new Triangle(triangle);

for (int i = 0; i < 5; i++)

{

shape[i]->showarea();

}

double total = 0;

for (int i = 0; i < 5; i++)

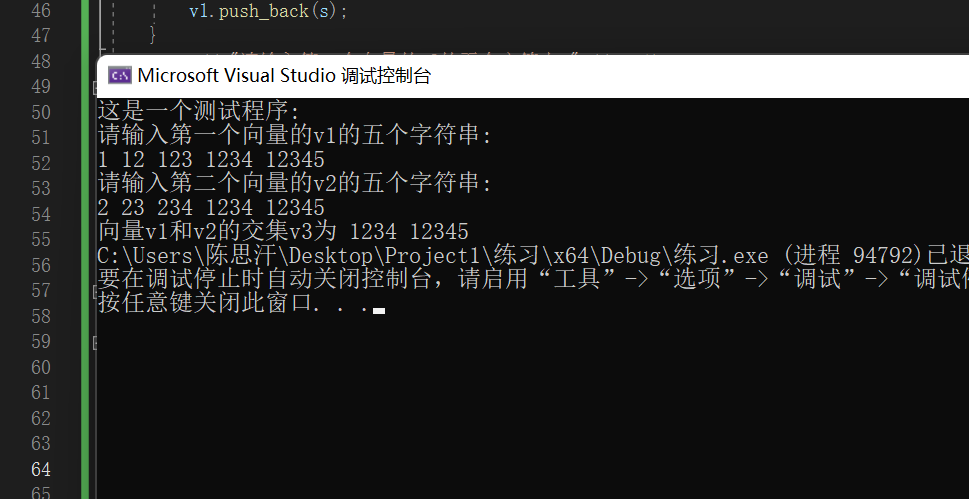
{

total += shape[i]->area();

}

cout << "总面积为:" << ' ' << total;

}

**5.**

#include<iostream>

#include<vector>

#include<string>

using namespace std;

template<typename T>

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

{

vector<T>v3;

for (int i = 0; i < v1.size(); i++)

{

for (int j = 0; j < v2.size(); j++)

{

if (v1[i] == v2[j])

{

if (v3.size() != 0)

{

int n = 0;//用于计数，以判断v3中是否已经有了这个元素

for (int m = 0; m < v3.size(); m++)

{

if (v3[m] == v1[i])

n++;

}

if (n == 0)

v3.push\_back(v1[i]);

}

else

v3.push\_back(v1[i]);

}

}

}

if (v3.size() == 0)

{

cout << "这两个向量没有交集!" << endl;

}

return v3;

}

int main()

{

cout << "这是一个测试程序:" << endl;

vector<string>v1, v2;

cout << "请输入第一个向量的v1的五个字符串:" << endl;

for (int i = 0; i < 5; i++)

{

string s;

cin >> s;

v1.push\_back(s);

}

cout << "请输入第二个向量的v2的五个字符串:" << endl;

for (int i = 0; i < 5; i++)

{

string s;

cin >> s;

v2.push\_back(s);

}

vector<string>v3 = intersection(v1, v2);

if (v3.size() != 0)

{

cout << "向量v1和v2的交集v3为" << ' ';

for (int i = 0; i < v3.size(); i++)

{

cout << v3[i] << ' ';

}

}

return 0;

}

**四、遇到的问题与解决办法**

基本没有遇到什么问题，主要是与关类的部分中一些规定及使用要求比较难记，所以在写程序的过程中会花掉一些时间去记忆。

**五、体会**

个人认为关于这部分内容的体会就重在于对面向对象这个思想的理解，从这一部分的内容中了解到了许多面向对象的思想，对它面向过程的编程的区别和这样编程的理由有了更深刻的认识！