# 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;

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);

15

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．实验结果如下

1. #include<iostream>

using namespace std;

class Time

{

private:

int hour;

int minute;

int sec;

public:

void shuru() {

cin >> hour >> minute >> sec;

}

void display() {

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

}

};

int main()

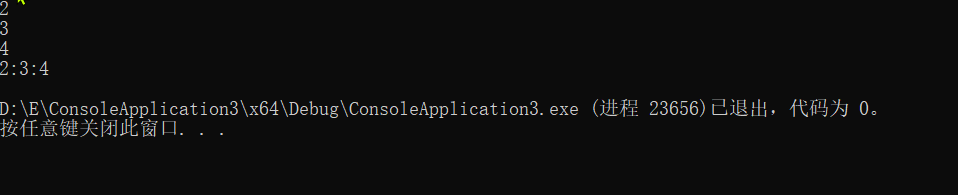
{

Time t1;//定义t1

t1.shuru();

t1.display();

return 0;



}

2. #include <iostream>

#include<string>

using namespace std;

class Student

{

private:

int num;

string Name;

char sex;

public:

void display();

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

this->num = num;

Name = name;

this->sex = sex;

}

Student(string name = "null", int num = 0, char sex = '0') {

this->num = num;

Name = name;

this->sex = sex;

}

};

void Student::display()

{

cout << "num：" << num << endl;

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

cout << "sex：" << sex << endl;

}

int main()

{

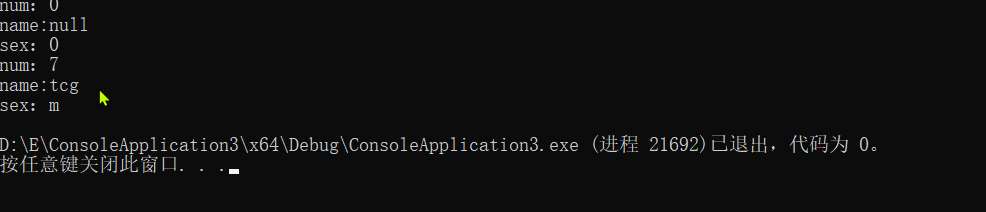
Student stud;

Student stud1("tcg", 7, 'm');

stud.display();

stud1.display();

return 0;

} 

3.

#include <iostream>

using namespace std;

class changfangti {

private:

int height;

int length;

int width;

public:

void set\_value() {

cin >> height >> length >> width;

}

void display() {

cout << height \* length \* width << endl;;

}

};

int main()

{

cout << "第一个长方体长宽高：";

changfangti t1;

t1.set\_value();

cout << "第二个长方体长宽高：";

changfangti t2;

t2.set\_value();

cout << "第三个长方体长宽高：";

changfangti t3;

t3.set\_value();

cout << "第一个长方体体积为：";

t1.display();

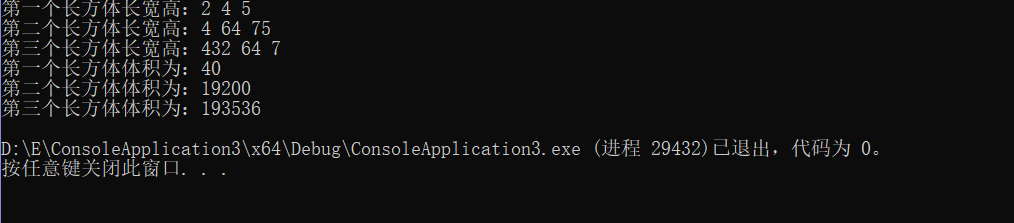
cout << "第二个长方体体积为：";

t2.display();

cout << "第三个长方体体积为：";

t3.display();

}



4. #include <iostream>

using namespace std;

/\*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。

\*/

class shape {

public:

virtual double area() const = 0;

virtual void show() = 0;

};

class Circle :public shape {

private:

double R;

public:

Circle(double R) {

this->R = R;

}

double area()const {

double Area;

Area = 3.14 \* R \* R;

return Area;

}

void show() {

cout << "圆面积为" << area() << endl;

}

};

class Square :public shape {

private:

int bianchang;

public:

Square(int Bianchang) {

this->bianchang = Bianchang;

}

double area()const {

double Area;

Area = bianchang \* bianchang;

return Area;

}

void show() {

cout << "正方形面积为" << area() << endl;

}

};

class Rectangle :public shape {

private:

int chang;

int kuan;

public:

Rectangle(int Chang, int Kuan) {

this->chang = Chang;

this->kuan = Kuan;

}

double area()const {

double Area;

Area = chang \* kuan;

return Area;

}

void show() {

cout << "长方形面积为" << area() << endl;

}

};

class Trapezoid :public shape {

private:

double shang;

double xia;

int h;

public:

Trapezoid(double Shang, double Xia, int H) {

this->shang = Shang;

this->xia = Xia;

this->h = H;

}

double area()const {

double Area;

Area = (shang + xia) \* h / 2;

return Area;

}

void show() {

cout << "梯形面积为" << area() << endl;

}

};

class Triangle :public shape {

private:

int bian1;

int bian2;

int bian3;

public:

Triangle(int Bian1, int Bian2, int Bian3) {

this->bian1 = Bian1;

this->bian2 = Bian2;

this->bian3 = Bian3;

}

double area()const {

double Area;

int p;

p = (bian1 + bian2 + bian3) / 2;

Area = sqrt(p \* (p - bian1) \* (p - bian2) \* (p - bian3));

return Area;

}

void show() {

cout << "三角形面积为" << area() << endl;

}

};

int main() {

shape\* area[5];

Circle circle(10.2);

Square square(3);

Rectangle rectangle(3, 4);

Trapezoid trapezoid(2.0, 4.5, 3);

Triangle triangle(4, 5, 6);

area[0] = &circle;

area[1] = &square;

area[2] = &rectangle;

area[3] = &trapezoid;

area[4] = &triangle;

double sum = 0;

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

{

area[i]->show();

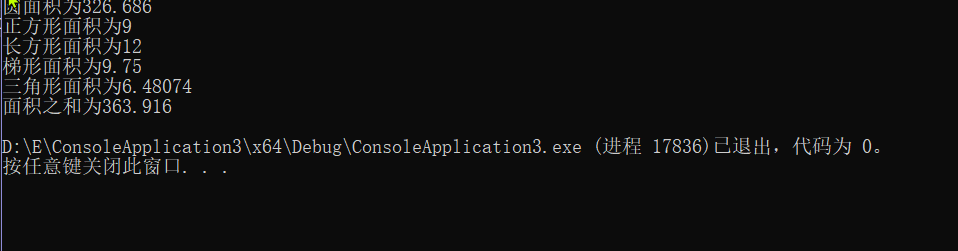
sum = sum + area[i]->area();

}

cout << "面积之和为" << sum << endl;

return 0;

}



5. #include<iostream>

#include<vector>

#include<algorithm>

#include<iterator>

using namespace std;

vector<int> intersection(vector<int>& v1, vector<int>& v2) {

vector<int>v;

sort(v1.begin(), v1.end());

sort(v2.begin(), v2.end());

set\_intersection(v1.begin(), v1.end(), v2.begin(), v2.end(), back\_inserter(v));

return v;

}

void print\_vector(vector<int> v) {

if (v.size() > 0) {

cout << "交集为：";

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

cout << v[i] << " ";

}

}

else {

cout << "交集为空集";

}

}

int main() {

vector<int>v1, v2, v;

cout << "输入第一个向量元素：" << endl;

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

int t = 0;

cin >> t;

v1.push\_back(t);

}

cout << "输入第二个向量元素：" << endl;

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

int t = 0;

cin >> t;

v2.push\_back(t);

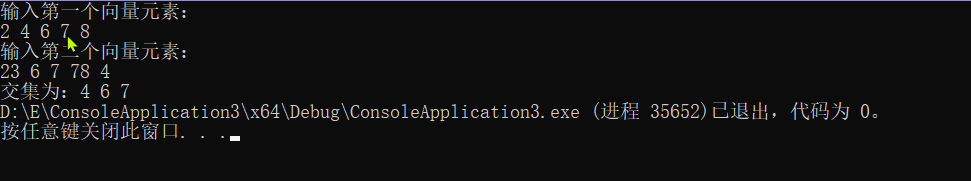
}

v = intersection(v1, v2);

print\_vector(v);

return 0;

}



遇到的问题与解决方案

我所遇到的问题在四和五，四在于不会使用visual stdio创建类文件，于是就把所有类放在一起，然后是虚函数的使用调用求面积的函数开始没用const修饰导致一直无法编译通过，便去掉了virtual后来网上查询要子类和父类都要加const修饰，第五题不会就网上找现成的函数避免多余操作发现set\_insertion

体会

C++学习除了课内知识外还要再课外补充相关习题学习编程思想，了解对应方法，重要是动手实践面对不会的要善于上网上查资料来弥补学习上的漏洞