**实验五 类与对象**

**一、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。

**二、实验内容**

**1、改写题目所给的程序，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?**

**程序：**

#include<iostream>

using namespace std;

class Time // 定义 Time 类

{

private: // 数据成员为私有的

int hour;

int minute;

int sec;

public:

void settime()

{

cout << "SetTime : hour minute second" << endl;

cin >> this->hour; //输入设定的时间

cin >> this->minute;

cin >> this->sec;

}

void showtime()

{

cout << "ShowTime : ";

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

}

};

int main()

{

Time t1; //定义 t1 为 Time 类对象

t1.settime();

t1.showtime();

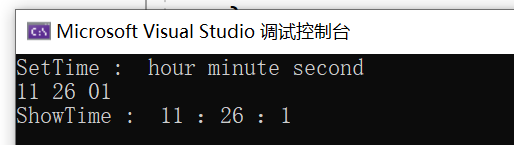
return 0;

}

**说明：**

1. **类内的数据成员;**
2. **可以调用私有的属性；**
3. **公有的方法或者类内仅作声明类外定义；**

**输出：**

****

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。

**程序：**

含类定义的头文件 student.h

#pragma once

#include <string>

using namespace std;

class Student //类声明

{

public: //公用成员函数原型声明

void display();

Student();

Student(int \_num,string \_name, char \_sex);

void set\_value();

private:

int num;

string name;

char sex;

};

包含成员函数定义的源文件 student.cpp

#include <iostream>

#include"student.h" //不要漏写此行，否则编译通不过

using namespace std;

Student::Student(){}

Student::Student(int \_num, string \_name, char \_sex)

{

this->num = \_num;

this->name = \_name;

this->sex = \_sex;

}

void Student::set\_value()

{

cout << "请输入学生的 num, name, sez: ";

cin >> this->num >> this->name >> this->sex;

}

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

{

cout << "num：" << num << endl;

cout << "name：" << name << endl;

cout << "sex：" << sex << endl;

}

包含主函数的源文件 main.cpp。

#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 stud1 对象的 display 函数

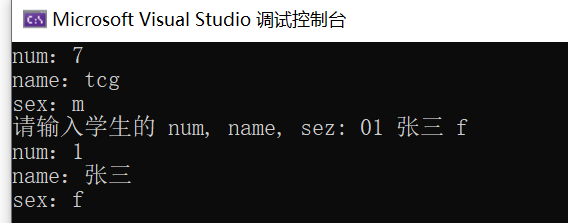
stud.set\_value();

stud.display();

return 0;

}

**输出：**

****

**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。**

**程序：**

#include <iostream>

using namespace std;

class Box

{

public:

void setValue();

float getVolume();

void display();

private:

float lengh;

float width;

float height;

};

void Box::setValue()

{

cout << "请输入长方体长, 宽, 高: " << endl;

cin >> this->lengh;

cin >> this->width;

cin >> this->height;

}

float Box::getVolume()

{

float volume;

volume = lengh \* width \* height;

return volume;

}

void Box::display()

{

cout << "该长方体体积是: " << this->getVolume() << endl;

}

void main()

{

Box box1,box2,box3;

box1.setValue();

box1.display();

box2.setValue();

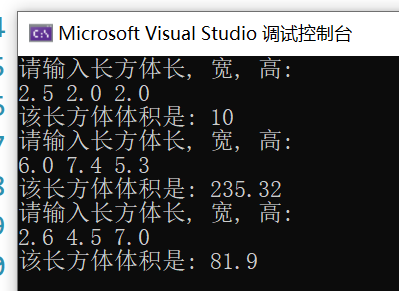
box2.display();

box3.setValue();

box3.display();

}

**输出：**

****

**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);**

**程序：**

#include <iostream>

#include <iomanip>

using namespace std;

class Shape //基类

{

public:

virtual double Area() = 0;

virtual void display() = 0;

};

class Rectangle :public Shape //矩形

{

public:

double a, b;

Rectangle(double \_a, double \_b)

{

this->a = \_a;

this->b = \_b;

}

virtual double Area()

{

return a \* b;

}

virtual void display()

{

cout << "矩形的面积为: " << this->Area() << endl;

}

};

class Circle :public Shape //圆

{

public:

double r;

Circle(double \_r)

{

this->r = \_r;

}

virtual double Area()

{

return 3.14 \* r \* r;

}

virtual void display()

{

cout << "圆的面积为: " << this->Area() << endl;

}

};

class Triangle :public Shape //三角

{

public:

double a, b ,c;

Triangle(double \_a, double \_b, double \_c)

{

this->a = \_a;

this->b = \_b;

this->c = \_c;

}

virtual double Area()

{

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

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

}

virtual void display()

{

cout << "三角形的面积为: " << this->Area() << endl;

}

};

class Square :public Shape //正方形

{

public:

double a;

Square(double \_a)

{

this->a = \_a;

}

virtual double Area()

{

return a \* a;

}

virtual void display()

{

cout << "正方形的面积为: " << this->Area() << endl;

}

};

class Trapezoid :public Shape //梯形

{

public:

double a, b, h;

Trapezoid(double \_a, double \_b, double \_h)

{

this->a = \_a;

this->b = \_b;

this->h = \_h;

}

virtual double Area()

{

return (a + b) \* h / 2.0;

}

virtual void display()

{

cout << "梯形的面积为: " << this->Area() << endl;

}

};

int main()

{

Circle circle(10.2);

Square square(3);

Rectangle rectangle(3, 4);

Trapezoid trapezoid(2.0, 4.5, 3);

Triangle triangle(4, 5, 6);

Shape\* pShape[5] = { &circle,&square,&rectangle,&trapezoid,&triangle };

double sum = 0;

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

{

pShape[i]->display();

sum += pShape[i]->Area();

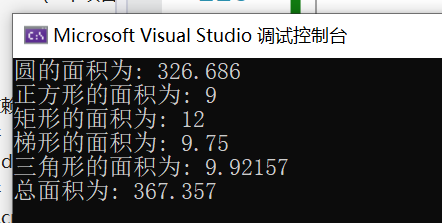
}

cout << "总面积为: " << sum << endl;

return 0;

}

**输出：**



**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。**

**程序：**

#include<iostream>

#include<vector>

using namespace std;

void print\_Vector(vector<string>& v) //vector打印(string)

{

if (v.size() != 0) //不为空

{

cout << "{ ";

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

{

cout << (\*it) << " ";

}

cout << "}" << endl;

}

else

{

cout << "{ }" << endl; //若为空打印空

}

}

template<typename T>

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

{

vector<T>v;

bool is = false;

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

{

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

{

if (v1.at(i) == v2.at(j))

{

if (v.size() != 0) //写入v

{

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

{

if (v1.at(i) == v.at(m)) //重复判断

{

break;

}

else

{

v.push\_back(v1.at(i));

}

}

}

else

{

v.push\_back(v1.at(i));

}

}

}

}

return v;

}

int main()

{

cout << "请输入第一个向量的数据: ";

vector<string>v1;

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

{

string complex;

cin >> complex;

v1.push\_back(complex);

}

cout << "请输入第二个向量的数据:" ;

vector<string>v2;

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

{

string complex;

cin >> complex;

v2.push\_back(complex);

}

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

cout << "v1和v2的交集为:";

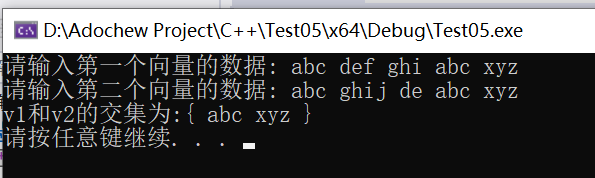
print\_Vector(v);

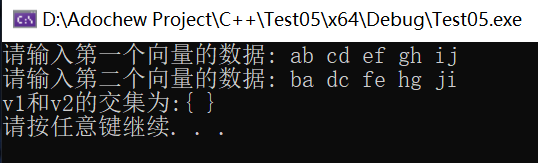
system("pause");

return 0;

}

**输出：**

****

****

**三、遇到的问题和解决方法**

**在第五题没有考虑重复元素的排除,在for遍历后的if判断是否重复,若重复则不插入,解决了该问题.同时vector相关函数不够熟练,需要加强训练.**

**四、心得与体会**

**通过本次实验我掌握声明了类的方法，类和类的成员的概念以及定义对象的方法.同时,初步了解用类和对象编制基于对象的程序,完成了本学期最后一次实验报告.回顾这一学期,我学习了面向对象的编程思想和cpp的程序设计基础,有收获有不足,希望在以后的学习中继续增长代码基础能力.**