实验报告

专业班级： 软件工程2205

学号：L209220532

姓名：丹尼

实验报告成绩：

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

批阅教师：

实验四

1. 实验目的

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。

1. 实验内容与要求
2. 有以下程序：

#include <iostream> **using namespace** std;  
  
**class** Time {  
**private**:  
 **int** hour;  
 **int** minute;  
 **int** sec;

}；

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;

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 “student.h”

#include <iostream>

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 vector intersection(const vector& v1, const vector& 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。

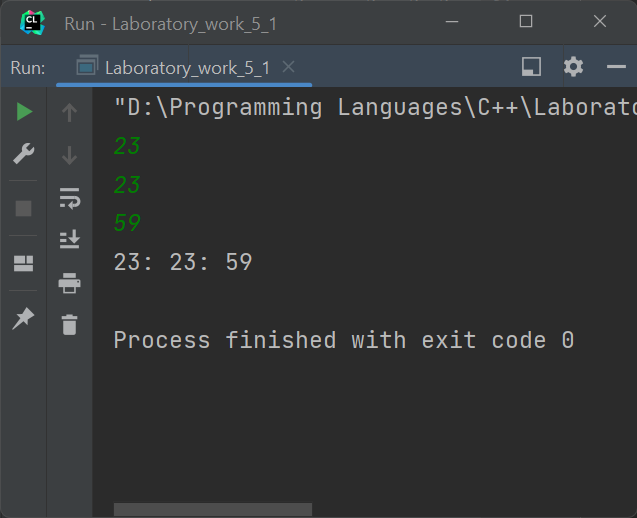
1. 实验步骤、算法与结果分析

501time.cpp

**程序代码**

#include **<iostream>  
using namespace** std;  
  
**class** Time {  
**private**:  
 **int** hour;  
 **int** minute;  
 **int** sec;  
**public**:  
 **int** getHour() **const** {  
 **return** hour;  
 }  
  
 **void** setHour(**int** hour) {  
 Time::hour = hour;  
 }  
  
 **int** getMinute() **const** {  
 **return** minute;  
 }  
  
 **void** setMinute(**int** minute) {  
 Time::minute = minute;  
 }  
  
 **int** getSec() **const** {  
 **return** sec;  
 }  
  
 **void** setSec(**int** sec) {  
 Time::sec = sec;  
 }  
};  
  
  
**int** main() {  
 Time t1;  
 **int** a;  
 cin>>a;  
 t1.setHour(a);  
 cin>>a;  
 t1.setMinute(a);  
 cin>>a;  
 t1.setSec(a);  
 cout << t1.getHour() << **": "** << t1.getMinute() << **": "** << t1.getSec() << endl;  
 **return** 0;  
}

**结果**

****

502student

**程序代码**

**main.cpp**

#include **<iostream>**#include **"student.h"  
int** main() {  
 Student stud;  
 Student stud1(007, **"tcg"**, **'m'**);  
 stud.display();  
 **return** 0;  
}

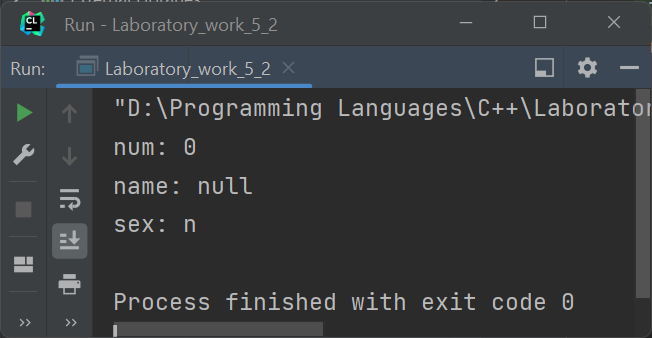
**student.cpp**

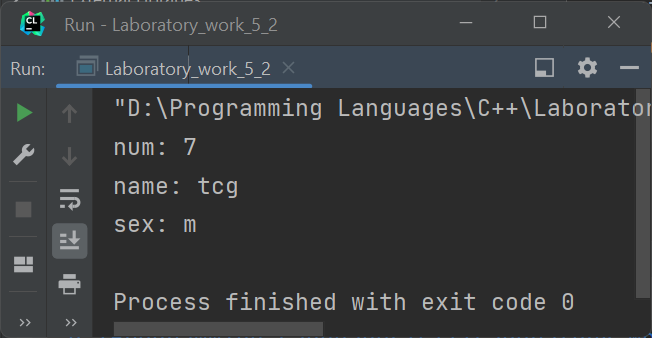
*//student.cpp*#include **<iostream>**#include **"student.h"  
  
void** Student::display() {  
 std::cout << **"num: "** << num << std::endl;  
 std::cout << **"name: "** << name << std::endl;  
 std::cout << **"sex: "** << sex << std::endl;  
}  
  
Student::Student(**int** num, **char** \*name, **char** sex) {  
 set\_value(num, name, sex);  
  
}  
  
Student::Student(){  
 set\_value();  
  
}

**student.h**

*//student.h***class** Student{  
**public**:  
 **void** display();  
 Student();  
 Student(**int** num, **char** \*name, **char** sex);  
  
**private**:  
 **int** num;  
 **char** name[20];  
 **char** sex;  
  
 **void** set\_value(**int** num = 0, **char** \*name = {**"null"**}, **char** sex = **'n'**){  
 **this**->num = num;  
 **this**->sex = sex;  
  
 **int** it = 0;  
 **while** (\*name != **'\0'**) {  
 **if** (it >= 20) **break**;  
 **this**->name[it] = \*name;  
 \*name++;  
 it++;  
 }  
 }  
};

**结果**

****

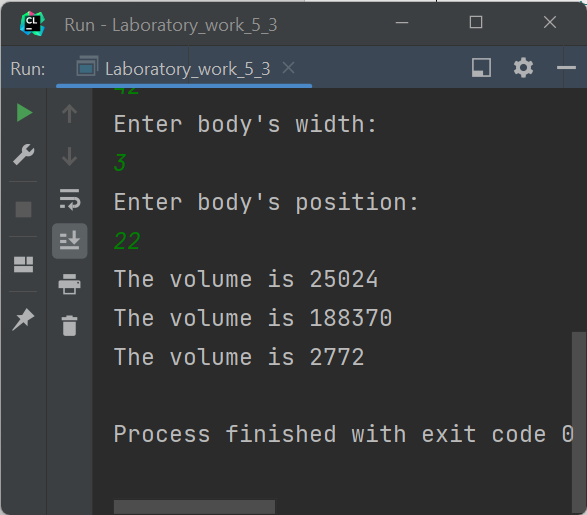
****

503bodyvolume.cpp

**程序代码**

#include **<iostream>  
  
  
class** Body {  
**public**:  
 Body(**int** length = 0, **int** width = 0, **int** height = 0) : length(abs(length)), width(abs(width)), height(abs(height)) {}  
  
 **int** getVolume() {  
 **return** length \* width \* height;  
 }  
  
**private**:  
 **int** length;  
 **int** width;  
 **int** height;  
};  
  
**enum** Parameters {  
 ***length***,  
 ***width***,  
 ***position***};  
  
  
  
**static** std::string stringBuild(Parameters s = ***length***){  
 **if** (s == ***length***) **return "Enter body's length: "**;  
 **else if** (s == ***width***) **return "Enter body's width: "**;  
 **else if** (s == ***position***) **return "Enter body's position: "**;  
}  
  
**static void** inputNumber(**int** &number, Parameters s){  
 std::cout << stringBuild(s) << std::endl;  
 std::cin >> number;  
 **while** (std::cin.fail()) {  
 std::cin.clear();  
 std::cin.ignore();  
 std::cout << stringBuild(s) << std::endl;  
 std::cin >> number;  
 };  
}  
  
**int** main() {  
  
 **constexpr int** bodyCount = 3;  
 Body array[bodyCount];  
  
  
 **for** (**int** it = 0; it < bodyCount; it++) {  
 **int** a, b, c;  
  
 inputNumber(a, ***length***);  
 inputNumber(b, ***width***);  
 inputNumber(c, ***position***);  
  
 array[it] = Body(a, b, c);  
  
 }  
 **for** (Body body : array) {  
 std::cout<<**"The volume is "**<<body.getVolume()<<std::endl;  
 }  
  
 **return** 0;  
}

**结果**

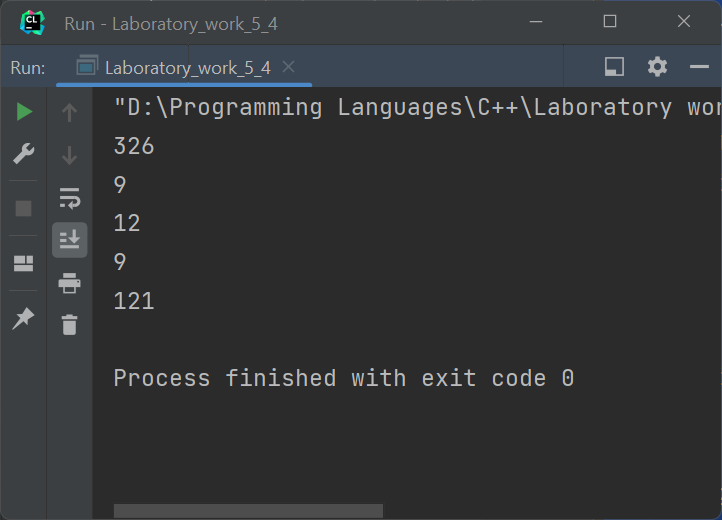
****

504shape.cpp

**程序代码**

#include **<iostream>**#include **<valarray>**#include **"math.h"  
  
class** Shape {  
**public**:  
 **virtual int** getArea() = 0;  
  
  
};  
  
**class** Circle : **public** Shape{  
  
**private**:  
 **double** radius;  
 **static constexpr double** pi = 3.14;  
**public**:  
  
  
  
 **explicit** Circle(**double** radius) : radius(radius) {}  
  
  
 **int** getArea() **override** {  
 **return** radius \* radius \* pi;  
 }  
};  
  
**class** Square : **public** Shape{  
  
**private**:  
 **double** side;  
**public**:  
  
 **explicit** Square(**double** side) : side(side) {}  
  
  
 **int** getArea() **override** {  
 **return** side \* side;  
 }  
};  
  
  
**class** Rectangle : **public** Shape{  
**private**:  
 **double** firstSide;  
 **double** secondSide;  
**public**:  
 Rectangle(**double** firstSide, **double** secondSide) : firstSide(firstSide), secondSide(secondSide) {}  
  
  
  
 **int** getArea() **override** {  
 **return** firstSide \* secondSide;  
 }  
};  
  
**class** Trapezoid : **public** Shape{  
**private**:  
 **double** firstSide;  
 **double** secondSide;  
 **double** height;  
  
**public**:  
  
 Trapezoid(**double** firstSide, **double** secondSide, **double** height) : firstSide(firstSide), secondSide(secondSide),  
 height(height) {}  
  
 **int** getArea() **override** {  
 **return** ((firstSide + secondSide) \* height) / 2;  
 }  
  
  
};  
  
**class** Triangle : **public** Shape {  
**private**:  
 **double** side1;  
 **double** side2;  
 **double** side3;  
  
**public**:  
 Triangle(**double** side1, **double** side2, **double** side3) : side1(side1), side2(side2), side3(side3) {}  
  
 **int** getArea() **override** {  
  
 **double** p = side1 + side2 + side3;  
  
 **return** sqrt(p \* (p - side1) \* (p - side2) \* (p - side3));  
 }  
  
};  
**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);  
  
 std::cout<<circle.getArea()<<std::endl;  
 std::cout<<square.getArea()<<std::endl;  
 std::cout<<rectangle.getArea()<<std::endl;  
 std::cout<<trapezoid.getArea()<<std::endl;  
 std::cout<<triangle.getArea()<<std::endl;  
  
 **return** 0;  
}

**结果**

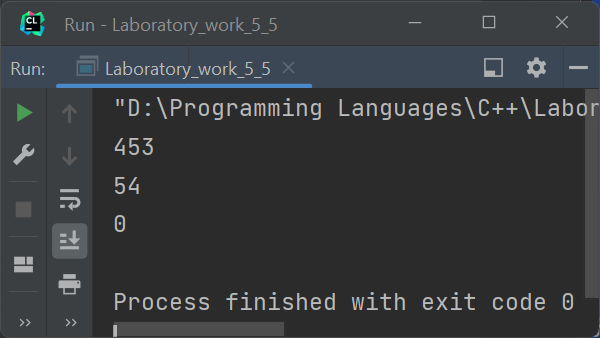
****

505template.cpp

**程序代码**

#include **<iostream>**#include **"vector"  
  
using** std::vector, std::string, std::cout;  
  
**template**<**typename** T>  
vector<T> intersection(**const** vector<T> &v1, **const** vector<T> &v2){  
 vector<T> v3;  
 **for** (**int** it = 0; it < v1.size(); it++){  
 **for** (**int** j = 0; j < v2.size(); j++){  
 **if** (v2[j] == v1[it]) v3.push\_back(v1[it]);  
 }  
 }  
 **return** v3;  
}  
  
**int** main() {  
  
 vector<string> s1 = **{"453"**, **"55"**, **"985"**, **"54"**, **"0"}**;  
 vector<string> s2 = **{"453"**, **"65"**, **"95"**, **"54"**, **"0"}**;  
  
 vector<string> s3 = intersection(s1, s2);  
  
 **for** (string it : s3){  
 cout<<it<<std::endl;  
 }  
  
 **return** 0;  
}

**结果**

****

四，遇到的问题与解决办法

实验过程中没有遇到任何问题。

五，体会

During this laboratory work I had a chance to practice working with classes in C++.

Classes are essential for object oriented programming, so it’s necessary for a programmer to know how to create a class, to call it and define. Classes make programs easier to build, read and rewrite. I really liked using classes in Java and in C++ too.