**实验五 类与对象**

专业班级： 软工6班

学 号： 8209230619

姓 名： 郑凯东

**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;

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、建立一个对象数组，内放5个学生的（学号，成绩），设立一个函数max，用指向对象的指针作函数参数，在max函数中找出5个学生的最高成绩者，并输出其学号。

5、设有一描述坐标点的类Point，其私有变量x和y代表一个点的(x,y)坐标值。请编写程序实现以下功能：利用构造函数传递参数，在定义对象时将x、y坐标值初始化为（60,80）；利用公有成员函数void setPoint(int i, int j)将坐标值修改为(60+i,80+j)；利用公有成员函数display()输出修改后的坐标值。主函数中通过定义对象，验证各个函数。

**【实验结果】**

**1.**

#include <iostream>

using namespace std;

class Time // 定义Time类

{

public: // 数据成员为公用的

  void setTime();

  void showTime();

private:

  int hour;

  int minute;

  int sec;

} t1;

void Time ::setTime()

{

  cin >> this->hour; // 输入设定的时间

  cin >> this->minute;

  cin >> this->sec;

}

void Time ::showTime()

{

  cout << this->hour << ':' << this->minute << ':' << this->sec << endl; // 输出时间

}

int main()

{

  Time tl; // 定义this为Time类对象

  t1.setTime();

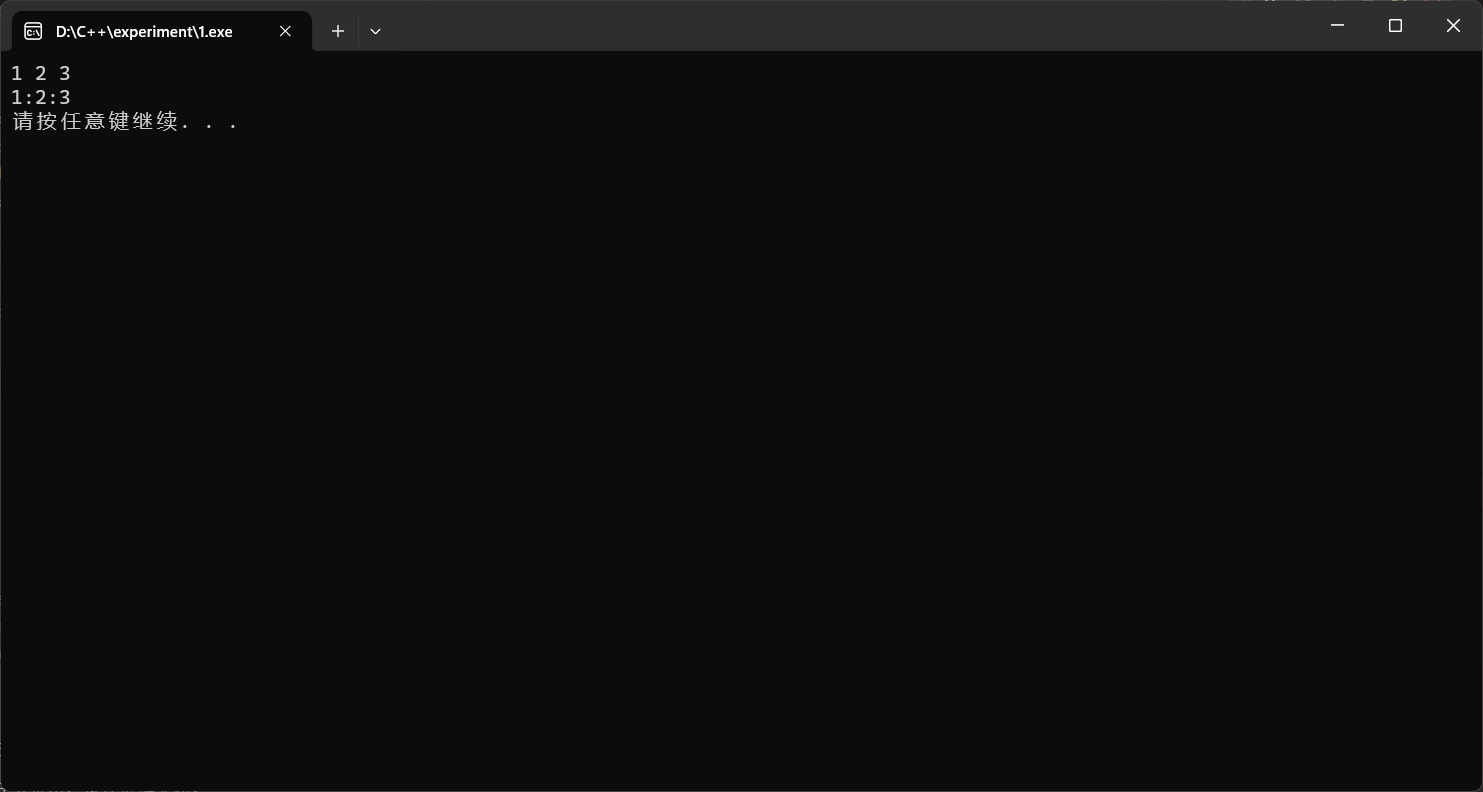
  t1.showTime();

  return 0;

}

（忘写）回答：获得数据，展现数据（get，set）函数，还有方法应该作为公有成员；

把类的属性作为私有成员，析构函数，构造函数应该在类内定义，友元函数，静态成员函数，成员函数应该在类外定义。

结果：

**2.**

//Student.h

#include <string>

using namespace std;

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);

private:

int num;

string name;

char sex;

};

//Student.cpp

#include <iostream>

#include <cstring>

#include "student.h"

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->name = Name;

this->sex = Sex;

}

//main.cpp

#include <iostream>

#include "student.h"

using namespace std;

int main()

{

Student stud; // 定义对象

string a = "tcg";

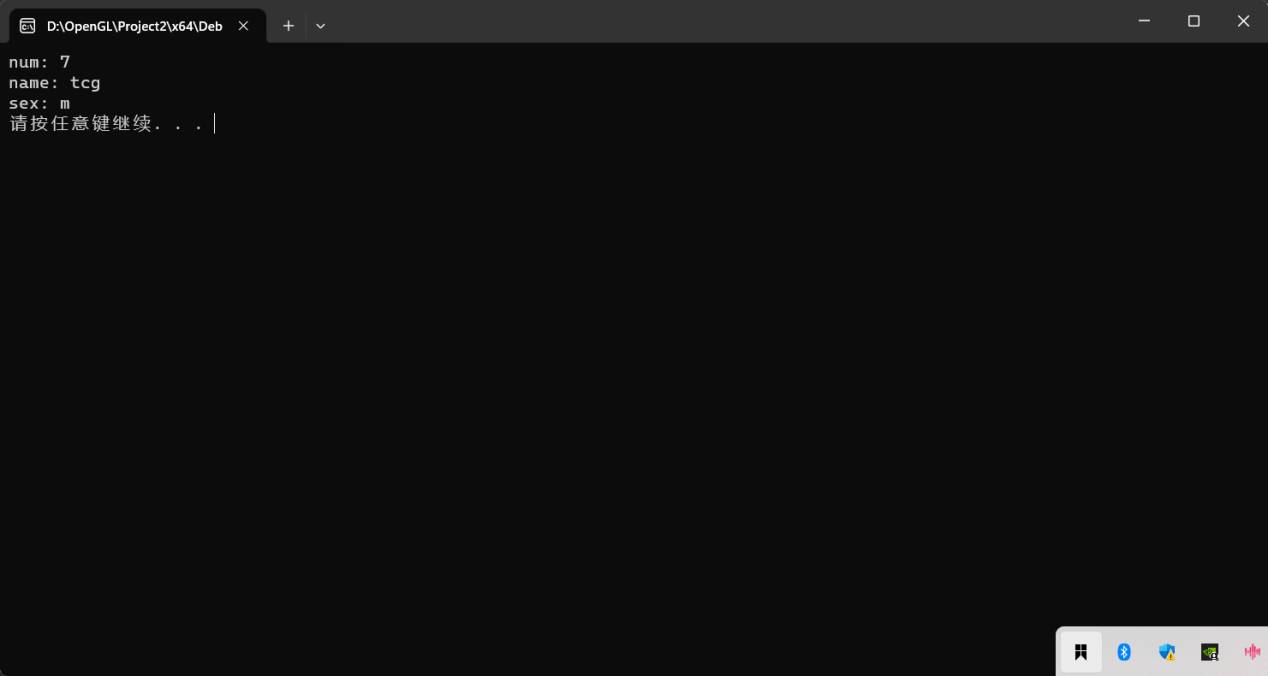
stud.set\_value(7, a, 'm');

stud.display(); // 执行stud对象的display函数

system("pause");

return 0;

}

结果：****

**3.**

#include <iostream>

using namespace std;

class Cuboid

{

private:

  int length;

  int width;

  int heigh;

  int volume;

  static int sumVolume;

public:

  Cuboid()

  {

    cin >> this->length >> this->width >> this->heigh;

  }

  void countVolume();

  static void showSumVolume()

  {

    cout << "the summary of all the cuboids is " << sumVolume << endl;

  }

};

void Cuboid::countVolume()

{

  this->volume = this->heigh \* this->width \* this->length;

  sumVolume += this->volume;

}

int Cuboid::sumVolume = 0;

int main()

{

  Cuboid c1, c2, c3;

  c1.countVolume();

  c2.countVolume();

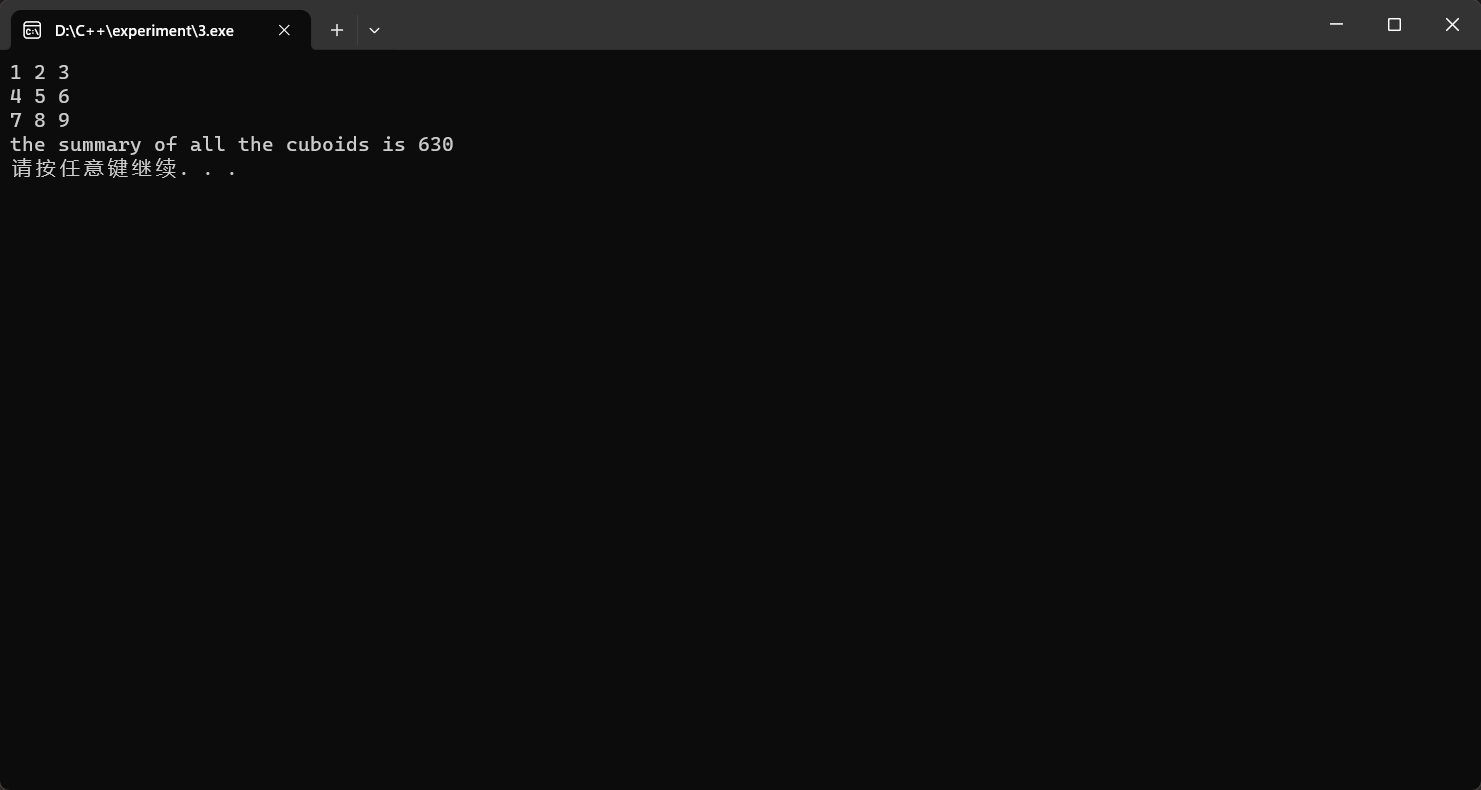
  c3.countVolume();

  Cuboid::showSumVolume();

  system("pause");

  return 0;

}

结果：

**4.**

#include <iostream>

using namespace std;

class Student

{

private:

  long long numb;

  int score;

  static int bestNumb;

  static int bestScore;

public:

  Student(int N, int S)

  {

    this->numb = N;

    this->score = S;

    if (this->score > bestScore)

    {

      bestNumb = this->numb;

      bestScore = this->score;

    }

  }

  static void showTheBestOne();

};

void Student ::showTheBestOne()

{

  cout << "The best one's numb is " << bestNumb << " , and his(her) score is " << bestScore << endl;

}

int Student::bestScore = 0;

int Student::bestNumb = 0;

int main()

{

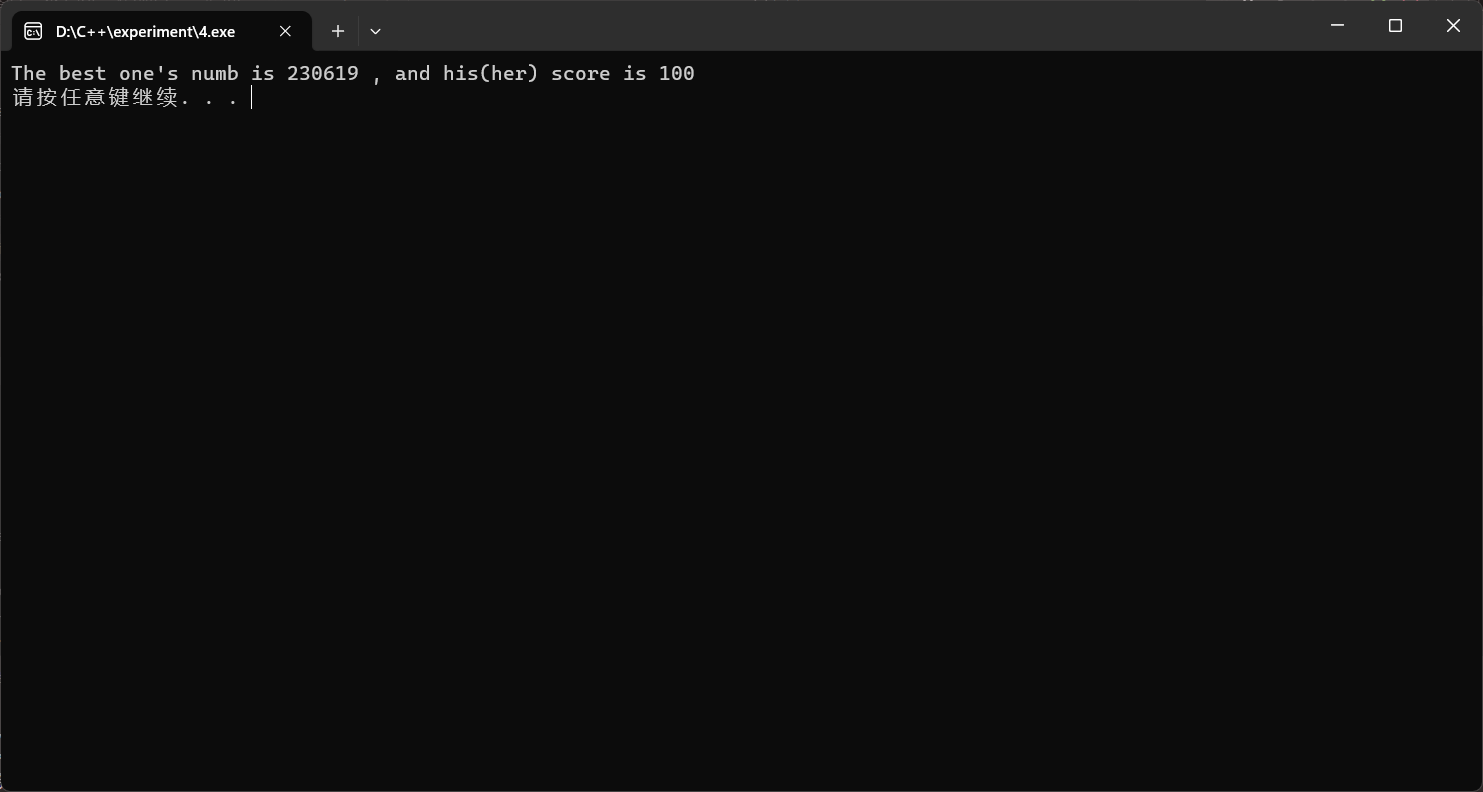
  Student s1(230619, 100), s2(230719, 10), s3(230819, 0);

  Student::showTheBestOne();

  system("pause");

  return 0;

}

结果：

**5.**

#include <iostream>

using namespace std;

class Point

{

private:

  int x;

  int y;

public:

  Point(int x, int y)

  {

    this->x = x;

    this->y = y;

    cout << "The original point is " << '(' << this->x << ", " << this->y << ')' << endl;

  }

  void setPoint(int i, int j);

  void display();

};

void Point::setPoint(int i, int j)

{

  this->x += i;

  this->y += j;

}

void Point::display()

{

  cout << "The changed point is " << '(' << this->x << ", " << this->y << ')' << endl;

}

int main()

{

  Point p(5, 6);

  p.setPoint(1, 1);

  p.display();

  system("pause");

  return 0;

}

结果：