**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

{

public:

void input()

{

cout << "请输入 hour:";

cin >> this->hour;

cout << "请输入 minute:";

cin >> this->minute;

cout << "请输入 second:";

cin >> this->sec;

}

void output()

{

cout << "现在的时间是: " << this->hour << ":" << this->minute << ":" << this->sec

<< "\t" << this->hour << "时" << this->minute << "分" << this->sec << "秒" << endl;

}

private:

int hour;

int minute;

int sec;

};

int main()

{

Time t1;

t1.input();

t1.output();

system("pause");

return 0;

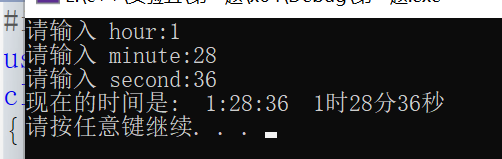
}

**说明：**

1. **相对于原来的程序，hour、minute、sec这样的成员都由公有改为了私有，提高了数据的隐私性；**
2. **原程序是在main函数中直接输入输出hour、minute、sec数据，更改后的程序提供了publi作用域下的input输入函数、output输出函数来实现对Time的输入和输出，将public成员函数与private成员相联系；**
3. **成员函数应该是公用的，hour、minute、sec成员应该是私有的；一个类特有的方法就放在类中定义，公共的方法就放在类外定义；在类的成员数量很大的时候，定义成员函数时最好在类内声明，类外定义；**

**输出：**

**→现在时间是1：28：36，尝试输入这个时间**

****

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：**

1. 含类定义的头文件 student.h

#include<iostream>

using namespace std;

class Student

{

public:

void set\_value(string Num,string Name, string Sex);

void display();

private:

string num;

string name;

string sex;

};

(2)包含成员函数定义的源文件 student.cpp

#include"student.h"

void Student::set\_value(string Num, string Name, string Sex)

{

this->num = Num;

this->name = Name;

this->sex = Sex;

};

void Student::display()

{

cout << "num: " << this->num << endl;

cout << "name: " << this->name << endl;

cout << "sex: " << this->sex << endl;

}

1. 包含主函数的源文件 main.cpp。

#include"student.h"

int main()

{

Student stud;

stud.set\_value("007", "tcg", "m");

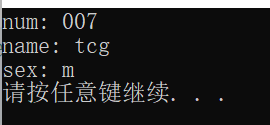
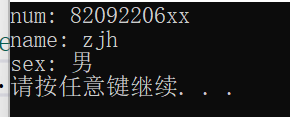
stud.display();

system("pause");

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 std::cout;

using std::cin;

using std::endl;

class Cuboid

{

public:

Cuboid(double length, double width, double height)

{

this->Length = length;

this->Width = width;

this->Height = height;

}

//构造函数访问私有成员定义长宽高

public:

double caculate()

{

return this->Length \* this->Width \* this->Height;

}

//提供计算体积的方法

private:

double Length;

double Width;

double Height;

};

//定义长方体类,并利用构造函数定义长宽高,而且提供计算体积的方法;

int main() {

cout << "请输入第一个立方体的长宽高:" << endl;

double L1 = 0, W1 = 0, H1 = 0;

cout << "长 Length1 = ";

cin >> L1;

cout << "宽 Width1 = ";

cin >> W1;

cout << "高 Heigth1 = ";

cin >> H1;

Cuboid c1 = Cuboid(L1, W1, H1);

cout << "第一个立方体的体积为:" << c1.caculate() << endl << endl;

//计算第一个立方体的体积;

cout << "请输入二个立方体的长宽高:" << endl;

double L2 = 0, W2 = 0, H2 = 0;

cout << "长 Length2 = ";

cin >> L2;

cout << "宽 Width2 = ";

cin >> W2;

cout << "高 Heigth2 = ";

cin >> H2;

Cuboid c2 = Cuboid(L2, W2, H2);

cout << "第二个立方体的体积为:" << c2.caculate() << endl << endl;

//计算第二个立方体的体积;

cout << "请输入第三个立方体的长宽高:" << endl;

double L3 = 0, W3 = 0, H3 = 0;

cout << "长 Length3 = ";

cin >> L3;

cout << "宽 Width3 = ";

cin >> W3;

cout << "高 Heigth3 = ";

cin >> H3;

Cuboid c3 = Cuboid(L3, W3, H3);

cout << "第三个立方体的体积为:" << c3.caculate() << endl << endl;

//计算第三个立方体的体积;

system("pause");

return 0;

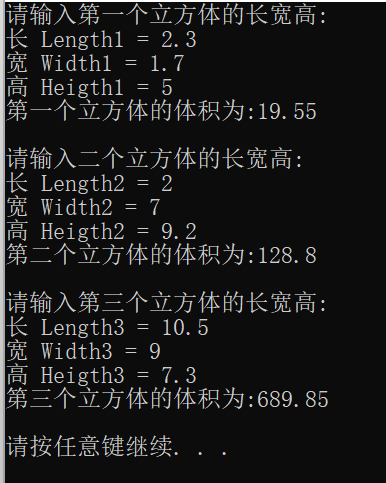
}

**分析：**

1. **对于length(长)、width(宽)、height(高)这样的成员属性，我们要将它们置private作用域下，同时利用构造函数为其赋初值；**
2. **类内提供成员函数caculate()来计算立方体的体积；**
3. **注意this->指针的利用；**

**输出：**

**→三个立方体的长宽高分别为：2.3、1.7、5；2、7、9.2；10.5、9、7.3；**

****

**由此得到了三个立方体的体积；**

**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<cmath>

using std::cin;

using std::cout;

using std::endl;

class Shape

{

public:

virtual double caculate() = 0;

};

//构造抽象基类Shape,并定义纯虚函数caculate()计算其面积;

class Circle : public Shape

{

public:

Circle(double R)

{

this->radius = R;

}

//构造函数给私有空间内的成员赋初值;

public:

virtual double caculate()

{

return 3.41 \* pow(this->radius, 2);

}

private:

double radius;

};

//子类Circle

class Square : public Shape

{

public:

Square(double L)

{

this->length = L;

}

public:

virtual double caculate()

{

return pow(this->length, 2);

}

private:

double length;

};

//子类Square

class Rectangle :public Shape

{

public:

Rectangle(double L, double W)

{

this->length = L;

this->width = W;

}

public:

virtual double caculate()

{

return this->length \* this->width;

}

private:

double length;

double width;

};

//子类Rectangle

class Trapezoid: public Shape

{

public:

Trapezoid(double up\_L, double down\_L, double H)

{

this->up\_length = up\_L;

this->down\_length = down\_L;

this->height = H;

}

public:

virtual double caculate()

{

return (this->up\_length + this->down\_length) \* this->height / 2;

}

private:

double up\_length;

double down\_length;

double height;

};

//子类Trapezoid

class Triangle: public Shape

{

public:

Triangle(double L1,double L2,double L3)

{

this->length1 = L1;

this->length2 = L2;

this->length3 = L3;

}

public:

virtual double caculate()

{

double p = (this->length1 + this->length2 + this->length3) / 2;

return sqrt(p \* (p - length1) \* (p - length2) \* (p - length3));

//海伦公式求面积

}

private:

double length1;

double length2;

double length3;

};

//子类Triangle

//

//以上代码均是利用继承,重写了纯虚函数caculate(),计算其各自的面积

int main()

{

Shape\* array[5];

//定义Shape的基类指针数组

Circle circle(10.2);

Square square(3);

Rectangle rectangle(3, 4);

Trapezoid trapezoid(2.0, 4.5, 3);

Triangle triangle(4, 5, 6);

//定义派生类对象

array[0] = &circle;

array[1] = &square;

array[2] = &rectangle;

array[3] = &trapezoid;

array[4] = &triangle;

//使其中的每一个元素指针指向一个派生类的对象

double sum = 0;

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

{

sum += array[i]->caculate();

}

//for循环求它们的面积之和;

cout << "五个图形的总面积是:" << sum << endl;

return 0;

}

**分析：**

1. **本体考察了多态中的抽象类以及虚函数的知识点；首先我们要创造一个虚基类Shape，再利用继承的方法创建五个派生类Circle（圆形）、Square（正方形）、Rectangle（ 长方形）、Trapezoid （梯形）和 Triangle （三角形）；**
2. **注意纯虚函数**virtual double caculate() = 0，**在之后的五个子类里都要求重写；**
3. **创建Shape类指针数组来存储五个具体的的对象；**

**输出：**

**→最后输出结果是:** ****

**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>

using namespace std;

#include<vector>

template<typename T>

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

{

vector<T>v; //这个向量用于储存v1和v2的交集

bool is = false; //用于判断两个向量是否有交集

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

{

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

{

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

{

is = true;

if (v.size() != 0) //判断v1,v2的这个交集是否已存在于v中

{

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

{

if (v1.at(i) == v.at(z))

{

break;

}

else

{

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

}

}

}

else

{

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

}

}

}

}

if (is==false)

{

cout << "这两个向量没有交集!" << endl;

return v;

}

else

{

return v;

}

}

void printStringVector(vector<string>& v)

{

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

{

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

}

cout << endl;

}

//函数printStringVector来打印string类型的vector向量,利用了迭代器

int main()

{

cout << "请输入第一个vector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vector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); //定义向量v储存v1和v2的交集

if (v.size() != 0)

{

cout << "v1和v2的交集为:";

printStringVector(v);

}

system("pause");

return 0;

}

**分析：**

1. **本体着重于考察vector的相关知识，总结如下：**

**vector<T>v;**

**a.v.size() 求容器的大小；**

**b.v.push\_back(T idx) 在容器末尾增加数据idx;**

**c.v.begin() v.end() 常用的迭代器**

**d.v.capacity() 求容器的容量**

**e.v[x] v.at（x） 都可以用于读取vector容器中的数据 ……**

**2.利用string来处理字符串更加便捷；**

**3.打印vector的多种方法要熟记；**

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

{

cout << \*it << endl;

}

（2）void print(int val)

{

cout << val << endl;

}

for\_each(v.begin(), v.end(), print); <\*\*\*利用STL中提供的遍历算法\*\*\*>

（3）while (itBegin != itEnd)

{

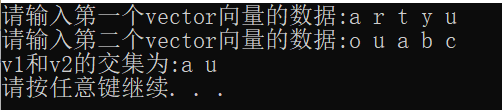
cout << \*itBegin << endl;

itBegin++;

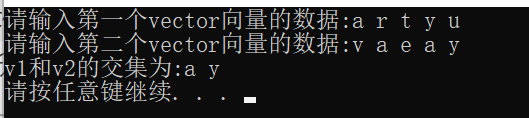
}

**输出：**

1. **当v1中的字符在v2中没有重复出现时：**

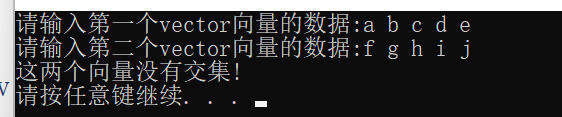
****

1. **当v1中的字符在v2中重复出现时：**



**→观察到没有重复打印已经重复的‘a’,符合预期结果；**

**3、当v1和v2没有重复元素时：**

****

**三、遇到的问题和解决方法**

**本次实验主要是考察关于类与对象的知识。要求我们构造函数、分文件的编写、虚基类的实现和vector容器的使用有深入的了解。在编写过程中，我遇到的主要问题是相关知识点掌握不太牢固，同时对于编写程序的格式，我应该要做到更加规范。**

**四、心得与体会**

**这是我们本期最后一次的实验报告，回顾这一学期对于c++的学习，有困惑、有疑问、有喜悦、当然也有许多收获。经历了这半年的学习，我们了解了面向对象的编程思想，能够利用计算机语言来解决生活中的实际问题。希望在之后的学习与生活中，我们能仍然保持积极进取的乐观心态、坚韧不拔的精神品质，在我们学习生涯中继续进取。**