《C++面向对象程序设计》
实验指导书

郑州轻工业学院

2018.9.15

目录

前言

《面向对象程序设计》在计算机科学中是一门专业基础课，面向对象软件开发方法吸收了软件工程领域有益的概念和有效的方法而发展起来的一种软件开发方法。它把数据和对数据的操作封装起来，集抽象性、封装性、继承性和多态性于一体，可以帮助人们开发出模块化、数据抽象程度高的、体现信息隐蔽、可复用、易修改、易扩充等特性的程序。面向对象程序设计方法及技术是对面向对象方法及思想的基本体现。该课程的具体目标如下：

（1）理解类和对象的基本概念和本质，掌握利用类来解决实际问题的实现方法。

（2）掌握继承、派生和多态性的概念及使用方法。

（3）掌握文本文件和二进制文件的输入/输出方法，掌握模板的使用方法和异常的解决方法。

（4）能对实际问题进行分析抽象，设计解决问题的算法，根据所设计的算法进行面向对象的程序设计，从而解决实际问题。

C++基础练习

[实验内容]

请编写一个 C++程序，完成以下基本功能：

1. 在程序头添加注释“Welcome! First C++ Program”；

2. 在主函数中输入两个整数 a、b，并通过函数调用求解以该两个数为边长的长方形的面积，并返回求解结果，进行输出。

3. 在主函数中输入两个浮点数 c、d，利用函数重载求解以这两个浮点数为边长的长方形的面积，并返回求解结果，进行输出。

4. 在主函数中输入一个整型常数 i，定义一个局部变量 area 存储以 i 为半径的圆的面积并输出，定义一个全局变量 area，用以存储以 i 为边的正方形的面积，并输出；

5. 在主函数中，动态申请一个双精度空间，并对其空间值赋值为 r,计算以 r 为半径的圆的面积并输出，完成后释放该空间。

[实验目的]

1. 掌握 C++程序的基本格式与规范，学会编写简单的 C++程序；
2. 熟悉 C++程序基本的输入/输出操作；
3. 掌握函数的说明、函数的定义以及函数的调用方法；
4. 掌握作用域运算符的基本使用方法和功能；
5. 掌握 C++内存的动态分配与释放方法；
6. 掌握 C++的类型变量、常量及修饰符的使用方法。

[上机代码提示]

第一步：在程序头添加注释“Welcome! First C++ Program”

```
// Welcome! First C++ Program
#include   <iostream>
using   namespace std;
int main()
{    cout<<” Welcome! First C++ Program \n”;
     return 0;
}
```

第二步：在主函数中输入两个整数 a、b，并通过函数调用求解以该两个数为边长的长方形的面积，并返回求解结果，进行输出。

```
#include <iostream>
using namespace std;
int main()
{
     int a, b, area;
     cout<<”请输入两个整数：”;
     cin>>a>>b;
    area=Area(a, b);//请自己写函数 Area
     cout<<”以输入的两个数为边长的正方形面积是：”<<area<<endl;
     return 0;
}
```

第三步：在 2 的基础上，主函数中输入两个浮点数 c、d，利用函数重载求解以这两个浮点数为边长的长方形的面积，并返回求解结果，进行输出。

```
#include <iostream>
using namespace std;
int main()
{
     int a, b, area1;
    float c,d,area2;
     cout<<"请输入两个整数：";
     cin>>a>>b;
    area1=Area(a, b);//请自己写函数 Area
     cout<<"以输入的两个整数为边长的正方形面积是："<<area1<<endl;

    cout<<"请输入两个浮点数：";
     cin>>c>>d;
    area2=Area(c, d);//请自己写重载函数 Area
     cout<<"以输入的两个浮点数为边长的正方形面积是："<<area2<<endl;

     return 0;
}
```

第四步：在主函数中输入一个整型常数 i，定义一个局部变量 area 存储以 i 为半径的圆的面积并输出，定义一个全局变量 area，用以存储以 i 为边的正方形的面积，并输出；

```
#include <iostream>
using namespace std;
int area;
int main()
{    int i, area;
    cout<<"请输入一个整数：";
     cin>>i;
    area=3.14*i*i;
    ::area=i*i;
     cout<<"以输入的整数为半径的圆的面积为："<<area<<endl;
    cout<<"以输入的整数为边的正方形的面积为："<<::area<<endl;
     return 0;}
```

第五步：在主函数中，动态申请一个双精度空间，并对其空间值赋值为 r,计算以 r 为半径的圆的面积并输出，完成后释放该空间。

```
#include <iostream>
using namespace std;
int main()
{    double *r;
     r = new double;
     cout<<"请输入半径：";
     cin>>*r;
     cout<<"面积是："<<3.14*(*r)*(*r)<<endl;
     delete r;
     return 0;}
```

实验一　类与对象（一）

[实验内容]

请大家发挥想象，声明一个股票类 Stock，为该类设计私有的数据成员和共有的成员函数，完成以下功能：

1. 定义 2 个股票类对象 stock1 和 stock2，它们分别为当月第 2 个交易日，和第 3 个交易日的信息，设置这两个交易日的当日最高价、当日最低价、当日开盘价、当日收盘价；

2、用第 2 个交易日和第 3 个交易日的收盘价计算第 3 个交易日的涨幅；

3、重新定义一个对象 stock3，使用构造函数设置该股票对象的交易日、当日最高价、当日最低价、开盘价和收盘价，并输出“构造函数使用中……”；

4、重载股票类的构造函数（不带参数），使交易日、当日最高价、当日最低价、开盘价和收盘价均为 0，且输出“不带参数的构造函数使用中……”；

5、在股票类的析构函数中输出“析构函数使用中……ByeBye!”；

6、定义拷贝构造函数，且在函数中输出“特殊的构造函数之拷贝构造函数使用中……”，并在程序中，使用拷贝构造函数建立新对象 stock4，让其值与 stock3 相同。

[实验目的]

1. 掌握类的定义方式；
2. 掌握类的数据成员和成员函数定义方式；
3. 掌握类成员的访问控制方式；
4. 理解对象的定义及定义和使用对象的方法；
5. 掌握构造函数和析构函数的定义和使用方法；
6. 掌握构造函数的重载；
7. 了解拷贝构造函数的定义和使用。

[上机代码提示]

```
#include <iostream>
using namespace std;
class Stock;
{
    private:
        int Number;
        float Max,Min,Begin,End;
    public:
        Stock(int n,float ma,float mi,float b,float e);
        void Set_Stock(int n,float ma,float mi,float b,float e);
        float Get_End();
        void Show_Stock();
};
Stock::Stock(int n, float ma,float mi,float b,float e)
{
    Number = n;
    Max = ma;
   Min = mi;
```

```
    Begin = b;
    End = e;
}
void Stock::Set_Stock(int n, float ma,float mi,float b,float e)
{
    Number = n;
    Max = ma;
   Min = mi;
    Begin = b;
    End = e;
}
float Stock::Get_End()
{
    return End;
}
void Stock::Show_Stock()
{
    cout<<Number<<"\t";
    cout<<Max<<"\t";
    cout<<Min<<"\t";
    cout<<Begin<<"\t";
    cout<<End<<endl;
}
int main()
{
    Stock s1(1, 4.18,4.00,4.05,4.09);
    Stock s2(2,4.41,4.03,4.04,4.40);
    cout<<"\n"<<(s2.Get_End()-s1.Get_End())/s1.Get_End()*100<<" %;
    return 0;
}
```

实验二　类与对象（二）

[实验内容]

请大家基于实验 1 中声明的股票类，完成以下功能：

1. 定义一个对象数组，连续存放七天的股票信息；

2. 编写一个主函数，根据前后两日 Stock 对象的当日收盘价计算当日收盘价的涨幅。用指针引用对象数组中的两个对象。在主函数中调用该函数计算从第 2 个交易日开始每天的当日涨幅；

3. 在 Stock 类中定义一个静态数据成员，记录当前 Stock 对象的数量；

4. 设计一个成员函数，实现对对象赋值，函数中的形参是对另一个对象的引用，使用 this 指针避免对自己的赋值，在主函数中显示用该成员函数赋值的对象；

5. 定义一个友元函数，计算 Stock 对象的当日开盘价与当日收盘价的高低，如果当日开盘价高于当日收盘价，则返回真，反之则返回假。

[实验目的]

1. 理解对象数组的定义和使用方法；
2. 理解对象指针的概念，掌握指针引用对象的方法；
3. 掌握 this 指针的工作方式；
4. 理解静态数据成员和静态成员函数的定义，并掌握其使用方法；
5. 了解友元与友元函数的概念、作用和使用方法。

[上机代码提示]

```
#include <iostream>
using namespace std;
const N=7;
class Stock
{
    private:
        int Number;
        float Max,Min,Begin,End;
    public:
        Stock() {};
        Stock(int n,float ma,float mi,float b,float e);
        void Set_Stock(int n,float ma,float mi,float b,float e);
        void Set_Stock();
        float Get_End();
        void Show_Stock();
};
Stock::Stock(int n, float ma,float mi,float b,float e)
{
    Number = n;
    Max = ma;
   Min = mi;
    Begin = b;
    End = e;
```

```
}
void Stock::Set_Stock(int n, float ma,float mi,float b,float e)
{
      Number = n;
      Max = ma;
     Min = mi;
      Begin = b;
      End = e;
}
float Stock::Get_End(){    return End;}
void Stock::Show_Stock()
{
      cout<<Number<<"\t";
      cout<<Max<<"\t";
      cout<<Min<<"\t";
      cout<<Begin<<"\t";
      cout<<End<<endl;
}
void Stock::Set_Stock()
{
      cout<<"Number:";
      cin>>Number;
      cout<<"Max:";
      cin>>Max;
      cout<<"Min:";
      cin>>Min;
      cout<<"Begin:";
      cin>>Begin;
      cout<<"End:";
      cin>>End;
}
int main()
{
      int i;
      Stock s1[100];
      Stock *p;
      for (i=0,p=s1;i<N;i++,p++)
      {
            p->Set_Stock();
}
for (i=0,p=s1;i<N;i++,p++)
      {
            p->Show_Stock();
}
for (i=0,p=s1+1;i<N;i++,p++)
cout<<"\n"<<(p->Get_End()-(p-1)->Get_End())/(p-1)->Get_End()*100;
cout<<" %";
return 0;
}
```

实验三　继承与派生

[实验内容]

定义一个家具基类 Furniture,数据成员包括家具编号、价格，函数成员包括输入和输出，请基于该基类完成以下功能：

1.定义一个沙发类 Sofa 是 Furniture 的派生类，在该类中增加数据成员：能够承担的重量 weight,增加函数成员：计算沙发的价格（通过“承重*10”实现，或自己增加数据成员计算价格）;

2.定义一个欧式沙发类EuroSofa是Sofa的派生类,在该类中增加数据成员：材质、沙发体积，增加函数成员：计算欧式沙发的价格（如果材质为 1 等，则沙发体积*100；如果材质为 2 等，则沙发体积*50；如果材质为 3 等，则沙发体积*20。同学们也可以自行增加数据成员，进行欧式沙发价格的计算）;

3.定义一个玩具类 Toy,数据成员包括玩具的材质、玩具本身的体积，增加函数成员：计算玩具类的价格（如果玩具的材质为 1 等品，则玩具体积*50；如果玩具材质为 2 等品，则玩具体积*30，如果玩具为 3 等品，则玩具体积*5）;

4.定义一个欧式沙发玩具，既继承了 EuroSofa 类，又继承了 toy 类，增加函数成员：计算欧式沙发玩具的价格（自己设计计算玩具沙发价格的计算方式）。

[实验目的]

1.掌握类的继承、派生的概念、声明和定义方式;
2.学会定义和使用类的继承关系;
3.掌握私有派生和共有派生的访问特性及使用方法;
4.了解虚基类在解决二义性问题中的作用。

[上机代码提示]

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
class Furniture{
    public:
        void CalculatePrice();
    private:
        float weight;
        float price;
};
```

实验四　运算符重载

[实验内容]

发挥想象，定义一个类，设计数据成员和成员函数，完成以下功能：

1. 重载算术运算符+，-，*，/；
2. 重载赋值运算符=
3. 重载输出运算符<<和输入运算符>>
4. 设计主函数调用重载运算符函数

也可定义四个类完成以上运算符重载！

[实验目的]

1. 了解运算符重载的定义和使用方法；
2. 掌握运算符重载应该注意的问题；
3. 掌握常用的几种运算符的重载；
4. 理解编译时的多态性和运行时的多态性。

[上机代码提示]

```
#include <iostream>
using namespace std;
class Rational;//声明 Rational 类
istream& operator>>(istream& istr,Rational &r);//声明重载>>
ostream& operator<<(ostream& ostr,const Rational& d);//声明重载<<
class Rational
{
    private:
        long num,den;
        void Standardize(void);
    public:
        Rational(int num=0,int denom=1);
        friend istream& operator>>(istream& istr,Rational &r);
friend ostream& operator<<(ostream& ostr,const Rational& d);
};
void Rational::Standardize(void)
{
    if(den<0)
        {
        //使分母大于 0，符号在分子上
            den=-den;
            num=-num;
}
}
Rational::Rational(int p,int q):num(p),den(q)
{
    if(den==0)
    {
```

```
            cerr<<"A Zero denominator is invalid"<<endl;
            exit(1);
}
}
//重载>>,以上 P/Q 的形式输入
istream& operator>>(istream& istr,Rational& r)
{
    char c;//读入操作符'/'
    //作为友元，">>"可以存取 r 的分子和分母
    istr>>r.num>>c>>r.den;
    //如果分母为 0 终止操作
    if(r.den==0)
        {
            cerr<<"A Zero denominator is invalid\n";
            exit(1);
}
//将 r 标准化，例如-2/-5 变成 2/5
r.Standardize();
return istr;
}
//重载<<,以 P/Q 的形式输出
ostream& operator<<(ostream& ostr,const Rational& r)
{
    ostr<<r.num<<'/'<<r.den;
    return ostr;
}
int main()
{
    Rational test;
    cin<<test;
    cout<<endl<<test;
    return 0;
```

实验五　文件操作

[实验内容]

请设计一个 C++程序，完成以下功能：

1.将字符串“I am a good student.”写入磁盘文件 good.dat 中，并将其独处显示在屏幕上；

2.将下表内容以二进制形式写入磁盘文件 good.dat 中，然后读入内存，并显示在显示器上。

商品名称	商品编号	商品价格
公主裙	A161	300
钓鱼竿	H380	200
西洋镜	L901	500

3.用 width()和 setw()控制显示在显示器上输出宽度，且修改为左对齐；

4.将商品价格以八进制、十进制和十六进制的形式显示出来。

[实验目的]

1. 掌握文本文件的输入/输出方法；
2. 掌握二进制文件的输入/输出方法；
3. 掌握控制输出宽度的函数使用方法；
4. 掌握设置进制输出的方法。

[上机代码提示]

```
int main()
{
    GoodClass Lin[3];//定义数组对象,请自行定义类
    Lin[0]=GoodClass(“公主裙”, “A161”,300);//构造对象
    Lin[1]=GoodClass(“钓鱼竿”, “H380”,200);// 构造对象
    Lin[2]=GoodClass(“西洋镜”, “L901”,500);// 构造对象
    ofstream Goodfile(“good.dat”,ios::binary);//打开二进制文件 good.dat
    Goodfile.write((char *)Lin,4*sizeof(GoodClass));
                    //写四个对象到文件 good.dat
    return 0;
}
```

实验六　综合设计

[实验内容]

利用 C++面向对象编程实现下列实际问题中的任意一题：

实际问题 1：设计管理高校教师的程序；

实际问题 2：设计模拟网上购书的结账功能；

实际问题 3：设计模拟 ATM 为用户提供服务；

实际问题 4：学生自选实际问题进行实现。

[实验目的]

1. 清楚面向对象程序设计软件开发过程中的各个阶段；
2. 学会进行实际问题的分析，抽象实际问题的功能模块；
3. 能够抽象出解决实际问题的对象和类，并对其进行设计与实现；
4. 能够采用 C++进行编程进行功能的实现；
5. 能够对 C++程序进行调试和改进。

[上机代码提示]

实际问题 1：设计管理高校教师的程序.

第一步：问题分析和功能定义。

根据实际情况，分析工资系统的数据组成和工资的计算方法。

第二步：对象设计。

设计不同的类。

```
class Person{
    public:
        void input();
    protected:
        int number;//工号
        string name;//姓名
       string sex;//性别
        double salary;//工资
};
Class Teacher:public virtual Person{
    public:
        void input();
        void pay();
    protected:
        int teaching_hours;//课时
        int title;//职称
};
class Temporary_workers:public virtual Person{
```

```
    public:
        Temporary_workers():hourlyPay(30){}
        void input();
        void pay();
    private:
        double hourlyPay;
        int workingHours;
};
class Staff: public virtual Person{
    public:
        Staff(){}
        void input();
        void pay();
    protected:
        double position;//职位
};
class Staff_Teacher:public Staff,public Teacher{
    public:
        Staff_Teacher(){};
        void input();
        void pay();
};
```

第三步：核心控制设计。

用一个简单菜单控制操作，并将工资的计算结果存入文件。

第四步：编写程序并调试。

```
#include <iostream>
#include <string>
#include <fstream>
using namespace std;
class Person{
    public:
        friend ostream &operator<<(ostream&,Person&);
        void input();
    protected:
        int number;//工号
        string name;//姓名
        string sex;//性别
        double salary;     //工资
};
class Teacher:public virtual Person{
    public:
        void input;
        void pay();
    protected:
        int teaching_hours;//课时
        int title;//职称
};
class Temporary_workers:public virtual Person{
    public:
```

```
        Temporary_workers():hourlyPay(30){}
        void input();
        void pay();
    private:
        double hourlyPay;
        int workingHours;
};
class Staff: public virtual Person
{
    public:
        Staff(){}
        void input();
        void pay();
    protected:
        double position;//职位
};
class Staff_Teacher:public Staff,public Teacher
{
    public:
        Staff_Teacher(){};
        void input();
        void pay();
};
ostream& operator<<(ostream &os,Person &emp){
    os<<"\t 工号\t 姓名\t 性别\t 收入"<<endl;
    os<<"\t"<<emp.number<<"\t"<<emp.name<<"\t"<<emp.sex;
    os<<"\t"<<emp.salary<<endl;
    os<<"\t+-------------------------------------------+"<<endl;
    return os;
}
void Person::input()
{
    cout<<"工号:";cin  >>number;
    cout<<"姓名:";cin  >>name;
    cout<<"性别:";cin  >>sex;
}
void Teacher::input()
{
    Person::input();
    cout<<"  课时:";cin >>teaching_hours;
    cout<<"1----------教授"<<endl;
    cout<<"2----------副教授"<<endl;
    cout<<"3----------讲师"<<endl;
    cout<<"4----------助教"<<endl;
    cout<<"  请选择职称:";cin >>title;
}
void Teacher::pay()
{
    if(title==1)//教授
```

```
            salary=teaching_hours*150+5000;
        else if(title==2)//副教授
            salary=teaching_hours*120+4000;
        else if(title==2)//讲师
            salary=teaching_hours*100+3000;
        else //助教
            salary=teaching_hours*80+2000;
}
void Temporary_workers::input()
{
    Person::input()’
    cout<<”当月工作时数”;cin >>workingHours;
}
void Temporary_workers::pay()
{
    salary=workingHours*hourlyPay;
}
void Staff::input()
{
    Person::input();
    cout<<”1----------院级”<<endl;
    cout<<”2----------处级”<<endl;
    cout<<”3----------科级”<<endl;
    cout<<”4----------一般工作人员”<<endl;
    cout<<” 请选择职位：”;cin >>position;
}
void Staff::pay()
{
    if(position==1)//院级
        salary=8000;
    if(position==2)//处级
        salary=6800;
    if(position==3)//科级
        salary=6600;
    if(position==4)//一般工作人员
        salary=5000;
}
void Staff_Teacher::input()
{
    Teacher::input();
    cout<<”1----------院级”<<endl;
    cout<<”2----------处级”<<endl;
    cout<<”3----------科级”<<endl;
    cout<<”4----------一般工作人员”<<endl;
    cout<<” 请选择职位：”;cin >>position;
}
void Staff_Teacher::pay()
{
```

```
    Staff::pay();
    if(title==1)//教授
        salary=teaching_hours*1.5+5000+salary;
    else if(title==2)//副教授
        salary=teaching_hours*1.2+4000+salary;
    else if(title==2)//讲师
        salary=teaching_hours*1.2+3000+salary;
    else //助教
        salary=teaching_hours*1.2+2000+salary;
}
void menu()
{
    system("cls");
    cout<<"**********欢迎使用工资管理系统**********"<<endl;
    cout<<"***              1.教师                    ***"<<endl;
cout<<"***              2.一般职员                ***"<<endl;
cout<<"***              3.临时工                  ***"<<endl;
cout<<"***              4.双肩挑教师              ***"<<endl;
cout<<"***              0.退出                    ***"<<endl;
cout<<"****************************************"<<endl;
cout<<"请选择："<<endl;
}
int main()
{
    int ch;
    Teacher *TeacherPtr;
    Temporary_workers *Temporary_workersPtr;
    Staff *StaffPtr;
    Staff_Teacher *Staff_TeacherPtr;
    ofstream outFile(".\\result.txt");
    while(true)
    {
        menu();
        cin>>ch;
        switch(ch)
        {
            case 1:
                TeacherPtr= new Teacher;
                TeacherPtr->input();
                TeacherPtr->pay();
                outFile<< *TeacherPtr;
                cout<< *TeacherPtr;
                delete TeacherPtr;
                break;
            case 2:
                Temporary_workersPtr = new Temporary_workers;
                Temporary_workersPtr->input();
                Temporary_workersPtr->pay();
                outFile<< *Temporary_workersPtr;
```

```
                cout<< *Temporary_workersPtr;
                delete Temporary_workersPtr;
                break;
            case 3:
                StaffPtr = new Staff;
                StaffPtr->input();
                StaffPtr->pay();
                outFile<< *StaffPtr;
                cout<< *StaffPtr;
                delete StaffPtr;
                break;
            case 4:
                Staff_TeacherPtr = new Staff_Teacher;
                Staff_TeacherPtr->input();
                Staff_TeacherPtr->pay();
                outFile<< *Staff_TeacherPtr;
                cout<< *Staff_TeacherPtr;
                delete Staff_TeacherPtr;
                break;
            case 0:
                outFile.close();
                return 0;
}
    }
}
```

实际问题 2：设计模拟网上购书的结账功能：请参考教材 311 页
实际问题 3：设计模拟 ATM 为用户提供服务；请参考教材 319 页

附：实验报告参考格式

郑州轻工业学院

实 验 报 告

课程名称：_____

姓　　名：_____

学　　号：_____

专业班级：_____

任课教师：_____

_____年_____月_____日

平时成绩登记表

评定项目	内　　容	满 分	评　分	总　分
平时作业	及时上交、答案正确、过程完整	50		
实验过程	验收及时、回答问题正确	30		
实验报告	报告书写规范、内容条理清楚、表达准确规范、上交及时。	20		
评语： **指导老师签字：**　　年　　月　　日				

实验报告正文

实验一

类与对象（一）

一、 实验目的

请同学们参考每次实验的指导说明，自己总结，本次实验的目的

二、 问题的本质和抽象描述

请同学们自己总结。

例如：要完成以上实验内容，我们需要设计一个类，该类中应该有 5 个数据成员，包含交易日、当日最高价、当日最低价、当日开盘价、当日收盘价，有函数成员，包括构造函数、不带参数的构造函数、析构函数、……

三、 测试

1. 方案

请同学们在此描述测试方案、测试模块、测试数据实例（文字数据、图或表等形式）……

例如：本实验计算第 2 个交易日对象和第 3 个交易日对象的股票信息，第 2 个交易日的收盘价为 5，开盘价为 3，当日最高价为 6，当日最低价为 2; 第 3 个交易日的收盘价为 8，开盘价为 5.5，当日最高价为 9，最低价为 5.5.

2. 结果

结合测试数据实例描述测试过程和测试结果，最好给出表示测试过程和结果的抓图，对测试结果进行分析并得出结论。

四、　　总结与讨论

可针对本设计谈体会、谈改进、谈设想等，展示你的概括、归纳和创新思维能力，看重的不是你的对与错，而是鼓励你的想象和创新思维。

五、附：程序模块的源代码