# 实验一 类和对象

## 一、实验目的与任务

（1）掌握类与对象、数据成员与成员函数、构造函数与析构函数等概念。

（2）掌握类的定义与实现方法，以及对象的定义与引用方法。

（3）掌握运算符重载的方法。

## 二、实验内容

（1）定义一个Book（图书）类，在该类定义中包括以下数据成员和成员函数。

数据成员： bookname（书名）、price（价格）和number（存书数量）。

成员函数： display()显示图书的情况；borrow()将存书数量减1，并显示当前存书数量；restore()将存书数量加1，并显示当前存书数量。

在main函数中，要求创建某一种图书对象，并对该图书进行简单的显示、借阅和归还管理。

#include<iostream.h>

class book

{

private:

char bookname[20];

int price;

int number;

public:

void display();//显示图书情况

void borrow();//借

void restore();//还

void ruku();//书入库

};

void book:: ruku(){cout<<"依次输入书名、数量、价格"<<endl;cin>>bookname>>number>>price;}

void book:: display(){cout<<"《"<<bookname<<"》"<<endl<<"当前存书数量:"<<number<<endl<<"价格:"<<price<<"元"<<endl;}

void book:: borrow(){number--;cout<<"成功借阅"<<endl;}

void book:: restore(){number++;cout<<"成功归还"<<endl;}

void main()

{

book d;

d.ruku();

int x;

d.display();

cout<<"借阅请输入1，归还请输入2"<<endl;

cin>>x;

switch(x)

{case 1 :d.borrow();break;case 2: d.restore();break;}

d.display();

}

（2）定义一个Box（盒子）类，在该类定义中包括以下数据成员和成员函数。

数据成员：length（长）、width（宽）和height（高）。

成员函数：构造函数Box，设置盒子的长、宽和高3个初始数据；成员函数setBox对数据成员置值；成员函数volume 计算盒子的体积。

在main函数中，要求创建Box对象，输入长、宽、高，输出盒子的体积。

#include<iostream.h>

class box

{

public:

box(int a,int b,int c);

void volume();

private:

int l;

int w;

int h;

};

//box::date (){int l;int w;int h;}

box::box(int a,int b,int c){l=a;w=b;h=c;}

void box::volume()

{

int v;

v=l\*w\*h;

cout<<"体积="<<v<<endl;

}

void main()

{

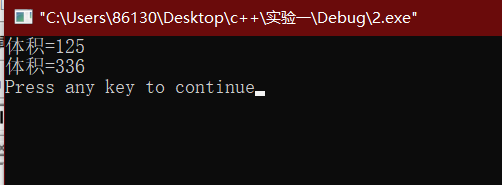
box d1(5,5,5);

box d2(6,7,8);

d1.volume();

d2.volume();

}



（3）定义一个Student类，在该类定义中包括：一个数据成员（分数score）及两个静态数据成员（总分total和学生人数count）；成员函数scoretotalcount(double s) 用于设置分数、求总分和累计学生人数；静态成员函数sum()用于返回总分；静态成员函数average()用于求平均值。

在main函数中，输入某班同学的成绩，并调用上述函数求全班学生的总分和平均分。

#include<iostream.h>

class st

{

public:

void stc(double s);

static void sum();//返回总分

static void ave();//

private:

double score;

static double total;

static int count;

};

double st::total=0;

int st::count=0;

void st::stc(double s)

{

total=0;

int c;

cout<<"输入学生人数:";

cin>>c;

count=c;

for(int i=1;i<=c;i++)

{

cout<<"学号："<<i<<endl;

cout<<"成绩:";

cin>>score;

//score=s;

total=total+score;

}

cout<<"学生数："<<count<<endl;

}

void st::sum(){cout<<"总分："<<total<<endl;}

void st::ave(){cout<<"平均分:"<<total/count<<endl;}

void main()

{

st d;

d.stc(10);

d.sum();

d.ave();

}



（4）定义一个表示点的结构类型Point和一个由直线方程*y* = *ax* + *b*确定的直线类Line。结构类型Point有两个成员x和y，分别表示点的横坐标和纵坐标。Line类有两个数据成员a和b，分别表示直线方程中的系数。Line类有一个成员函数print用于显示直线方程。友元函数setPoint(Line &l1,Line &l2)用于求两条直线的交点。在main函数中，建立两个直线对象，分别调用print函数显示两条直线的方程，并调用函数setPoint求这两条直线的交点。

#include<iostream.h>

struct point{int x,y;};//交点

class line

{

public:

int a,b,c,d;//系数

void print();//显示值线方程

friend void setpoint();//球两值线交点

};

void line::print()

{ line p;

point g;

cout<<"输入l1参数:"<<endl;

cin>>p.a>>p.b;

cout<<"输入l2参数:"<<endl;

cin>>p.c>>p.d;

cout<<"两直线方程为："<<endl<<"y="<<p.a<<"x+"<<p.b<<endl<<"y="<<p.c<<"x+"<<p.d<<endl;

}

void setpoint()

{

point g;

line p;

g.x=(p.d-p.b)/(p.a-p.c);

g.y=p.a\*((p.d-p.b)/(p.a-p.c))+p.b;cout<<"交点为"<<"("<<g.x<<","<<g.y<<")"<<endl;

}

void main()

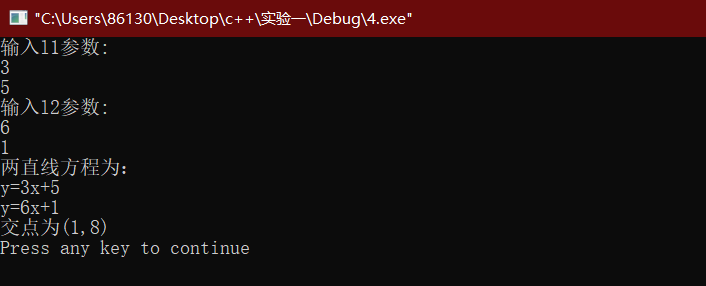
{

line p;

p.print();

setpoint();

}



（5）定义一个复数类，用运算符重载的方式实现复数的输入、输出、基本的数学运算。

#include<iostream.h>

class fu

{

public:

void output();

void input();

void set();

void operator++();

void operator--();

void operator\*();

//void operator%();

int x1,x2,M1,M2,M3;

double s1,s2,M4;

private:

int xu1,xu2,m1,m2,m3;

double shi1,shi2,m4;

};

void fu::set(){shi1=s1;xu1=x1;shi2=s2;xu2=x2;m1=M1;m2=M2;m3=M3;m4=M4;}

void fu::input()

{

cout<<"请依次输入两个复数的实部 虚部(1：i/-1：-i)"<<endl;

cin>>shi1>>xu1>>shi2>>xu2;

//cout<<shi1<<xu1<<"+"<<shi2<<xu2<<endl;

}

void fu::output()

{

cout<<shi1<<"i+"<<shi2<<"i="<<m1<<"i"<<endl;

cout<<shi1<<"i-"<<shi2<<"i="<<m2<<"i"<<endl;

cout<<shi1<<"i\*"<<shi2<<"i="<<m3<<endl;

cout<<shi1<<"i/"<<shi2<<"i="<<m4<<endl;

}

void fu::operator++()

{

m1=shi1+shi2;

}

/\*cout<<shi1<<"i+"<<shi2<<"i"<<endl;

/\*if(xu1==0 && xu2==0)

{m=shi1+shi2;

cout<<m<<endl;}

if(xu1==0 && xu2==1)

{cout<<shi1<<"+"<<shi2<<"i"<<endl;}

if(xu1==1 && xu2==0)

{cout<<shi1<<"i+"<<shi2<<endl;}

if(xu1==1 && xu2==1)

{m=shi1+shi2;cout<<m<<"i"<<endl;}

if(xu1==1 && xu2==-1)

{m=shi1+shi2;cout<<m<<"i"<<endl;}

if(xu1==-1 && xu2==1)

{cout<<"-"<<m<<"i"<<endl;}\*/

void fu::operator--()

{

m2=shi1-shi2;

}

void fu::operator\*()

{m3=shi1\*shi2\*(-1);

m4=shi1/shi2\*(-1);

/\*if(xu1==1 && xu2==1)

{m3=shi1\*shi2\*(-1);}

if(xu1==-1 && xu2==-1)

{m3=shi1\*shi2\*(-1);}

if(xu1==1 && xu2==-1)

{m3=shi1\*shi2}\*/

}

//void fu::operator%()

//{m4=shi1/shi2\*(-1);}

void main()

{

fu d;

d.set ();

d.input();

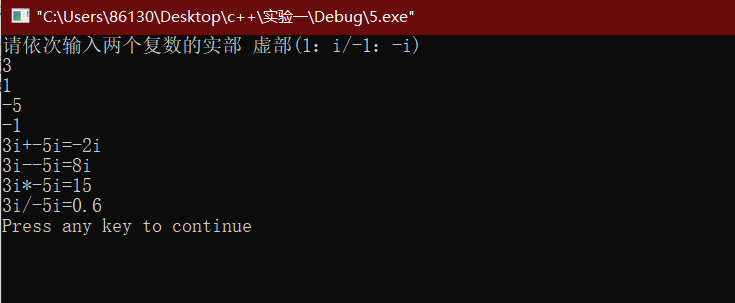
++d;

--d;

\*d;

d.output();

}



（6）假设有向量X = ( x1, x2,…, xn) 和 Y = ( y1, y2, …, yn )，它们之间的加、减和乘法分别定义为：

X + Y = ( x1 + y1, x2 + y2,…, xn + yn )

X − Y = ( x1 − y1, x2 − y2,…, xn − yn )

X \* Y = x1 \* y1 + x2 \* y2 +,…,+ xn \* yn

编写程序定义向量类Vector，重载运算符“+”、“−”、“\*”和“=”，实现向量之间的加、减、乘、赋值运算；重载运算符“>>”、“<<”实现向量的输入、输出功能。注意检测运算的合法性。

#include<iostream.h>

class vector

{

public:

vector(int n);

vector(double \*,int);

~vector();

double & operator[](int i);

vector & operator=(vector &);

friend vector operator+(vector &,vector &);

friend vector operator-(vector &,vector &);

friend vector operator\*(vector &,vector &);

friend ostream & operator<<(ostream &output,vector &);

friend istream & operator>>(istream &input,vector &);

private:

int len;double \*v;

};

vector::vector(int n)

{

v=new double[n];

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

v[i]=0;

len=n;

}

vector::vector(double \*C,int n)

{

v=new double[n];

len=n;

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

v[i]=C[i];

}

vector::~vector(){delete[]v;v=NULL;len=0;}

double &vector::operator[](int i)

{

if(i>=0&&i<len)

return v[i];

else

{

cout<<"长度为："<<i<<endl;

}

}

vector &vector::operator=(vector &C)

{

if(len==C.len)

{

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

v[i]=C[i];

return \*this;

}

else

{

cout<<"operator=fail!\n";

}

}

vector operator+(vector &A,vector &B)

{

int n=A.len;

double \*T=new double[n];

if(n==B.len)

{

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

T[i]=A[i]+B[i];

}

}

vector operator-(vector & A,vector &B)

{

int n=A.len;

double \*T=new double[n];

if(n==B.len)

{

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

T[i]=A[i]-B[i];

return vector(T,n);

}

}

double operator\*(vector &A,vector &B)

{

int n=A.len;

double s=0;

if(n==B.len)

{

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

s+=A[i]\*B[i];

return s;

}

}

ostream & operator<<(ostream &output,vector & A)

{

int i;

output<<'(';

for(i=0;i<A.len-1;i++)

output<<A[i]<<',';

output<<A[i]<<',';

return output;

}

istream & operator>>(istream &input,vector & A)

{

for(int i=0;i<A.len;i++)

{

input>>A[i];

return input;

}

}

void main()

{

int k1,k2,k3;

double t;

cout<<"输入数组元素个数：";

cin>>k1;

vector A(k1);

cout<<"X:";

cin>>A;

cout<<"输入数组元素个数：";

cin>>k2;

vector B(k2);

cout<<"Y:";

cin>>B;

cout<<"C=";

cin>>k3;

vector C(k3);

cout<<"X="<<A<<endl;

cout<<"Y="<<B<<endl;

C=A+B;

cout<<"X+Y="<<C<<endl;

C=A-B;

cout<<"X-Y="<<C<<endl;

t=A\*B;

cout<<"A\*B="<<t<<endl;

}

## 三、通过实验回答以下问题

（1）构造函数的作用是什么？构造函数的调用时机？

构造函数是在定义类对象时有程序du自动调用的为类提供初始化操作的函数。构造函数一般是用来初始化对象的。

（2）一个类所有对象之间共享信息的机制是什么？

个对象通过发送消息来请求另一个对象为其服务。

（3）运算符重载的实现机制是什么？为什么引入运算符重载？

自定义运算符的运算规则，为一些类定义各种运算符，使它的记法形式和计算方式更符合人们的习惯。