实验五 友元函数、友元类与运算符重载

1. 实验目的

掌握友元函数和友元类的说明方法，理解友元函数和友元类的使用特点。理解运算符重载的概念和实质，掌握运算符重载函数的定义方法，掌握将运算符重载为类的成员函数和友元函数的方法，掌握和运用几种常用的用C++语言运算符的重载进行解决问题的方法。

1. 实验内容

(1) 设计一个班级类（专业名，班级号，10个学生的姓名，10个学生的成绩），再设计一个友元函数求该班级学生中的的最高分和最低分，并输出对应的分数和姓名。

#include <iostream>

#include <string>

using namespace std;

class Banji

{

public:

Banji(string m,int n, string \*a,int \*s);

friend void compare(Banji &);

private:

string major;

int num;

string name[10];

int score[10];

}

Banji::Banji(string m,int n, string \*a,int \*s)

{

int i,j;

major=m;

num=n;

for(i=0;i<10;i++)

name[i]=a[i];

for(j=0;j<10;j++)

score[j]=s[j];

}

void compare(Banji &c)

{

int min,max,i,j,k,g;

max=c.score[0];

min=c.score[0];

for(i=0;i<10;i++)

{

if(c.score[i]<min)

k=i;

}

for(j=0;j<10;j++)

{

if(c.score[j]>max)

g=j;

}

cout<<"最高分:"<<c.name[g]<<" "<<c.score[g]<<endl;

cout<<"最低风:"<<c.name[k]<<" "<<c.score[k]<<endl;

}

int main()

{

string b[10]={"a","b","c","d","e","f","g","h","i","j"};

int s1[10]={91,90,87,88,78,76,98,99,100,56};

Banji x("math",001,b,s1);

compare(x);

return 0;

}

(2) 日期类Date包含日期的年、月、日三个数据成员，编写一个友元函数，求两个日期之间相差的天数。

#include <iostream>

#include "Date.h"

using namespace std;

class Date

{

public:

Date(int y,int m,int d);

friend void math(Date a,Date b);

public:

int year;

int month;

int day;

}

Date::Date(int y,int m,int d)

{

year=y;

month=m;

day=d;

}

void math(Date a,Date b)

{

int judge(int year);

int m[]={31,28,31,30,31,30,31,31,30,31,30,31};

int i,j,h,g,k,s1=0,s2=0;

for(i=0;i<a.year;i++)

{

if(judge(i))

s1=s1+366;

else

s1=s1+365;

}

for(j=0;j<a.month-1;j++)

s1=s1+m[j];

if(judge(a.year)&&a.month>2)

s1=s1+a.day+1;

else

s1=s1+a.day;

for(h=0;h<b.year;h++)

{

if(judge(h))

s2=s2+366;

else

s2=s2+365;

}

for(g=0;g<b.month-1;g++)

s2=s2+m[g];

if(judge(b.year)&&b.month>2)

s2=s2+b.day+1;

else

s2=s2+a.day;

k=abs(s1-s2);

cout<<"两个时间相差:"<<k<<"天"<<endl;

}

#include <iostream>

#include "Date.h"

using namespace std;

int judge(int year)

{

if(year%4==0&&year%100!=0||year%400==0)

return 1;

else

return 0;

}

int main()

{

Date c(1998,4,23),d(2000,12,29);

math(c,d);

return 0;

}

(3) 设计一个复数类Complex，重载运算符“\*”，“/”，使之能用于复数的乘除。运算符重载函数作为Complex类的成员函数。

#include <iostream>

#include <math.h>

#include "omplex.h"

using namespace std;

class Complex

{

public:

Complex(double r,double i);

Complex operator \*(Complex &c2);

Complex operator /(Complex &c2);

void display();

public:

double real;

double imag;

}

Complex::Complex(double r,double i)

{

real=r;

imag=i;

}

void Complex::display()

{

if(real!=0)

{

cout<<real;

if(imag>0)

cout<<"+"<<imag<<"i"<<endl;

else if(imag==0)

cout<<endl;

else if(imag<0)

cout<<imag<<"i"<<endl;

}

if(real==0&&imag>0)

cout<<imag<<"i"<<endl;

if(real==0&&imag==0)

cout<<imag<<endl;

if(real==0&&imag<0)

cout<<imag<<"i"<<endl;

}

Complex Complex::operator \*(Complex &c2)

{

Complex c;

c.real=real\*c2.real-imag\*c2.imag;

c.imag=real\*c2.imag+imag\*c2.real;

return c;

}

Complex Complex::operator /(Complex &c2)

{

Complex c;

c.real=(real\*c2.real+imag\*c2.imag)/(c2.real\*c2.real+c2.imag\*c2.imag);

c.imag=(imag\*c2.real-real\*c2.imag)/(c2.real\*c2.real+c2.imag\*c2.imag);

return c;

}

int main()

{

Complex c1(3,4),c2(5,-10),c3,c4;

c3=c1\*c2;

c4=c1/c2;

cout<<"c1=";

c1.display();

cout<<"c2=";

c2.display();

cout<<"c1\*c2=";

c3.display();

cout<<"c1/c2=";

c4.display();

return 0;

}

(4) 设计一个长方形类Rectangle，包含长和宽两个私有数据成员。要求重载运算符“+”，以实现多个矩形对象的面积之和（即要能够连加）。

#include <iostream>

using namespace std;

class Rectangle

{

public:

Rectangle(int l,int w);

Rectangle operator +(Rectangle &t2);

void display();

private:

int length;

int width;

int area;

};

Rectangle::Rectangle(int l,int w)

{

length=l;

width=w;

area=l\*w;

}

Rectangle Rectangle::operator +(Rectangle &t2)

{

Rectangle s;

s.area=area+t2.area;

return s;

}

void Rectangle::display()

{

cout<<area<<endl;

}

int main()

{

Rectangle k;

Rectangle r1(3,4),r2(2,3),r3(4,5),r4(6,3);

k=r1+r2+r3+r4;

cout<<"四个长方体面积之和为:";

k.display();

return 0;

}

3、实验小结