**实验五 静态成员、友元、对象成员**

**1、实验目的：**

1）掌握静态数据成员的定义及使用；

2）掌握静态成员函数的定义及使用。

3）掌握友员函数的定义和使用。

4）理解对象成员的概念

5）掌握对象成员的初始化方法

**2、实验内容：**

**2.1 分析下面的程序，指出程序运行的结果。**

1) **P132** **习题3.14，3.15，3.20，3.23，3.24，3.25，3.26**

**代码：**

3.14

#include <iostream>

using namespace std;

class B{

public:

B(){}

B(int i,int j){

x=i;

y=j;

}

void printb(){

cout<<x<<","<<y<<endl;

}

private:

int x,y;

};

class A{

public:

A(){}

A(int i,int j);

void printa();

private:

B c;

};

A::A(int i,int j):c(i,j){}

void A::printa(){

c.printb();

}

int main(){

A a(7,8);

a.printa();

return 0;

}

3.15

#include <iostream>

using namespace std;

class A{

public:

void set(int i,int j){

x=i;

y=j;

}

int get\_y(){

return y;

}

private:

int x,y;

};

class box{

public:

void set(int l,int w,int s,int p){

length=l;

width=w;

label.set(s,p);

}

int get\_area(){

return length\*width;

}

private:

int length,width;

A label;

};

int main(){

box b;

b.set(4,6,1,20);

cout<<b.get\_area()<<endl;

return 0;

}

3.20

#include <iostream>

using namespace std;

class aClass{

public:

aClass(){

total++;

}

~aClass(){

total--;

}

int gettotal(){

return total;

}

private:

static int total;

};

int aClass::total=0;

int main(){

aClass o1,o2,o3;

cout<<o1.gettotal()<<"objects in existence\n";

aClass \*p;

p=new aClass;

if(!p){

cout<<"Allocation error\n";

return 1;

}

cout<<o1.gettotal();

cout<<"objects in existence after allocation\n";

delete p;

cout<<o1.gettotal();

cout<<"objects in existence after allocation\n";

return 0;

}

3.23

#include <iostream>

using namespace std;

class Test

{

private:

int val;

public:

Test()

{cout<<"default."<<endl;}

Test(int n)

{

val=n;

cout<<"Con."<<endl;

}

Test(const Test& t)

{

val=t.val;

cout<<"Copy con."<<endl;

}

};

int main()

{

Test t1(6);

Test t2=t1;

Test t3;

t3=t1;

return 0;

}

3.24

#include <iostream>

using namespace std;

class N{

private:

int A;

static int B;

public:

N(int a){

A=a;

B+=a;

}

static void f1(N m);

};

void N::f1(N m){

cout<<"A="<<m.A<<endl;

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

}

int N::B=0;

int main(){

N P(5),Q(9);

N::f1(P);

N::f1(Q);

return 0;

}

**3.25**

#include <iostream>

using namespace std;

class M

{

int x,y;

public:

M()

{x=y=0;}

M(int i,int j)

{x=i; y=j;}

void copy(M \* m);

void setxy(int i,int j)

{x=i;y=j;}

void print()

{cout<<x<<","<<y<<endl;}

};

void M::copy(M \*m)

{x=m->x;

y=m->y;}

void fun(M m1,M \*m2)

{

m1.setxy(12,15);

m2->setxy(22,25);

}

int main()

{

M p(5,7),q;

q.copy(&p);

fun(p,&q);

p.print();

q.print();

return 0;

}

3.26

#include <iostream>

using namespace std;

class M{

int A;

static int B;

public:

M(int a){

A=a;

B+=a;

cout<<"Constructing"<<endl;

}

static void f1(M m);

~M(){

cout<<"Destructing"<<endl;

}

};

void M::f1(M m){

cout<<"A="<<m.A<<endl;

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

}

int M::B=0;

int main(){

M P(5),Q(10);

M::f1(P);

M::f1(Q);

return 0;

}

**程序运行结果：**

**3.14**

**7,8**

**3.15**

**24**

**3.20**

**3objects in existence**

**4objects in existence after allocation**

**3objects in existence after allocation**

**3.23**

Con.

Copy con.

Default.

3.24

**A=5**

**B=14**

**A=9**

**B=14**

**3.25**

5，7

22，25

3.26

Constructing

Constructing

A=5

B=15

Destructing

A=10

B=15

Destructing

Destructing

Destructing

Press any key to continue

2）

#include<iostream.h>

class Sample

{

int a,b;

static int c;

public:

Sample()

{

a=b=0;

c++;

}

void show()

{

cout<<"a="<<a<<','<<"b="<<b<<','<<"c="<<c<<endl;

}

};

int Sample::c=1;

void main()

{

Sample a1,a2;

a1.show();

a2.show();

}

**程序运行结果如下：**

a=0,b=0,c=3

a=0,b=0,c=3

3)

#include<iostream.h>

#include<stdio.h>

class B

{

private:

int a,b;

public:

B()

{}

B(int i,int j)

{

a=i;

b=j;

}

void showB()

{

cout<<"a="<<a<<","<<'\t'<<"b="<<b<<endl;

}

};

class A

{

private:

B c;

public:

A(){}

A(int i,int j):c(i,j)

{}

void showA()

{

c.showB();

}

};

void main()

{

A a1(5,6);

a1.showA();

}

**程序运行结果:**

a=5, b=6

2．2编写并调试程序。

**1）**、编写一个名为score的类，用于统计一个班的学生成绩。其中使用一个静态成员数据tals来存储总分及一个静态成员函数re\_totals()返回该总分。并编写相应的测试程序。

程序编写如下：

#include<iostream.h>

class score

{

int mark[10];//保存10个学生的成绩

public:

static int tals;//记录总分

static int num;//记录学生人数

static int re\_tals()//返回总分

{ return tals; }

void set()//从键盘输入10个学生的成绩，保存到数组中，并用tals记录总分，num统计人数

{

for(num=0;num<10;num++){

cin>>mark[num];

tals=tals+mark[num];

}

}

int get\_num()//获取学生成绩个数

{ return num; }

};

int score::tals=0;

int score::num=0;

void main()

{

score t;

cout<<"Please input the scores of the class:"<<endl;

t.set();

cout<<"The number of the scores is: "<<t.get\_num()<<endl;

cout<<"The total score of the class is: "<<t.re\_tals()<<endl;

}**程序运行结果**：

Please input the scores of the class:

5 5 6 3 6 3 6 3 6 3

The number of the scores is: 10

The total score of the class is: 46

**2）**、编写一个程序求直角坐标中一个点到直线的距离。要求设计一个点类point，含有两个公有成员数据x和y，代表点的坐标。另一个类为直线类line，它有三个私有成员数据a,b,c。分别代表直线方程ax+by+c=0的系数。在类line中设计一个友元函数distance计算一个点到直线的距离。点（x,y）到直线ax+by+c=0的距离distance的计算公式如下：

distance=|ax+by+c|/sqrt(a\*a+b\*b)

请根据下面给出的部分程序编写直线类line及其成员函数，以组成完整的程序。

#include<iostream.h>

#include<math.h>

class line;

class point{

public:

float x,y;

point(float i,float j){

x=i;

y=j;

}

double distance(point &,line &);

};

class line{

private:

int a;

int b;

int c;

public:

line(int aa,int bb,int cc){

a=aa;

b=bb;

c=cc;

}

friend double distance(point &,line &);

};

double distance(point &p1,line &l1){

return fabs(p1.x\*l1.a+p1.y\*l1.b+l1.c)/sqrt(l1.a\*l1.a+l1.b\*l1.b);

}

void main()

{

point P1(2,4);

line L1(1,2,3);

cout<<"distance="<<distance(P1,L1)<<endl;

}

**程序运行结果:**

distance=5.81378

**3)、**编写一类Cpoint表示一个点的信息。在此基础上编写一个表示三角形的类tria,顶点为其对象成员。编写完整的程序输出三角形的面积。**要求用友元函数来实现三角形面积的输出**。

程序如下:

#include<iostream.h>

#include<math.h>

class Cpoint

{public:

float a1,b1;

Cpoint(float i,float j)

{

a1=i;

b1=j;

}

};

class tria

{

double L1,L2,L3,s;

public:

double area;

Cpoint A,B,C;

tria(float a,float b,float c,float d,float e,float f):A( a,b),B(c,d),C(e,f)

{

L1=sqrt((A.a1-B.a1)\*(A.a1-B.a1)+(A.b1-B.b1)\*(A.b1-B.b1));

L2=sqrt((A.a1-C.a1)\*(A.a1-C.a1)+(A.b1-C.b1)\*(A.b1-C.b1));

L3=sqrt((B.a1-C.a1)\*(B.a1-C.a1)+(B.b1-C.b1)\*(B.b1-C.b1));

s=((L1+L2+L3)/2);

area=sqrt(s\*(s-L1)\*(s-L2)\*(s-L3));

}

friend void print(tria D);

};

void print(tria D)

{

cout<<D.area<<endl;//三角形面积的输出

}

void main()

{

tria D(4.0,2.0,3.0,4.0,5.0,6.0);

print(D);

}

**程序运行结果:**

**3**

**4）、已定义了类CDate:**

**#include <iostream>**

**#include <string>**

**using namespace std;**

**class CDate**

**{**

**int year,month,date;**

**public:**

**CDate(int,int,int);**

**void print();**

**};**

**CDate:: CDate (int y,int m,int d)**

**{**

**year=y;**

**month=m;**

**date=d;**

**}**

**void CDate::print()**

**{**

**cout<<month<<'\\'<<date<<'\\'<<year<<endl;**

**}**

在此基础上编写一个student类，包含：

(1)私有数据name,id,birthday. Birthday为CDate对象；

(2)设计构造函数，成员函数print()打印学生信息；成员函数modify,修改学生的name,id;

(3)定义主函数验证以上类。

**代码**：

**#include <iostream>**

**#include <string>**

**using namespace std;**

**class CDate**

**{**

**int year,month,date;**

**public:**

**CDate(int,int,int);**

**void print();**

**};**

**CDate:: CDate (int y,int m,int d)**

**{**

**year=y;**

**month=m;**

**date=d;**

**}**

**void CDate::print()**

**{**

**cout<<month<<'\\'<<date<<'\\'<<year<<endl;**

**}**

**class Student{**

**string name;**

**int id;**

**CDate birthday;**

**public:**

**Student(int year,int month,int date,string n,int i):birthday(year,month,date){**

**name=n;**

**id=i;**

**}**

**void print(){**

**cout<<"id："<<id<<"姓名："<<name<<"生日";**

**birthday.print();**

**}**

**void modify(string m,int d){**

**name=m;**

**id=d;**

**cout<<"修改后的信息"<<"id："<<id<<"姓名："<<name<<"生日";**

**birthday.print();**

**}**

**};**

**int main(){**

**CDate bir(1997,8,13);**

**Student stu(1997,8,13,"陈子枫",7);**

**stu.print();**

**stu.modify("刘阳",3);**

**return 0;**

**}**

**输出结果:**

**id：7姓名：陈子枫生日8\13\1997**

**修改后的信息id：3姓名：刘阳生日8\13\1997**