例 4.3

```
/*Filename:exam4_3.cpp*/
#include <iostream.h>
void main()
{   int year,rem4,rem100,rem400;
    cout << "输入年份:";
    cin >> year;
    rem400=year%400;
    rem100=year%100;
    rem4=year%4;
    if ((rem400==0) || ((rem4==0) && (rem100!=0)))
        cout << year << "是闰年" << endl;
    else
        cout << year << "不是闰年" << endl;
}
```

例 4.5

```
/*Filename:exam4_5.cpp*/
#include <iostream.h>
void main()
{   char ch;
    cout << "课程代号:";cin >> ch;
    switch (ch)
    {
    case 'm': case 'M':  case 'w':  /为什么 case 后没有任何语句，请思考!
    case 'W':cout << "8 学分" << endl;
             break;   为什么使用 break 语句，请思考!
    case 'p': case 'P': case 'c':
    case 'C':cout << "5 学分" << endl;
            break;
    case 'e':
    case 'E':cout << "6 学分" << endl;
             break;
    default:cout << "输入的课程代号不正确" << endl;
    }
}
```

例 4.6

```
/*Filename:exam4_6.cpp*/
#include <iostream.h>
void main()
{   int choice;
    cout << "1.Visual C++" << endl;
    cout << "2.Visual Basic" << endl;
    cout << "3.Visual Foxpro" << endl;
    cout << "请选择:";  cin >> choice;
    switch (choice)
    {
    case 1:cout << "你的选择是 Visual C++" << endl;
           break;
    case 2:cout << "你的选择是 Visual Basic" << endl;
           break;
    case 3:cout << "你的选择是 Visual Foxpro" << endl;
           break;
```

```
        default:cout << "输入错误" << endl;
        }
    }
```

例 5.6

```
/*Filename:exam5_6.cpp*/
#include <iostream.h>
void fun1();
void fun2();
void main()
{   fun1();
    fun2();
}
void fun1()
{   int n=1;          //局部变量
    n+=10;
    cout << "fun1:n=" << n << endl;
}
void fun2()
{   int n=2;          //局部变量
    n+=20;
    cout << "fun2:n=" << n << endl;
}
```

例 5.16

```
/*Filename:exam5_16.cpp*/
#include <iostream.h>
void hanio(int,char,char,char);
void main()
{   char A='A',B='B',C='C';
    int n=3;
    hanio(n,A,B,C);
}
void hanio(int n,char A,char B,char C)
{   if (n==1)
        cout <<"将第" << n << "个盘片从" << A << "柱搬到" << C << "柱上" << endl;
    else
    {   hanio(n-1,A,C,B);
        cout <<"将第" << n << "个盘片从" << A << "柱搬到" << C << "柱上" << endl;
        hanio(n-1,B,A,C);
    }
}
```

例 5.18

```
/*Filename:exam5_18.cpp*/
#include <iostream.h>
int min(int a,int b);
int min(int a,int b,int c);
int min(int a,int b,int c,int d);
void main()
{   cout << min(13,5,4,9) << endl;
    cout << min(-2,8,0) << end;
}
```

```
int min(int a,int b)
{
    return a<b ? a : b;
}
int min(int a,int b,int c)
{   int t=min(a,b);
    return min(t,c);
}
int min(int a,int b,int c,int d)
{   int t1=min(a,b);
    int t2=min(c,d);
    return min(t1,t2);
}
```

例 6.2

```
/*Filename:exam6_2.cpp*/
#include <iostream.h>
#define N 10
void main()
{   int a[]={0,1,2,3,4,5,6,7,8,9},k;
    int low=0,high=N-1,mid;
    cout << "k:";
    cin >> k;
    while (low<=high)
    {   mid=(low+high)/2;
        if (a[mid]==k)
        {   cout << "a[" << mid << "]=" << k << endl;
            return;
        }
        if (a[mid]>k)  high=mid-1;
        else  low=mid+1;
    }
    cout << k << "未找到" << endl;
}
```

例 6.3

```
/*Filename:exam6_3.cpp*/
#include <iostream.h>
#define N 10
void main()
{   int a[]={6,3,9,8,1,5,4,10,2,7};
    int i,j,tmp,exchange;
    cout << "排序前:";
    for (i=0;i<N;i++)
        cout << a[i] << " ";
    cout << endl;
    for (i=0;i<N-1;i++)
    {   exchange=0;
        for (j=N-2;j>=i;j--)
            if (a[j+1]<a[j])
            {   tmp=a[j+1];     //a[j+1]<->a[j]
                a[j+1]=a[j];
                a[j]=tmp;
                exchange=1;
            }
```

```
            if (!exchange)          本趟未发生交换,排序完成,退出 for 循环
                break;
        }
        cout << "排序后:";
        for (i=0;i<N;i++)
            cout << a[i] << " ";
        cout << endl;
    }
```

例 6.5

```
/*Filename:exam6_5.cpp*/
#include <iostream.h>
#define M 200
#define N 200
int akmerman(int m,int n)
{   int akm[M][N];
    int i,j;
    for (j=0;j<N;j++)
        akm[0][j]=j+1;
    for (i=1;i<M;i++)
    {   akm[i][0]=akm[i-1][1];
        for (j=1;j<N;j++)
            akm[i][j]=akm[i-1][akm[i][j-1]];
    }
    return akm[m][n];
}
void main()
{
    cout << akmerman(3,2) << endl;
}
```

例 6.6

```
/*Filename:exam6_6.cpp*/
#include <iostream.h>
#include <iomanip.h>
void fun(int i);
const int N=10;
void main()
{   int i,j,count=1;
    int a[N][N];
    for (i=1;i<N;i++)                   //列和对角线元素均为 1
    {
        a[i][i]=1;a[i][1]=1;
    }
    for (i=3;i<N;i++)                   //求第 3—N 行的元素值
        for (j=2;j<=i-1;j++)
            a[i][j]=a[i-1][j-1]+a[i-1][j];
    for (i=1;i<N;i++)                   //输出数序
    {   fun(3*(N-i)/2);
        for (j=1;j<=i;j++)
            cout << setw(3) << a[i][j];
        cout << endl;
    }
}
void fun(int i)                         //输出 i 个空格
{   int j;
```

```
    for (j=0;j<i;j++)
    cout << " ";
}
```

例 6.9

```
/*Filename:exam6_9.cpp*/
#include <stdio.h>
#include <iostream.h>
#define N 100
void main()
{   int i,j,k;
    char s1[N],s2[N];
    puts("主字符串:");
    gets(s1);
    puts("子字符串:");
    gets(s2);
    for (i=0;s1[i];i++)
    {   for (j=i,k=0;s1[j] && s2[k] && s1[j]==s2[k];j++,k++);
        if (!s2[k])     /*若 s2 比较完毕,表示它是 s1 的子串*/
        {   cout << "位置=" << i << endl;
            return;
        }
    }
    puts("未找到");
}
```

例 7.12

```
/*Filename:exam7_12.cpp*/
#include <iostream.h>
int add(int,int);
int sub(int,int);
int mul(int,int);
int div(int,int);
int (*pfun)(int,int);
void main()
{ int i=8,j=2;
    pfun=add;  //函数指针指向 add 函数
    cout << i << "+" << j << "=" << pfun(i,j) << endl;
    pfun=sub;  //函数指针指向 sub 函数
    cout << i << "-" << j << "=" << pfun(i,j) << endl;
    pfun=mul;  //函数指针指向 mul 函数
    cout << i << "*" << j << "=" << pfun(i,j) << endl;
    pfun=div;  //函数指针指向 div 函数
    cout << i << "/" << j << "=" << pfun(i,j) << endl;
}
int add(int m,int n)  //加法函数
{
    return m+n;
}
int sub(int m,int n)  //减法函数
{
    return m-n;
}
int mul(int m,int n)  //乘法函数
{
```

```
    return m*n;
}
int div(int m,int n)   //整除函数
{
    return m/n;
}
```

例 8.1

```
/*Filename:exam8_1.cpp*/
#include <iostream.h>
#include <string.h>
struct Stud                    //声明结构体类型 Stud
{   int no;                    //学号
    char *name;                //姓名
    char sex[3];               //性别
    float score;               //分数
};
void main()
{   struct Stud st1,st2;       //定义结构体变量 st1 和 st2
    st1.no=102;                //成员赋值
    st1.name=new char[10];     //指针成员分配所指空间
    strcpy(st1.name,"李萍");//成员赋值
    strcpy(st1.sex,"女");  //成员赋值
    st1.score=92.5; // 成 员 赋 值
    cout << st1.no << "," << st1.name << "," << st1.sex << ","
<< st1.score << endl;
    st2=st1;                   //将 st1 整体赋给 st2
    cout << st2.no << "," << st2.name << "," << st2.sex << ","
 << st2.score << endl;
}
```

例 8.2

```
/*Filename:exam8_2.cpp*/
#include <iostream.h>
#pragma pack(1)      //见下面的说明，表示进行紧凑编译
struct Date          //声明日期结构体类型 Date
{  int  y;
    int m;
    int d;
};
struct Person        //声明人员结构体类型 Person
{   int no;
    char name[10];
    struct Date birthday;
};
void main()
{   struct Person p;
    cout << "no:" << sizeof(p.no) << endl;
    cout << "name:" << sizeof(p.name) << endl;
    cout << "birthday:" << sizeof(p.birthday) << endl;
    cout << "Person:" << sizeof(Person) << endl;
    cout << "Date:" << sizeof(Date) << endl;
}
```

例 8.3

```
/*Filename:exam8_3.cpp*/
#include <iostream.h>
#include <iomanip.h>
struct Stud1                    //声明结构体类型 Stud1
{   int no;                     //学号
    char name[16];              //姓名
    char sex;                   //性别,'m':男,'f':女
    float score;                //分数
};
void main()
{   int i,num=0;
    float avg,sum=0;
    struct Stud1 s[5] = {{101,"Li ping",'m',45},
                         {102,"Zhang ping",'m',62.5},
                         {103,"He fang",'f',92.5},
                         {104,"Cheng ling",'f',87},
                         {105,"Wang ming",'m',58}};
    for(i=0;i<5;i++)            //累加总分和不及格的人数
    {   sum+=s[i].score;
        if (s[i].score<60) num+=1;
    }
    cout.setf(ios::fixed);
    cout << "总分:" << setprecision(1) << sum << endl;
    avg=sum/5;                  //求平均分
    cout<< " 平 均 分 :" << avg << endl;
    cout << "不及格人数:" << num << endl;
}
```

例 8.4

```
/*Filename:exam8_4.cpp*/
#include <iostream.h>
struct Stud2
{   int num;
    char name[20];
    char sex;
    float score;
};
void main()
{   struct Stud2 s1={102,"Zhang ping",'m',92.5},*ps;
    ps=&s1;
    cout << "学号:" << s1.num << " 姓名:" << s1.name;
    cout << " 性别:" << s1.sex << " 分数:" << s1.score << endl;
    cout << "学号:" << ps->num << " 姓名:" << ps->name;
    cout << " 性别:" << ps->sex << " 分数:" << ps->score << endl;
    cout << "学号:" << (*ps).num << " 姓名:" << (*ps).name;
    cout << " 性别:" << (*ps).sex << " 分数:" << (*ps).score << endl;
}
```

例 8.5

```
/*Filename:exam8_5.cpp*/
#include <iostream.h>
#include <iomanip.h>
struct Stud2
```

```
{   int num;
    char name[20];
    char sex;
    float score;
};
void main()
{   struct Stud2 s1[5]=
{   {101,"Zhou ping",'m',45},
        {102,"Zhang ping",'m',62.5},
        {103,"Liou fang",'f',92.5},
        {104,"Cheng ling",'f',87},
        {105,"Wang ming",'m',58} };
    struct Stud2 *ps;
    cout << setw(6) << "学 号" << setw(21) << "姓 名";
    cout << setw(6) << "性别" << setw(7) << "分数" << endl;
    for (ps=s1;ps<s1+5;ps++)
    {   cout << setw(6) << ps->num;
        cout << setw(21) << ps->name;
        cout << setw(6) << ps->sex;
        cout << setw(7) << ps->score << endl;
    }
}
```

例 8.6

```
/*Filename:exam8_6.cpp*/
#include <iostream.h>
#include <iomanip.h>
struct Stud3
{   int num;
    char name[20];
    char sex;
    float score;
};
void avg(struct Stud3 *ps);
void main()
{   struct Stud3 s1[5]={{101,"Li ping",'m',45},
    {102,"Zhang ping",'m',62.5},
    {103,"He fang",'f',92.5},
    {104,"Cheng ling",'f',87},
    {105,"Wang ming",'m',58}};
    struct Stud3 *ps=s1;
    avg(ps);
}
void avg(struct Stud3 *ps)
{   int c=0,i;
    float sum=0;
    for (i=0;i<5;i++,ps++)
    {   sum+=ps->score;
        if (ps->score<60) c+=1;
    }
    cout << "不及格人数:" << c << endl;
    cout << "平均分:" << sum/5 << endl;
}
```

例 9.3

```
/*Filename:exam9_3.cpp*/
#include <iostream.h>
class TPoint            //声明类 TPoint
{   int x,y;
public:
    void setpoint(int x1,int y1)    //内联成员函数 setpoint
    {  x=x1;y=y1; }
    void dispoint()                 //内联成员函数 dispoint
    {  cout << "(" << x << "," << y << ")" << endl; }
};
void main()
{   TPoint a;                  // 定义一般对象
TPoint *p=new TPoint;   //*定义对象指针 p，并使用 new TPoint
                                建立该指针所指的匿名对象
    a.setpoint(12,6);
    cout << "First point=>";
    a.dispoint();
    p->setpoint(5,12);
    cout << "Second point=>";
    p->dispoint();
}
```

例 9.4

```
/*Filename:exam9_4.cpp*/
#include <iostream.h>
#include <string.h>
#pragma pack(1)     //第 8 章的说明，表示进行紧凑编译
class Student
{   int no;
    char name[10];
public:
    void setvalue(int xh,char xm[]);
    void display();
};
void Student::setvalue(int xh,char xm[])
{   no=xh;
    strcpy(name,xm);
}
void Student::display()
{   cout << "学号:" << no << ",姓名:" << name << endl;  }
void main()
{   Student s1,s2;
    s1.setvalue(101,"张三");
    s2.setvalue(108,"李四");
    s1.display();
    s2.display();
    cout << sizeof(s1) << endl;
    cout << sizeof(s2) << endl;
}
```

例 9.7

```
/*Filename:exam9_7.cpp*/
#include <iostream.h>
class TPoint2
{   int x,y;
```

```
public:
    TPoint2() { }                                  //默认构造函数
    TPoint2(int x1,int y1) {  x=x1;y=y1; }  /重载构造函数
    void setvalue(int x1,int y1) {  x=x1;y=y1; }
    void dispoint() {  cout << "(" << x << "," << y << ")" << endl; }
};
void main()
{   TPoint2  a(12,16),b;
    cout << "First point=>";
    a.dispoint();
    b.setvalue(5,18);
    cout << "Second point=>";
    b.dispoint();
}
```

例 9.8

```
/*Filename:exam9_8.cpp*/
#include <iostream.h>
class MyClass4
{   int value;
public:
    MyClass4() { value=0; }                 //构造函数
    MyClass4(int v) { value=v; }            //重载构造函数
    int getvalue() { return value; }
    void setvalue(int v) { value=v; }
};
void main()
{   MyClass4 a[10]={0,1,2,3,4,5,6,7,8,9},b[10];
    cout << "输出 a:" << endl;
    for (int i=0;i<10;i++)
    {   cout << "a[" << i << "]=" << a[i].getvalue() << " ";
        if ((i+1)%5==0)
            cout << endl;
    }
    cout << "\n输出 b:" << endl;
    for (i=0;i<10;i++)
    {   cout << "b[" << i << "]=" << b[i].getvalue() << " ";
        if ((i+1)%5==0)
            cout << endl;
    }
    cout << endl;
}
```

例 9.9

```
/*Filename:exam9_9.cpp*/
#include <iostream.h>
class TPoint3
{   int x,y;
public:
    TPoint3() { }                //默认构造函数
    void setvalue(int x1,int y1) {  x=x1;y=y1; }
    void dispoint() {  cout << "(" << x << "," << y << ")" << endl; }
```

```
};
void main()
{   TPoint3 *p=new TPoint3[3];
    p[0].setvalue(1,2);
    p[1].setvalue(2,3);
    p[2].setvalue(3,4);
    p[0].dispoint();
    p[1].dispoint();
    p[2].dispoint();
    delete [3] p;
}
```

例 9.10

```
/*Filename:exam9_10.cpp*/
#include <iostream.h>
class TPoint4
{   int x,y;
public:
    TPoint4(int x1,int y1) {  x=x1;y=y1; }                //构造函数
    TPoint4(const TPoint4 &obj) {  x=obj.x;y=obj.y; }   //复制构造函数
    void dispoint() {  cout << "(" << x << "," << y << ")" << endl; }
};
void main()
{   TPoint4 a(12,16),b(a);
    cout << "First point=>";
    a.dispoint();
    cout << "Second point=>";
    b.dispoint();
}
```

例 9.11

```
/*Filename:exam9_11.cpp*/
#include <iostream.h>
#include <string.h>
class MyClass5
{   int no;
    char *pname;
public:
    MyClass5(int n,char *p)
    {   no=n;
        pname=new char[10]; //不用 new 分配内存空间时出错
        strcpy(pname,p);
    }
    void display() {  cout << "no:" << no << ",name:" << pname << endl; }
};
void main()
{   MyClass5 s(10,"Mary");
    s.display();
}
```

例 9.12

```
/*Filename:exam9_12.cpp*/
#include <iostream.h>
class TPoint5
{   int x,y;
public:
TPoint5(int x1,int y1) {  x=x1;y=y1; }              //构造函数
~TPoint5() {  cout << "调用析构函数." << endl; }          //析构函数
void dispoint() {  cout << "(" << x << "," << y << ")" << endl; }
};
void main()
{   TPoint5 a(12,6),*p=new TPoint5(5,12);   //对象指针指向创建的匿名对象
    cout << "First point=>";
    a.dispoint();
    cout << "Second point=>";
    p->dispoint();
delete p;
}
```

例 9.13

```
/*Filename:exam9_13.cpp*/
#include <iostream.h>
#include <string.h>
class Student1
{   int no;
    char *pname;
public:
    Student1() {}            //默认构造函数
    Student1(int n,char *p) //重载构造函数
    {   no=n;
        pname=new char[10]; //不用 new 分配内存空间时出错
        strcpy(pname,p);
    }
    Student1(Student1 &s)   //复制构造函数
    {   no=s.no;
        pname=s.pname;
    }
    void display()
    {   cout << "no:" << no << ",name:" << pname << endl;
        cout << "  pname:" << (int)pname << endl;    //输出 pname 指针值
    }
};
void main()
{   Student1 s(10,"Mary"),t(s);
    cout << "s:";s.display();
    cout << "t:";t.display();
}
```

例 9.14

```
/*Filename:exam9_14.cpp*/
#include <iostream.h>
#include <string.h>
class Student2
```

```
{   int no;
    char *pname;
public:
    Student2() {}            //默认构造函数
    Student2(int n,char *p) //重载构造函数
    {   no=n;
        pname=new char[10]; //不用 new 分配内存空间时出错
        strcpy(pname,p);
    }
    Student2(Student2 &s)   //复制构造函数
    {   no=s.no;
        pname=new char[strlen(s.pname)+1];
        strcpy(pname,s.pname);
    }
    void display()
    {   cout << "no:" << no << ",name:" << pname << endl;
        cout << "  pname:" << (int)pname << endl;   //输出 pname 指针值
    }
    ~Student2() {  delete pname; }  /析构函数
};
void main()
{   Student2 s(10,"Mary"),t(s);
    cout << "s:";s.display();
    cout << "t:";t.display();
}
```

例 9.21

```
/*Filename:exam9_21.cpp*/
#include <iostream.h>
#include <string.h>
#define N 4
class Student3
{   int no;
    char name[10];
    int deg1;           //语文成绩
    int deg2;           //数学成绩
    int deg3;           //英语成绩
    static int sum1;  //语文总分
    static int sum2;  //数学总分
    static int sum3;  //英语总分
    static int sn;  // 总 人 数
public:
    Student3(int n,char na[],int d1,int d2,int d3) //构造函数
    {   no=n;
        strcpy(name,na);
        deg1=d1;deg2=d2;deg3=d3;
        sum1+=deg1;sum2+=deg2;sum3+=deg3;
        sn++;
    }
    static double avg1() { return (sum1*1.0)/sn; } //静态成员函数
    static double avg2() { return (sum2*1.0)/sn; } //静态成员函数
    static double avg3() { return (sum3*1.0)/sn; } //静态成员函数
    void disp()
    { cout << " " << no << "," << name << "," << deg1 << ","
<< deg2 << "," << deg3 << endl;
    }
};
int Student3::sum1=0;   //静态数据成员置初值
int Student3::sum2=0;   //静态数据成员置初值
```

```
int Student3::sum3=0; //静态数据成员置初值
int Student3::sn=0; //静态数据成员置初值
void main()
{   Student3 s1(1,"Li",67,89,90);   //创建第一个学生对象
    Student3 s2(2,"Ma",67,89,90);   //创建第二个学生对象
    Student3 s3(3,"Zheng",67,89,90);    //创建第三个学生对象
    Student3 s4(4,"Chen",67,89,90);     //创建第四个学生对象
    cout << "输出结果" << endl;
    s1.disp();
    s2.disp();
    s3.disp();
    s4.disp();
    cout << " 语文平均分:" << Student3.avg1() << endl;
    cout << " 数学平均分:" << Student3.avg2() << endl;
    cout << " 英语平均分:" << Student3.avg3() << endl;
}
```

例 9.27

```
/*Filename:exam9_26.cpp*/
#include <iostream.h>
class B1
{
public:
    B1() { cout << "B1:Constructor" << endl; }
    ~B1() { cout << "B1:Destructor" << endl; }
};
class B2
{
public:
    B2() { cout << "B2:Constructor" << endl; }
    ~B2() { cout << "B2:Destructor" << endl; }
};
class B3
{
public:
    B3() { cout << "B3:Constructor" << endl; }
    ~B3() { cout << "B3:Destructor" << endl; }
};
class A
{  B1  b1;
    B2 b2;
    B3 b3;
public:
    A():b3(),b2(),b1() { cout << "A:Constructor" << endl; }
    ~A() { cout << "A:Destructor" << endl; }
};
void main()
{
    A a;
```

例 9.32

```
/*Filename:exam9_30.cpp*/
#include <iostream.h>
class MyClass15
{   int n;
public:
    MyClass15() {}
    MyClass15(int m) { n=m; }
```

```
    MyClass15 add(MyClass15 s1,MyClass15 s2)
    {   this->n=s1.n+s2.n;
        return (*this);
    }
    void disp() {  cout << "n=" << n << endl; }
};
void main()
{   MyClass15 s1(10),s2(5),s3;
    cout << "s1:";
    s1.disp();
    cout << "s2:";
    s2.disp();
    s3.add(s1,s2);
    cout << "执行 s3.add(s1,s2)\ns3:";
    s3.disp();
}
```

例 9.33

```
/*Filename:exam9_31.cpp*/
#include <iostream.h>
class CDate                          //声明类日期类 CDate
{
private:
    int m_nDay;                      //日
    int m_nMonth;                    //月
    int m_nYear;                     //年
public:
    CDate();                         //默认构造函数
    CDate(int,int,int);              //重载构造函数
    void Display();                  //输出日期
    void AddDay();                   //返回加 1 后的日期
    void SetDate(int,int,int);       //设置日期
    ~CDate();                        //析构函数
private:
    bool IsLeapYear();               //判断该年是否为闰年
};
CDate::CDate() { }
CDate::CDate(int year,int month,int day)
{   m_nDay=day;
    m_nMonth=month;
    m_nYear=year;
}
void CDate::Display()
{
    cout << m_nYear << "年" << m_nMonth << "月" << m_nDay << "日" << endl;
}
void CDate::AddDay()                 //返回加 1 后的日期
{   m_nDay++;                        //先将日号增 1
    if (IsLeapYear())                //处理 2 月份:闰年的情况
    {   if (m_nMonth==2 && m_nDay==30)
        {   m_nMonth++;
            m_nDay=1;
            return;
        }
    }
    else                             //处理 2 月份:不是闰年的情况
```

```
{   if (m_nMonth==2 && m_nDay==29)
    {   m_nMonth++;
        m_nDay=1;
        return;
    }
}
if (m_nDay>31)              //以下处理日号大于 31 的情况
{   if (m_nMonth==12)       //为 12 月份时
    {   m_nYear++;
        m_nMonth=1;
        m_nDay=1;
    }
    else                    //为非 12 月份时
    {   m_nMonth++;
        m_nDay=1;
    }
}
else if (m_nDay==31)   //处理日号为 30 的情况
{   if (m_nMonth==4 || m_nMonth==6 || m_nMonth==9 || m_nMonth==11)
        {   //4,6,9,11 月份只有 30 天
            m_nMonth++;
            m_nDay=1;
        }
    }
}
void CDate::SetDate(int year,int month,int day)   //设置日期
{   m_nDay=day;
    m_nMonth=month;
    m_nYear=year;
}
CDate::~CDate() { }          //析构函数
bool CDate::IsLeapYear()     //判断是否闰年
{   bool bLeap;
    if(m_nYear%4!=0) bLeap=false;
    else if(m_nYear%100!=0) bLeap=true;
    else if(m_nYear%400!=0) bLeap=false;
    else bLeap=true;
    return bLeap;
}
void main()
{   CDate date;              //定义 CDate 类对象 date
    int y,m,d;
    cout << "年月日:";
    cin >> y >> m >> d;
    date.SetDate(y,m,d);     //调用设置日期成员函数
    cout << "当前日期:";
    date.Display();          //调用输出日期成员函数
    date.AddDay();
    cout << "当前日期加 1:";  //调用日期加 1 成员函数
    date.Display();
}
```

例 10.6

```
/*Filename:exam10_6.cpp*/
#include <iostream.h>
#include <iomanip.h>
#include <string.h>
typedef struct
{   int no;          //学号
    char name[20];   //姓名
```

```
    float deg1;     //分数 1
    float deg2;     //分数 2
    float deg3;     //分数 3
    float deg4;     //分数 4
    float deg5;     //分数 5
    float avg;      //平均分数
} StudType;         //自定义类型
void average(StudType &rs)      //求平均分
{
    rs.avg=(rs.deg1+rs.deg2+rs.deg3+rs.deg4+rs.deg5)/5;
}
void display(StudType s1)       //输出学生记录
{   cout << s1.no << "  " << s1.name << endl;
    cout.precision(1);          //精度设为 1
    cout.setf(ios::fixed);      //以浮点数输出
    cout.setf(ios::left);       //左对齐
    cout << setw(8) << s1.deg1 << setw(8) << s1.deg2 <<
setw(8) << s1.deg3 << endl;
    cout << setw(8) << s1.deg4 << setw(8) << s1.deg5 <<
setw(8) << s1.avg << endl;
}
void main()
{   StudType s;
    s.no=102;
    strcpy(s.name,"李明");
    s.deg1=92.6;
    s.deg2=82;
    s.deg3=74;
    s.deg4=84.5;
    s.deg5=79.5;
    average(s);
    display(s);
}
```

例 10.7

```
/*Filename:exam10_7.cpp*/
#include <iostream.h>
class Sample
{   int x,y;
public:
    Sample() { x=y=0; }
    Sample(int x1,int y1){ x=x1;y=y1; }
    void copy(Sample &s) { x=s.x;y=s.y; }
    void setxy(int x1,int y1) { x=x1;y=y1; }
    void disp() {  cout << "(" << x << "," << y << ")" << endl; }
};
void fun(Sample s1,Sample &s2)
{   s1.setxy(12,18);
    s2.setxy(23,15);
}
void main()
{   Sample s(5,10),t;
    s.disp();
    t.copy(s);
    t.disp();
    fun(s,t);
    s.disp();
    t.disp();
}
```

例 10.10

```
/*Filename:exam10_10.cpp*/
#include <iostream.h>
#include <math.h>
int solve(double a,double b,double c,double &x1,double &x2)
{   double d;
    int count;
    d=b*b-4*a*c;
    if (d>0)
    {   count=2;
        x1=(-b+sqrt(d))/(2*a);
        x2=(-b-sqrt(d))/(2*a);
    }
    else if (d==0)
    {   count=1;
        x1=(-b)/(2*a);
    }
    else count=0;
    return count;
}
void disp(double a,double b,double c)
{   double x1,x2;
    cout << "(" << a << "," << b << "," << c << "):";
    switch (solve(a,b,c,x1,x2))
    {
    case 0:cout << "无实根" << endl;
           break;
    case 1:cout << "x=" << x1 << endl;
           break;
    case 2:cout << "x1=" << x1 << ",x2=" << x2 << endl;
           break;
    }
}
void main()
{   disp(2,6,4);
    disp(2,4,2);
    disp(5,2,3);
}
```

例 10.11

```
/*Filename:exam10_11.cpp*/
#include <iostream.h>
#include <string.h>
class Teacher
{   char name[10];      //姓名
    char prof[10];      //职务
    Teacher *leader;    //指向领导的指针
public:
    Teacher() { strcpy(name,"\0"); }
    Teacher(char n[],char p[])
    {   strcpy(name,n);
        strcpy(prof,p);
        leader=new Teacher;
    }
    void setleader(Teacher &p) {  leader=&p; } //对象引用作为参数
    char *getname() { return name; }
```

```
    Teacher *getleader() { return leader; }
    void disp(){ cout << "  " << name << "," << prof << " "; }
};
void main()
{   Teacher p[]={Teacher("王华","室主任"),Teacher("李明","职工"),
        Teacher("陈强","系主任"),Teacher("章城","职工"),
    Teacher("张伟","室主任"),Teacher("许源","职工")};
    p[0].setleader(p[2]);
    p[1].setleader(p[0]);
    p[3].setleader(p[4]);
    p[4].setleader(p[2]);
    p[5].setleader(p[4]);
    cout << "输出结果" << endl;
    cout << "  姓名,职务  领导姓名" << endl;
    cout << "  ------------------" << endl;
    for (int i=0;i<6;i++)
    {   p[i].disp();
        cout << (p[i].getleader())->getname() << endl;
    }
}
```

例 11.1

```
/*Filename:exam11_1.cpp*/
#include <iostream.h>
#include <string.h>
#include <iomanip.h>
class Stud
{   char name[10];
    int deg;
    char level[7];
public:
    Stud(char na[],int d) {  strcpy(name,na); deg=d; }
    char *getname() { return name; }
    friend void trans(Stud &s)
    {   if (s.deg>=90) strcpy(s.level,"优");
        else if (s.deg>=80) strcpy(s.level,"良");
        else if (s.deg>=70) strcpy(s.level,"中");
        else if (s.deg>=60) strcpy(s.level,"及格");
        else strcpy(s.level,"不及格");
    }
    void disp()
    {
        cout << "  " << name << "," << deg << "," << level << endl;
    }
};
void main()
{   Stud st[]={Stud("王华",78),Stud("李明",92),
Stud("张伟",62),Stud("孙强",88)};
    cout << "输出结果" << endl;
    for (int i=0;i<4;i++)
    {   trans(st[i]);
        st[i].disp();
    }
}
```

例 11.2

```
/*Filename:exam11_2.cpp*/
#include <iostream.h>
class Stack;                        //预先声明
class Node                          //结点类声明
{   int data;                       //结点值
    Node *next;                     //指向下一结点的指针
public:
    Node(int d)                     //构造函数:建立一个结点,其 data 为 d,next 为空
    {   data=d;
        next=NULL;
    }
    friend class Stack;             //声明友元类 Stack
};
class Stack                         //栈类声明
{   Node *top;                      //栈指针:指向栈顶结点
public:
    Stack(){  top=NULL; }           //构造函数:设置一个空栈
    void push(int d)                //入栈成员函数
    {   Node *p=new Node(d);        //新建一个结点*p
        if (top!=NULL)
            p->next=top;            //若原栈不空,将*p 作为链栈的第一个结点
        top=p;                      //由 top 指向该新结点
    }
    int pop(int &c)                 //出栈成员函数
    {   Node *p=top;                //p 指向栈顶结点
        if (top!=NULL)
        {   c=p->data;              //c 取原栈顶结点的 data 值
            top=top->next;          //top 后移一个结点
            delete p;               //释放*p 结点
            return 1;
        }
        else return 0;
    }
};
void main()
{   Stack s;
    int c;
    s.push(1);                      //4 个元素入栈
    s.push(2);
    s.push(3);
    s.push(4);
    cout << "出栈次序:";
    while (s.pop(c))                //出栈所有元素
        cout << c << " ";
    cout << endl;
}
```

例 11.3

```
/*Filename:exam11_3.cpp*/
#include <iostream.h>
class BBank;            //这里预先声明,类 BBank 在后面声明
class GBank;            //这里预先声明,类 GBank 在后面声明
class CBank             //声明中国银行类 CBank
{
```

```
    int balance;    //存放在银行中的存款数
public:
    CBank() {  balance=0; };        //构造函数
    CBank(int b) {  balance=b; }    //重载构造函数
    void getbalance()
    {   cout << "输入中国银行存款数:";
        cin >> balance;
    }
    void disp() {  cout << "中国银行存款数:" << balance << endl; }
    friend void total(CBank,BBank,GBank);   //友元函数声明
};
class BBank     //声明工商银行类 BBank
{   int balance;
public:
    BBank() {  balance=0; };        //构造函数
    BBank(int b) {  balance=b; }    //重载构造函数
    void getbalance()
    {   cout << "输入工商银行存款数:";
        cin >> balance;
    }
    void disp() {  cout << "工商银行存款数:" << balance << endl; }
    friend void total(CBank,BBank,GBank);
};
class GBank     //声明农业银行类 GBank
{   int balance;
public:
    GBank() {  balance=0; };        //构造函数
    GBank(int b) {  balance=b; }    //重载构造函数
    void getbalance()
    {   cout << "输入农业银行存款数:";
        cin >> balance;
    }
    void disp() {  cout << "农业银行存款数:" << balance << endl; }
    friend void total(CBank,BBank,GBank);
};
void total(CBank A,BBank B,GBank C)     //定义友元函数
{   cout << "  总存款数:" <<  A.balance+B.balance+C.balance
         << endl;
}
void main()
{   CBank X(100);
    BBank Y;
    GBank Z;
    X.disp();
    Y.disp();
    Z.disp();
    Y.getbalance();
    Z.getbalance();
    total(X,Y,Z);
}
```

例 11.4

```
/*Filename:exam11_4.cpp*/
#include <iostream.h>
const int MAX=20;
class numset
{   int count;          //数序中整数个数
    int a[MAX];         //存放本数序的整数
```

```
public:
    numset() { count=0; }
    void addnum(int n)   /添加一个元素 n,并使之有序
    {   int i=0,j;
        while (i<count && n>=a[i] ) i++;     //找到 n 应放的位置 i
        if (n==a[i-1]) return;               //相同的元素不添加
        for (j=count;j>=i;j--)               //a[i]之后元素后移,腾空 a[i]
            a[j+1]=a[j];
        a[i]=n;                              //i 位置处放置 n
        count++;                             //元素总数增 1
    }
    void disp()      //显示所有元素
    {   for (int i=0;i<count;i++)
            cout << a[i] << " ";
        cout << endl;
    }
    int geti(int i) {  return a[i]; }        //返回位置 i 处的元素
    friend numset unionset(numset s1,numset s2)   /合并整数数序 s1 和 s2
    {   numset s;
        int i=0,j=0;
        while (i<s1.count && j<s2.count)
        {   int v1=s1.geti(i);
            int v2=s2.geti(j);
            if (v1<v2) {  s.addnum(v1); i++; }
            else if (v1>v2) {  s.addnum(v2); j++; }
            else {  s.addnum(v1); i++; j++; }
        }
        while (i<s1.count)
        {   int v1=s1.geti(i);
            s.addnum(v1);
            i++;
        }
        while (j<s2.count)
        {   int v2=s2.geti(j);
            s.addnum(v2);
            j++;
        }
        return s;
    }
};
void main()
{   numset set1,set2,set3;
    set1.addnum(1);
    set1.addnum(9);
    set1.addnum(5);
    set1.addnum(9);
    cout << "数序一:";
    set1.disp();
    set2.addnum(8);
    set2.addnum(2);
    set2.addnum(6);
    set2.addnum(5);
    cout << "数序二:";
    set2.disp();
    set3=unionset(set1,set2);     /调用友元函数
    cout << "合并:";
    set3.disp();
}
```

例 12.1

```
#include <iostream.h>
class MyClass
{   int n;
public:
    MyClass() {}
    MyClass(int m) { n=m; }
    MyClass operator +(MyClass s)   //运算符"+"重载成员函数
    {   MyClass tmp;
        tmp.n=n+s.n;
    return tmp;
    }
    void disp() {  cout << "n=" << n << endl; }
};
void main()
{   MyClass s1(10),s2(20),s3;
    cout << "s1:";s1.disp();
    cout << "s2:";s2.disp();
    s3=s1+s2;
 cout << "s3:";s3.disp();
}
```

例 12.5

```
#include <iostream.h>
#include <string.h>
const int Max=256;
class MyClass4
{   char name[Max];
public:
    MyClass4() {}                                   //默认构造函数
    MyClass4(char* str) {  strcpy(name,str); }   //重载构造函数
    ~MyClass4() {}                                  //析构函数
    MyClass4 operator+(const MyClass4 &);//重载运算符“＋”成员函数
    void display() { cout << name << endl; }
};
MyClass4 MyClass4::operator+(const MyClass4 &s)
{   char str[Max];
    strcpy(str,name);
    strcat(str,s.name);
    return MyClass4(str);   //调用重载构造函数建立一个匿名对象并返回该对象
}
void main()
{   char *str;
    str=new  char[Max];
    MyClass4 s1("Visual C++ ");
    MyClass4 s2("6.0");
    cout << "s1:";s1.display();
    cout << "s2:";s2.display();
    MyClass4 s3=s1+s2;
    cout << "s3:";s3.display();
    cout << "s1:";s1.display();
    cout << "s2:";s2.display();
    delete str;
}
```

例 12.26

```
/*Filename:exam12_2.cpp*/
#include <iostream.h>
int day_tab[2][12]={{31,28,31,30,31,30,31,31,30,31,30,31},
      {31,29,31,30,31,30,31,31,30,31,30,31}};
      //day_tab二维数组存放各月天数,第一行对应非闰年，第二行对应闰年
class Date
{   int year,month,day;                    //分别为年、月、日号
    bool leap(int);
    int dton(Date &);
    Date ntod(int);
public:
    Date() {}                                                    //默认构造函数
    Date(int y,int m,int d) {  year=y;month=m;day=d; } //重载构造函数
    Date operator+(int days)
    {   static Date date;
        int number=dton(*this)+days;
        date=ntod(number);
        return date;
    }
    Date operator-(int days)
    {   static Date date;
        int number=dton(*this);
        number-=days;
        date=ntod(number);
        return date;
    }
    int operator-(Date &b)
    {   int days=dton(*this)-dton(b)-1;
        return days;
    }
    void disp() {  cout << year << "." << month << "." << day << endl; }
};
bool Date::leap(int year)
{   if (year%4==0 && year%100!=0 || year%400==0)  //是闰年
        return true;
    else                                          不是闰年//
        return false;
}
int Date::dton(Date &d)
{   int y,m,days=0;
    for (y=1;y<=d.year;y++)
if (leap(y)) days+=366;
        else days+=365;
    for (m=0;m<d.month-1;m++)
        if (leap(d.year)) days+=day_tab[1][m];
        else days+=day_tab[0][m];
    days+=d.day;
    return days;
}
Date Date::ntod(int n)
{   int y=1,m=1,d,rest=n,lp;
    while (1)
    {   if (leap(y))
        {   if (rest<=366) break;
            else rest-=366;
        }
        else
        {   if (rest<=365) break;
            else rest-=365;
        }
```

```
        y++;
    }
    y--;
    lp=leap(y);
    while (true)
    {   if (lp)
        {   if (rest>day_tab[1][m-1]) rest-=day_tab[1][m-1];
            else break;
        }
        else
        {   if (rest>day_tab[0][m-1]) rest-=day_tab[0][m-1];
            else break;
        }
        m++;
    }
    d=rest;
    return Date(y,m,d);
}
void main()
{   Date now(2008,6,12),then(2010,2,10);
    cout << "now:";now.disp();
    cout << "then:";then.disp();
    cout << "相差天数:" << (then-now) << endl;
    Date d1=now+100,d2=now-100;
    cout << "now+100:";d1.disp();
    cout << "now-100:";d2.disp();
}
```

例 12.7

```
/*Filename:exam12_3.cpp*/
#include <iostream.h>
class Vector
{   int x,y;
public:
    Vector() { };                              //默认构造函数
    Vector(int x1,int y1) {  x=x1; y=y1; }  //重载构造函数
    Vector operator +(Vector v)                //成员函数方式重载运算符“+”
    {   Vector tmp;                 //定义一个 tmp 对象
        tmp.x=x+v.x;
        tmp.y=y+v.y;
        return tmp;                 //返回 tmp 对象
    }
    Vector operator -(Vector v) //成员函数方式重载运算符“－”
    {   Vector tmp;                 //定义一个 tmp 对象
        tmp.x=x-v.x;
        tmp.y=y-v.y;
        return tmp;                 //返回 tmp 对象
    }
    void display() { cout << "(" << x << "," << y << ")" << endl;}
};
void main()
{   Vector v1(6,8),v2(3,6),v3,v4;
    cout << "v1=";v1.display();
    cout << "v2=";v2.display();
    v3=v1+v2;
    cout << "v1+v2=";v3.display();
    v4=v1-v2;
```

```
    cout << "v1-v2=";v4.display();
}
```

例 12.8

```
/*Filename:exam12_4.cpp*/
#include <iostream.h>
class Vector
{   int x,y;
public:
    Vector() { };
    Vector(int x1,int y1) { x=x1;y=y1; }
    friend Vector operator +(Vector v1,Vector v2) /友元函数方式重载运算符“+”
    {   Vector tmp;    //定义一个 tmp 对象
        tmp.x=v1.x+v2.x; tmp.y=v1.y+v2.y;
        return tmp;    //返回 tmp 对象
    }
    friend Vector operator -(Vector v1,Vector v2) /友元函数方式重载运算符“一”
    {   Vector tmp;     /定义一个 tmp 对象
        tmp.x=v1.x-v2.x; tmp.y=v1.y-2.y;
        return tmp;      /返回 tmp 对象
    }
    void display() {  cout << "(" << x << "," << y << ")" << endl; }
};
void main()
{   Vector v1(6,8),v2(3,6),v3,v4;
    cout << "v1=";v1.display();
    cout << "v2=";v2.display();
    v3=v1+v2;
    cout << "v1+v2=";v3.display();
    v4=v1-V2;
    cout << "v1-v2=";v4.display();
}
```

例 12.9

```
/*Filename:exam12_5.cpp*/
#include <iostream.h>
class Date1
{   int year,month,day;          //分别为年、月、日号
public:
    Date1() {}                   //默认构造函数
    Date1(int y,int m,int d) {  year=y;month=m;day=d; } //重载构造函数
    Date1(const Date1 &d1)       //复制构造函数
    {   year=d1.year;
        month=d1.month;
        day=d1.day;
    }
    bool operator < (const Date1 d1)
    {   if ((year<d1.year) || (year==d1.year && month<d1.month)
            || (year==d1.year && month==d1.month && day<d1.day))
            return true;
        else
            return false;
```

```
    }
    bool operator == (const Date1 d1)
    {   if ((year==d1.year && month==d1.month && day==d1.day))
            return true;
        else
            return false;
    }
    bool operator > (const Date1 d1)
    {   if ((year>d1.year) || (year==d1.year && month>d1.month)
            || (year==d1.year && month==d1.month && day>d1.day))
            return true;
        else
            return false;
    }
    void disp() {  cout << year << "." << month << "." << day << endl; }
};
void main()
{   Date1 d1(2008,10,1),d2(2010,10,1),d3(d1);
    cout << "d1:";d1.disp();
    cout << "d2:";d2.disp();
    cout << "d3:";d3.disp();
    cout << "d1<d2:" << (d1<d2) << endl;
    cout << "d1>d2:" << (d1>d2) << endl;
    cout << "d1==d2:" << (d1==d2) << endl;
    cout << "d1==d3:" << (d1==d3) << endl;
}
```

例 12.10

```
/*Filename:exam12_6.cpp*/
#include <iostream.h>
class Vector1
{   int x,y;
public:
    Vector1() { };
    Vector1(int x1,int y1) { x=x1;y=y1; }
    friend Vector1 operator +=(Vector1 v1,Vector1 v2) //友元函数方式实现
    {   v1.x+=v2.x;
        v1.y+=v2.y;
        return v1;
    }
    Vector1 operator -=(Vector1 v) //成员函数方式实现
    {    Vector1 tmp;   //定义一个 tmp 对象
        tmp.x=x-v.x;
        tmp.y=y-v.y;
        return tmp;     //返回 tmp 对象
    }
    void display() {cout << "(" << x << "," << y << ")" << endl;}
};
void main()
{   Vector1 v1(6,8),v2(3,6),v3,v4;
    cout << "v1=";v1.display();
    cout << "v2=";v2.display();
    v3=v1+=v2;
    cout << "v3=v1+=v2 后,v3=";v3.display();
    v4=v1-=v2;
    cout << "v4=v1-v2 后,v4=";v4.display();
```