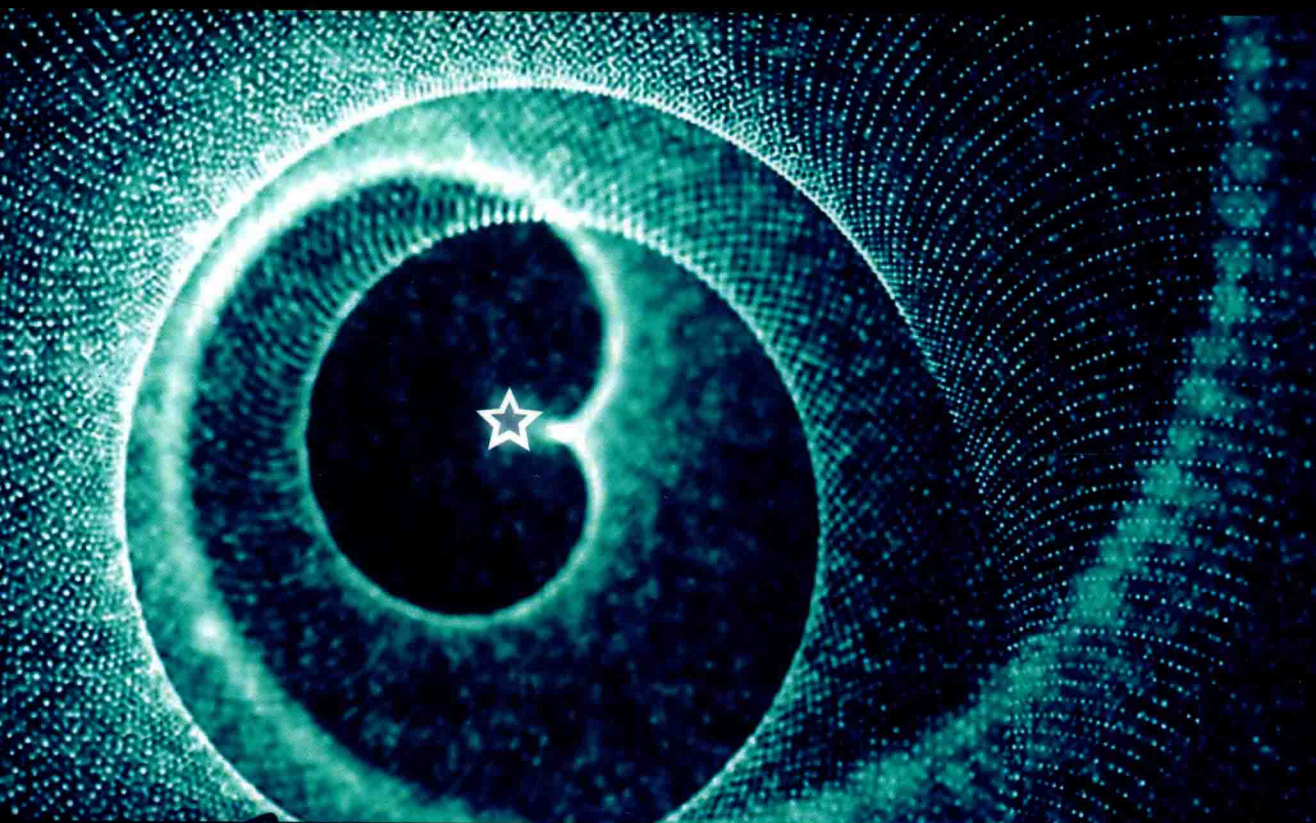


多次修订，精益求精，算法新；方便实用，数值计算的好伴侣

常用算法程序集

(C++描述) (第6版)

徐士良◎编著

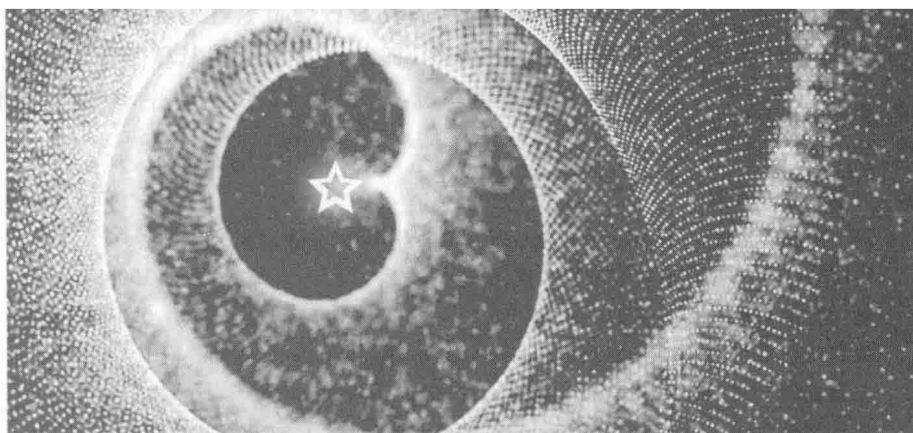


清华大学出版社

常用算法程序集

(C++描述) (第6版)

徐士良◎编著



清华大学出版社
北京

内 容 简 介

本书是针对工程中常用的行之有效的算法而编写的,其主要内容包括封装的四个基本运算类(复数运算类、实系数与复系数多项式运算类以及产生随机数类),矩阵运算,矩阵特征值与特征向量的计算,线性代数方程组,非线性方程与方程组的求解,插值与逼近,数值积分,常微分方程组的求解,数据处理,极值问题的求解,数学变换与滤波,特殊函数的计算,排序等。

书中所有的算法程序均用 C++ 描述,源代码可从清华大学出版社网站(www.tup.com.cn)下载。

本书可供广大科研人员、工程技术人员及管理工作者阅读使用,也可作为高等院校师生的参考书。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

常用算法程序集: C++ 描述/徐士良编著. —6 版. —北京:清华大学出版社,2019

ISBN 978-7-302-50542-6

I. ①常… II. ①徐… III. ①工程计算程序—程序设计 ②C 语言—程序设计 IV. ①TP319
②TP312.8

中国版本图书馆 CIP 数据核字(2018)第 141944 号

责任编辑:白立军 张爱华

封面设计:杨玉兰

责任校对:梁 毅

责任印制:杨 艳

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座

邮 编:100084

社总机:010-62770175

邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质量反馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课件下载: <http://www.tup.com.cn>, 010-62795954

印 刷 者:北京富博印刷有限公司

装 订 者:北京市密云县京文制本装订厂

经 销:全国新华书店

开 本:185mm×260mm

印 张:35.5

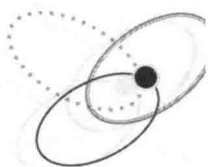
字 数:838 千字

版 次:1989 年 11 月第 1 版 2019 年 5 月第 6 版

印 次:2019 年 5 月第 1 次印刷

定 价:89.00 元

产品编号:080113-01



在本次修订中,所有的算法程序均采用 C++ 语言描述,并逐个进行了重新调试,对原来的程序做了较大的修改。对于有些问题,为了便于读者直接使用,在使用面向过程的 C++ 语言描述基础上,还使用了面向对象的 C++ 语言描述,将若干同类算法封装在一个类中。例如,在本书的第 1 章中,分别将复数运算封装成一个类,实系数多项式运算封装成一个类,复系数多项式运算封装成一个类,产生随机数运算封装成一个类;在第 12 章和第 13 章中分别将特殊函数与数据排序封装成类。由于在第 1 章中定义了复数运算类,因此在 2.1 节中矩阵相乘包括了实矩阵与复矩阵的相乘,2.2 节中的矩阵求逆包括了实矩阵与复矩阵的求逆。

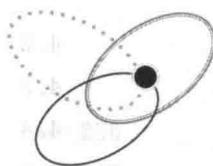
本书是针对工程中常用的行之有效的算法而编写的,并且根据算法的分类以及使用特点做了精心的组织和安排。本书具有以下特点。

- (1) 书中除收集了传统的算法外,还根据作者的工作经验和近年来数值计算的发展,选取了一些新的、实用的算法。可以说,书中各章几乎都有一些新的算法。
- (2) 书中所有的算法程序都经过认真的调试(在 Visual C++ 6.0 环境下)。
- (3) 书中收集的算法都是行之有效的,基本可以满足解决工程中各种实际问题的需要。限于作者水平,书中难免有疏漏之处,恳请读者批评指正。

作 者

2018 年 3 月

目 录



第 1 章 基本运算类

1

1.1 复数运算类	1
1.2 实系数多项式运算类	9
1.3 复系数多项式运算类	14
1.4 产生随机数类	18

第 2 章 矩阵运算

23

2.1 矩阵相乘	23
2.2 矩阵求逆	27
2.3 对称正定矩阵的求逆	33
2.4 托伯利兹矩阵求逆的特兰特方法	35
2.5 求一般行列式的值	39
2.6 求矩阵的秩	42
2.7 对称正定矩阵的乔里斯基分解	44
2.8 矩阵的三角分解	46
2.9 一般实矩阵的 QR 分解	50
2.10 一般实矩阵的奇异值分解	54
2.11 求广义逆的奇异值分解法	66

第 3 章 矩阵特征值与特征向量的计算

70

3.1 约化对称矩阵为对称三对角阵的豪斯荷尔德变换法	70
3.2 求对称三对角阵的全部特征值与特征向量	75
3.3 约化一般实矩阵为赫申伯格矩阵的初等相似变换法	79
3.4 求赫申伯格矩阵全部特征值的 QR 方法	82
3.5 求实对称矩阵特征值与特征向量的雅可比法	89
3.6 求实对称矩阵特征值与特征向量的雅可比过关法	95
3.7 乘幂法	99

第 4 章 线性代数方程组

104

4.1 求解方程组的全选主元高斯消去法	104
---------------------------	-----

4.2	求解方程组的全选主元高斯-约当消去法	109
4.3	求解三对角线方程组的追赶法	114
4.4	求解一般带型方程组	117
4.5	求解对称方程组的分解法	123
4.6	求解对称正定方程组的平方根法	127
4.7	求解托伯利兹方程组的列文逊方法	130
4.8	高斯-赛德尔迭代法	135
4.9	求解对称正定方程组的共轭梯度法	138
4.10	求解线性最小二乘问题的豪斯荷尔德变换法	141
4.11	求解线性最小二乘问题的广义逆法	144
4.12	求解病态方程组	147

第5章 非线性方程与方程组的求解

151

5.1	求非线性方程实根的对分法	151
5.2	求非线性方程一个实根的牛顿迭代法	154
5.3	求非线性方程一个实根的埃特金迭代法	157
5.4	求非线性方程一个实根的试位法	159
5.5	求非线性方程一个实根的连分式法	162
5.6	求实系数代数方程全部根的QR方法	166
5.7	求代数方程全部根的牛顿下山法	168
5.8	求非线性方程组一组实根的梯度法	176
5.9	求非线性方程组一组实根的拟牛顿法	180
5.10	求非线性方程组最小二乘解的广义逆法	185
5.11	求非线性方程一个实根的蒙特卡罗法	191
5.12	求实函数或复函数方程一个复根的蒙特卡罗法	194
5.13	求非线性方程组一组实根的蒙特卡罗法	198

第6章 插值与逼近

202

6.1	拉格朗日插值	202
6.2	连分式插值	204
6.3	埃尔米特插值	208
6.4	埃特金逐步插值	211
6.5	光滑插值	213
6.6	三次样条函数插值、微商与积分	220
6.7	二元插值	230
6.8	最小二乘曲线拟合	233
6.9	切比雪夫曲线拟合	239
6.10	最佳一致逼近的里米兹方法	243
6.11	矩形域的最小二乘曲面拟合	248

第7章 数值积分

256

7.1 变步长梯形求积法	256
7.2 变步长辛卜生求积法	258
7.3 自适应梯形求积法	261
7.4 龙贝格求积法	264
7.5 计算一维积分的连分式法	266
7.6 高振荡函数求积法	271
7.7 勒让德-高斯求积法	275
7.8 拉盖尔-高斯求积法	278
7.9 埃尔米特-高斯求积法	280
7.10 切比雪夫求积法	282
7.11 计算一维积分的蒙特卡罗法	285
7.12 变步长辛卜生二重积分法	287
7.13 计算二重积分的连分式法	291
7.14 计算多重积分的高斯方法	295
7.15 计算多重积分的蒙特卡罗法	299

第8章 常微分方程组的求解

302

8.1 积分一步的变步长欧拉方法	302
8.2 积分一步的变步长龙格-库塔方法	306
8.3 积分一步的变步长基尔方法	309
8.4 积分一步的变步长默森方法	314
8.5 积分一步的连分式法	319
8.6 积分一步的变步长特雷纳方法	325
8.7 积分一步的变步长维梯方法	331
8.8 全区间积分的双边法	335
8.9 全区间积分的阿当姆斯预报校正法	339
8.10 全区间积分的哈密方法	343
8.11 积分刚性方程组的吉尔方法	347
8.12 求解二阶初值问题的欧拉方法	361
8.13 求解二阶初值问题的连分式法	366
8.14 求解二阶边值问题的差分法	371
8.15 求解二阶边值问题的试射法	374
8.16 求解二阶边值问题的连分式法	379

第9章 数据处理

386

9.1 随机样本分析	386
9.2 一元线性回归分析	391

9.3	多元线性回归分析	394
9.4	逐步回归分析	399
9.5	半对数数据相关	409
9.6	对数数据相关	412
第10章 极值问题的求解		416
10.1	一维极值连分式法	416
10.2	n 维极值连分式法	419
10.3	不等式约束线性规划问题	424
10.4	求 n 维极值的单形调优法	429
10.5	求约束条件下 n 维极值的复形调优法	435
第11章 数学变换与滤波		444
11.1	傅里叶级数逼近	444
11.2	快速傅里叶变换	447
11.3	快速沃什变换	454
11.4	五点三次平滑	456
11.5	离散随机线性系统的卡尔曼滤波	459
11.6	$\alpha\beta\gamma$ 滤波	465
第12章 特殊函数的计算		469
12.1	伽马函数	469
12.2	不完全伽马函数	471
12.3	误差函数	475
12.4	第一类整数阶贝塞尔函数	476
12.5	第二类整数阶贝塞尔函数	482
12.6	变形第一类整数阶贝塞尔函数	487
12.7	变形第二类整数阶贝塞尔函数	491
12.8	不完全贝塔函数	495
12.9	正态分布函数	498
12.10	t 分布函数	500
12.11	χ^2 分布函数	502
12.12	F 分布函数	503
12.13	正弦积分	505
12.14	余弦积分	507
12.15	指数积分	509
12.16	第一类椭圆积分	512
12.17	第二类椭圆积分	515
12.18	特殊函数类	517

第 13 章 排序

538

13.1 冒泡排序	538
13.2 快速排序	540
13.3 希尔排序	544
13.4 堆排序	546
13.5 数据排序类	549

参考文献

555

第 1 章

基本运算类

1.1 复数运算类

【功能】

1. 计算两个复数的和
2. 计算两个复数的差
3. 计算两个复数的乘积
4. 计算两个复数的商
5. 计算复数的整数次幂
6. 计算复数的 n 次方根
7. 计算复数的指数
8. 计算复数的自然对数
9. 计算复数的正弦值
10. 计算复数的余弦值
11. 计算复数的模
12. 计算复数的幅角

在本节中, $j = \sqrt{-1}$ 。

【方法说明】

1. 计算两个复数的和

设

$$u + jv = (a + jb) + (c + jd)$$

则

$$u = a + c, \quad v = b + d$$

2. 计算两个复数的差

设

$$u + jv = (a + jb) - (c + jd)$$

则

$$u = a - c, \quad v = b - d$$

3. 计算两个复数的乘积

通常, 计算两个复数的乘积需要 4 次实数乘法, 下面的算法只需要 3 次乘法。

设

$$u + jv = (a + jb)(c + jd)$$

令

$$p = ac, \quad q = bd, \quad s = (a + b)(c + d)$$

则

$$u = p - q, \quad v = s - p - q$$

4. 计算两个复数的商

设

$$u + jv = (a + jb)/(c + jd)$$

令

$$p = ac, \quad q = -bd, \quad s = (a + b)(c - d), \quad w = c^2 + d^2$$

则

$$u = (p - q)/w, \quad v = (s - p - q)/w$$

5. 计算复数的整数次幂

设

$$u + jv = (x + jy)^n$$

令

$$z = x + jy = r(\cos\theta + jsin\theta)$$

其中

$$r = \sqrt{x^2 + y^2}, \quad \theta = \arctan \frac{y}{x}$$

则

$$u + jv = z^n = (x + jy)^n = r^n(\cos n\theta + jsin n\theta)$$

即

$$u = r^n \cos n\theta, \quad v = r^n \sin n\theta$$

6. 计算复数的 n 次方根

设复数为

$$z = x + jy = r(\cos\theta + jsin\theta)$$

其中

$$r = \sqrt{x^2 + y^2}, \quad \theta = \arctan \frac{y}{x}$$

则

$$\begin{aligned} u_k + jv_k &= z^{\frac{1}{n}} = (x + jy)^{\frac{1}{n}} \\ &= r^{\frac{1}{n}} \left(\cos \frac{2k\pi + \theta}{n} + jsin \frac{2k\pi + \theta}{n} \right), \quad k = 0, 1, \dots, n-1 \end{aligned}$$

其中

$$u_k = r^{\frac{1}{n}} \cos \frac{2k\pi + \theta}{n}, \quad v_k = r^{\frac{1}{n}} \sin \frac{2k\pi + \theta}{n}$$

7. 计算复数的指数

设复数为

$$z = x + jy$$

则

$$u + jv = e^z = e^{x+jy} = e^x (\cos y + jsiny)$$

其中

$$u = e^x \cos y, \quad v = e^x \sin y$$

8. 计算复数的自然对数

设复数为

$$z = x + jy$$

则

$$u + jv = \ln z = \ln(x + jy) = \ln \sqrt{x^2 + y^2} + j \arctan \frac{y}{x}$$

其中

$$u = \ln \sqrt{x^2 + y^2}, \quad v = \arctan \frac{y}{x}$$

9. 计算复数的正弦值

设复数为

$$z = x + jy$$

则

$$\begin{aligned} u + jv &= \sin z = \sin(x + jy) = \sin x \cos(jy) + \cos x \sin(jy) \\ &= \left(\frac{e^y + e^{-y}}{2} \right) \sin x + j \left(\frac{e^y - e^{-y}}{2} \right) \cos x \end{aligned}$$

其中

$$u = \left(\frac{e^y + e^{-y}}{2} \right) \sin x, \quad v = \left(\frac{e^y - e^{-y}}{2} \right) \cos x$$

10. 计算复数的余弦值

设复数为

$$z = x + jy$$

则

$$\begin{aligned} u + jv &= \cos z = \cos(x + jy) = \cos x \cos(jy) - \sin x \sin(jy) \\ &= \left(\frac{e^y + e^{-y}}{2} \right) \cos x - j \left(\frac{e^y - e^{-y}}{2} \right) \sin x \end{aligned}$$

其中

$$u = \left(\frac{e^y + e^{-y}}{2} \right) \cos x, \quad v = - \left(\frac{e^y - e^{-y}}{2} \right) \sin x$$

11. 计算复数的模

设复数为

$$z = x + jy$$

则复数模为

$$r = |x + jy| = \sqrt{x^2 + y^2}$$

12. 计算复数的幅角

设复数为

$$z = x + jy$$

则复数的幅角为

$$\theta = \arctan \frac{y}{x}$$

【数据成员与函数成员】

类名: complex

数据成员	说明
double R	复数的实部
double I	复数的虚部

函数成员	说明
complex(double, double)	构造函数
void prt()	复数输出。形式为(实部, 虚部)
complex operator+(complex&.)	复数加法。重载运算符+
complex operator-(complex&.)	复数减法。重载运算符-
complex operator*(complex&.)	复数乘法。重载运算符*
complex operator/(complex&.)	复数除法。重载运算符/
complex cpower(int)	复数乘幂
void croot(int, complex *)	复数的 n 次方根
complex cexp()	复数指数
complex clog()	复数对数
complex csin()	复数正弦
complex ccos()	复数余弦
double cfabs()	复数模
double angle()	复数幅角

【程序】

```
//复数运算类.h
#include <iostream>
#include <cmath>
using namespace std;
```



```

class complex
{
private:
    double R;
    double I;
public:
    complex(double real=0, double imag=0) //构造函数
    { R=real; I=imag; }
    void prt() //复数输出
    {
        cout << "(" << R << ", " << I << ")" ;
        //输出形式为 (实部, 虚部)

        return;
    }
    double cfabs() //复数模
    {
        double y;
        y=sqrt(R*R+I*I);
        return y;
    }
    double angle() //复数幅角
    {
        double y;
        y=atan2(I, R);
        return(y);
    }
    complex operator + (complex& c2) //复数加法
    {
        complex c;
        c.R=R+c2.R; c.I=I+c2.I;
        return c;
    }
    complex operator - (complex& c2) //复数减法
    {
        complex c;
        c.R=R-c2.R; c.I=I-c2.I;
        return c;
    }
    complex operator * (complex& c2) //复数乘法
    {
        complex c;
        double p, q, s;
        p=R*c2.R; q=I*c2.I;
        s=(R+I)*(c2.R+c2.I);
    }

```

```

        c.R=p-q; c.I=s-p-q;
        return c;
    }

    complex operator / (complex& c2)           //复数除法
    {
        complex c;
        double p, q, s, w;
        p=R*c2.R; q=-I*c2.I;
        s=(R+I)*(c2.R-c2.I);
        w=(c2.R)*(c2.R)+(c2.I)*(c2.I);
        if (w+1.0!=1.0)
        {
            c.R=(p-q)/w; c.I=(s-p-q)/w;
        }
        else
        {
            c.R=1e+300; c.I=1e+300;
        }
        return c;
    }

    complex cpower (int n)                     //复数乘幂
    {
        complex c;
        double r, q;
        q=atan2(I, R);
        r=sqrt(R*R+I*I);
        if (r+1.0!=1.0)
        { r=n*log(r); r=exp(r); }
        c.R=r*cos(n*q); c.I=r*sin(n*q);
        return c;
    }

    void croot (int n, complex *p)             //复数的 n 次方根
    {
        complex c;
        int k;
        double r, q, t;
        if (n<1) return;
        q=atan2(I, R);
        r=sqrt(R*R+I*I);
        if (r+1.0!=1.0)
        { r=(1.0/n)*log(r); r=exp(r); }
        for (k=0; k<n; k++)
        {
            t=(2.0*k*3.1415926+q)/n;

```

```

        c.R=r*cos(t); c.I=r*sin(t);
        p[k]=c;
    }
}

complex cexp() //复数指数
{
    complex c;
    double p;
    p=exp(R);
    c.R=p*cos(I); c.I=p*sin(I);
    return c;
}

complex clog() //复数对数
{
    complex c;
    double p;
    p=R*R+I*I;
    p=log(sqrt(p));
    c.R=p; c.I=atan2(I, R);
    return c;
}

complex csin() //复数正弦
{
    complex c;
    double p, q;
    p=exp(I); q=exp(-I);
    c.R=sin(R) * (p+q)/2;
    c.I=cos(R) * (p-q)/2;
    return c;
}

complex ccos() //复数余弦
{
    complex c;
    double p, q;
    p=exp(I); q=exp(-I);
    c.R=cos(R) * (p+q)/2;
    c.I=-sin(R) * (p-q)/2;
    return c;
}
};

```

【例】 首先输入两个复数，分别执行加、减、乘、除运算，并输出结果；然后重新输入一个复数，分别求该复数的-3次方、5次方根、指数、对数、正弦和余弦，并输出结果。

主函数程序如下：

```

//复数运算类例.cpp
#include <iostream>
#include <cmath>
#include "复数运算类.h"
using namespace std;

int main() //主函数
{
    int i;
    double a, b;
    complex c1, c2, c3, c, p[5];
    cin >> a >> b; //输入复数 c1 的实部与虚部
    c1 = complex(a, b); cout << "c1 = "; c1.prt(); cout << endl;
    cin >> a >> b; //输入复数 c2 的实部与虚部
    c2 = complex(a, b); cout << "c2 = "; c2.prt(); cout << endl;
    cin >> a >> b; //输入复数 c3 的实部与虚部
    c3 = complex(a, b); cout << "c3 = "; c3.prt(); cout << endl;
    c = c1 + c2;
    cout << "c1 + c2 = "; c.prt(); cout << endl;
    c = c1 - c2;
    cout << "c1 - c2 = "; c.prt(); cout << endl;
    c = c1 * c2;
    cout << "c1 * c2 = "; c.prt(); cout << endl;
    c = c1 / c2;
    cout << "c1 / c2 = "; c.prt(); cout << endl;
    c = c3.cpower(-3);
    cout << "c3 的-3次方 = "; c.prt(); cout << endl;
    cout << "c3 的5次方根为: " << endl;
    c3.croot(5, p);
    for (i=0; i<5; i++)
    {
        p[i].prt(); cout << endl;
    }
    c = c3.cexp();
    cout << "cexp(c3) = "; c.prt(); cout << endl;
    c = c3.clog();
    cout << "clog(c3) = "; c.prt(); cout << endl;
    c = c3.csin();
    cout << "csin(c3) = "; c.prt(); cout << endl;
    c = c3.ccos();
    cout << "ccos(c3) = "; c.prt(); cout << endl;
    return 0;
}

```

运行结果为(其中第1行为键盘输入)