

科学与工程数值算法系列丛书

科学与工程数值算法 (Java版)

丁军 杨丽丽 编著



清华大学出版社

TP312JA
15

科学与工程数值算法系列丛书

科学与工程数值算法 (Java 版)

丁 军 杨丽丽 编著

清华大学出版社

(京)新登字 158 号

内 容 简 介

本书介绍了在科学与工程实际工作中常用的数值计算算法的原理和 Java 编程方法。本书包括复数运算、矩阵运算、线性代数方程组的求解、非线性方程与方程组的求解、插值、数值积分、常微分方程初值问题的数值解法和特殊函数 8 章, 涉及使用频率非常高的 120 多个基本算法。在介绍每一算法原理的基础上, 讨论了算法的 Java 编程方法, 并用常用的数据描述算法程序的调用示例。

本书适合涉及科学与工程数值计算工作的科研人员、工程技术人员、管理人员以及大专院校相关专业的师生参考阅读。

版权所有, 翻印必究。

本书封面贴有清华大学出版社激光防伪标签, 无标签者不得销售。

图书在版编目(CIP)数据

科学与工程数值算法(Java 版)/丁军, 杨丽丽编著. —北京: 清华大学出版社, 2002

(科学与工程数值算法系列丛书)

ISBN 7-302-06073-8

I. 科... II. ①丁...②杨... III. Java 语言—程序设计—数值计算—计算方法 IV. TP312

中图版本图书馆 CIP 数据核字(2002)第 087158 号

出 版 者: 清华大学出版社(北京清华大学学研大厦, 邮编 100084)

<http://www.tup.com.cn>

<http://www.tup.tsinghua.edu.cn>

责任编辑: 刘 颖

印 刷 者: 北京通州区大中印刷厂

发 行 者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 21 字数: 499 千字

版 次: 2003 年 1 月第 1 版 2003 年 1 月第 1 次印刷

书 号: ISBN 7-302-06073-8/TP·3624

印 数: 0001~5000

定 价: 30.00 元

前 言

随着计算机技术的发展，科学计算已与科学理论、科学实验等并列为现代科学的三大组成部分。越来越多的非计算机专业的科研人员和工程技术人员，在各自的专业领域内设计实用的专业应用软件。在设计这些软件中，数值计算是必不可少的。本书以 Java 2 为平台，简单介绍算法的数学原理。在算法的选择上，我们除了选择有效的经典算法以外，也挑选了一些新的算法。目的是帮助读者理解和掌握各个算法的全过程，使读者对算法的原理、构造和应用有较深刻的理解。

读者对象

如果您需要使用 Java 进行数值计算方面的编程，那么本书可能就是您寻觅已久的助手。

本书假定读者具有一定的 Java 编程经验。如果您还不熟悉如何用 Java 来创建工程，我们建议您在阅读一本 Java 编程的教科书之后再阅读本书。

内容结构

本书介绍了 120 多个实用经典算法，每一算法都独立成节。每节都包括算法原理、算法实现和示例 3 部分。算法原理部分分别讨论了每一算法的计算原理；算法实现部分讨论了用 Java 实现算法的技巧，并给出了完整的算法函数源程序；示例部分介绍了算法函数的调用方式，给出了调用算法的示例源程序、验证算法的示例数据和运算结果。本书所有源程序都可以从 www.wenyuan.com.cn 网站下载。

本书包括复数运算、矩阵运算、线性方程组的求解、非线性方程与方程组的求解、插值、数值积分、常微分方程初值问题的数值解法和特殊函数等 8 章。各章都是在介绍算法的基础上用 Java 实现其算法，其中包括如下内容：

- 第1章 复数运算。本章介绍了与复数运算相关的算法。
- 第2章 矩阵运算。本章在介绍各种矩阵运算原理的基础上实现了矩阵基础运算。
- 第3章 线性代数方程组的求解。本章分析了线性代数方程组的常用求解算法。
- 第4章 非线性方程与方程组的求解。本章介绍了非线性方程和方程组的求解方法。
- 第5章 插值。本章介绍了常用的插值算法。
- 第6章 数值积分。本章介绍了常用的数值积分算法。
- 第7章 常微分方程初值问题的数值解法。本章介绍了常微分方程初值问题的数值解法。
- 第8章 特殊函数。本章介绍了一些特殊函数的算法。

致谢

我要感谢清华大学出版社所有对本书的出版付出过劳动的人，特别是彭欣女士，她为

本书的策划和组织付出了大量的劳动并提供了很多很好的建议。

限于笔者的能力，本书中错误之处在所难免，请读者批评指正。

作者

2002年8月19日

目 录

第 1 章 复数运算	1
1.1 复数乘法	1
1.2 复数除法	3
1.3 复数的模	5
1.4 复数的根	7
1.5 复数的实幂指数	10
1.6 复数的自然对数	11
1.7 复数的复幂指数	13
1.8 复数的正弦	15
1.9 复数的余弦	17
1.10 复数的正切	18
第 2 章 矩阵运算	21
2.1 矩阵基础运算	21
2.2 实矩阵求逆的全选主元高斯-约当法	25
2.3 复矩阵求逆的全选主元高斯-约当法	29
2.4 对称正定矩阵的求逆	32
2.5 托伯利兹矩阵求逆的特兰特方法	34
2.6 求行列式值的全选主元高斯消去法	37
2.7 求矩阵秩的全选主元高斯消去法	39
2.8 对称正定矩阵的乔里斯基分解与行列式的求值	41
2.9 矩阵的三角分解	43
2.10 一般实矩阵的 QR 分解	45
2.11 一般实矩阵的奇异值分解	48
2.12 求广义逆的奇异值分解法	54
2.13 约化对称矩阵为对称三对角阵的豪斯荷尔德变换法	56
2.14 实对称三对角阵的全部特征值与特征向量的计算	59
2.15 约化一般实矩阵为赫申伯格矩阵的初等相似变换法	61
2.16 求赫申伯格矩阵全部特征值的 QR 方法	63
2.17 求实对称矩阵特征值与特征向量的雅可比法	69
2.18 求实对称矩阵特征值与特征向量的雅可比过关法	72
2.19 求矩阵特征值和特征向量幂法	74
2.20 带位移的反幂法求矩阵特征值和特征向量	76

第 3 章 线性代数方程组的求解	80
3.1 全选主元高斯消去法.....	80
3.2 全选主元高斯-约当消去法.....	83
3.3 复系数方程组的全选主元高斯消去法.....	86
3.4 复系数方程组的全选主元高斯-约当消去法.....	89
3.5 求解三对角线方程组的追赶法.....	91
3.6 一般带型方程组的求解.....	94
3.7 求解对称方程组的分解法.....	97
3.8 求解对称正定方程组的平方根法.....	99
3.9 求解大型稀疏方程组的雅可比迭代法.....	102
3.10 求解托伯利兹方程组的列文逊方法.....	105
3.11 高斯-赛德尔迭代法.....	109
3.12 超松弛迭代法.....	111
3.13 求解对称正定方程组的共轭梯度法.....	113
3.14 求解线性最小二乘问题的豪斯荷尔德变换法.....	115
3.15 求解线性最小二乘问题的广义逆法.....	117
3.16 病态方程组的求解.....	119
3.17 范得蒙方程组的解法.....	121
第 4 章 非线性方程与方程组的求解	124
4.1 求非线性方程实根的对分法.....	124
4.2 求非线性方程一个实根的牛顿法.....	125
4.3 求非线性方程一个实根的埃特金迭代法.....	127
4.4 求非线性方程一个实根的连分式解法.....	128
4.5 求实系数代数方程全部根的 QR 方法.....	130
4.6 求实系数代数方程全部根的牛顿下山法.....	132
4.7 求复系数代数方程全部根的牛顿下山法.....	136
4.8 求非线性方程组一组实根的梯度法.....	139
4.9 求非线性方程组一组实根的拟牛顿法.....	142
4.10 求非线性方程组最小二乘解的广义逆法.....	144
4.11 求非线性方程一个根的蒙特卡洛法.....	147
4.12 求实函数或复函数方程一个复根的蒙特卡洛法.....	149
4.13 求非线性方程组一组实根的蒙特卡洛法.....	151
第 5 章 插值及数据拟合	154
5.1 一元全区间不等距插值.....	154
5.2 一元全区间等距插值.....	156
5.3 一元三点不等距插值.....	159
5.4 一元三点等距插值.....	160
5.5 连分式不等距插值.....	162

5.6	连分式等距插值.....	163
5.7	埃尔米特不等距插值.....	165
5.8	埃尔米特等距插值.....	166
5.9	埃特金不等距逐步插值.....	168
5.10	埃特金等距逐步插值.....	170
5.11	光滑不等距插值.....	171
5.12	光滑等距插值.....	173
5.13	第一种边界条件的三次样条函数插值、微商与积分.....	175
5.14	第二种边界条件的三次样条函数插值、微商与积分.....	179
5.15	第三种边界条件的三次样条函数插值、微商与积分.....	182
5.16	二元三点插值.....	186
5.17	二元全区间插值.....	188
5.18	最小二乘法曲线拟合.....	190
5.19	Chebyshev 曲线拟合.....	194
5.20	绝对值偏差最小的直线拟合.....	199
第 6 章	数值积分	203
6.1	变步长梯形求积法.....	203
6.2	变步长辛卜生求积法.....	205
6.3	自适应梯形求积法.....	206
6.4	龙贝格求积法.....	208
6.5	计算一维积分的连分式法.....	211
6.6	高振荡函数求积法.....	213
6.7	高斯-勒让德求积法.....	215
6.8	高斯-切比晓夫求积法.....	219
6.9	高斯-拉盖尔求积法.....	221
6.10	高斯-埃尔米特求积法.....	223
6.11	变步长 Simpson 法求二重积分.....	226
6.12	连分式法求二重积分.....	229
6.13	计算多重积分的高斯法.....	232
6.14	计算多重积分的蒙特卡洛方法.....	234
第 7 章	常微分方程初值问题的数值解法	237
7.1	Euler 公式.....	237
7.2	梯形公式.....	239
7.3	改进的 Euler 方法.....	241
7.4	经典 Runge-Kutta 方法.....	243
7.5	Adams 显式法.....	245
7.6	修正的 Adams 预测-校正公式.....	248
7.7	使用经典 Runge-Kutta 方法解一阶微分方程组.....	251

7.8	维梯方法	254
7.9	基尔方法	257
7.10	墨森方法	261
7.11	双边法	264
7.12	特雷纳方法	267
7.13	差分法求解二阶微分方程边值问题	271
第8章	特殊函数	275
8.1	伽马函数和贝塔函数	275
8.2	不完全伽马函数、误差函数	278
8.3	不完全贝塔函数	280
8.4	整数阶的第一类贝塞尔函数	282
8.5	整数阶的第二类贝塞尔函数	287
8.6	整数阶的第一类变型贝塞尔函数	292
8.7	整数阶的第二类变型贝塞尔函数	296
8.8	分数阶的第一、二类贝塞尔函数	300
8.9	正态分布函数	305
8.10	χ^2 分布函数	306
8.11	t 分布函数	307
8.12	F分布函数	309
8.13	正弦积分	310
8.14	余弦积分	312
8.15	第一类椭圆积分	314
8.16	第二类椭圆积分	317
8.17	指数积分	319
8.18	定指数积分	322


```
// 功能: 计算复数的乘积数:
// 参数: a ---Complex类, 乘数
//       b ---Complex类, 被乘数
//       c --- Complex类,返回积
// 返回值: Complex类, 乘积
////////////////////////////////////

public static void cheng( Complex a, Complex b, Complex c)
{ double p, q, s;
  c.real = a.real * b.real - a.image * b.image;
  c.image = a.real* b.image+ a.image* b.real;
}
//复数的其他成员函数。
}
```

3. 示例

调用函数 `cheng` 来求解 $(9 + 11i)(56+3i)$, 其程序代码如下:

```
////////////////////////////////////
// 类 名: Example1_1
// 功 能: 复数乘法的示例函数
// 使用方法: 在文本区text1,text2,text3,text4中依次输入乘数的实部、虚部和被乘数的
// 实部、虚部, 按下(计算乘积)按钮, 在文本框text5中输出结果
////////////////////////////////////

public class Example1_1 extends Applet implements ActionListener
{ TextField text1,text2,text3,text4;
  TextArea text5;
  Button button1;
  public void init()
  { text1=new TextField(10);text2=new TextField(10);
    text3=new TextField(10);text4=new TextField(10);
    text5=new TextArea(10,20);
    button1=new Button("计算乘积");
    add(text1);add(text2);add(text3);add(text4);
    add(button1);add(text5);text5.setEditable(false);
    button1.addActionListener(this);
  }
  public void actionPerformed( ActionEvent e )
  { double n1,n2,n3,n4;
    Complex a,b,c;
    if(e.getSource()==button1)
    { n1=Double.valueOf(text1.getText()).doubleValue();
      n2=Double.valueOf(text2.getText()).doubleValue();
      n3=Double.valueOf(text3.getText()).doubleValue();
      n4=Double.valueOf(text4.getText()).doubleValue();
      a=new Complex();Complex.init(n1,n2,a);
      b=new Complex();Complex.init(n3,n4,b);
      c=new Complex();Complex.init(0,0,c);
      text5.setText("");
      text5.append("复数 A="+a.real+")"+"a.image+")I\n");
    }
  }
}
```

```

text5.append("复数 B=("+b.real+")"+"+b.image+") I\n");
Complex.cheng(a,b,c);
text5.append(它们的乘积 A*B=("+c.real+")"+"+c.image+") I\n");
}
}

```

其输出结果如图 1.1 所示。

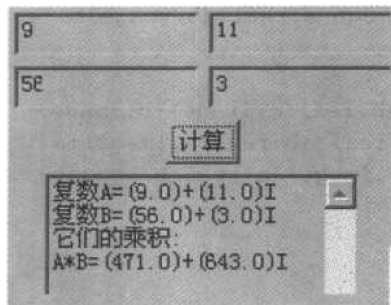


图 1.1 程序输出结果

1.2 复数除法

1. 算法原理

对复数 $z_1 = a + ib$, $z_2 = c + id$, z_1 除以 z_2 可用如下公式计算:

$$\frac{z_1}{z_2} = \frac{a + ib}{c + id} = \frac{(ac + bd) + i(bc - ad)}{c^2 + d^2} = \begin{cases} \frac{\left(a + b\frac{d}{c}\right) + j\left(b - a\frac{d}{c}\right)}{c + d\frac{d}{c}} & |c| \geq |d| \\ \frac{\left(a\frac{c}{d} + b\right) + j\left(b\frac{c}{d} - a\right)}{c\frac{c}{d} + d} & |c| < |d| \end{cases}$$

后面的等式用于防止上溢和下溢。

2. 算法实现

根据上述算法, 可以实现计算复数的除法的 Java 函数 `chu`, 其代码如下:

```

////////////////////////////////////
// 函数名: chu (从属于Complex类)
// 功能: 计算复数的商
// 参数: a ---Complex类, 被除数
//       b ---Complex类, 除数
//       c --- Complex类, 返回商
// 返回值: Complex类, 乘积
////////////////////////////////////

class Complex

```

```

{ double real,image;
//复数的除法函数
public static void chu ( Complex a, Complex b, Complex c)
{   double w;
    if(Math.abs(b.real)>Math.abs(b.image))
    {   w=b.real+b.image*(b.image/b.real);
        c.real=(a.real+a.image*(b.image/b.real))/w;
        c.image=(a.image-a.real*(b.image/b.real))/w;
    }
    else
    {   w=b.real*(b.real/b.image)+b.image;
        c.real=(a.real*(b.real/b.image)+a.image)/w;
        c.image=(a.image*(b.real/b.image)-a.real)/w;
    }
}
}

```

3. 示例

调用函数 `Chu` 来求解 $(1+2i)/(3+4i)$, 其程序代码如下:

```

////////////////////////////////////
// 类 名: Example1_2
// 功 能: 复数除法的示例函数
// 使用方法: 在文本区text1,text2,text3,text4中依次输入除数的实部、虚部和被除数的
//           实部、虚部, 按下(计算商)按钮, 在文本框text5中输出结果
////////////////////////////////////

public class Example1_2 extends Applet implements ActionListener
{   TextField text1,text2,text3,text4;
    TextArea text5;
    Button button1;
    //界面初始化函数
public void init()
    {   text1=new TextField(10);text2=new TextField(10);
text3=new TextField(10);text4=new TextField(10);
text5=new TextArea(10,50);button1=new Button("计算商");
    add(text1);add(text2);add(text3);
add(text4);add(button1);add(text5);
text5.setEditable(false);
button1.addActionListener(this);
    }
    //按钮监听事件
public void actionPerformed(ActionEvent e)
    {   double n1,n2,n3,n4;
        Complex a,b,c;
        if( e.getSource()==button1 )
    {   n1=Double.valueOf(text1.getText()).doubleValue();
        n2=Double.valueOf(text2.getText()).doubleValue();
        n3=Double.valueOf(text3.getText()).doubleValue();
        n4=Double.valueOf(text4.getText()).doubleValue();
        a=new Complex();b=new Complex();c=new Complex();
        Complex.init(n1,n2,a);Complex.init(n3,n4,b);
    }
}

```

```

Complex.init(0,0,c);text5.setText("");
text5.append("复数A="+a.real+")"+"+"+a.image+"I\n");
text5.append("复数B="+b.real+")"+"+"+b.image+"I\n");
Complex.chu(a,b,c);
text5.append("它们的商: \nA/B="+c.real+")"+"+"+c.image+"I\n");
}
}
}

```

其输出结果如图 1.2 所示。

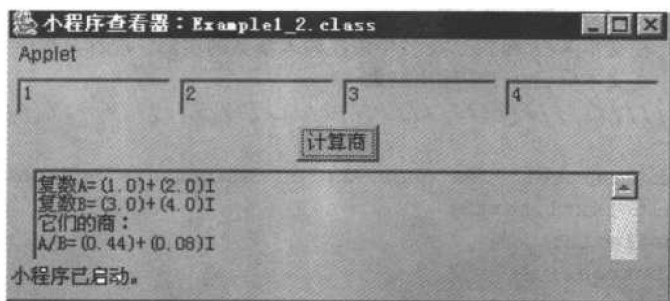


图 1.2 程序输出结果

1.3 复数的模

1. 算法原理

对复数 $z = x + iy$ ，其模 $|z|$ 定义如下：

$$|z| = \sqrt{x^2 + y^2} = \begin{cases} |x| \sqrt{1 + \left(\frac{y}{x}\right)^2}, & |x| > |y| \\ |y| \sqrt{1 + \left(\frac{x}{y}\right)^2}, & |x| \leq |y| \end{cases}$$

后面的等式用于防止上溢和下溢。

2. 算法实现

根据上述算法，可以实现计算复数的模的 Java 函数 `abs`，其代码如下：

```

////////////////////////////////////
// 函数名: abs (从属于Complex类)
// 功能: 计算复数的模
// 参数: a---Complex类 复数
// 返回值: double 型, 模的数值
////////////////////////////////////

public static double( Complex a)
{ double x,y,r;
  x=Math.abs(a.real);y=Math.abs(a.image);

```

```

        if(x>y) r=x*Math.sqrt(1+(y/x)*(y/x));
        else   r=y*Math.sqrt(1+(x/y)*(x/y));
        return r;
    }

```

3. 示例

调用函数 `abs` 来求解复数 $12.3 + 45.6i$ 的模，其程序代码如下：

```

////////////////////////////////////
// 类 名: Example1_2
// 功 能: 复数乘法的示例函数
// 使用方法: 在文本区text1,text2中依次输入复数的实部、虚部，按下（计算模）按钮，
//           在文本框text3中输出结果
////////////////////////////////////

public class Example1_3 extends Applet implements ActionListener
{
    TextField text1,text2;
    TextArea text3;
    Button button1;
//界面初始化函数
public void init()
    {
        text1=new TextField(10); text2=new TextField(10);
        text3=new TextArea(10,50);button1=new Button("计算模");
        add(text1);add(text2);add(button1);add(text3);
        text3.setEditable(false);
        button1.addActionListener(this);
    }
//按钮监听事件
public void actionPerformed(ActionEvent e)
    {
        double n1,n2,r;
        Complex a;
        if(e.getSource()==button1)
        {
            n1=Double.valueOf(text1.getText()).doubleValue();
            n2=Double.valueOf(text2.getText()).doubleValue();
            a=new Complex();Complex.init(n1,n2,a);
            text3.setText("");
            text3.append("复数A="+a.real+" "+a.image+" I\n");
            r=Complex.abs(a);text3.append("它的模: |A|="+r+"\n");
        }
    }
}

```

其输出结果如图 1.3 所示。

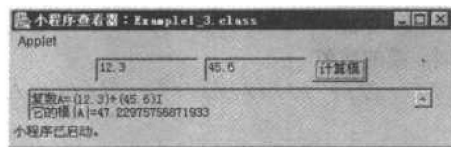


图 1.3 运行结果

1.4 复数的根

1. 算法原理

对复数 $z = x + iy$, 其全部 n 次根的计算方法如下:

$$\begin{aligned}
 z^{\frac{1}{n}} &= (x + iy)^{\frac{1}{n}} = [r(\cos \phi + i \sin \phi)]^{\frac{1}{n}} \\
 &= r^{\frac{1}{n}} \left(\cos \frac{\phi + 2k\pi}{n} + i \sin \frac{\phi + 2k\pi}{n} \right) \\
 &= r^{\frac{1}{n}} (\cos (\theta + k\omega) + i \sin (\theta + k\omega)) \\
 &= r^{\frac{1}{n}} \cos (\theta + k\omega) + i r^{\frac{1}{n}} \sin (\theta + k\omega) \\
 &= r^{\frac{1}{n}} (\cos \theta \cos k\omega - \sin \theta \sin k\omega) + i r^{\frac{1}{n}} (\sin \theta \cos k\omega + \cos \theta \sin k\omega)
 \end{aligned}$$

其中,

$$\begin{aligned}
 k &= 0, 1, \dots, n-1 \\
 \omega &= \frac{2\pi}{n} \\
 r &= \sqrt{x^2 + y^2} \\
 \theta &= \begin{cases} 0 & a > 0, b = 0 \\ \frac{\pi}{n} & a < 0, b = 0 \\ \frac{\pi}{2n} & a = 0, b > 0 \\ -\frac{\pi}{2n} & a = 0, b < 0 \\ \frac{\arctan(\frac{b}{a})}{n} & a > 0, b \neq 0 \\ \frac{\arctan(\frac{b}{a}) + \pi}{n} & a < 0, b \neq 0 \end{cases}
 \end{aligned}$$

因此复数 $z = x + iy$ 的全部 n 次根可表示为:

$$x_k + iy_k, \quad k = 0, 1, \dots$$

其中,

$$\begin{aligned}
 x_k &= r^{\frac{1}{n}} (\cos \theta \cos k\omega - \sin \theta \sin k\omega) \\
 y_k &= r^{\frac{1}{n}} (\sin \theta \cos k\omega + \cos \theta \sin k\omega)
 \end{aligned}$$

注: 在 Java 中没有开 n 次方的函数, $r^{\frac{1}{n}}$ 改用 $e^{\frac{\ln r}{n}}$ 来计算。

2. 算法实现

根据上述方法,可以实现计算复数的根的Java函数Root。为了减少正弦和余弦的运算量,Root函数使用了如下的递推公式:

$$\begin{cases} \sin k\omega = \sin(k-1)\omega \cos \omega + \cos(k-1)\omega \sin \omega \\ \cos k\omega = \cos(k-1)\omega \cos \omega - \sin(k-1)\omega \sin \omega \end{cases}$$

Root函数代码如下:

```

////////////////////////////////////
// 函数名: Root (从属于Complex类)
// 功能: 计算复数的根
// 参数: a---Complex类复数
//       b---Complex类复数数组,用于返回所有的根
//       n---整数,开方次数
// 返回值: 无
////////////////////////////////////

public static void Root( Complex a, Complex[] b, int n)
{   double r,sita,w=2*Pi/n,c,s,csita,ssita,cw,sw,tc;
    r=Complex.abs(a);
    if(a.image==0)
    {   if(a.real>=0)sita=0;
        else sita=Math.PI/n;
    }
    else if(a.real==0)
    {   if(a.real>0)sita= Math.PI /2/n;
        else sita=- Math.PI/2/n;
    }
    else
    {   if(a.real>0) sita=Math.atan(a.image/a.real)/n;
        else sita=(Math.atan(a.image/a.real)+Pi)/n;}
        csita=Math.cos(sita);ssita=Math.sin(sita);
        cw=Math.cos(w);sw=Math.sin(w);
        c=1;s=0;r=Math.sqrt(Math.sqrt(r));
        for(int k=0;k<n;k++)
        { b[k].real=r*(csita*c-ssita*s);
          b[k].image=r*(ssita*c+csita*s);
          tc=c;c=c*cw-s*sw;s=s*cw+tc*sw;
        }
    }
}

```

3. 示例

调用函数Root求解复数 $12+5i$ 的4次根,其程序代码如下:

```

////////////////////////////////////
// 类   名: Example1_3
// 功   能: 复数开方示例函数
// 使用方法: 在文本区text1,text2中依次输入复数的实部、虚部,在文本区text3输入开方次
//           数,按下(计算根)按钮,在文本框text4中输出结果
////////////////////////////////////

```