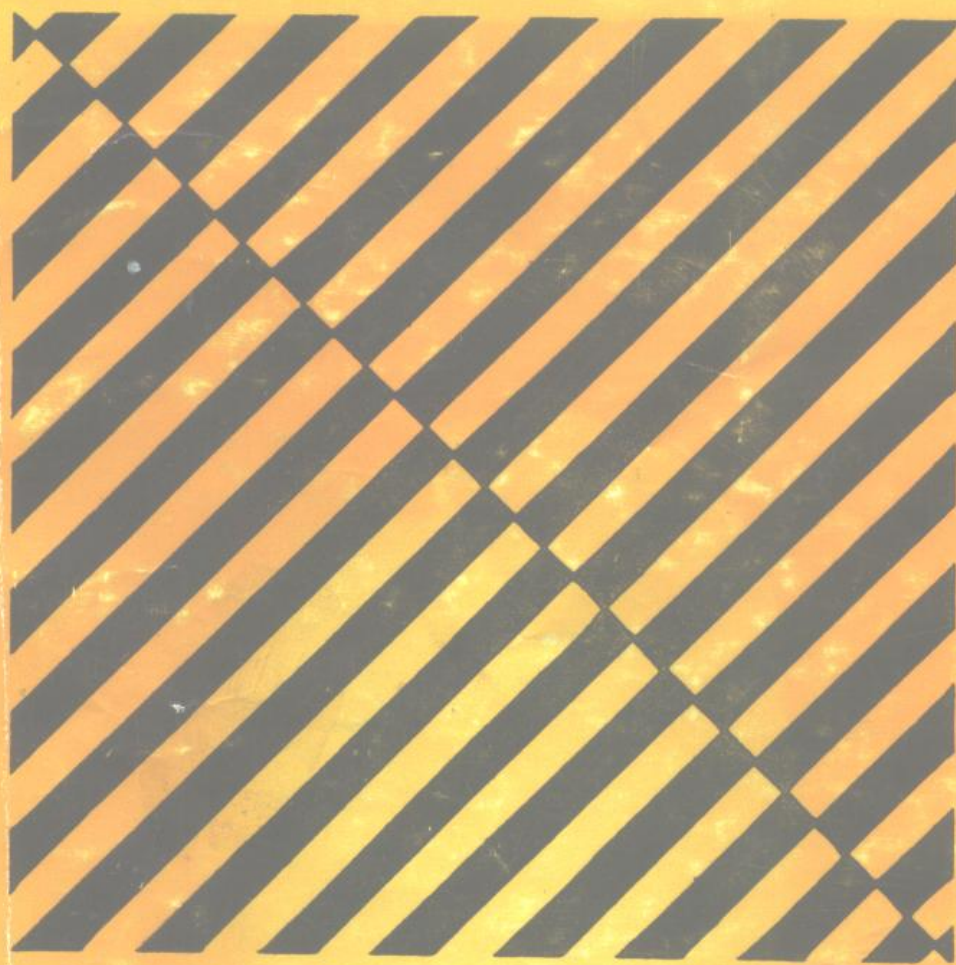


PASCAL

管理程序库



杨保群 邓幼强 李树良 编著

西南交通大学出版社

PASCAL 管 理 程 序 库
PASCAL GUANLI CHENG XU KU

杨保群 邓幼强 李树良 编著

西南交通大学出版社

内 容 提 要

本书由 55 个实用的 PASCAL 管理程序组成, 包括数值计算、矩阵运算、运筹学、回归分析、概率统计、技术经济学等方面。

书中程序按标准 PASCAL 语法编写, 全部上机通过, 可不加修改地在 IBM-PC 和 APPLE II 微机上运行, 每一程序的使用方法都结合例题作了详尽介绍。程序中的人机对话和注释均用汉字。各程序严格遵循结构化设计规则, 采用模块结构, 具有可读性好、编程风格优美的特点。

本书适用于大专院校本科生、研究生和企业管理人员、科技人员作为学习 PASCAL 语言的参考书和案头程序库, 也可供企业在建立管理信息系统时作为模型库使用。

75220/107

PASCAL 管 理 程 序 库 PASCAL GUANLI CHENG XU KU

杨保群 邓幼强 李树良 编

西南交通大学出版社出版

四川省新华书店发行

西南交通大学出版社印刷厂印刷

开本: 787×1092 1/16 印张: 19.56

1988 年 4 月第一版 1988 年 4 月第一次印刷

字数: 501 千字 印数: 1~3500 册

TSBN 7-81022-023-3/TP 003

定价: 3.60 元

前 言

PASCAL 是一种较新的在国际上广泛流行的高级语言。它具有丰富完备的数据类型, 简单易懂的编程语句, 清晰明了的模块结构, 自由灵活的书写格式, 以及编译的目标码质量好、运行效率高和程序设计风格优美等特点。近几年来, PASCAL 语言在我国正日益受到重视, 越来越多的高等院校选用 PASCAL 作为计算机教学的第一语言。

然而, 我们在实际工作中发现, 许多人尽管学习了 PASCAL 语言, 但在解决实际问题时仍然转而采用 BASIC 或 FORTRAN 程序。究其原因, 是这些年来已经开发了许多解决各种具体问题的现成 BASIC 或 FORTRAN 程序, 使用起来非常方便。而 PASCAL 语言由于在我国推广较晚, 除了在教科书中可以找到为理解语句功能而编写的各种示范程序外, 真正有实用价值的程序还很少见。因此, 编写实用的 PASCAL 程序库, 以便推动计算机技术的广泛应用和 PASCAL 语言的进一步普及, 无疑是十分必要的。

本书汇集和编写了在日常管理工作中常用的 55 个 PASCAL 程序, 内容包括数值计算、矩阵运算、运筹学、回归分析、概率统计、技术经济学等各个方面, 为了使程序能使用于不同机型, 这些程序在编写时均采用标准 PASCAL 语法, 避免使用各种不同 PASCAL 版本中扩充的语句。本书程序全部在 IBM-PC 微机上通过, 也可不加修改地在 APPLE II 微机上运行。此外, 本书程序经过汉化, 程序中有许多汉字注释和提示信息(没有汉字处理功能的计算机, 可将汉字改成英文或汉语拼音), 具有很好的可读性和易理解性。

书中各个程序严格遵循结构化程序设计规则, 采用模块结构, 充分发挥了 PASCAL 语言作为结构化程序设计语言的特点, 表现出编程风格严谨、设计思路清晰、程序文体优美、书写格式明快的编程技巧。学习和使用 PASCAL 语言的读者, 不仅可以从中领悟出解决实际问题的思路, 直接使用各个程序的功能, 而且从本书提供的几百个程序模块中, 还可以方便地拼装、构造出各种新程序来。

本书中每个程序不仅给出了程序清单, 而且还对所解决的问题作了详细讨论, 指出了该程序所解决问题的适用范围和条件、必需的输入数据和运行后的输出结果。每个程序都通过实际例子说明使用步骤。

本书程序有繁有简, 有难有易, 适用面广, 无论是 PASCAL 语言的初学者还是有较高水平的读者均可以从中获益。本书适用于大专院校本科生、研究生和企业管理人员、科技人员作为学习 PASCAL 语言的参考书和案头程序库, 也可供企业在建立管理信息系统时作为模型库使用。

本书第一、五、六章由邓幼强同志编写, 第二、三、四章由李树良同志编写。杨保群教授指导了本书的编写工作, 仔细审阅了全部书稿并统纂全书。

衷心希望读者提出批评和建议。

编 者

1987年11月

目 录

第一章 数值计算程序

1.1	解线性方程组	1
1.2	曲线插值	7
1.3	牛顿插值	11
1.4	解一元二次方程	16
1.5	牛顿法求高次方程的实根	19
1.6	半区间搜索法求高次方程的实根	25
1.7	求三角多项式的值	34
1.8	辛普生公式求定积分	41
1.9	梯形公式求定积分	45
1.10	高斯公式求定积分	49
1.11	龙格公式求定积分	53
1.12	求已知点的导数	57

第二章 矩阵运算程序

2.1	矩阵的加减, 数与矩阵相乘	60
2.2	矩阵相乘	67
2.3	矩阵求逆	72

第三章 运筹学程序

3.1	线性规划问题	78
3.2	计划评审技术 (PERT)	90
3.3	马尔柯夫分析	102
3.4	贝叶斯决策分析	112
3.5	最佳订货量 (EOQ)	121
3.6	最佳生产量 (EPQ)	127
3.7	运输问题	130
3.8	排队问题	142
3.9	决策分析	151

第四章 回归分析程序

4.1	线性相关系数	160
-----	--------	-----

4.2	一元线性回归分析.....	164
4.3	多元线性回归分析.....	169
4.4	N阶多项式回归分析.....	177
4.5	幂函数回归分析.....	185
4.6	指数回归分析.....	191
4.7	绘制函数图形.....	196

第五章 概率统计程序

5.1	均值、方差和标准差.....	202
5.2	几何均值与离差.....	207
5.3	二项分布.....	210
5.4	泊松分布.....	214
5.5	正态分布.....	218
5.6	χ^2 -分布.....	222
5.7	“学生氏” t -分布.....	226
5.8	t -检验.....	231
5.9	F -分布.....	237

第六章 技术经济学程序

6.1	非线性盈亏分析.....	243
6.2	一次投资的未来值.....	250
6.3	系列等额投资的未来值.....	254
6.4	预期未来值的等额投资资金.....	257
6.5	等额回收序列资金.....	261
6.6	回收等额序列资金的投资现值.....	264
6.7	预期未来值的初始投资.....	268
6.8	投资本息表.....	271
6.9	年折旧率.....	280
6.10	年折旧额.....	283
6.11	折余残值.....	287
6.12	贷款的分期偿还金额.....	290
6.13	贷款的偿还期.....	294
6.14	分期还款清单.....	297
6.15	求两日期之间的天数.....	305

第一章 数值计算程序

1.1 解线性方程组

本程序用于求解线性方程组，要求方程组中变量的个数与方程式的个数必须相等。在程序的常量说明中，Maxequations 规定了方程组最多可求解的方程式个数（本程序指定为 50 个），如欲增加，可改变 Maxequations 的值，同时须使常量 Plusmax 的值为 Maxequations+1。每一个方程的系数与常数值，应逐一输入。

计算实例

解线性方程组：

$$\begin{cases} 2x_1 + 3x_2 - 4x_3 + x_4 & = 14 \\ x_1 + 2x_2 + x_3 - 3x_4 + x_5 & = 4 \\ 3x_1 + x_2 + 2x_3 + 2x_4 + 2x_5 & = 41 \\ x_1 + x_2 & + x_5 = 14 \\ 3x_1 & + 4x_3 + x_5 = 20 \end{cases}$$

运行过程

****解线性方程组****

输入方程式个数(限于整数1..50) 5

输入方程式的系数和常数

方程 1

系数 1: 2

系数 2: 3

系数 3: -4

系数 4: 1

系数 5: 0

常数 : 14

方程 2

系数 1: 1

系数 2: 2

系数 3: 1

系数 4: -3

系数 5: 1

常数 : 4

方程 3

系数 1: 3
系数 2: 1
系数 3: 2
系数 4: 2
系数 5: 2
常数 : 41

方程 4

系数 1: 1
系数 2: 1
系数 3: -1
系数 4: 0
系数 5: 1
常数 : 14

方程 5

系数 1: 3
系数 2: 0
系数 3: 4
系数 4: 0
系数 5: 1
常数 : 20

答案

x 1 = 2.000
x 2 = 3.000
x 3 = 1.000
x 4 = 5.000
x 5 = 10.000

本程序需要继续运行吗? (Y/N)N

运算结束, 再见!

源程序清单

```
PROGRAM Simlequations(INPUT,OUTPUT);  
{本程序用于求解线性方程组}  
  
CONST  
  Maxequations=50; {方程式数的最大值,可改变}  
  Plusmax=51;      {其值为 Maxequations+1 }  
  
TYPE  
  index=1..Maxequations;
```

```

MatrixARRAY=ARRAY[index,1..Plusmax] OF REAL;

VAR
  Matrix: MatrixARRAY;
  Numequations: index;
  Nosolution: BOOLEAN;

PROCEDURE readdata; (接收数据模块)

TYPE
  set0=SET OF index;
VAR
  i,j,k: INTEGER;
  coeff: REAL;
  boo: BOOLEAN;
  set1:set0;

BEGIN
  set1:= [ 1..maxequations ];

REPEAT
  boo:=TRUE;
  WRITE('输入方程式个数(限于整数1..50) ');
  readln(k);
  IF k IN set1 THEN Numequations:=k
  ELSE
    BEGIN
      WRITELN;
      WRITE('输入有错,请重新输入!');
      WRITELN;
      boo:=FALSE;
      END;
    UNTIL boo;
  WRITELN('输入方程式的系数和常数 ');
  FOR i:=1 TO Numequations DO
    BEGIN
      WRITELN('方程',i:2);
      FOR j:=1 TO Numequations DO
        BEGIN
          WRITE('系数':8,j:2,' ');
          READLN(coeff);

```

```

    Matrix[i,j]:=coeff
END;
    WRITE('常数 : ':11);
    READLN(coeff);
    Matrix[i,j]:=coeff
END
END;

```

PROCEDURE swap(i,j:index); {交换模块}

```

VAR
    count: INTEGER;
    temp: REAL;

BEGIN
    FOR count:=1 TO (Numequations+1) DO
        BEGIN
            temp:=Matrix[i,count];
            Matrix[i,count]:=Matrix[j,count];
            Matrix[j,count]:=temp
        END;
    END;
END;

```

PROCEDURE subtract(computed:index); {行对应元素相减}

```

VAR
    i,k,count: INTEGER;
    temp: REAL;

BEGIN
    temp:=1/Matrix[computed,computed];
    FOR count:=1 TO (Numequations+1) DO
        Matrix[computed,count]:=temp * Matrix[computed,count];
    FOR i:=1 TO Numequations DO
        IF i < computed THEN
            BEGIN
                temp:=(-Matrix[i,computed]);
                FOR k:=1 TO (Numequations+1) DO
                    Matrix[i,k]:=Matrix[i,k]+(temp * Matrix[computed,k])
                END
            END;
        END;
    END;
END;

```

END;

PROCEDURE calculate; {计算模块}

VAR

 computed, count: index;

BEGIN

 Nosolution:=FALSE;

 computed:=1;

 WHILE (computed <= Numequations) AND (NOT Nosolution) DO

 BEGIN

 Nosolution:=TRUE;

 count:=computed;

 WHILE (count <= Numequations) AND (Nosolution) DO

 IF Matrix[count,computed] <> 0 THEN

 Nosolution:=FALSE

 ELSE

 count:=count+1;

 IF NOT Nosolution THEN

 BEGIN

 swap(computed, count);

 subtract(computed)

 END;

 computed:=computed+1

 END

 END;

PROCEDURE print; {打印模块}

VAR

 i, j: INTEGER;

BEGIN

 Writeln('答案');

 FOR i:=1 TO Numequations DO

 Writeln(' x', i:2, ' =':3, Matrix[i, Numequations+1]:10:3);

END;

```

FUNCTION NotAgain: BOOLEAN;

VAR
  c: CHAR;

BEGIN
  WRITE(' 本程序需要继续运行吗? (Y/N)');
  READLN(c);
  WHILE NOT (c IN ['y', 'Y', 'n', 'N']) DO
    BEGIN
      WRITE('需继续运行, 键入 Y 或 YES, 否则键入 N 或 NO');
      READLN(c);
    END;
    NotAgain := c IN ['n', 'N'];
  END;

```

```

BEGIN (主程序)
  WRITELN;
  WRITELN('****解线性方程组**** ');
  REPEAT;
    readdata;
    calculate;
    IF Nosolution THEN
      BEGIN
        WRITELN; WRITELN('无唯一解 ! ');
        WRITELN
      END
    ELSE
      print;
  UNTIL NotAgain;
  WRITELN('运算结束, 再见!');
END.

```

1.2 曲线插值

本程序用于计算曲线插值。在一些实际问题里用到的函数 $y = f(x)$ 往往很复杂, 难以写出其具体的表达式, 但通过实验观察, 可得到该曲线上某区间 $[x_0, x_n]$ 的若干点 (x_0, y_0) 、 (x_1, y_1) 、 $\dots$ 、 (x_n, y_n) , 依据这些点的数据, 如何求得该区间上任意点 x 的函数值 y ? 这就是曲线插值要解决的问题。

本程序按拉格朗日插值法进行计算。程序运行时, 需输入曲线上已知点的数目, 并逐一输入各点的坐标值。需注意, 坐标值中不允许有 $x_i = x_j (i \neq j)$ 的情况。

程序对已知点数目限定最多为 50, 如欲增加, 可修改常量 maxpoints 的值。

计算实例

已知曲线过如下各点: $(-3, -15)$ 、 $(1, 1)$ 、 $(-2, 1)$ 、 $(2, 5)$ 、 $(-1, 5)$ 、 $(3, 21)$ 、 $(0, 3)$, 求 $x = -1.65$ 和 $x = 0.2$ 时的 y 值。

运行过程

***** 曲线插值 *****

请输入以下数据

已知点数目: 7

第1点

X = -3

Y = -15

第2点

X = 1

Y = 1

... ..

第7点

X = 0

Y = 3

插值计算: X = -1.65

Y = 3.45788

继续在该曲线上插值吗?(Y/N) Y

插值计算: X = 0.2

Y = 2.40800

继续在该曲线上插值吗?(Y/N) N

本程序需要继续运行吗? (Y/N)N

源程序清单

```
PROGRAM Interpolate(INPUT,OUTPUT);
{ 本程序用于计算曲线插值}

CONST
TYPE
    index=1..maxpoints;
    arraycord=ARRAY[index] OF REAL;

VAR
    Xcord,Ycord:arraycord;
    Amount: INTEGER;
    answer: CHAR;

FUNCTION ReadInt(VAR i: INTEGER; low,high: INTEGER): BOOLEAN;
{ 读入整型数函数}

BEGIN
    i:=0;
    WHILE NOT eoln AND NOT (INPUT^ IN ['0'..'9']) DO
        GET(INPUT);
    IF eoln THEN ReadInt:=FALSE
    ELSE
        BEGIN
            WHILE INPUT^ IN ['0'..'9'] DO
                BEGIN
                    i:=i*10+ORD(INPUT^)-ORD('0');
                    GET(INPUT);
                END;
            ReadInt:=(i>=low) AND (i<=high)
        END;
    READLN
END;
```

```

FUNCTION NotAgain: BOOLEAN;

VAR
    c: CHAR;

BEGIN
    WRITE('本程序需要继续运行吗? (Y/N)');
    READLN(c);
    WHILE NOT (c IN ['y', 'Y', 'n', 'N']) DO
        BEGIN
            WRITE('需继续运行, 键入 Y 或 YES, 否则键入 N 或 NO');
            READLN(c);
        END;
    NotAgain:=c IN ['n', 'N']
END;

PROCEDURE ReadData;

VAR
    i: index;
    X, Y: REAL;

BEGIN
    WRITELN; WRITELN('请输入以下数据'); WRITELN;
    REPEAT
        WRITE('    已知点数目: ');
    UNTIL ReadInt(Amount, 1, maxpoints);
    FOR i:=1 TO Amount DO
        BEGIN
            WRITELN('    第', i:0, '点');
            WRITE('X = ':12);
            READLN(X);
            Xcord[i]:=X;
            WRITE('Y = ':12);
            READLN(Y);
            Ycord[i]:=Y
        END;
    WRITELN;
END;

```

```

PROCEDURE Calculate;

VAR
    Alpha, Beta, IntPol: REAL;
    i, j: index;

BEGIN
    WRITE('插值计算: X = ');

    READLN(IntPol);
    Beta:=0;
    FOR i:=1 TO Amount DO
        BEGIN
            Alpha:=1;
            FOR j:=1 TO Amount DO
                IF i<>j THEN
                    Alpha:=(IntPol-Xcord[j])/
                        (Xcord[i]-Xcord[j]);
                    Beta:=Beta+(Alpha*Ycord[i])
                END;
            WRITELN('Y = ':14, Beta:1:5);
            WRITELN
        END;

BEGIN (主程序)
    WRITELN;
    WRITELN('*****曲线插值*****');
    REPEAT
        WRITELN;
        ReadData;
        REPEAT
            WRITELN;
            Calculate;
            WRITE('继续在该曲线上插值吗?(Y/N) ');
            READLN(answer);
            UNTIL answer IN ['N', 'n'];
            WRITELN;
        UNTIL NotAgain
    END.

```

1.3 牛 顿 插 值

本程序使用牛顿向前插值公式求给定函数的插值。

程序运行时,需输入以下数据:欲插入点左右两侧(相对数轴上的位置而言)相邻点的 x 值;欲插入点的 x 值,要求的计算精度的位数(本程序指定为6位),左右两侧相邻点 x 对应的 y 值。程序打印出按上述值计算的一阶差分,并要求再输入前向结点的坐标,逐次打印出新的差分值,直到新的差分满足预定的精度要求,程序便输出计算结果。

计算实例

从三角函数表中查得正弦函数值如下:

x	0.6	0.7	0.8	0.9
$\sin(x)$	0.564642	0.644218	0.717358	0.783327

求 0.63 的正弦函数值。

运行过程

*****牛顿向前插值公式计算插值*****

请输入以下数据

插值点左侧 x 值: 0.6

插值点右侧 x 值: 0.7

插值点的 x 值: 0.63

精度要求的小数点位数(从0到6): 6

输入 x 为0.60时的 y 值: 0.564642

输入 x 为0.70时的 y 值: 0.644218

计算插值

1阶差分值 = 0.079576

输入 x 为0.80时的 y 值: 0.717358

计算插值

2阶差分值 = -0.006435

计算结果

经2阶差分的插值 = 0.589191

本程序需要继续运行吗? (Y/N)N