

使用微机的材料力学

〔日〕 岩佐哲夫 樱井惠三 著
中野将治 宫井义裕

科学出版社

使用微机的材料力学

〔日〕 岩佐哲夫 櫻井惠三 著
中野将治 宫井义裕

樊发兴 译

科 学 出 版 社

1988

内 容 简 介

随着微机的普及, 计算机已逐渐渗透到日常生活和教学的各个领域。在发达国家, 尤其日本, 有人设想把微机引进基础理论教学, 开发“教学软件”, 使学生在学基础理论的同时, 学会用 BASIC 语言设计程序, 反过来, 通过程序设计更加加深对基础理论的认识。其目的是使学生兼备专业基础和计算机知识。本书就是这一设想的产物。

本书把材料力学的基本公式及其应用都用 BASIC 语言进行了程序设计。学过材料力学的广大工程技术人员将发现, 本书使你的专业知识与计算机结合起来, 使你发挥更大潜力; 学过计算机的人通过本书可使你知道怎样把你的知识用于实际工作、解决实际问题, 初学者和教育工作者将会发现这是一本不可多得的好教材。

岩佐哲夫 櫻井惠三 中野将治 宫井義裕

パソコンを活用した材料力学

日刊工業新聞社, 1983

使用微机的材料力学

〔日〕岩佐哲夫・櫻井惠三 著
中野将治 宫井義裕

樊发兴 译

责任编辑 杨 龄 李成香

科学出版社出版

北京朝阳门内大街137号

中国科学院印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

1988年4月第 一 版 开本: 787×1092 1/32

1988年4月第一次印刷 印张: 4 3/4

印数: 0001—3,210 字数: 106,000

ISBN 7-03-000084-6/O·23

定价: 1.30元

。版变典一第变需。和算因性用解由出
管些益向新会册地指普中。一普送册下送送普本。刊显
出代现大册国译业工册日序公册新言册下刊公及划译册新册
意情心要示民

1986年10月出版

随着个人用计算机（微机）的普及，计算机已成为日常生活中常用的工具。但另一方面，由于制造商们的竞争日益激烈，造成了一个错觉，好象谁都能轻而易举地掌握计算机似的。因此，有人以为，既然微型计算机能够改善自己的工作，价格又不贵，何不将它买来。然而，一经使用，他们就会发现不如想象的那样简单了。他们要么只依靠计算机系统公司的程序使用计算机，用过之后，便放在办公室一隅去蒙受尘土；要么买了现成的程序，想对不适合工作需要的部分进行改造，但却不懂内容。如此等等常有所闻。

实际上，在实际工作中，为了充分发挥微机的功能，通晓专业的人总希望按实际情况自己来编制程序。然而，无论对谁，编制复杂程序仍是困难的事。因此，本书对机械设计中强度计算等方面常用的材料力学的基本公式及其各个项目，都用BASIC语言进行了程序设计。这样做，从熟练运用计算机特性这一角度来考虑，多少会有些不妥之处，但相信本书将在实际工作者和计算机之间起到桥梁作用。

首先，对正在学习材料力学的学生来说，通过正确运用本书的程序，就可以通过计算机的屏幕学到很多具体的知识。

其次，本书的程序固然可以单独使用，也可以把各个单独程序进行组合，编出各个设计领域的计算系统。作者们希望以本书为阶梯，在这方面发挥作用。

另外，请注意，本书程序要用PC8001型微机运算，如

改用他种机型运算时，需要作一些变动。

最后，本书参考了很多著作，作者借此机会谨向这些著作的作者以及给予本书出版机会的日刊工业新闻社大阪分社表示衷心谢意。

昭和57年5月¹⁾

著者

1) 即公元1982年5月。——译者注

目 录

序言

1. 程序设计基础知识	1
1.1 程序的设计	1
1.2 数据表示法与四则运算	1
1.3 条件语句	4
1.4 转移方法	9
1.5 变量下标加字法	10
1.6 循环方法	14
1.7 函数的使用法	17
1.8 字符串	20
1.9 子程序	21
1.10 BASIC语法汇总	23
2. 平面图形的性质及材料性质的程序	25
2.1 平面图形的几何性质	25
2.2 金属材料的机械性质	33
2.3 平面图形及材料性质的子程序	40
3. 关于应力及应变的程序	45
3.1 拉伸应力及应变	45
3.2 阶梯形杆的拉伸	49
3.3 受自重作用的杆	53
3.4 冲击载荷作用下的应力	56
3.5 两端刚性固定的杆	59
3.6 垂直于刚性壁安装的桁架	64

4. 关于梁的程序	70
4.1 受 n 个集中力作用的悬臂梁	70
4.2 集中力作用下悬臂梁的挠度	76
4.3 在 n 个集中力作用下悬臂梁的挠度	80
4.4 受均布载荷作用的悬臂梁	85
4.5 受 n 个集中载荷作用的简支梁	91
4.6 n 个集中载荷作用下简支梁的挠度	97
4.7 受均布载荷作用的简支梁	102
4.8 一端固定, 一端铰支梁	107
5. 关于扭转的程序	114
5.1 圆轴及中空圆轴的扭转	114
5.2 轴的传递马力	118
5.3 圆柱形螺旋弹簧	122
5.4 承受弯曲和扭转的轴	127
6. 关于长柱屈曲的程序	132
6.1 长柱的临界载荷	132
7. 机械设计方面的应用	142
参考文献	145
附录	145

1. 程序设计基础知识

1.1 程序的设计

利用微机进行工作，须用一种特殊的语言，并使之有序。这种有序语言所表述的指令集合叫做程序。

微机通常用 BASIC 语言作为指令。因此，在编制程序时，BASIC语言的知识是不可缺少的。然而，更重要的是必须具备有关学科的专业知识。计算机是按程序运行的，只有充分理解工作的内容，系统地掌握并表现这些内容，才能编制出正确的计算机程序。

本书汇集了机械和结构强度计算等方面常用的材料力学基本内容，旨在使读者了解用 アルゴリズム ¹⁾ 和 BASIC 等语言描述问题和利用微机处理问题的程序技巧，从而更深入地掌握材料力学的内容。

本章将说明作为程序设计基础的 BASIC 语言的梗概。初学 BASIC 语言的读者，把文法严密但并不过分拘泥的程序作为使用微机的例题，切切实实地进行练习，这是理解 BASIC 语言的捷径。对 BASIC 语言已有一定程度掌握的读者，可从第二章读起。

1.2 数据表示法与四则运算

我们用一个包含加、减、乘、除运算和数据读写的简单

1) 请注意为解决这些问题而建立程序的手续。

程序作为例题，例中集中地示出了数值写法、算式及输入输出的表示方法等基本内容。

图1.1为当变量 A 、 B 分别取5.2和3.3时，它们的和、差、积、商的计算程序。

```
10 REM 四则运算
20 A=5.2
30 B=3.3
40 S=A+B
50 D=A-B
60 P=A*B
70 Q=A/B
80 PRINT A,B
90 PRINT S,D,P,Q
100 END
```

图 1.1 四则运算的程序

采用BASIC语言书写程序，所有行的前面都要有行号。由于计算机运行的原则是按行号由小到大顺序执行的，所以行号起到运行顺序的重要作用。另一方面，考虑到在行号后面有时需要增加新的行，因而不必作成连续编号。下面就图1.1所示的程序逐行加以说明。

- 10行，注释行，行号后面写上REM (注释)，其后面的内容并不是使计算机产生动作的指令，而仅是为了便于理解程序内容的注释。
- 20行， $A=5.2$ ，意味着给变量 A 赋值5.2，变量名可以自由选定，但要遵守下面的规则：
 - 1) 变量名用一个或二个字符表示。
 - 2) 变量名的第一个字必须用英文字母，第二个字使用英文字母或数字均可¹⁾。需要说明的是：这里所用的等号(=)，其意义是：将等号右边的数值或算式的计算结果代入左边的变量(赋值)，这与数学里的等号意义不同；其次，对于同一个变量可用任何数值代都行，但其功能是：新数据进入，老数据被冲掉(排挤)。
- 30行，与20行相同，这里是给变量 B 赋值3.3。
- 40，50，60，70行，右端分别表示 A 、 B 两个数的相加、相

1) 注意，不同的变量不能用相同的变量名。——译者注

减、相乘和 A 被 B 除的形式，左端分别用字母 S, D, P, Q 表示二数的和、差、积和商。这些行的右边，包括运算符号 $+, -, *, /$ 等在内叫做算术表达式。

变量=算术表达式

把整个上述形式叫做算术表达式。其意义是：计算右边表达式，把所得的值赋给“=”左边的变量。

- 80, 90行，为输出语句，在PRINT（打印）词后面写上被贮存的变量 A, B 及 S, D, P, Q 。PRINT语句表示把这些被贮存的变量值在CRT（cathode-ray tube）屏幕上显示的指令。所以执行完80行语句，就在CRT上显示出 A, B 变量的值为5.2及3.3，然后另起一行。同样，执行完90行语句，将在CRT上显示出变量 S, D, P, Q 中贮存着的值。
- 100行，END（终止）表示程序执行结束。

Run			
5.2	3.3		
8.5	1.7	17.16	1.57576
Ok			

图 1.2 运算结果

在图1.2中，示出了这个程序的运算结果。即使程序再运算一次，屏幕上显示出的值也不会改变。在此，如果对图1.2所示程序的20行及30行作如下修改¹⁾，即

```
20 INPUT A
30 INPUT B
```

记号INPUT（输入）为键盘输入语句，其后面写上从外部向变量输入数值或字符串等信息，由微机的键盘输入这些

1) 程序的修改方法，请参阅附录1。

信息，经这样修改后，从键盘上可以向变量 A, B 输入任意的信息。

把数据赋与变量的另一（比较紧凑的输入）方法，是在程序中使用 **READ, DATA** 语句，这个方法的本质是把程序中包含的信息，在微机运行时赋给变量¹⁾。

用 **READ, DATA** 语句来表示图1.1所示的程序，如图1.3所示。

```
10 REM 四则运算
20 READ A,B
30 S=A+B
40 D=A-B
50 P=A*B
60 Q=A/B
70 PRINT A,B
80 PRINT S,D,P,Q
90 DATA 5.2,3.3
100 END
```

READ, DATA 语句的一般表示形式为：

READ 变量名, 变量名, ..., 变量名
DATA 常数, 常数, ..., 常数

图 1.3 使用 **READ, DATA** 语句时的四则运算程序

在 **READ** 语句后面，写上读入信息的变量名，在 **DATA** 语句后面，则写上常数，而这些常数

书写的顺序与上述欲分配的变量的顺序要一一对应。

在图1.3的程序里，依次把常数5.2和3.3分配给变量 A 及 B ，关于 **DATA** 语句在程序中的位置，放在任何一行都可以。

1.3 条件语句（判断方法）

如前节所述，计算机执行程序的原则，是按行号由小到大的顺序进行的，但是，也会有这样的情况出现，即根据计算的结果，要求改变程序中某一段的公式（与前面的不同），或者是根据输入信息，要求改变计算处理的方法，对应于这

1) 此种语句称为读数语句。——译者注

些要求，必须相应地改变程序的流程。这种根据所给条件，改变程序流程的做法称为判断。

下面以二次方程式求根的简单程序为例，说明判断条件的基本内容。图1.4为二次方程式 $ax^2 + bx + c = 0$ ，按求根公式，求实根的程序。方程中系数 a, b, c 由INPUT 输入。

图1.5为这个程序的计算流程图¹⁾。从这个流程图，可看到如下情形：根据判别式 $D = b^2 - 4ac$ 的不同值（正、零和负）。二次方程式的根相应地可为2个实根、重根和虚根，然而各个根所用的计算公式是不同的。因此，在这种情况下，程序中必须作出判断。下面将按照图1.4中的行号逐一说明。

```
10 REM 二次方程式的根
20 INPUT A,B,C
30 IF A=0 THEN 160
40 D=B^2-4*A*C
50 IF D<0 THEN 140
60 IF D>0 THEN 100
70 X=-B/(2*A)
80 PRINT" 重根 ";X
90 GOTO 170
100 X1=(-B+SQR(D))/(2*A)
110 X2=(-B-SQR(D))/(2*A)
120 PRINT" 2个实根 ";X1,X2
130 GOTO 170
140 PRINT"虚根 "
150 GOTO 170
160 PRINT"a等于零"
170 END
```

图 1.4 二次方程式

- 10行，注释行。
- 20行，用INPUT语句输入变量 A, B, C 。
- 30行，IF（如果） $A=0$ THEN（则）160，是判断语句，根据条件改变计算流程的指令，式中表示当 $A=0$ 的条件成立，则运行转移到160行；当 $A=0$ 的条件不成立时，就不转移，仍按原顺序往下执行40行。

1) 关于流程图中的符号意义，请参阅附录2。

- 40行, 把变量 A, B, C 的值代入判别式 D 中, 并计算 D 值。

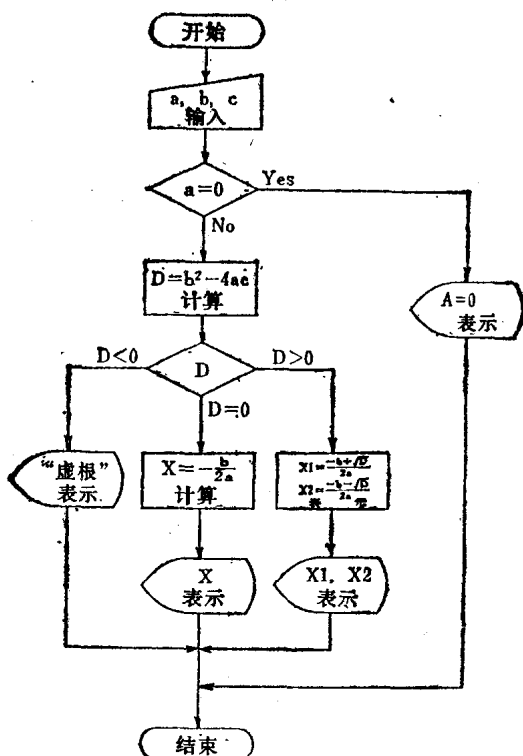


图 1.5 二次方程式的流程图

式中 B^2 表示 B 的自乘, 符号“ $\wedge$ ”表示乘幂, 关于运算的优先次序如下:

(1) $\wedge$ (2) $*$ / (3) $+$ -

- 50行, 与30行功能类似, 即当判别式的值 $D < 0$ (即 D 为负)的条件成立, 则计算流程转移到140行; 如果 $D = 0$ 或为正时, 仍然按原顺序往下执行60行。
- 60行, 也同样, 当 $D > 0$ 条件成立, 计算流程转移到100

行, 如果 $D > 0$ 条件不成立, 则继续执行下面的行。

- 70行, 由50行和60行可见, 判别式 D 的值既非 $D < 0$, 亦非 $D > 0$, 则只有 $D = 0$, 此时方程式的根便为重根, X 的值由70行表示。这是一个把变量 A , B 的值代入变量 X 中的算术表达式, 式中的圆括弧 $()$ 为运算顺序的指令(多重括弧也行), 与40行中所指出的运算次序相比, 此括号是最优先的。因此, $X = -B / (2 * A)$ 可表示成:

$$X = \frac{-B}{2A}$$

在上式中, 如果没有圆括号, 即 $X = -B / 2 * A$, 则将变成如下的形式:

$$X = \frac{-B}{2} \times A$$

对此必须加以注意。

- 80行, 用PRINT表示的输出语句, 把计算的结果在CRT上显示, 在PRINT语句后面引用符号“^^”, 在“^^”中的内容仍原封不动地显示在CRT上, 在这一行里表示重根, 下面用分号“;”与 X 隔开, 不留空白, 表示出 X 的值。
- 90行, GOTO (转向) 表示无条件转移语句, GOTO 170的作用将在1.4节内叙述, 而写在GOTO后面的行号为运行中无条件转移到该行的指令。
- 100, 110行, 是计算二个实根的算术表达式, 式中的SQR(D)表示了判别式 D 的平方根 $\sqrt{D}$ 。对于BASIC语句还备有一些常用的函数¹⁾。
- 120行, PRINT表示输出语句, 它把变量 $X1$, $X2$ (根)的值, 显示在CRT上。

1) 请参阅1.7节。

- 130行, 无条件转移到170行运行。
- 140行, 用PRINT表示的输出语句, 在CRT上显示出虚根字样。
- 150行, 与130行类似, 表示无条件转移到170行运行。
- 160行, 当 $A=0$ 时, 用PRINT表示的输出语句, 在CRT上显示A 等于零。
- 170行, 程序运行结束语句。

对于判断条件语句IF...THEN, 一般采用如下的形式

IF 条件 THEN 行号

如果条件满足, 就转移到THEN后面的行号去继续执行, 若条件不满足, 则不转移并继续往下执行下一语句。所谓条件, 是指用等号或不等号表示的比较式, 这些比较式所能使用的符号表示在表1.1中, 还可将这些简单判断进行组合来处理更为复杂的判断。

表 1.1 表示大小比较的符号

数学符号	BASIC符号	例	意 义
=	=	$A=B$	A等于B
<	<	$A<B$	A小于B
≤	<=	$A<=B$	A小于B或等于B
>	>	$A>B$	A大于B
≥	>=	$A>=B$	A大于B或等于B
≠	<>	$A<>B$	A不等于B

条件式与算术表达式不同, 在下面的例子中, 比较号的左边或右边可以是变量、常量或算式。

(例) IF $A+B=C$ THEN 200

IF $5.7<A*B$ THEN 150

IF $A + B/2 \geq B - C$ THEN 700

1.4 转 移 方 法

IF...THEN语句的意义是：根据相应的条件，转移到前面或后面指定的语句去继续执行，而在现有少数的简单转移的方法中，有以下三种语句：

1) GOTO 行号

2) ON 变量 GOTO 行号1, 行号2, ..., 行号 n

3) GOSUB 行号

1) 中的GOTO表示无条件转移，GOTO这一语句，在图1.4的程序中已经出现过，当计算机接到这个指令时，便无条件地转移到被写在GOTO词后面的行号去，但是这个行号必须在该程序中存在。

2) 中的ON...GOTO语句，其意义是：当变量的值为1时，计算机将转向行号1运行，当变量的值为2时，则转向行号2运行，同样，当变量的值为 n 时，则转向行号 n 运行。

例如：

ON K GOTO 100, 110, 120

此种写法表示当 K 的值为1时，计算机将转向100行运行； K 的值为2时，则转向110行运行； K 的值为3时，则转向120行运行。

3) 中的GOSUB（转子程序）语句，是使运行转向子程序的指令¹⁾。在一个程序中，经常会遇到必须对同一形式的内容进行多次反复的处理和计算，在这种情况下处理的方法，是把这样一些相同的内容集中起来，单独编成一个相对

1) 亦称转子语句，关于子程序，请参阅1.9节。——译者注

独立的程序，这样的程序称为子程序。在计算中，若需要调用这样的子程序，就使用GOSUB 指令，写在GOSUB 词后面的行号是被调用子程序最前面的行号。计算机一执行子程序中的RETURN 返回语句，就又自动返回相应于转出时GOSUB语句的下一行，继续执行原程序¹⁾。

1.5 变量下标加字法²⁾

在程序中，有时会遇到这样的问题，例如，需要对100个变量求和，此时也就要有100个不同类别的变量，而在数据的处理以及输出输入中也必须进行多次操作，对这些具有同一特性的成批的数据进行处理，最好使用数组³⁾表示，就比较方便，并且可以编出更为一般化的程序。

上述问题，在数学中被表示成如下形式：

$$S = \sum_{i=1}^{100} a_i$$

在这种情况下，显得特别重要的是添下标*i*的问题。譬如，上述例题中，用*A*来表示变量的话，那末就有*A*(1)，*A*(2)，…，*A*(*I*)，…，*A*(100)的形式。把这样一种在变量名后面括号内加了数字的变量(例如：*A*(1)，*A*(2)，…，*A*(*I*))称为下标变量⁴⁾。

下面用一个具有20个数据求和的程序来具体说明使用下

1) 这种包含有调用子程序的原程序，相对于子程序来说，就称为主程序。

——译者注

2) 即下标变量的下标含义。——译者注

3) 原文称排列型变量。——译者注

4) 亦称数组元素，原文称添字变量。——译者注