

科海培训中心
系列教材

QUICK BASIC

最新高级程序设计

(4.5 版)

作者 秦笃烈

中国科学技术出版社

北京科海培训中心系列教材

Quick BASIC最新高级 程序设计(4.5版)

[美] D.Inman B.Albrecht A.Jamtgaard著

秦笃烈 译

中国科学技术出版社

内 容 提 要

本书是针对著名的Microsoft公司推出的Quick BASIC最新的4.5版写出的高级程序设计专著,它以大量生动的实例介绍各种高级程序设计技术。内容包括Quick库建立、图形着色技术和贴面图案设计、动画技术、屏幕变换、窗口技术、索引文件和关键词文件、关系文件、无结构文件等。特别是所介绍的关系文件和无结构文件,体现了建立不定长字段、字段数不加限制记录文件的崭新程序设计方法,它具有重要的实用价值。这一先进的思想也适用于其它新型BASIC语言True BASIC和Turbo BASIC,甚至也适用于任何高级语言。本书读者对象为业余和专业程序设计人员、大学生、研究生,计算机用户、程序设计教师以及有关工程技术人员。

Quick BASIC最新高级程序设计

[美] D. Inman B. Albrecht A. Jamtgaard 著

秦笃烈 译

责任编辑:朱桂兰

封面设计:王林通

*

中国科学技术出版社出版(北京海淀区魏公村白石桥路32号)

新华书店北京发行所发行 各地新华书店经销

北京科海培训中心特约联系

(海淀路黄庄文化馆平房)

河北蔚县印刷厂印刷

*

开本:850×1168毫米 1/32 印张:11.6 字数:244千字

1990年9月第一版 1990年9月第一次印刷

印数:1—3,000册 定价:8.00元

ISBN 7-5046-02086-/TP.10

序 言

BASIC语言是世界上使用者最多的计算机语言。由于历史条件的限制，它的许多旧版本有一些缺点。随着计算机科学的发展，BASIC语言也在不断发展。由于BASIC语言的发明人肯梅尼和库尔茨等人的努力，BASIC语言有了美国国家标准并推出了功能强大结构新颖的True BASIC语言。实力强大的著名公司Boland和Microsoft也推出了Turbo BASIC和Quick BASIC。Quick BASIC 4.5是它的最新版本，本书是为4.5版而写的Quick BASIC程序设计专著。它篇幅虽然不大，但内容相当丰富。掌握任何计算机语言的捷径是首先学习和运行各种典型的样本程序然后自己创作程序。本书提出的关系文件和无结构文件内容十分精彩。掌握以后，不难自行构造出字段数不限、字段长不限的数据库。所介绍的方法可适用于各种计算机语言。

本书由Osborne McGraw-Hill出版公司1989年出版。由于发现它的实用性和趣味性，立即译出以飨读者。

我谨向北京科海培训中心和中国科学技术出版社的支持和合作深表谢意。

如果本书对传递最新信息、繁荣出版事业有所裨益，将感到不胜欣慰。

凡需要本书程序盘的读者可与北京科海培训中心或译者联系。

秦笃烈

北京右安门外，首都医学院
生物医学工程系100054

于1990.8.29

目 录

序言

1. 共同的起点	(1)
1.1 关于Quick BASIC的一些说明	(1)
1.2 BASIC保留字的用法	(3)
1.3 其它必要的预备知识	(3)
1.4 Quick BASIC概略性复习	(4)
2. 惯用法、风格和升级	(30)
2.1 记号惯用法	(30)
2.2 程序设计惯用法和风格	(32)
2.3 惯用法示范	(38)
2.4 Quick BASIC 4.5和其它版本的比较	(45)
3. Quick库	(54)
3.1 Quick库的优点	(54)
3.2 在建立Quick库之前	(55)
3.3 建立Quick库	(55)
3.4 使用Quick库UTIL1	(64)
3.5 增加一个Quick库	(65)
4. 图形基础	(76)
4.1 文本模式图形	(76)
4.2 目光转向窗口	(83)
4.3 其它字符形状	(86)
4.4 传送数据	(87)
4.5 文本和象素位置相互关系	(91)
5. 图形:着色和复杂图案设计技术	(105)
5.1 加上颜色	(105)
5.2 拼置贴面图案	(110)
5.3 程序主要功能分析	(123)

5.4	使用贴面图案·····	(127)
6.	小精灵和动画·····	(141)
6.1	小精灵诞生·····	(141)
6.2	使用小精灵·····	(153)
6.3	使用多个小精灵·····	(156)
6.4	改变文本颜色·····	(158)
6.5	飞碟·····	(165)
7.	屏幕变换·····	(174)
7.1	比例因子·····	(174)
7.2	窗口·····	(176)
7.3	裁裁·····	(180)
7.4	图形的放大或缩小·····	(181)
7.5	屏幕内的视见窗·····	(182)
7.6	利用WINDOW和VIEW语句调整比例·····	(193)
7.7	对二维对象作变换·····	(194)
8.	查找和排序·····	(204)
8.1	排序·····	(204)
8.2	各种排序算法·····	(206)
8.3	查找·····	(217)
9.	索引文件和关键词文件·····	(224)
9.1	索引数据文件·····	(224)
9.2	关键词数据文件·····	(246)
10.	关系文件·····	(255)
10.1	连接关系文件·····	(255)
10.2	修改程序·····	(257)
10.3	小结·····	(274)
11.	无结构文件·····	(277)
11.1	无结构顺序文件·····	(277)
	附录A Quick BASIC 4.5的保留字·····	(294)
	附录B ASCII代码表·····	(296)

1. 共同的起点

本节的读者已具有广泛的计算机方面的修养，作者假定这些读者已达到某个共同的起点，假定读者已经对 QuickBASIC 的早期版本特别是 Quick BASIC 4.0 版有相当了解。如果他们 对 Turbo Basic, MBASIC, ZBASIC, True BASIC 或其它曾经流行的 BASIC 语言版本有过使用经验，也就具有了本书必要的基础。

如果尚未有这样的经历，建议读者阅读关于 Quick BASIC 4.5 的入门书（如由 Bob Albrecht 等著的 Quick BASIC Made Easy, Osborne McGraw-Hill 1989 年版），也可以阅读关于 QuickBASIC 4.0 的中级读物（如由 Dom Inman 等著的 Using QuickBASIC, Osborne McGraw-Hill 1988 年版）。

1.1 关于 QuickBASIC 的一些说明

任何作家把所作的讨论分为初级、中级和高级三种类型的意图都应向读者作出解释。因此，本章要对这种先进讨论方式，提出一个共同的出发点。

QuickBASIC 4.5 是 Microsoft 公司研制的 QuickBASIC 的最新版本。它比解释型的 BASIC 版本运行更快但仍然保留了这种语言的交互性。它把解释型 BASIC 的交互性优点和编译程序的速度以及结构化模块化语言的功能结合起来。

QuickBASIC 4.5 功能和 QuickBASIC 4.0 类似，但作了一些修改，使得初学的程序设计人员更容易使用。例如，建立过程提供更简单更全面的 SETUP 程序，它采用树型结构菜单。利用这个程序，在开始实际安装 QuickBASIC 4.5 以前完成自己需要的配置选择。这些选择存在文件 QB.INI 中，SETUP 能够记住它并当再次运行它时能显示所作的配置选择。

QuickBASIC 提供如下的两级菜单：

- **简便菜单 (Easy Menus)** 它们提供最简单的用户界面。提供高级的或庞杂功能的菜单项以及可能使初学者引起混淆的菜单项均不出现。某些对话框也加以简化。
- **完备菜单 (Full Menus)** 它们提供为高级程序设计人员使用的全部功能。

程序SETUP有如下两级安装方法:

- **第一级** 这一级为简便菜单所用。它使用程序SETUP的缺省设置。需要时可通过Options菜单加以修改。
- **第二级** 这一级提供为高级程序设计人员需要的全部功能。

因为本书是为高级程序设计人员而写,除了为讨论所必须,一般将简便菜单和第一级SETUP忽略。

QuickBASIC包括以下特征:

- 和BASICA及GW-BASIC兼容
- 内装功能齐全的编辑、编译和调试程序
- 包含下拉菜单、对话框和在线帮助
- 通过键盘或鼠标器选择命令
- 在内存中编译以节省时间
- 输入程序时同时进行语法检查
- 支持结构化程序设计: 字母数字标识符, 结构化逻辑语句, 子程序以及多行函数
- 支持程序模块组成的库
- 支持建立.EXE文件
- 支持图形、BLOAD、BSAVE、声音、音乐和事件陷阱。

由上可见, QuickBASIC具有作为Microsoft推出的BASIC特征的强有力然而易于使用的语言环境。QuickBASIC有许多高级功能照顾到了和BASICA以及GW-BASIC的兼容性。当然, 如果已用过QuickBASIC以前版本, 对新版本就更易掌握。

QuickBASIC 4.5增加的功能也使得很容易从GW-BASIC和其它语言转为QuickBASIC。通过实践, 就能逐步从初学水

平转到要求运用QuickBASIC的高级功能。这时很自然会要求使用以Microsoft BASIC编译程序的各种强有力功能。

1.2 BASIC保留字用法

因为QuickBASIC 4.5和BASICA和GW-BASIC兼容,许多以ASCII形式储存的BASICA或GW-BASIC程序可由QuickBASIC装入、编译和运行。然而,有些BASIC和GW-BASIC语句和函数不能使用,有些需要修改,有些则可继续使用。这节要讨论对这些语句和函数进行讨论。

(1) 不能使用的保留字

下面所列出的BASICA和GW-BASIC语句和函数在QuickBASIC 4.5程序中不能使用,因为它们涉及到源文件的编辑,程序执行界面,盒式录音机,或者和QuickBASIC环境提供的语句功能重复:

AUTO	DELETE	LLIST	MOTOR	SAVE
CONT	EDIT	LOAD	NEW	USR
DEF USR	LIST	MERGE	RENUM	

(2) 可修改的保留字

包含下列语句的BASICA和GW-BASIC程序在由QuickBASIC 4.5编译和运行以前要进行修改:

BLOAD	CHAIN	DIM	RESUME
BSAVE	COMMON	DRAW	RUN
CALL	DEF	PLAY	

具体的修改方法可参看QuickBASIC 4.5的相应手册中说明。本书按正确用法使用这些语句。

附录A给出QuickBASIC 4.5给出保留字的完整清单。

1.3 其它必要的预备知识

读者应该知道如何使用全部菜单以及和它们有关的对话框。

QuickBASIC允许利用键盘和鼠标器两种方法进行菜单和对话框选择。手册Learning to Use QuickBASIC(学会使用QuickBASIC)详细解释了如何利用键盘和鼠标器在菜单和对话框之间移动。因此,本书假定读者已经对这些内容有所了解。

读者也应该对QuickBASIC编辑程序和调试工具有一定了解和使用经验。应该熟悉QuickBASIC 4.5的多数关键字。本书侧重对新的QuickBASIC 4.5关键字进行讨论。

本章将对QuickBASIC的中级水平内容作一些概略性复习。如果对这些材料不熟悉,可以去查阅QuickBASIC手册或通读一本入门性的中级读物。

1.4 QuickBASIC概略性复习

要理解QuickBASIC术语,需要对称为模块的程序单元有明确的概念。模块是包含QuickBASIC语句的个别文件。模块可以是一个完整的程序,也可以是包含一个或多个可以由其它模块调用的SUB或FUNCTION过程。包含多于一个模块的程序称为多模块程序。当储存多模块程序时,每个模块分别地存在一个文件中。

每个QuickBASIC程序有一个主模块。在一个多模块程序中,主模块包含程序运行时要执行的第一个语句。一个模块可以是许多不同程序的组成部分。然而,一个模块可以仅仅是一个程序(即对模块名发生作用的程序)。在单模块程序中,包含任何SUB和FUNCTION过程的程序就是主模块。

过程SUB...END SUB 以及FUNCTION...END FUNCTION对QuickBASIC程序特别重要。这些过程类似于子程序和DEF FN语句,但是它们和老的结构相比有独到的优点。利用这些过程进行程序设计可有如下三大好处:

- 允许把程序分成每个分散的逻辑单元。每个单元可以比不包含这样的单元和程序更方便地进行测试和纠错。
- 一旦对过程进行调试时,它们可以在一个或其它程序中作

为段块使用。

- 因为它们只有一个入口点并且任何在它们内部声明的变量缺省地是该过程的局部变量，所以一般说来，它们比较可靠。

(1) 用户定义的函数

可以利用DEF FN语句为自己的函数进行命名和定义。可以像GW-BASIC和BASICA那样使用单行函数定义法也可以使用更强有力的GW-BASIC和BASICA不提供的多行函数定义法。只能在所定义的模块中使用DEF FN函数。

程序1-1使用了以下三个单行函数定义：

```
DEF FNcent#      把一个双精度数四舍五入到二位小数
DEF FNmin!       求两个单精度数的最小值
DEF FNsrch%      在字符串中查找字符串
```

注意，DEF FN函数必须在该函数使用以前定义。也就是说，必须把函数定义的位置放在源文本中任何参考DEF FN的语句之前。最好在函数使用以前就在程序中把函数定义放在一起。

当运行程序1-1时，先要求输入两个双精度数。然后将输入的数赋给变量first#和second#。这些变量在DEF FNcent#中用来把数变成两位小数。

接着，程序1-1要求输入两个单精度数。输入的值分别赋给n1!和n2!。单精度变量n1!和n2!在DEF FNmin!中用到。这个函数求n1!和n2!的最小值并作为单精度值和四舍五入以后的双精度值first#和second#一起在屏幕上加以显示。

最后，要求输入一个字符串和一个子串。计算机对输入的子串查找输入的字符串。可以用大写、小写或大小写混合的方式输入字符串。这里的查找和大小写无关。如果子串找不到，计算机在屏幕上显示这个结论。如果子串找到，计算机显示子串在字符串中出现的开始位置。

```
REM ** DEMONSTRATION OF DEF FN **
```

```
* Program 1-1. File: PR00101.BAS
```

```
REM ** Define Function FNcent# **
```

```
* Rounds a double precision number to two places
```

```
DEF FNcent# (mn#) = SGN(mn#) * INT(100 * ABS(mn#) + .5) / 100
```

```
REM ** Define Function FNmin! **
```

```
* Finds the minimum of two single precision numbers
```

```
DEF FNmin! (n1!, n2!) = n1! * ABS(n1! <= n2!) + n2! * ABS(n1! > n2!)
```

```
REM ** Define Function FNsrch% **
```

```
* Searches a string for a substring
```

```
DEF FNsrch% (st$, sbst$) = INSTR(UCASE$(st$), UCASE$(sbst$))
```

```
REM ** Input Numbers **
```

```
DEFINT A-Z: CLS
```

```
INPUT "First double precision number, please "; first#
```

```
INPUT "Second double precision number, please "; secnd#
```

```
PRINT : INPUT "First single precision number, please "; n1!
```

```
INPUT "Second single precision number, please "; n2!
```

```
PRINT
```

```
REM ** Call Numeric Functions and Print Results **
```

```
PRINT "Rounded value of first number is "; FNcent#(first#)
```

```
PRINT "Rounded value of second number is "; FNcent#(secnd#)
```

```
PRINT
```

```
PRINT "Minimum of single precision numbers is "; FNmin!(n1!, n2!)
```

```
PRINT
```

```
REM ** Input Strings, Search, and Print **
```

```
LINE INPUT "String to be searched, please? "; st$
```

```
LINE INPUT "String to search for, please? "; sbst$
```

```
REM ** Call String Function and Print Results **
```

```
PRINT : found = FNsrch%(st$, sbst$)
```

```
IF found = 0 THEN
```

```
PRINT CHR$(34); sbst$; CHR$(34); " not found in the string."
```

```
ELSE
```

```
PRINT CHR$(34); sbst$; CHR$(34); " starts at position";
```

```
PRINT found; "in the string."
```

```
END IF
```

```
END
```

程序 1-1 演示DEF FN

这个查找功能是由DEF FNsrch%完成的，它以变量st\$代表字符串并以sbst\$代表子串。这三个函数的定义如下所示：

```
DEF FNcent*(mn*) = SGN(mn*) * INT(100 * ABS(mn*) + .5) / 100
```

```
DEF FNmin!(n1!, n2!) = n1! * ABS(n1! <= n2!) + n2! *  
ABS(n1! > n2!)
```

```
DEF FNsrch%(st$sbst$) = INSTR(UCASE$(st$), UCASE$(sbst$))
```

程序 1-1 的典型输出如图1.1

```
First double precision number, please ? 1234567.898123
Second double precision number, please ? 354.865123456789

First single precision number, please ? 123.4567
Second single precision number, please ? 198.1234

Rounded value of first number is 1234567.89
Rounded value of second number is 354.87

Minimum of single precision numbers is 123.4567

String to be searched, please? This is the string to be searched.
String to search for, please? Ring

"Ring" starts at position 15 in the string.

Press any key to continue
```

图 1-1 程序1-1的输出

QuickBASIC的多行函数定义比单行函数定义更强有力。可以如下述列表所示把一个语句段放在函数名、参数表行以及函数的结束部之间。垂直的圆点代表定义函数的语句段。

```

REM ** WORD COUNTER & SPACE SQUEEZER **
* Program 1-2 file: PRO0102.BAS

REM ** Define FUNCTIONS **

* FNcount% counts words in a string
DEF FNcount% (strng$)
    words = 0
    FOR num = 1 TO LEN(strng$)
        char$ = MID$(strng$, num, 1)
        IF char$ = " " THEN
            words = words + 1
        END IF
    NEXT num
    FNcount = words + 1
END DEF

* FNsqueezes squeezes spaces from a string
DEF FNsqueezes (strng$)
    squeezes$ = ""
    FOR num = 1 TO LEN(strng$)
        char$ = MID$(strng$, num, 1)
        IF char$ <> " " THEN
            squeezes$ = squeezes$ + char$
        END IF
    NEXT num
    FNsqueezes$ = squeezes$
END DEF

REM ** Use the Function Definitions **
DEFINT A-Z: CLS
LOCATE 2, 10: PRINT "Type Q and press ENTER to quit."
VIEW PRINT 5 TO 24
DO
    LINE INPUT "String, please ? "; teststring$
    IF UCASE$(teststring$) = "Q" THEN
        EXIT DO
    END IF
    PRINT
    PRINT "The word count is "; FNcount%(teststring$)
    PRINT : PRINT "Your squeezed string is"
    PRINT FNsqueezes$(teststring$)
    PRINT
LOOP
VIEW PRINT: CLS
END

```

程序 1-2 词计数以及空格压缩

```
DEF FNfunctionnam(parameterlist)
```

```
.  
. .  
. .
```

```
END DEF
```

程序1-2使用两个多行函数定义不可能压缩成一行，因为复杂的函数。在DEF FN函数定义内使用的变量缺省地说，对当前模块而言是全局变量。然而，可以通过设置一个STATIC语句使DEF FN中的变量成为局部。在程序1-2中没有使用STATIC语句。因此，所有定义中使用的变量是全局变量。

函数定义往往在函数名之后跟有一个用括弧括起来的参数表。参数是传送给函数的变元的变量名。进入的参数可受到保护以免在DEF FN函数中被修改。以下列表表明变数string\$在程序1-2中作为参数使用。

```
DEF FNcount%(strng$)  
DEF FNsqueeze$(strng$)
```

调用该函数的语句包含一张变元表。变元是调用它时传送给函数的常数、变量或表达式。DEF FN函数可以通过在QuickBASIC语句中使用它们的方式调用。在变元表中变量按值传送给DEF FN函数。从技术角度说，当使用DEF FN函数时，建立变量的临时拷贝，然后把该拷贝的地址传送给函数。正如以下列表所示，程序1-2调用每一个DEF FN函数并且每次都传送输入的字符串teststring\$的值。

```
PRINT "The word count is" ; FNcount%(teststring$)  
PRINT FNsqueeze$(teststring$)
```

注意在程序1-2中使用了语句段IF...THEN...ELSE。这

种格式提供明确的具有可读性的结构。在带有注释语句内容

“Use the Function Definition”（使用函数定义）的程序段中，加了一个DO...LOOP，它的作用是在按键Q时可有次序地从DO...LOOP退出。DO...LOOP也允许在DO或LOOP语句中使用UNTIL或WHILE子句。

VIEW PRINT语句用来指定文件屏幕的“卷屏”区域。在第二行上印出第一个指示：“Type Q and press ENTER to quit”然后VIEW PRINT语句指定从第5行到第24行作为今后打印的卷屏区。因此，该指示在该区域填充时并不卷离屏幕。在程序即将结束时，另一个VIEW PRINT语句规定不把卷屏区域扩充到全屏幕。

程序1-2的典型输出如图1-2和图1-3。

```

Type Q and press ENTER to quit.

String, please ? This is a test string to count words and squeeze spaces.
The word count is 11
Your squeezed string is
Thisisateststringtountwordsandsqueezespaces.
String, please ?
```

图 1-2 程序1-2：第一个字符串

Type Q and press ENTER to quit.

String, please ? This is a test sentence to count words and squeeze spaces,

The word count is 11

Your squeezed string is

Thisistestsentencecountwordsandsqueezespaces,

String, please ? Now, here is a second sentence,

The word count is 6

Your squeezed string is

Now,hereisasecondsentence.

String, please ? q

图 1-3 程序1-2：准备退出

(2) FUNCTION...END FUNCTION过程

多行DEF FN函数对单行DEF FN函数来说是一种改进。然而，FUNCTION...END FUNCTION过程可以完成同样的事并有一些好处。

在FUNCTION过程中使用的所有变量对该过程来说是局部的。然而，可以使用带有SHARED语句的全局变量或使用其它语句的SHARED属性。

程序1-3使用两个FUNCTION过程。除了开始和结束行以外，在FUNCTION中的语句段和程序1-2的DEF FNsqueeze\$函数定义中所用的语句段是完全一样的。在开始行和两个结尾行方面的差别如下所示：

```
FUNCTION Squeezer$(strng$)  DEF FNsqueeze$(strng$)
```