

〔美〕 戴 维 R·塞 西 尔 著
周烈强 译 王懋江 校

怎样调试 BASIC程序



DEBUGGING



BASIC



PROGRAMS



北京科学技术出版社

怎样调试BASIC程序

〔美〕 戴维R·塞西尔 著
周烈强 译 王懋江 校

北京科学技术出版社

DEBUGGING BASIC PROGRAMS

BY DAVID R. CECIL

TAB BOOKS Inc. 1984.

怎样调试BASIC程序

〔美〕戴维 R. 塞西尔 著

周强烈 译 王懋江 校

*

北京科学技术出版社出版

(北京西直门外南路19号)

新华书店首都发行所发行 各地新华书店经售

北京通县马驹桥印刷厂印刷

*

787×1092毫米 32开本 11.25印张 247千字

1988年4月第一版 1988年4月第一次印刷

印数 1—3,000册

ISBN 7-5304-0212-9/T·35 定价: 2.50元

内 容 简 介

本书是一本调试BASIC程序的实用指南,共九章。全书针对微型计算机广泛使用的 BASIC 语言,系统而全面地介绍了调试程序的一般原则,详细分析了BASIC程序中各种错误的类型(如算术错、逻辑错、字符串错、输入/输出错、磁盘BASIC错等),错误的表现形式,出错的原因和修正错误的方法与技巧。此外,书末还给出了TRS-80机、Apple 机和IBM PC机中的错误信息对照表。

无论对职业程序员还是初学者来说,本书将会帮助您大大节省调试程序的时间和提高工作效率。

本书适用面广,既可供一般专业程序员使用,也可作为具有高中以上文化程度的初学微型计算机的人员和青少年计算机爱好者的参考书和上机指南。

译 者 的 话

随着微型计算机的普及，BASIC语言得到了迅速的推广。但是，会用这种语言写程序是一回事，写出的程序能正确运行则是另一回事。BASIC程序看起来似乎很简单，然而从程序第一次上机敲入到能正确运行，即使对于一个有经验的程序员来说，也要经过多次反复的调试。在算法正确的前提下，大量的调试工作就是排除程序书写不当而引起的各种各样的错误，而其中有不少错误并不明显。尽快地发现程序中的错误，确定出错的位置，查出出错的原因并予以修正，是缩短调试时间的重要途径。

本书作者是美国微型计算机专家，得克萨斯 A & T 大学数学系系主任，他已发表过好几本有关微型计算机的专著。除本书外，其他还有《80 Micro》、《Personal Computing》和《TRS 80 Newsletter》。本书虽无高深哲理，但它总结了作者多年实践的经验，故颇为实用。书中既叙述了作为一个程序员应该掌握的一般调试程序的原则，又系统而全面地列举了BASIC程序中可能出现的各种错误。书中详细地给出了包含错误的程序和程序片断，计算机的运行结果和相应的出错信息，并详细地分析了出错的原因和改正错误的具体方法和技巧。书中还对怎样防止错误，怎样找出生错误和怎样处理用一般方法无能为力的错误也做了具体介绍。所以，本书无论对入门的程序员还是对有经验的程序员来说，都不失为一本实用的 BASIC 程序调试指南。译者相信，只要掌

握了这本优秀新指南所提供的忠告和经验，作为一个程序员，写出和调试出没有错误的程序也就不难了。

由于译者水平有限，译文中可能有不当之处，敬请读者不吝赐教。

周烈强 王懋江

1986年8月于北京

前 言

计算机程序未必总是象预期的那样运行。有时，它们甚至根本不运行，而有时它们却会陷入无休止的循环而不能停止。

从发现和识别错误的线索，到具体的错误类型，到“当所有其他办法都无效时的措施”和以给出错误信息结束，本书阐述了怎样消除用BASIC语言写的程序中的各种错误。

书中使用了若干小的程序段，来识别和改正一些具体的错误。虽然所有这些例子是在 Radio Shack TRS-80 I 型上用Microsoft公司的BASIC实现的，但对大多数微型计算机的BASIC也同样适用。

第一章介绍了发现和识别错误的线索。这一章叙述的各种方法和提出的几条原则，将有助于确定BASIC程序中经常出现错误的位置和类型。

第二章列举了大多数常见的错误（打字错，语法错，赋值错等等），将给出了怎样避免这些错误的详细办法。

第三章至第七章分别着重讨论了各种类型的错误：逻辑错、算术错、字符串错、输入／输出错和磁盘BASIC错。

为了对付那些有时出现的顽固性错误，第八章列举了“当所有其他办法都无效时的措施”。

第九章给出了 TRS-80 I 型和 II 型上的非磁盘BASIC和磁盘BASIC的所有出错信息。这些信息对Apple(苹果)机和IBM公司的机器几乎完全适用。所以，不要因为是为 TRS-80

的信息而忽略这一章。

掌握怎样识别、定位和改正错误，不但在于消除存在的问题，而且还希望更容易地写出错误较少的程序。

目 录

第一章 调试的基本知识	1
1.1 一组通用的调试规则	1
1.2 错误的种类	3
1.3 一般的调试原则	9
1.4 几种调试措施	11
1.5 利用软件调试工具	14
1.6 进一步的措施	25
第二章 一般性错误	27
2.1 变量	27
2.2 错误的替换	50
2.3 标点错	57
2.4 行引用和行号	65
2.5 其他的常见错误	82
2.6 由小错误引起大错误	83
第三章 逻辑错	90
3.1 转移	92
3.2 逻辑运算符和逻辑运算	109
3.3 循环	111
3.4 子程序	139
第四章 算术错	145
4.1 数的类型	145
4.2 有关数的比较	158

4.3	初始化	162
4.4	下标变量	165
4.5	运算的次序	168
4.6	其他函数	178
第五章	字符串错	195
5.1	字符串的存储	195
5.2	CLEAR N 语句	199
5.3	重新规定数组的长度	206
5.4	字符串中引号的使用	207
5.5	字符串的比较	216
5.6	其他一些串函数	227
第六章	输入/输出错	233
6.1	键盘	233
6.2	显示屏幕的使用	250
6.3	屏幕的若干特性	268
6.4	端口	273
第七章	磁盘BASIC错	282
7.1	使用磁盘的注意事项	282
7.2	文件说明	283
7.3	文件的建立和缓冲区分配	285
7.4	顺序存取	289
7.5	随机存取	296
7.6	文件中的直接语句	308
第八章	当所有方法都无效时的措施	310
8.1	核对清单	310
8.2	无用单元收集	314
8.3	在计算机复位以后	314

第九章 错误信息	316
9.1 非磁盘BASIC错.....	316
9.2 磁盘BASIC错.....	339
9.3 DOS错.....	348

第一章 调试的基本知识

程序中的错误是指某些障碍、不完整和缺陷，它会导致错误的或不可靠的结果。而调试则是指找出与修正错误及其有关问题的技巧。调试程序时，虽然要应用科学的方法，但归根结底，这是一种技术和技巧。如果缺乏这种技术和技巧，则调试就不一定能达到目的。让我们来考察一下开发一个计算机程序的所有阶段——设计程序、编写程序、调试程序和测试程序。其中，调试程序比所有其他阶段合起来所花的时间要多得多。

为了对调试有所帮助，本章首先提出了一些调试程序的规则、要览和方法。然后，讨论特定类型的错误，并提出一些有效地调试程序的措施。

1.1 一组通用的调试规则

1. 切不可认为任何事情都是理所当然的。即使程序目前能运行，并不意味着它可能没有错误；即使某个算法总能正确工作，也不意味着它对于下一组条件和数据也是一个恰当的算法；即使某一个子程序或代码块已经用了多年，也不意味着它无错误。我们能保证任何时候都不改变程序吗？

2. 应当相信错误是确实存在的。我们是否错误地预计了可能出现的情况？我们预料的结果是应该得到的结果吗？

3. 首先，应检查出最简单和最明显的事项。如果存在这样一些事项，则应按先易后难的次序予以研究。对于微型

计算机程序来说，屏幕上所显示出的错误信息确定了首先应检查的项目。

4. 所有的措施都必须有计划有条理地实行，并且使用合乎情理的常识。

5. 当进行调试时，对所有的工作和所采取的措施都应做出记录。特别是务必包括所有测试过的情况。读一读这种调试注解会对我们有所帮助。这种调试注解使我们明白，当下一次遇到类似情况时，应该避免什么和为什么这样解决。困难的测试情况的经验对于我们将来设计其他程序也许是很有用的。

6. 请记住，修正一个错误的任一尝试可能会引起更多的原来程序没有的新错误。在修改程序的逻辑时请特别小心。让我们来考察一个“等于则转移”的条件语句。例如

```
100 IF A=B THEN 300
```

我们发现它不能正确工作。于是又试一试

```
100 IF A<>B B THEN 350
```

但是又出现一组新的错误。为此，再试一试计数器的情况

```
100 IF A=12 THEN 300 ELSE A=A+1
```

然而，我们未进行初始化。CLEAR 语句的使用可以做为另一个例子。由于未保留足够的字符串空间，于是就按 CLEAR 键，但是这样做会把变量清零。

7. 一次只修正一个且仅一个错误。如果同时进行许多修改，我们就不知道哪一个修改，纠正了哪一个问题，也不知道其究竟。一定要在修改上花点时间。力求调试技巧尽善尽美，会使以后所需的修改更明了。

8. 出错检测点(表现有错误的点)可能不是错误的源发

点（产生出错条件的点），检测点可能在某个很后面的语句行上。DATA语句就是一个很好的例子。如

```
10 DATA 15,20,25,30,35
20 FOR I=1 TO 10
30 READ A(I)
40 NEXT I
READY
OUT OF DATA IN 30
```

这里的错误不是由行30引起的，而是由行10引起的，在行10中只能找到5个数据。而不是所要求的10个。

9. 请记住，当你敲 BASIC 程序时，系统对其语法不予检查。直到运行（敲RUN）时才会出现语法错。所以，在按〈ENTER〉键之后，即使屏幕上显示出了READY信息，也不要认为所敲入的内容都是正确的。

10. 请阅读有关的手册。每一种型号的微型计算机均有自己的特性。有的要求用LET；有的必须用一个END语句；有的（如TRS 100型）不用双精度。就打开磁盘文件的方法而言，几乎各种型号的微型计算机都稍有差别。

1.2 错误的种类

现在，我们已经有了—些调试规则，那么，哪些类型的错误会钻到程序中来呢？首先，与其他高级语言不同，BASIC中不存在错误的等级。即不存在警告性的、致命性的或终止性的错误。要么完全检测不出错误（因为你使用了一个错误的算法而不知其错误所在），要么程序执行到显示出出错信息而告终，要么机器“挂起”。有关TRS-80 I型和Ⅱ型的出错信息表和大多数典型微型计算机的出错信息请参见第九章。

错误类型可以描述为次要的、主要的和严重的，同时还

可用介于这些类型的中间类型来描述。这里所用的方法是把错误分为语法错、逻辑错、数据错、变量错、数组错、算术错、逻辑运算错、输入/输出错和其他错。

语法错

1. 印刷上的错误。如把 0 当作 O ,把 I 当作 l。
2. 标点错。如把逗号当作分号。
3. 引号或括号不配对。
4. 错误地使用空格。例如在数据项中使用空格。
5. 把保留字用作变量名。
6. 把注释用作 GOTO或GOSUB 语句的目标(只有去掉注释之后, 这才不算是一个错误)。
7. 对于行号的其他不正确的引用。

逻辑(控制流程)错

1. 转移到不正确的行号。
2. 未考虑所有可能的条件和提供相应的转移。
3. 在转移语句之后出现不正确的缺省路径或失败路径。
4. 程序不能终止。
5. 未正确地使用嵌套IF语句。
6. 循环不适当地初始化。
7. 不适当的循环下标。
8. 循环不能终止。
9. 循环完成之后少循环一次。
10. 在Microsoft公司的 basic 中,所有循环至少执行一次。当使用循环代替条件转移时, 这是所希望的吗?
11. 不正确的循环出口。
12. 从程序的另一部分转移到循环之内。
13. 调用了不应当调用的子程序。

i4. 未调用应当调用的子程序。

i5. 子程序不正确的退出。

数据错

1. 数据格式不合适。例如，对于日期来说，当要求dd/mm/yy或mm-dd-yy时，使用了mm/dd/yy。

2. 输入的值太少或太多。

3. DATA和READ 语句匹配错。

4. 程序控制转移到数据语句，数据语句当作程序语句执行。

5. 数据值解释为行号。

变量错

1. 变量说明与数据类型不一致。

2. 使用了未初始化的变量。

3. 以多于一种以上的意义使用同一变量。

4. 用不同的方法拼写同一变量名。

5. 使用了类似的变量名，而计算机把它们解释为同一变量。如PI和PIE，ACCOUNT 和ACCUMULATES。

6. 计数器不适当地设置零。

7. 下标变量未假定其值的整个范围。

8. 对于字符串而言，没有清除足够的存储单元。

数组（矩阵）错

1. 某个数组的维数未正确说明。

2. 某个数组未全部清除为零，或未用适当的值初始化。

3. 下标越界。

4. 下标不为整数。

5. 下标是从0开始标，还是从1开始标？它与变址(indexing)匹配吗？

6. 下标引用了错误的数组元素。
7. 颠倒了下标的次序。如当要求 $A(1,2)$ 时，却使用了 $A(2,1)$ 。
8. 不正确的下标个数。

算术错

1. 试图进行混合型运算。
2. 误解数的类型转换精度。
3. 对字符串进行计算。
4. 出现上溢或下溢。
5. 出现截断或舍入。
9. 未考虑到十进制数表示为二进制数时的不精确性。
7. 误解或误用了运算的层次（次序）。
8. 企图用零除。
9. 使用了无定义的形式。例如 $0/0$ 或 0 的 0 次幂。
10. 误解了算术函数的定义域或范围。如使用了负数的平方根。
11. 不正确地使用了复杂的计算方法。

逻辑运算错

1. 布尔运算 NOT、AND 和 OR 使用不正确，或者使用的优先顺序不对。
2. “+”运算符用作字符串的并量，这是所希望的吗？
3. 没有进行正确的比较，没有考虑空格和对齐小数点。
4. 数字值与字符值进行比较。
5. 未正确地实现比较关系。误解了术语“至多”，“不多于”，“至少”，“不少于”的含义。
6. <、AND 等的层次是什么？使用正确吗？如把 $(I < A) \text{ AND } (I < B)$ 写成 $I < (A \text{ AND } B)$ 。