

管理软件程序设计方法150例



汉字 FoxBASE+2.10  
管理软件程序设计方法 150 例

姚国祥 编著

武汉工业大学出版社



# 目 录

|                          |      |
|--------------------------|------|
| 1 程序设计概述 .....           | (1)  |
| 1.1 程序和程序设计 .....        | (1)  |
| 1.1.1 程序 .....           | (1)  |
| 1.1.2 程序设计 .....         | (1)  |
| 1.2 程序设计的步骤 .....        | (1)  |
| 1.3 程序质量的标准 .....        | (1)  |
| 1.4 结构化程序设计方法 .....      | (2)  |
| 1.4.1 三种基本控制结构 .....     | (2)  |
| 1.4.2 结构化程序设计特点 .....    | (3)  |
| 1.5 结构化程序设计原则 .....      | (4)  |
| 1.5.1 模块化 .....          | (4)  |
| 1.5.2 自顶向下 .....         | (4)  |
| 1.5.3 逐步求精 .....         | (4)  |
| 1.6 算法的描述工具——流程图 .....   | (5)  |
| 2 程序的调试和维护 .....         | (7)  |
| 2.1 程序错误类型 .....         | (7)  |
| 2.1.1 语法错误 .....         | (7)  |
| 2.1.2 逻辑错误 .....         | (7)  |
| 2.1.3 调试步骤 .....         | (8)  |
| 2.2 程序调试方法 .....         | (8)  |
| 2.2.1 保留和查看历史 .....      | (8)  |
| 2.2.2 跟踪程序执行 .....       | (9)  |
| 2.2.3 断点设置和恢复 .....      | (10) |
| 2.2.4 模拟键盘输入 .....       | (11) |
| 2.3 错误捕获与键盘侦测 .....      | (12) |
| 2.3.1 出错处理 .....         | (12) |
| 2.3.2 按下 Esc 键后的处理 ..... | (12) |
| 2.3.3 ON KEY 命令 .....    | (13) |
| 3 系统的安全设置 .....          | (14) |
| 3.1 系统的逻辑功能 .....        | (14) |
| 3.2 保持系统的安全性 .....       | (15) |
| 3.2.1 口令的设置 .....        | (15) |
| 3.2.2 设置复杂的口令 .....      | (16) |
| 3.2.3 双保险的口令字 .....      | (16) |
| 3.2.4 动态口令 .....         | (17) |
| 3.2.5 设置用户名和口令字 .....    | (17) |
| 3.2.6 权限的设置 .....        | (18) |
| 3.3 限制系统的使用 .....        | (19) |

|       |                    |      |
|-------|--------------------|------|
| 3.3.1 | 确定系统的工作时间          | (19) |
| 3.3.2 | 限制系统使用的试用期         | (20) |
| 3.3.3 | 限制系统的试用次数          | (20) |
| 3.4   | 数据库的保护             | (21) |
| 3.4.1 | 使用区位码为空的汉字         | (21) |
| 3.4.2 | 在数据库文件名中插入空格       | (21) |
| 3.4.3 | 利用空格和不显示汉字区位码作文件名  | (21) |
| 3.5   | 利用库结构实现数据库的保护      | (21) |
| 3.5.1 | DBF 文件的结构          | (21) |
| 3.5.2 | 对 DBF 文件进行加密       | (22) |
| 3.5.3 | 修改文件属性             | (23) |
| 3.6   | 程序文件的安全            | (23) |
| 3.6.1 | 防止修改程序             | (23) |
| 3.6.2 | 防止复制               | (23) |
| 3.7   | 子目录及其文件的保护         | (23) |
| 4     | 菜单程序设计             | (26) |
| 4.1   | 点式菜单               | (26) |
| 4.2   | 光条式菜单              | (27) |
| 4.3   | 用数据库实现动态光条式菜单      | (29) |
| 4.4   | 阵列式菜单              | (30) |
| 4.5   | 下拉式菜单              | (32) |
| 4.6   | 弹出式菜单              | (40) |
| 4.7   | 叠加式菜单              | (41) |
| 4.8   | 通用多方式菜单            | (46) |
| 5     | 数据输入程序设计           | (49) |
| 5.1   | 高效简便的数据输入法         | (49) |
| 5.2   | 格式输入设计             | (49) |
| 5.2.1 | 直接向数据库输入数据         | (49) |
| 5.2.2 | 连续输入数据             | (50) |
| 5.2.3 | 利用屏幕格式文件输入数据       | (51) |
| 5.2.4 | 单屏多记录数据输入          | (51) |
| 5.2.5 | 多屏输入               | (52) |
| 5.2.6 | 间接向数据库输入           | (52) |
| 5.3   | 实现快速输入             | (53) |
| 5.3.1 | 预测输入的数据            | (53) |
| 5.3.2 | 使用变量替换方法实现         | (54) |
| 5.3.3 | 使用功能键加快数据输入        | (54) |
| 5.3.4 | 用 INKEY 命令实现灵活快速输入 | (56) |
| 5.3.5 | 根据输入字段分别选用词组       | (56) |
| 5.3.6 | 利用字典库自动替换汉字词组      | (58) |
| 5.3.7 | 利用 SYS(18) 提示数据编码  | (58) |

|       |                            |      |
|-------|----------------------------|------|
| 5.4   | 全屏幕数据输入                    | (59) |
| 5.4.1 | 带提示的全屏幕输入                  | (59) |
| 5.4.2 | 开窗口上卷式数据输入                 | (61) |
| 5.4.3 | 仿全屏幕窗口命令 BROWSE            | (62) |
| 5.5   | 数据输入中的校验                   | (65) |
| 5.5.1 | 数据检验                       | (65) |
| 5.5.2 | 数据输入中的纠错                   | (66) |
| 5.5.3 | 对数据范围进行限制                  | (66) |
| 5.5.4 | 日期的限制                      | (66) |
| 5.5.5 | 代码库检验                      | (67) |
| 5.5.6 | 重复输入的检测                    | (67) |
| 5.5.7 | "双工" 输入数据                  | (68) |
| 5.6   | 通用数据输入程序                   | (68) |
| 5.6.1 | 生成数据字典信息库                  | (68) |
| 5.6.2 | 通用数据输入程序                   | (69) |
| 6     | 数据修改和删除程序                  | (71) |
| 6.1   | 数据修改                       | (71) |
| 6.1.1 | 使用 BROWSE 命令修改数据           | (71) |
| 6.1.2 | 利用格式文件修改数据                 | (71) |
| 6.1.3 | 成批修改数据                     | (72) |
| 6.1.4 | 用一个库的数据去修改另一个库的数据          | (72) |
| 6.2   | 数据删除                       | (73) |
| 6.2.1 | 连续地删除数据                    | (73) |
| 6.2.2 | 有保护地删除数据                   | (74) |
| 6.2.3 | 成批地删除数据                    | (75) |
| 6.2.4 | 对删除记录的恢复                   | (75) |
| 6.2.5 | 以假删除提高大库的删除速度              | (76) |
| 6.3   | 数据修改和删除共用程序                | (77) |
| 6.3.1 | 利用 GET/READ 命令修改、删除数据      | (77) |
| 6.3.2 | 在同一屏幕上修改多条记录               | (78) |
| 6.3.3 | 卷屏式窗口数据修改和删除               | (79) |
| 6.3.4 | 窗口全屏幕数据修改和删除               | (82) |
| 6.3.5 | 在窗口之外修改剩余字段                | (84) |
| 7     | 查询显示程序设计                   | (86) |
| 7.1   | 简单的查询程序                    | (86) |
| 7.1.1 | 用 LOCATE 和 CONTINUE 命令进行查询 | (86) |
| 7.1.2 | 用过滤器命令实现查询                 | (87) |
| 7.1.3 | 用 SEEK 命令实现快速查询            | (87) |
| 7.1.4 | 用索引文件实现不等式条件的快速查询          | (88) |
| 7.2   | 多字段组合查询                    | (89) |
| 7.2.1 | 用 LOCATE 命令实现多字段组合查询       | (90) |

|       |                            |       |
|-------|----------------------------|-------|
| 7.2.2 | 用 ON ERROR 功能实现组合查询 .....  | (90)  |
| 7.2.3 | 用 SEEK 命令实现多字段组合快速查询 ..... | (91)  |
| 7.2.4 | 按姓氏笔划顺序查询 .....            | (92)  |
| 7.2.5 | 任意字段组合条件查询 .....           | (93)  |
| 7.3   | 模糊查询 .....                 | (94)  |
| 7.3.1 | 利用不完全匹配比较实现模糊查询 .....      | (94)  |
| 7.3.2 | 用 \$ 功能实现模糊查询 .....        | (94)  |
| 7.3.3 | 最长子串单词匹配 .....             | (95)  |
| 7.3.4 | 汉字名称缩写匹配时的模糊查找 .....       | (96)  |
| 7.3.5 | 利用通配符 * 和 ? 进行模糊查找 .....   | (98)  |
| 7.4   | 数据显示 .....                 | (99)  |
| 7.4.1 | 字段名显示 .....                | (99)  |
| 7.4.2 | 动态显示 .....                 | (100) |
| 7.4.3 | 表格方式显示 .....               | (101) |
| 7.4.4 | 竖向列表显示 .....               | (102) |
| 7.4.5 | 窗口动态命令显示技术 .....           | (104) |
| 7.4.6 | 全方位窗口移动显示技术 .....          | (106) |
| 7.4.7 | 按输入顺序倒序显示 .....            | (109) |
| 7.4.8 | 超长汉字句子的显示 .....            | (110) |
| 7.5   | 数据显示中的转换技术 .....           | (110) |
| 7.5.1 | 汉字日期的转换 .....              | (111) |
| 7.5.2 | 汉字月份的转换 .....              | (111) |
| 7.5.3 | 汉字星期的转换 .....              | (111) |
| 7.5.4 | 大写金额的转换 .....              | (112) |
| 7.5.5 | 汉字数据的转换 .....              | (113) |
| 7.6   | 通用条件查询程序 .....             | (114) |
| 7.6.1 | 确定查询字段值范围 .....            | (115) |
| 7.6.2 | 生成完整的条件表达式 .....           | (116) |
| 7.6.3 | 通用条件查询程序 .....             | (117) |
| 8     | 数据统计程序 .....               | (117) |
| 8.1   | 直接利用系统提供的统计命令 .....        | (117) |
| 8.1.1 | 求总人数、工资总额和平均工资 .....       | (117) |
| 8.1.2 | 求最高工资和最低工资者 .....          | (118) |
| 8.1.3 | 汇总产量的旬小计 .....             | (118) |
| 8.1.4 | 计算实发工资 .....               | (118) |
| 8.2   | 日期时间的统计 .....              | (119) |
| 8.2.1 | 年龄的统计 .....                | (119) |
| 8.2.2 | 时间的相加 .....                | (129) |
| 8.2.3 | 时间的相减 .....                | (120) |
| 8.3   | 复杂的数据统计 .....              | (120) |
| 8.3.1 | 分类汇总 .....                 | (120) |
| 8.3.2 | 利用数组进行统计 .....             | (121) |
| 8.3.3 | 利用数据库关联进行多库统计 .....        | (122) |

|        |                            |       |
|--------|----------------------------|-------|
| 8.3.4  | 利用 UPDATA 命令进行多库统计 .....   | (123) |
| 8.3.5  | 分类计数 .....                 | (123) |
| 8.4    | 提高统计速度的措施 .....            | (124) |
| 8.4.1  | 用 SEEK 代替 COUNT 计数 .....   | (124) |
| 8.4.2  | 用过滤器命令实现条件求和,求平均数 .....    | (124) |
| 8.4.3  | 用 COPY 命令实现条件求和,求平均数 ..... | (125) |
| 8.4.4  | 利用库关联来提高逻辑比较速度 .....       | (125) |
| 8.5    | 通用随机统计程序 .....             | (126) |
| 9      | 打印程序设计 .....               | (127) |
| 9.1    | 打印人事档案表 .....              | (127) |
| 9.1.1  | 打印人事档案表 .....              | (127) |
| 9.1.2  | 通用字型变化打印程序 .....           | (128) |
| 9.1.3  | 实线表格的打印 .....              | (129) |
| 9.1.4  | 报表的分页打印 .....              | (129) |
| 9.1.5  | 打印机跑纸控制 .....              | (130) |
| 9.1.6  | 清除打印缓冲区 .....              | (130) |
| 9.1.7  | 打印机准备好检测 .....             | (131) |
| 9.1.8  | 随时中断打印 .....               | (131) |
| 9.1.9  | 数字为零时不打印 .....             | (131) |
| 9.1.10 | 一个完整的人事档案表 .....           | (131) |
| 9.2    | 程序清单及文稿打印 .....            | (133) |
| 9.2.1  | 程序清单的单页打印 .....            | (133) |
| 9.2.2  | 程序清单的双页打印 .....            | (134) |
| 9.2.3  | 多个程序排序连续打印 .....           | (136) |
| 9.2.4  | 文本文件的文稿格式打印 .....          | (137) |
| 9.3    | 通用数据库报表打印程序 .....          | (139) |
| 9.3.1  | 生成打印信息库 .....              | (139) |
| 9.3.2  | 通用报表打印程序 .....             | (140) |
| 10     | 数据绘图程序 .....               | (142) |
| 10.1   | 清屏技术 .....                 | (142) |
| 10.1.1 | 自右至左的清屏 .....              | (142) |
| 10.1.2 | 自下而上清屏 .....               | (142) |
| 10.1.3 | 自右下角至左上角清屏 .....           | (143) |
| 10.1.4 | 自两侧向中间清屏 .....             | (143) |
| 10.1.5 | 自中间向四周清屏 .....             | (143) |
| 10.2   | 画方框技术 .....                | (144) |
| 10.2.1 | 画单线,双线方框 .....             | (144) |
| 10.2.2 | 画多字符组合的实方框 .....           | (144) |
| 10.3   | 画直方图技术 .....               | (145) |
| 10.3.1 | 各种颜色测试程序 .....             | (145) |
| 10.3.2 | 用制表符绘制直方图 .....            | (146) |

|        |                               |       |
|--------|-------------------------------|-------|
| 10.3.3 | 用卷屏命令绘制直方图                    | (147) |
| 10.3.4 | 用字符绘制横向直方图                    | (148) |
| 10.4   | 动画技术                          | (148) |
| 10.4.1 | 单个汉字的游动                       | (149) |
| 10.4.2 | 将一行文字从屏幕下方推至顶部                | (149) |
| 10.4.3 | 将一行文字分半由屏幕上下推至中间              | (150) |
| 10.4.4 | 在屏幕底部循环显示一行汉字                 | (150) |
| 10.4.5 | 放射和收缩式显示文字                    | (150) |
| 10.5   | 使用汉字 2.13 系统实现通用数据绘图程序        | (152) |
| 10.5.1 | 汉字的放大显示                       | (152) |
| 10.5.2 | 通用直方图程序                       | (153) |
| 10.5.3 | 通用折线图程序                       | (154) |
| 10.5.4 | 通用饼形图程序                       | (156) |
| 11     | 系统维护与文档建设                     | (160) |
| 11.1   | 文件备份                          | (160) |
| 11.1.1 | 成批地备份当日最新文件                   | (160) |
| 11.1.2 | 仿 BACKUP 备份文件                 | (161) |
| 11.1.3 | DOS 命令仿真                      | (162) |
| 11.2   | 源程序处理                         | (162) |
| 11.2.1 | 给源程序添加行号                      | (162) |
| 11.2.2 | 结构程序格式(缩进式)自动生成               | (163) |
| 11.2.3 | 系统调用关系表                       | (164) |
| 11.3   | 文档的自动生成                       | (167) |
| 11.3.1 | 系统数据库文件表                      | (167) |
| 11.3.2 | 打印库结构表                        | (168) |
| 11.3.3 | 生成库结构文本文件                     | (169) |
| 11.3.4 | 使用说明书的快速生成                    | (170) |
| 附录 A   | 常用汉字区位码表                      | (171) |
| 附录 B   | SET COLOR 命令颜色代码表             | (172) |
| 附录 C   | INKEY 函数接受的无法显示的键与 ASCII 码对应表 | (172) |
| 附录 D   | READKEY 函数值一览表                | (173) |
| 附录 E   | ON KEY 命令控制键与 ASCII 码对应表      | (174) |
| 附录 F   | FoxBASE+ 出错信息一览表              | (174) |

# 程序设计概述

## 1.1 程序和程序设计

### 1.1.1 程序

数据库管理系统的使用有两种方式：

1) 单命令方式 在圆点提示符下,单个的执行命令,每执行一步可以得到相应的结果,这种方式在我们初学时有用,但开发一个大的应用系统则不行。

2) 程序执行方式 首先编写程序,形成程序文件,再在系统下执行,由于程序形成了程序文件,故能反复执行,不满意还可以修改,为数据的管理提供了极大的方便。

什么叫程序呢?程序是某种计算机语言指令的有序集合,它规定了计算机执行的规则和步骤,其功能是解决某个实际问题。

### 1.1.2 程序设计

编写、研制程序的过程,就是程序设计。

例如:我们小时候学习语文,首先学字、词,后来是遣词造句,最后是写作文写诗等,学习 FoxBASE 的各种命令、语句的用法,相当于学字、词和造句,而编写程序相当于写作文。

作文或写诗是作者与读者之间交流信息、交流思想感情,而程序则是人与机器之间的信息交流。

## 1.2 程序设计的步骤

程序设计一般经过以下四个步骤:

1) 问题分析 对用户的需求进行具体分析,明确编程目标,在这一过程中人们常用 O—I—P 的顺序来确定应解决的问题。

其中:O 为 output(输出),I 为 input(输入),P 为 process(处理)。

即先明确输出要求,之后看有哪些输入,再确定该怎样处理。

2) 确定算法 算法即解决问题的方法和步骤,方法多种多样,要选择其中效率最优的方法解决上面所提出的问题。

3) 编写程序 按事先选择好的计算机语言(FoxBASE)、按照确定的算法编写相应程序。

4) 调试程序 这个过程不断反复,直到结果正确为止。将编好的程序输入计算机后,并提交机器执行,看是否得到所需要的结果,在得不到结果或结果不对时,还需要修改程序,有关程序的调试方法,将在第二章中详细叙述。

## 1.3 程序质量的标准

一个程序设计完成后,还要不断改进,使其成为最优秀的程序,那么怎样的程序才是高质量的、

最好的程序呢？有以下6个衡量标准：

1) 正确性 一个好程序首先必须保证在执行后得到正确结果，在计算机硬件环境正常的情况下，计算机出错的主要原因是软件不完善，即程序有错，程序有错，则结果有错，因而毫无价值，所以保证正确性是第一位的。

2) 结构性 要求程序各个功能部分(模块)相互独立，彼此间联系简单，这对多人合编程序很有利。

3) 易读性(可读性) 程序应能易读和理解，且有一个清晰的结构，因为程序是需要维护的，为了自己或他人日后修改程序，必须保证易读性，常用方法是在程序中多写说明语句，程序体按缩进方式书写。

4) 高效率 所谓高效率是指程序运行时间要短和占用存储空间要少两方面，即要求时间效率与空间效率都高，因为时间和空间是一对矛盾，所以它们不可兼得，但一个好的程序应该根据所解决问题的实际要求和条件来决定两者的最佳配合。

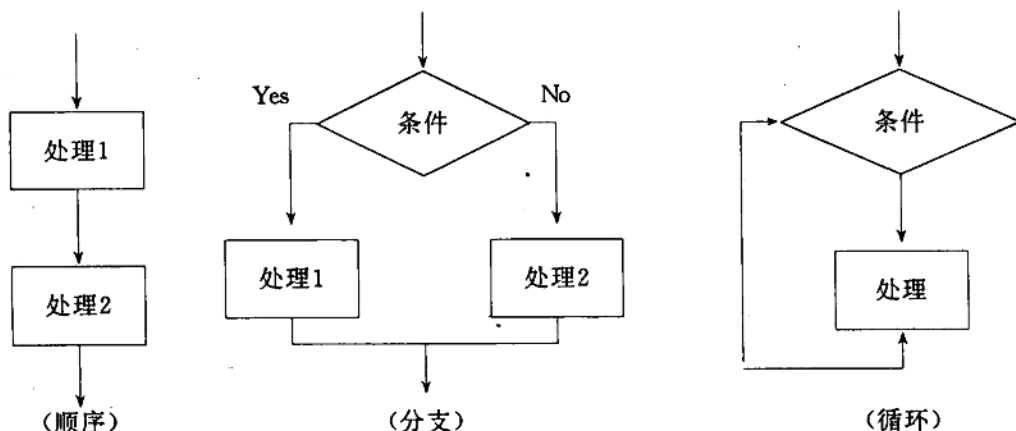
5) 适应性 这里包括两方面的含义：其一是指程序适应范围的广泛性；其二是指程序对环境和需求变化的适应程度，适应性要求程序具有抗干扰的能力，当环境或条件变化时程序运行不被破坏，具有检错和纠错能力，因此，程序研制时要充分考虑到可能的意外情况。

6) 用户友好性 程序设计出来是给用户使用的，用户界面要符合自然，充分考虑用户心理，让其感觉亲切、友好，如采用菜单技术，窗口技术，彩色图案等。

7) 操作简单性 尽可能地给用户方便，对用户的要求、约束越少越好，不给用户增加过多的负担，尽可能地提供帮助，如代码提示，操作说明等。

## 1.4 结构化程序设计方法

结构化程序设计方法是指仅用三种基本控制结构(顺序、分支、循环)就足以表示任何形式程序的方法，这三种基本控制结构如下图所示：



### 1.4.1 三种基本控制结构

1) 顺序结构 这是一种按语句的排列先后顺序依次执行的控制结构。

2) 分支结构 又称选择结构，根据判断条件成立与否而选择程序的执行方向，某一时刻只执行其中一支。



## 1.5 结构化程序设计原则

结构化程序设计的主要原则是：模块化、自顶向下、逐步求精。

### 1.5.1 模块化

就是把一个大算法按功能分为较小的算法，称之为模块，这一点与系统结构设计时的模块划分相同，模块应有以下特点：

1) 一个模块只能完成一项具体的功能，称为功能模块，如数据输入、查询、修改等，如果一个模块包括多个功能时，可以对其进一步划分，使每一模块与一具体的工作相对应。

2) 模块应具有独立性，包括：模块被调用后，不产生副作用，返回到调用程序，模块的正确性可以单独测试。

模块划分的结果是一个层次结构图，它与系统结构图相同。

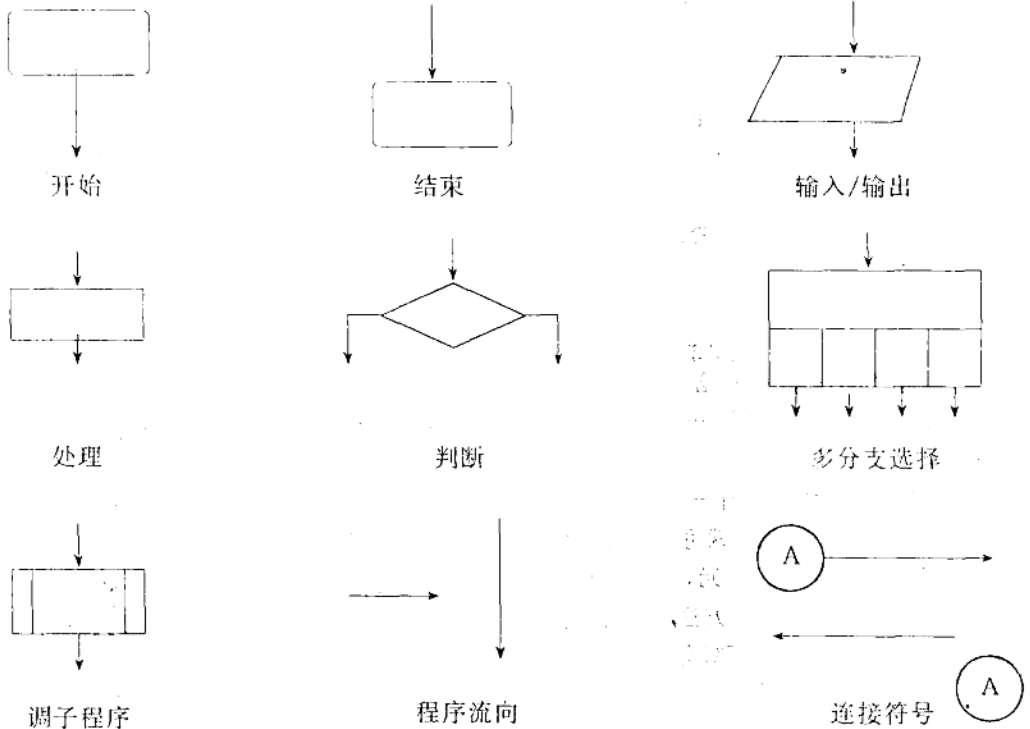
在 FoxBSE 中，模块对应一个子程序、过程，用户自定义函数，调用关系是纵向的，严格遵循上级调用下级制，其它关系均不能直接调用。

### 1.5.2 自顶向下

在程序设计时，由最上层模块开始，一层一层地往下进行，这一原则的核心是保证全局。

### 1.5.3 逐步求精

一个模块最初的功能比较简单，之后对它一步步地精确化，最终得到完善的程序，一般说来，自顶向下的模块设计过程就是逐步求精的过程。



## 1.6 算法的描述工具——流程图

在弄清问题之后,不能马上动手编程序,应先有算法(解决问题的具体方法);有了算法,如何表示出来(即描述算法),描述算法的工具较多,如文字描述、流程图等,较为常用的方法是流程图。流程图是一种描述程序处理过程(算法)的图示方法,图示符号一般规定如下:

举例:

求一元二次方程  $AX^2+BX+C=0$  ( $A \neq 0$ ) 的解

文字描述解决问题的方法:

- 1) 输入三个系数 A,B,C
- 2) 判断  $A=0$ ? 若是,则  $X=-C/B$ ,结束; 否则转 3
- 3) 求判别式  $W=B^2-4AC$

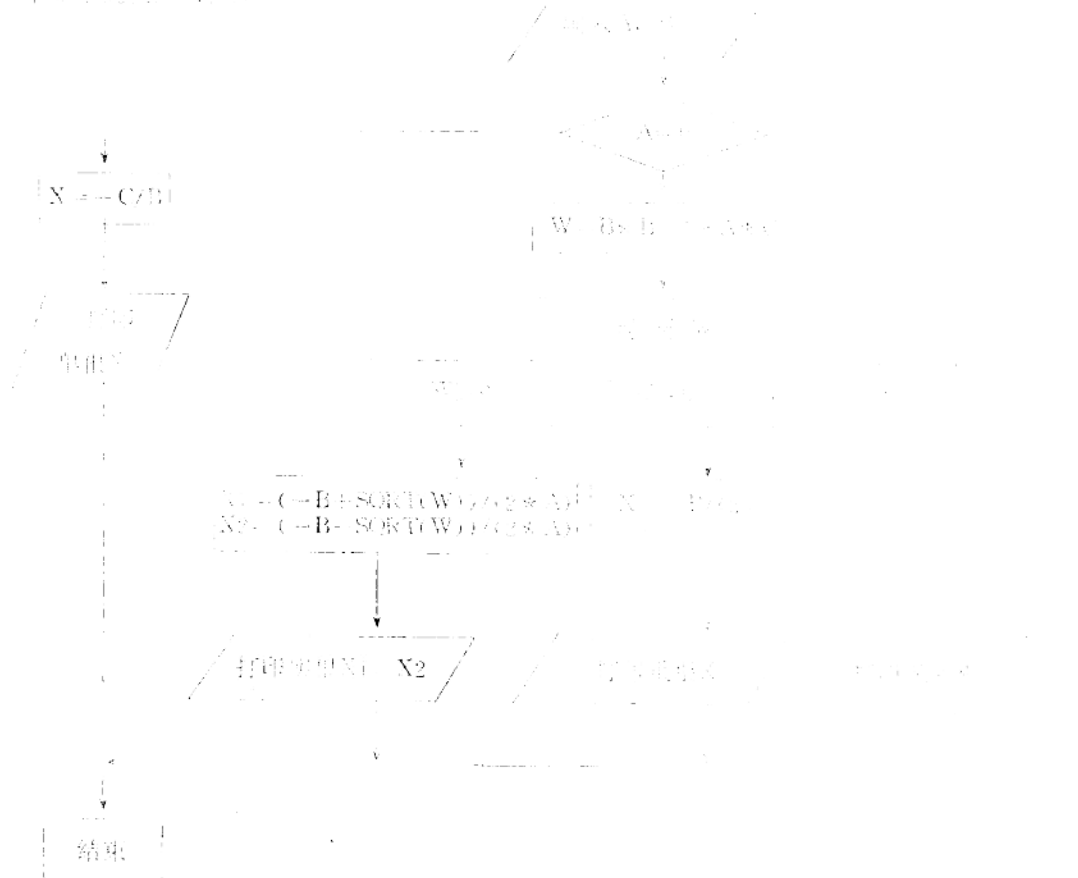
有三种情况:

- (1)  $W > 0$ : 实根  $X_1, X_2, X_1 = (-B + \sqrt{W}) / (2 * A)$
- (2)  $W = 0$ : 实根  $X_1 = X_2 = -B / (2 * A)$
- (3)  $W < 0$ : 无实根

流程图描述:

流程图:

下面流程图描述如下:



下面根据框图编写成程序:

```
*      求解一元二次方程      P1-1.PRG
SET TALK OFF
CLEAR
INPUT "输入第一系数: " TO A
INPUT "输入第二系数: " TO B
INPUT "输入第三系数: " TO C
IF A=0
    X=-C/B
    ? "仅有一个根为: ",X
ELSE
    W=B*B-4*A*C
    DO CASE
        CASE W>0
            X1=(-B+SQRT(W))/(2*A)
            X2=(-B-SQRT(W))/(2*A)
            ? "实根: X1=",X1
            ? "      X2=",X2
        CASE W=0
            X=-B/(2*A)
            ? "重根: X1=X2=",X
        CASE W<0
            ? "没有实根"
    ENDCASE
ENDIF
RETURN
```

## 2 程序的调试和维护

### 2.1 程序错误类型

手工编制的程序总是有这样或那样的错误,因此,调试纠错工作必不可少,程序错误分为两类,即语法错误和逻辑错误。

#### 2.1.1 语法错误

语法错误是指那些违反 FoxBASE 语法规则的错误,这种错误在程序执行时能被 FoxBASE 指出来。

例如: .S=3.14159\*(R1\*\*2-R2\*\*2)

漏掉括号,因而括号不配对。

.R=-10

.X=SQRT(R)

负数不能开平方。

.X="武汉"

.Y=98

.Z=X+Y

类型不匹配,字符型变量 X 和数字型变量 Y 相加。

.CLEAE

系统显示 "不可识别的命令动词",命令拼写错误,应该是 CLEAR。

语法错误比较容易找,执行时会显示出来,一般来说,屏幕显示箭头处的命令有错误。

#### 2.1.2 逻辑错误

逻辑错误指那些不违反 FoxBASE 语法规则,但却不合逻辑或不合题意的错误,这种错误无法由系统检测出来。

例如: 1) 结果不对

求环形面积?

.R1=20

.R2=10

.S=3.14159\*(R1^2-R2^2)

程序中漏了\*,将平方写成了乘号形式对,但结果不对。

2) 死循环

.X=1

.DO WHILE X<5

(循环体中既没有对变量 X 增值,又无退出循环语句)

.ENDDO

### 3) 结构语句不配对

结构语句有: IF ... ELSE ... ENDIF  
DO CASE ... ENDCASE  
DO WHILE ... ENDDO

### 4) 嵌套交叉错误

|              |              |
|--------------|--------------|
| DO WHILE ... | IF ...       |
| IF ...       | DO WHILE ... |
| ENDDO        | ENDIF        |
| ENDIF        | ENDDO        |

## 2.1.3 调试步骤

程序调试前,先应作好准备工作,准备工作包括:

- 1) 程序清单(短程序可不用);
- 2) 供调试使用的数据(或数据库),包括正常数据和异常(错误)数据;
- 3) 支援程序(相关程序)。

程序调试工作与程序设计工作相反,应该自下而上进行,即先调试最低层的小模块,再调试高层的大模块。

在一个模块调试用另外的模块时,要特别注意接口的工作情况,变量参数等。

在调试当中还要注意:

- 1) 先纠正语法错误,再找逻辑错误,即先调“通”。
- 2) 要采用不同的数据进行调试,正常数据,边界数据,异常数据都要试。

正常数据,检验结果对否?

边界数据,检验能否正常运行?

异常数据,检验程序是否有检错、纠错等功能?

- 3) 对于程序中的分支,要测试每条分支的执行情况,并作出记录,以免遗漏,如求解一元二次方程,对于单根、重根、无实根,都要测试一遍。

## 2.2 程序调试方法

为了便于用户查找程序中的错误,FoxBASE 提供了下列调试方法:保留和查看历史、跟踪程序执行、设置折点、模拟键盘输入等。

### 2.2.1 保留和查看历史

什么叫“历史”?在 FoxBASE 中,用于保存已执行过的命令的缓冲区,称为“历史”,通常情况下,历史中存放 20 条命令,当历史中达到 20 条命令时,就清除最初存放的命令。

历史有什么用途呢?基本用途有三个:

- 1) 按顺序保存命令以及它们的格式,以便用户进行查错。在园点提示符下,当用户输入完命令行,按回车键之后,将自动存入历史中,必要时用户可以对历史的命令进行查看。

- 2) 重新使用曾执行过的命令,减少按键次数,利用上下编辑键“↑”、“↓”从历史中重新显示以前执行的命令,顺序为后进先出。当显示到所需命令时,按回车键,系统便执行该命令。