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

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。

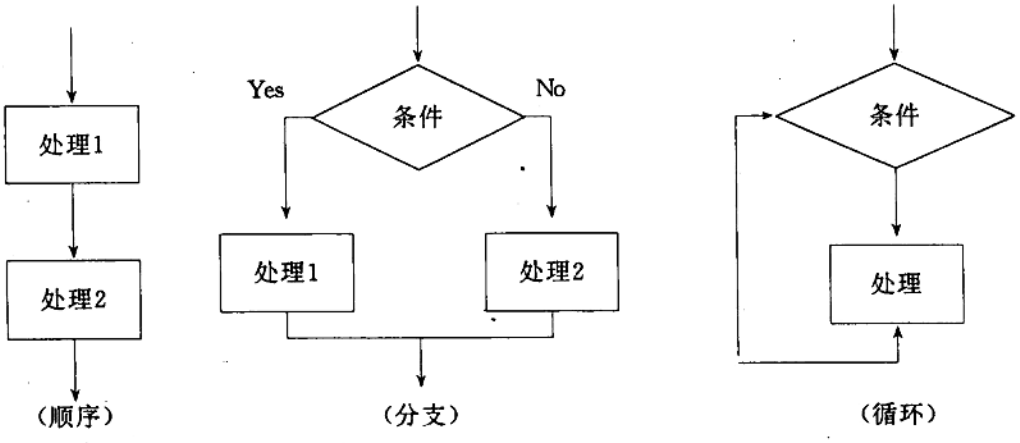
5）适应性　这里包括两方面的含义：其一是指程序适应范围的广泛性；其二是指程序对环境和需求变化的适应程度，适应性要求程序具有抗干扰的能力，当环境或条件变化时程序运行不被破坏，具有检错和纠错能力，因此，程序研制时要充分考虑到可能的意外情况。

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　尽可能地给用户提供方便，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。

3）循环结构　又称重复结构，根据判断条件，若成立则反复执行循环中某一功能，直到条件不成立，退出循环结构为止。

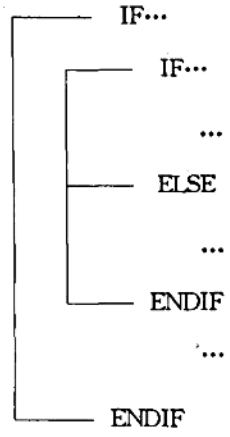
1.4.2 结构化程序设计特点：

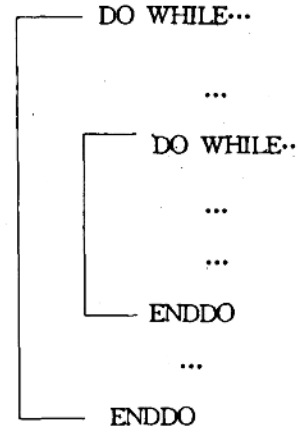
1）每种结构都严格地只有一个入口和一个出口，在入口和出口之间构成了控制结构。

2）可以形成嵌套，即三种结构可以自身嵌套或相互嵌套使用。

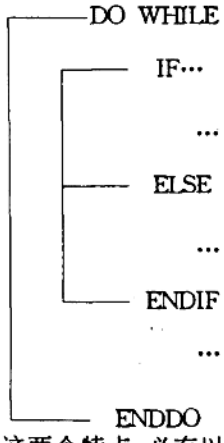
例如：

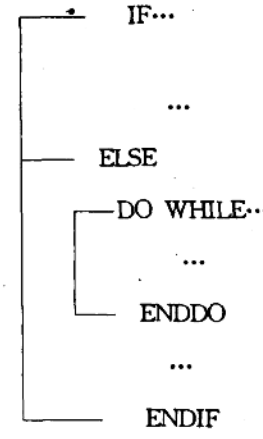
自身嵌套





相互嵌套





由于这两个特点，必有以下优点：

1）由于每种结构仅有一个入口和一个出口，具有相对的独立性，使得程序的编写和调试比较容易。

2）由于每种结构仅有一个入口和一个出口，程序执行时的控制流程与程序的静态描述相对应，利用静态描述的方法，就可以正确地理解程序动作的模拟结果。

3）能以控制结构为单元，从上到下地顺序地阅读程序文本。

与非结构化程序设计方法相比较，结构化程序设计不需要使用语句标号，没有跳转语句 GO-TO（在 FoxBASE 中 GOTO 语句是移动指针语句）。

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。

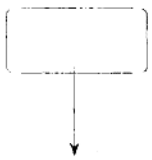
在 FoxBSE 中，模块对应一个子程序、过程，用户自定义函数，调用关系是纵向的，严格遵循上级调用下级制，其它关系均不能直接调用。

1.5.2 自顶向下

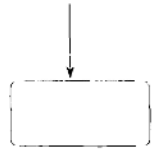
在程序设计时，由最上层模块开始，一层一层地往下进行，这一原则的核心是保证全局。

1.5.3 逐步求精

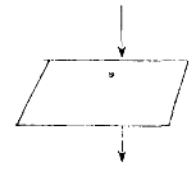
一个模块最初的功能比较简单，之后对它一步步地精确化，最终得到完善的程序，一般说来，自顶向下的模块设计过程就是逐步求精的过程。



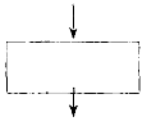
开始



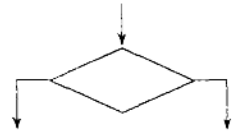
结束



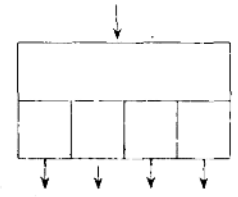
输入/输出



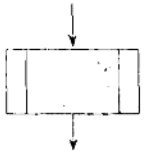
处理



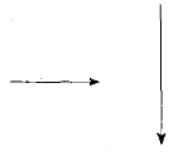
判断



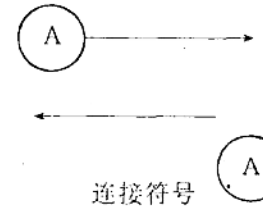
多分支选择



调子程序



程序流向



连接符号

2.6 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）的图示方式，符号含义一般规定如下图：

举例：

求 [illegible] 的解

[illegible]，下面是解决问题的方法：

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； [illegible]

（2） W＝0； [illegible]

（3） W＜0； [illegible]

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 语法规定的错误，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 语法规定，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、模拟健盘输入等。

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。