巧编通用数据处理程序

陈　伟　（安徽省合肥市统计局计算站）

使编写出的程序具有良好的操作环境和通用性，是众多程序编写者追求的目标，可是由于许多编程者对通用程序分析不够全面，使现有的许多程序具有某些方面的缺陷，比如：有些通用制表打印软件对打印机型号和汉字操作系统的要求十分苛求，以至于在很大程度上无法达到通用的功能。因此本文在介绍经验之前，首先分析一下数据处理程序的组成。

一个数据处理程序主要分为三个部分：输入、数据处理和输出。那么要编写通用程序就必须针对这些方面来考虑，本文就以编写程序时应注意的问题来谈一谈。

一、输入

读者不难发现，如今程序设计技术已有了较大的发展，从过去的程序适用单一性，到现在的通用性，直到发展成为将来的程序自我生成。其实它们之间也有着内在的联系和区别，所以现在有些同志把杂志上刊登的一些生成程序拿来使用在自己的通用程序中的方法是错误的，因为通用性程序的复杂度和执行时间都要比自动编程小的多，因此用户完全可以编写出比自动编程满意的通用程序，就拿输入方面来说：自动编程系统Auto-dBASE有一个具有极强通用性的输入程序，它的设计思路是首先通过用户的操作给定一些必要的参数，然后系统再通过这些参数去分析执行，但是它没有考虑到用户的操作环境，所以程序编写的并不十分完善。而通用性程序就没有Auto-dBASE的某些执行步骤（如：给定参数），所以无法也无需去套用Auto-dBASE的输入部分。

笔者在分析了众多的输入程序后，认为实现通用输入程序的办法只有两种：①dBASEⅢ中有COPY STRU EXTE TO 文件名和CREA 数据库 FROM 文件名两条截取和生成数据库结构的指令，它们给编程者分析数据库的构造带来了巨大的方便。第一种输入方案和上面说到的Auto-dBASE就是建立在这两条指令的基础上，而且许多杂志上刊登的通用输入方案也就以此为思路来编制程序的。但这种输入方案对字段较多且长度参差不齐的数据库来说，就显得十分脆弱，因为它将把用户带入到一个屏幕格式杂乱的输入状态下。②brow是dBASE Ⅲ中一条很有特色的指令，以至于后来的FoxBASE对它进行了大幅度的修改，使其具有极强的输入功能，可是由于借口“数据的保密性”，在编译dBASE中未能提供该项功能，这样似乎就使许多编程者“遗忘”了该条指令，其实用程序实现的该条指令不仅能在解释和编译两种状态下运行，而且它具有极强的通用性和良好的操作环境。至于“保密性”，完全可以在编程时限定用户的输入范围，使一些字段不能被用户修改或只能显示。用户在使用brow指令时会发现，该条指令将占据整个屏幕的空间，这样将会破坏屏幕的整体性，并且将使一些必要的输入提示内容无法显示，而用程序实现的该条指令可以在屏幕上任意区域开出一个窗口供用户输入和修改数据，但不影响窗口以外的任何信息，这样就给编程者为用户提供输入信息带来了方便。

作为模拟brow命令的子程序，在调用时需要一些必要的运行参数。TS：编程者给用户输入时的提示，DBF：存放输入库名，X__TAB、Y__TAB：窗口的起始横、纵坐标，X__TAB1、Y__TAB1：窗口的终止横、纵坐标，F__EOF：用来判断输入库DBF中是否存放内容，在调用前赋以.F.，如果返回时为.T.则表明DBF为空库，BR__FIELDS：存放输入和显示的字段名，区别输入和显示的标志是每个字段名后跟着一个逗号或点号，它们即作为字段名之间的分隔符，也作为判断符，逗号(,)表示相应的字段只显示而不能被用户修改，并且在屏幕上、下、左、右滚动时，这些字段一直锁定在屏幕的左边，点号(.)表示该字段是用户要求输入的字段。程序要求编程者给BR__FIELDS赋值时，显示的字段名只能出现在

```
USE &DBF
IF EOF()
  ?? CHR(7)
  @ L__S,5 SAY DBF+'是空库,无法操作!'
  TEMP=INKEY(3)
  @ L__S,0 SAY SPACE(80)
  F__EOF=.T.
  RETURN
ENDIF
@ X__TAB,Y__TAB TO X__TAB1,Y__TAB1
@ L__S,3 SAY TS
TEMP1=BR__H
TEMP=BR__FIELDS
STOR 0 TO SU,I
DO WHILE (AT('.',TEMP)<>0).OR.(AT(',',TEMP)<>0)
  X1=AT('.',TEMP)
  X2=AT(',',TEMP)
  I=I+1
  IF I<=9
    J=STR(I,1)
  ELSE
    J=STR(I,2)
  ENDIF
  IF (X1=0.OR.X2<X1).AND.X2<>0
    X1=X2
    BR__T&J=.F.
  ELSE
    SU=SU+1
    BR__T&J=.T.
  ENDIF
  BR__FIE&J=TRIM(LTRIM(LEFT(TEMP,X1-1)))
  IF LEN(TEMP)=X1
    TEMP=''
  ELSE
    TEMP=SUBSTR(TEMP,X1+1)
  ENDIF
  X3=AT(',',TEMP1)
  BR__H&J=TRIM(LTRIM(LEFT(TEMP1,X3-1)))
  IF LEN(TEMP1)=X3
    TEMP1=''
  ELSE
    TEMP1=SUBSTR(TEMP1,X3+1)
  ENDIF
ENDDO
NOTDO=.T.
STOR .F. TO X__RE,X__END
STOR 1 TO P__RE,FIRST,P__BE,X__SAY
COPY TO TEM STRUCTURE EXTENDED
SELE 3
USE TEM
SELE 1
DO WHILE NOTDO
  @ X__TAB+1,Y__TAB+1 CLEAR TO X__TAB1-1,Y__TAB1-1
  GO P__BE
  M=X__TAB+1
  DO WHILE M<=X__TAB1-1
    N=1
    K=0
    MN=Y__TAB+2
    DO WHILE N<=I
      IF N<=9
        J=STR(N,1)
      ELSE
        J=STR(N,2)
      ENDIF
      N=N+1
      IF BR__T&J
        K=K+1
      ENDIF
      IF (.NOT.BR__T&J).OR.K>=X__SAY
        SELE 3
        LOCATE ALL FOR FIELD__NAME=BR__FIE&J
        SELE 1
        IF Y__TAB1-MN-1<=C->FIELD__LE
          X__SAY1=K-1
          EXIT
        ENDIF
        IF N>I
          X__END=.T.
        ELSE
          X__END=.F.
        ENDIF
        X__SAY1=K
        CW=BR__FIE&J
        @ M,MN SAY &CW
      ENDIF
      MN=COL()+2
    ENDDO
    SKIP
    IF EOF()
      EXIT
    ENDIF
    M=M+1
  ENDDO
  DO WHILE .T.
    RE=RECCOUNT()
    M1=P__BE+P__RE-1
    IF M1<=RE
      GO M1
    ELSE
      GO RE
      P__RE=RE-P__BE+1
      M1=RE
    ENDIF
    X__RE=.F.
    DO WHILE .T.
      N=1
      SU1=0
      MN=Y__TAB+2
      DO WHILE N<=I
        IF N<=9
          J=STR(N,1)
        ELSE
          J=STR(N,2)
        ENDIF
        N=N+1
        SELE 3
        LOCATE ALL FOR FIELD__NAME=BR__FIE&J
        SELE 1
        IF .NOT.BR__T&J
          CW=BR__FIE&J
          @ X__TAB+P__RE,MN SAY &CW
        ELSE
          SU1=SU1+1
          IF FIRST=SU1.AND.X__SAY<=
            FIRST.AND.FIRST<=X__SAY1
            MN1=MN
            X__RE=.T.
            CW=BR__FIE&J
            @ X__TAB+P__RE,MN SAY &CW
            ROW=ROW()
            COL=COL()
            CW1=BR__FIE&J
            @ L__S,41 SAY '字段:'
            @ L__S,48 SAY SPACE(31)
            SET COLOR TO &L__GG1.&L__DD/&L__GG
              .&L__CC,&L__GG.&L__CC/&L__GG1
```

```
            .&L__DD,&L__GG2.&L__BB
          @ L__S,48 SAY BR__H&J
          SET COLOR TO &L__GG.&L__CC / &L__GG1
              .&L__DD,&L__GG1.&L__DD / &L__GG
              .&L__CC,&L__GG2.&L__BB
          @ ROW,COL SAY ''
        ELSE
          IF BR__T&J
            IF SU1 > = X__SAY
              IF SU1 > X__SAY1
                EXIT
              ENDIF
              CW = BR__FIE&J
              @ X__TAB+P__RE,MN SAY &CW
            ENDIF
          ENDIF
        ENDIF
      ENDIF
      MN = COL()+2
    ENDDO
    IF X__RE
      @ X__TAB+P__RE,MN1 GET &CW1
      READ
      CW = &CW1
      @ X__TAB+P__RE,MN1 SAY CW
      EXIT
    ELSE
      EXIT
    ENDIF
    LOOP
  ENDDO
  IF X__RE
    PKEY = MOD(READKEY(),256)
    IF PKEY = 12
      PKEY = 14
    ENDIF
    IF PKEY = 15
      PKEY = 1
    ENDIF
  ELSE
    PKEY = INKEY(0)
    DO CASE
      CASE PKEY = 30
        PKEY = 35
      CASE PKEY = 31
        PKEY = 34
      CASE PKEY = 23
        PKEY = 14
      OTHER
        PKEY = 1000
    ENDCASE
  ENDIF
  DO CASE
    CASE PKEY = 0
      IF (FIRST = 1.AND.P__BE < > 1).OR.(FIRST =
          X__SAY.AND.P__BE < > 1)
        IF P__RE = 1
          P__BE = P__BE-1
          EXIT
        ELSE
          P__RE = P__RE-1
        ENDIF
      ELSE
        IF P__BE < > 1.OR.(P__BE = 1.AND.FIRST < > 1)
          IF FIRST-1 > = X__SAY
            FIRST = FIRST-1
          ENDIF
        ENDIF
      ENDIF
    CASE PKEY = 1
      RE = RECCOUNT()
      IF FIRST = SU.OR.FIRST = X__SAY1
        IF M1 < > RE
          FIRST = X__SAY
        ENDIF
        IF P__RE = X__TAB1-X__TAB-1.AND.M1 < > RE
          P__BE = P__BE+1
          EXIT
        ELSE
          IF M1 < > RE
            P__RE = P__RE+1
          ENDIF
        ENDIF
      ELSE
        FIRST = FIRST+1
      ENDIF
    CASE PKEY = 2
      P__RE = P__BE
      FIRST = X__SAY
    CASE PKEY = 3
      P__RE = X__TAB1-X__TAB-1
      FIRST = X__SAY1
    CASE PKEY = 4
      IF P__RE = 1.AND.P__BE < > 1
        P__BE = P__BE-1
        EXIT
      ELSE
        IF P__RE < > 1
          P__RE = P__RE-1
        ENDIF
      ENDIF
    CASE PKEY = 5
      RE = RECCOUNT()
      IF M1 < > RE.AND.P__RE = X__TAB1-X__TAB-1
        P__BE = P__BE+1
        EXIT
      ELSE
        IF M1 < > RE
          P__RE = P__RE+1
        ENDIF
      ENDIF
    CASE PKEY = 6
      IF P__BE-(X__TAB1-X__TAB-1) < = 1
        P__BE = 1
      ELSE
        P__BE = P__BE-(X__TAB1-X__TAB-1)
      ENDIF
      EXIT
    CASE PKEY = 7
      RE = RECCOUNT()
      IF P__BE+X__TAB1-X__TAB-1 < = RE
        P__BE = P__BE+X__TAB1-X__TAB-1
        EXIT
      ENDIF
    CASE PKEY = 14
      NOTDO = .F.
      EXIT
    CASE PKEY = 34
      IF X__SAY < > 1
        X__SAY = X__SAY-1
        EXIT
      ENDIF
    CASE PKEY = 35
      IF .NOT.X__END
        X__SAY = X__SAY+1
        EXIT
      ENDIF
```

```
    ENDCASE
  ENDDO
ENDDO
   L__S,0 SAY SPACE(80)
SELE 3
USE
ERASE TEM.DBF
SELE 1
RETURN
```

输入字段名前面。BR__H: BR__FIELDS 的汉文名称，L__S: 屏幕的最底行，L__DD、L__CC、L__BB: 屏幕的前景、后台和边框的颜色。

子程序具体实现步骤如下: (1)首先判断 DBF 是否为空库; (2)分割字段名并且判断它们相应的属性; (3)在屏幕上显示字段并等待输入; (4)通过对输入键值的判断，执行相应的操作。在这些步骤中，实现最为困难但也最为重要的就是第 4 步，用程序能否完成该子程序主要取决与此，值得庆幸的是 FoxBASE 和编译 dBASE 中都提供了 INKEY 和 READKEY(在编译 DBASE 中是 LASTKEY)函数，它们给程序的编制带来极大的方便。(由于篇幅所限，笔者对子程序进行了一些必要的删减，以减少程序行数，在实际使用中用户可以根据需要适当添加功能。)

二、数据处理

这个环节是通用程序中最为重要也最为复杂的一个部分，当然涉及到这个部分的内容也十分广泛，所以这部分的程序不能简单地归并在几种运算操作中，可是做为编程者不可能把用户的所有要求合并起来，编写出一种万能的数据处理程序，而只能做到尽可能编制出一些常用的运算操作模块，当用户使用时，可以合理的组织这些功能来达到目的。

数据处理程序中还有一些环节必不可少: ①输入数据处理条件的模块，实现该功能的方法很多: 有借用字处理或行编辑软件的; 有编写小型字处理程序的; 还有通过菜单选择的。而在具体编写程序时，我们选用的方法应主要注意编程时所用的语言。②分析数据处理条件的模块，该模块是连接数据处理条件和数据处理的桥梁，它的主要功能是通过分析处理条件来执行运算操作模块。

三、输出

常用的程序输出一共有两种: ①屏幕输出; ②打印输出，而其中打印输出最为常见，所以本文就以打印输出为代表来说明输出部分。

在本文的一开始，笔者就提到了打印方面的问题，笔者认为在众多的通用程序中，打印输出的效果并非不佳，而最为主要的方面是“通用”未得到体现，但作为一个通用程序如果不体现“通用”方面的能力，那么这个程序几乎是失败的。

要编写通用的打印输出程序，我们就必须首先分析国内的打印机和汉字操作系统，在国内汉字操作系统和打印机的操作码各不相同，而且又都各自拥有自己的用户，所以我们在编写程序时，绝不能以一种汉字操作系统或一种打印机作为最终目标来编写通用程序，最好的方法是建立打印机库，在这个库中存放一些在程序使用中可能遇到的打印机操作码信息，当用户在使用时，可以针对自己的打印机来具体定义系统的打印机信息。下面笔者就提供用这种思路实现的打印程序，由于篇幅所限，笔者只提供其中的主要部分:

```
USE PRINTER
* 打开打印机信息库
LOCATE ALL FOR 选中标志='1'
* 查找系统所使用的打印机信息
@ PROW(),PCOL() SAY &表头控制符
@ PROW()+1,26 SAY '一九九一年冶金产品产量计划'
@ PROW(),1 SAY &副表头
@ PROW()+1,55 SAY '单位: 吨'
@ PROW(),PCOL() SAY &表体主栏
.
.
.
* 打印表体主栏
@ PROW(),PCOL() SAY &打印末
SET DEVI TO SCREEN
SET PRINT OFF
RETURN
```

通用打印程序所要处理的另一方面就是编写通用数据库的打印程序，这在大部分的应用软件中都得到了完美的解决，笔者就不再介绍了。

综上所述，要编写出一个好的程序，关键在于分析问题的深度，当今用程序可以完成各种复杂的数据处理，这样就使过去注重的程序实现方法，转变到全面分析问题的深度上来。

作者已经编写好该文中未提供的 Auto-dBASEⅢ源程序，如需要请与合肥市统计局计算站作者联系。邮编: 230001。