写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（1）

这部分我也循序渐进的讲解如何 RPG 里头更项的含义。

先说说 RPG 一般用来做什么吧，举例说：
1. 整个 RPG 程序不包含任何外部程序，只是做些简单的数学运算，或者对数据区、DATAQ 等非文件目标进行操作，这属于相对最简单的情况；
2. RPG 程序对物理文件或者逻辑文件进行读、写、更新、删除操作；
3. RPG 程序结合物理文件或逻辑文件、以及显示文件进行操作，这属于比较复杂的情况；
4. RPG 程序结合物理文件或逻辑文件、显示文件、以及 ICF 文件（即通讯文件）进行操作，这种情况也比较复杂。
当然，以上的情况只是平时使用的归纳，你也可以根据自己的实际需要结合各种情况，此处不再深入探讨。

以下结合例子简单说说各种情况，假设以下源码文件存放的路径是 库 MYLIB、源物理文件 MYSRCPF
1. (1)做个简单的运算，比如 AXB/(C-D)+E，有加减乘除等运算，代码如下：

```
.....CL0N01N02N03Factor1+++OpcdeFactor2+++ResultLenDHHiLoEqComments++++
*************** Beginning of data ************************************
     C                                   Z-ADD2                 A
10
     C                                   Z-ADD10                B             2
0
     C                                   Z-ADD5                 C
10
     C                                   Z-ADD1                 D
10
     C                                   Z-ADD2                 E
10
     C                                   Z-ADD0                 TEM           30

     C                                   Z-ADD0                 RST           30

     C                     A             MULT  B                TEM

     C                     C             SUB   D                RST

     C                     TEM           DIV   RST              RST

     C                                   MVR                    TEM2          3
0
     C                     RST           ADD   E                RST

     C                     RST           DSPLY
```

```
.....CL0N01N02N03Factor1+++OpcdeFactor2+++ResultLenDHHiLoEqComments++++
     C                                        SETON
       LR
****************** End of data ************************************
```

【注】

(a) Z-ADD 表明把 Factor2 处的变量值赋给 Result 处的变量，值得注意的是这两个变量都是数值型的，例如，"Z-ADD2　　A　　10"表示把 2 赋给变量 A，后面的"10"中的“1”表示变量 A 的长度是 1，“0”表示小数位长度是 0，即 A 是一个长度为 1 整型变量；

(b) MULT 表示做乘法运算，“A　　MULT B　　TEM”表示 AXB，把结果赋给整型变量 TEM；

(c) SUB 表示做减法运算，“C　　SUB　D　　RST”表示 C-D，把结果赋给整型变量 RST；

(d) DIV 表示做除法运算，“TEM　　DIV　RST　　RST”表示 TEM/RST，把结果赋给整型变量 RST；

(e) MVR 表示做取余数运算，“MVR　　TEM2　　30”表示把余数赋给整型变量 TEM2；

(f) ADD 表示做取加法运算，“RST　　ADD　E　　RST”表示 RST+E,把结果赋给整型变量 RST，这句一般写成“ADD　E　　RST”也可以。

(g) DSPLY 表示显示变量值，“RST　　DSPLY”表示显示变量 RST 的值；

(h) SETON 表示给指示器置为*ON 的状态，即'1'，“SETON　　LR”表示把 LR 指示器置为*ON，该句的意思表示最后一条记录，即程序结束。

按 F3 保存，假设该文件名是 MYRPG，所在的库是 MYLIB，源物理文件是 MYSRCPF。

程序写好之后，应该进行编译了。用 WRKMBRPDM FILE(MYLIB/MYSRCPF)，找到刚才的文件 MYRPG，输入 14，按 F4，进入“Create RPG/400 Program (CRTRPGPGM)” 画面，参数：

(i) Program： 生成的目标名（假如为 MYOBJ）；

(ii) Library：　生成的目标存放的库（假如为 MYLIB）。

参数填好之后，按确认键即进行编译，如果程序没错，则生成目标文件，假设为 MYOBJ。

在命令行输入 CALL MYLIB/MYOBJ，这时命令上应该显示 RST 的值。

(【注】目标即可执行文件)

续《写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（1）》

上文中,只举例说明了如何进行简单的数学运算,下面再举一例子讲解如何操作数据区(DATAARA)和数据队列（DTAQ）。

==========

1. (1) 如何建立数据区（DATAARA）

【注：下面只建立*CHAR 型的数据区，其他类型的有兴趣可以自己试试】：

在系统命令行输入 CRTDTAARA，按 F4，进入“Create Data Area (CRTDTAARA)”画面，参数如下：

(a) Data area： 数据区的名字，这里假设为 MYDTAARA；

(b) Library：　存放数据区的库，默认值是*CURLIB，即当前库，这里我们输入 MYLIB；

【注】MYLIB 库之前的文章里头已建立，这里不再重复讲述了，如果还不知道如何建立，请参考《写给刚接触 AS/400 的朋友 --- 如何建立属于自己的库、文件和成员》

(c) Type：　　数据区的类型，
*DEC 数值型，
*CHAR 字符型，
*LGL 逻辑型，
*DDM 分布式数据管理（这个我没用过），
这里我们选择*CHAR；

(d) Length：　数据区的长度，这里假设是 20，当然你可以根据自己需要输入对应的长度值；

(e) Decimal positions：　指小数位位数，一般是针对*DEC 型的才需要设置；

(f) Initial value：　　数据区初始值，可填可不填，我们这里选择不填，在程序里进行更改；

(g) Text：　　注释。

整个命令请参考如下：

```
CRTDTAARA DTAARA(MYLIB/MYDTAARA) TYPE(*CHAR) LEN(20)
TEXT('建立属于自己的数据区 MYDTAARA')
```

(2)在 RPG 里如何操作数据区，假设该源码文件名为 DTAARAR，存放在 MYLIB/MYFILE 下：

```
 Columns . . . :    1  71            Edit                        MYLIB/MYFILE
 SEU==>                                                            DTAARAR
 FMT DS .....IDsname....NODsExt-file++.............OccrLen+...................
********** Beginning of data ********************************
0001.00       I#DTA             DS                                      20

FMT J      ......I.................................PFromTo++DF i e l d+L1M1FrP1MnZr.
0002.00       I                                          1       8
CURDAT
0003.00       I                                         11      16
CURTME
FMT C      .....CL0N01N02N03Factor1+++OpcdeFactor2+++ResultLenDHHiLoEq
0004.00       C                       *NAMVAR     DEFN
MYDTAARA  #DTA
0005.00       C                       *LOCK         IN       #DTA

0006.00       C                                    TIME    DATTME
  140
0007.00       C                                    MOVE
DATTME        CURDAT
0008.00       C                                    MOVELDATTME       CURTME

0009.00       C                                    OUT     #DTA

0010.00       C                                    UNLCK#DTA
```

```
0011.00        C                                                SETON
                        LR
    ****************** End of data*********************************
```

【注】

(a) 第一行#DTA 是字段名，与上面的 Dsname 左对齐，

DS 表示#DTA 是个字段，与上面的 Ds 对齐，

20 表示字段#DTA 的长度，与上面的 Len+右对齐，注意，由于我们需要用它来定义数据区

MYDTAARA 的格式，所以长度要和 MYDTAARA 相同，

整行的意思是定义一个长度 20 的字段#DTA；

(b) 第二第三行进一步详细定义字段#DTA 的内部结构，

第二行的 CURDAT 是字段名，与 Field+左对齐，1 与 From 右对齐，8 与 To++右对齐，整行的意思是

把大字段#DTA 的第 1～8 位定义位另一个字段，第三行同样的意思，在此不在赘述；

(c) 第四行的 DEFN 表示定义，一般用来定义字段和数据区，

*NAMVAR 是定义数据区格式时必用的参数，

整行的意思就是把数据区 MYDTAARA 的格式按照字段#DTA 定义，以后对字段#DTA 进行操

作就等于对数据区 MYDTAARA 进行操作；

(d) 第五行的 IN 用来读数据区，*LOCK 表示操作时把数据区 MYDTAARA 锁住，一般只有对数据区进行更

新操作才需要设置才参数，如果只是读数据区，则不用该参数；

(e) 第六行的 TIME 是取系统当前日期和时间，整行的意思是把系统当前日期和时间赋给一个 14 位长 0 位

小数的数值型变量 DATTME 中，由于系统当前的日期和时间是按照“hhmmssMMDDYYYY”存放，所

以甘好也是 14 位；

(f) 第七行意思是把变量 DATTME 右移给字段 CURDAT，所以 CURDAT 的值甘好是日期“MMDDYYYY”；

(g) 第八行意思是把变量 DATTME 左移给字段 CURTME，所以 CURTME 的值甘好是日期“hhmmss”；

(h) 第九行的 OUT 表示对数据区进行写操作，记住 Factor 处是我们程序开始定义的大字段#DTA；

(i) 第十行的 UNLCK 表示解锁，因为我们在第五行使用了参数*LOCK；

(j) 第十一行表示程序结束。

==========

按 F3 退出编辑器并保存，进行编译，生成的程序名假设为 DTAARAR，
那么在命令行 CALL DTAARAR，那么数据区已经成功更改了，
使用命令 DSPDTAARA MYDTAARA 就可以查看数据区内容了。

==========

到此为止，你已经基本懂得如何对数据区进行操作了！

写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（2）

续《写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（1）》

【上文摘要】

RPG 一般用来做什么吧，举例说：

1. 整个 RPG 程序不包含任何外部程序，只是做些简单的数学运算，或者对数据区、DATAQ 等非文件目标进行操作，这属于相对最简单的情况；

2. RPG 程序对物理文件或者逻辑文件进行读、写、更新、删除操作；

3. RPG 程序结合物理文件或逻辑文件、以及显示文件进行操作，这属于比较复杂的情况；

4. RPG 程序结合物理文件或逻辑文件、显示文件、以及 ICF 文件（即通讯文件）进行操作，这种情况也比较复杂。

当然，以上的情况只是平时使用的归纳，你也可以根据自己的实际需要结合各种情况，此处不再深入探讨。

.....

上文中讲到了第 1 点，现在讲解第 2 点。

【注】假设以下源码文件存放的路径是：库 MYLIB、源物理文件 MYSRCPF

==========

2. (1)建立物理文件：

使用 WRKMBRPDM FILE(MYLIB/MYSRCPF)进入源码存储处，按 F6，进入“Start Source Entry Utility (STRSEU)”画面，“Source member”为源码文件名，此处是我们要建立的物理文件名，假设为 MYPF，“Source type”为文件类型，这里我们输入 PF 表示该文件是物理文件源码，按确认键，进入编辑器。

代码如下：

```
.....A.....T.Name++++++RLen++TDpB.....Functions++++++++++++++++++
++++
*************** Beginning of data
**************************************
    A          R FMYPF
    A            PFFLD1                 4A          TEXT('FIELD1')
    A                                                COLHDG('FIELD1')
    A            PFFLD2                 5P 0        TEXT('FIELD2')
    A                                                COLHDG('FIELD2')
****************** End of data
**************************************
```

【注】

(a) 第一行的 R 处在 T 处，表示 Record，即声明 FMYPF 是该物理文件的记录名；

(b) 第二行中 PFFLD1 位于 Name 处，T 处为空，表示这是字段名。该句定义一个叫 PFFLD1 的字段，长度为 4，字段属性是 A，表示字符型。TEXT('FIELD1')给字段做注解，COLHDG('FIELD1')给字段指定显示时候的列表题；

(c) 第三行定义一个叫 PFFLD2 的字段，长度为 5，字段属性是 P，表示数值型，0 表示小数位是 0 位。

保存文件，编译生成目标文件 MYPF。

(2)建立逻辑文件：

建立步骤和物理文件一样，不过值得注意的是，“Source type”必须为 LF，表示该文件是逻辑文件，这里假设文件名为 MYLF，代码如下：

```
.....A..........T.Name++++++.Len++TDpB......Functions++++++++++++++++++++
++
*************** Beginning of data
************************************
    A          R FMYPF                          PFILE(MYPF)
    A          K PFFLD1
****************** End of data
****************************************
```

【注】

(a)第一行中 R 依然表示 FMYPF 是记录名，一般和物理文件的一样；

PFILE(MYPF)是声明物理文件名。

(b)第二行中 K 的位置也是处于 T 处，表示该处字段 PFFLD1 是逻辑文件的键值。

(3)建立对物理文件或者逻辑文件进行简单操作的 RPG 程序：

建立步骤和物理文件一样，不过值得注意的是，“Source type”必须为 RPG，表示该文件是 RPG 源程序文件，这里假设文件名为 MYRPG，代码如下：

```
.....FFilenameIPEAF........L..I........Device+......KExit++Entry+A....U
*************** Beginning of data
************************************
    FMYPF      O    E              DISK
    FMYLF      IF   E        K     DISK
.....F.....................Ext-record.......................RcdnbrKOptionEntry+++....
    F                 FMYPF                        KRENAMEFMYLF
.....CL0N01N02N03Factor1+++OpcdeFactor2+++ResultLenDHHiLoEqComments+
+++
    C                          MOVE 'RCD1'       PFFLD1
    C                          Z-ADD1            PFFLD2
    C                          WRITEFMYPF
    C                *LOVAL    SETLLFMYLF
    C                          READ FMYLF                          90
    C                *IN90     IFEQ '0'
    C                PFFLD1    DSPLY
    C                PFFLD2    DSPLY
    C                          ENDIF
    C                          SETON                               LR

****************** End of data
****************************************
```

【注】

(a) 第一行 MYPF 是我们上面建立的物理文件名，记住，不是记录名，别搞错了；
O 位于 IP 的 I 处，表示该文件是只允许写入；
E 位于 F 处，表示该文件是外部文件；
DISK 位于 Device 处，表示该文件是磁盘文件；
(b) 第二行 MYLF 是我们上面建立的逻辑文件名，记住，不是记录名，别搞错了；
I 位于 IP 的 I 处，表示该文件是只允许读取；
后面的 F 表示该文件是全过程文件，记住一般都要写这个 F 的就行了；
E 位于 F 处，表示该文件是外部文件；
K 位于 I..L 的 L 处，表示该文件有键值；
DISK 位于 Device 处，表示该文件是磁盘文件；
(c) 第三行的 FMYPF 位于 Ext-record 处，是逻辑文件的记录名；
“KRENAMEFMYLF”其实由三部分组成：“K”、“RENAME”和“FMYLF”。
"K"表示该句和上一行是连续的，即对上一行的补充，“RENAME”表示重命名，“FMYLF”是任意指定的。这行的意思就是把逻辑文件 MYLF 的记录名 FMYPF 重命名为 FMYLF，因为它原本的记录名和物理文件相同，所以必须重命名。
(d) 第四行中的 MOVE 表示右移，该行意思是把常量'RCD1'右移给字段 PFFLD1；
(e) 第五行中的 Z-ADD 表示对变量置 0，再赋值，该行意思是把常量 1 赋给字段 PFFLD2；
(f) 第六行中的 WRITE 表示写一笔记录，后面的 FMYPF 是物理文件记录名，记住，不是物理文件名，别搞错了，该行意思是把 PFFLD1、PFFLD2 组成的记录写入物理文件；
(g) 第七行中的 SETLL 表示给文件设置低界，后面紧跟着的 FMYLF 是逻辑文件记录名，这个是我们重命名后的名字，*LOVAL 是系统常量，表示最小值，该行的意思是把文件指针指向文件头；
(h) 第八行中的 READ 表示读一笔记录，后面的 FMYLF 是逻辑文件记录名，这个是我们重命名后的名字，
90 位于 Eq 处，表示指示器，该行的意思是读逻辑文件的一笔记录，读到的话指示器置为'0'，否则，置为'1'；
(i) 第九行的 IFEQ 表示如果等于的意思，该句的意思就是如果指示器 90 等于‘0’，即读到记录；
(j) 第十行中的 DSPLY 用来显示字段变量，该行的意思是显示字段变量 PFFLD1 的值；
(k) 第十一行中的 ENDIF 和前面的 IF 匹配，这是必须的；
(l) 第十二行中的 SETON 用来把指示器的状态改为'1'，该句意思是把指示器 LR 置为‘1’。

到此为止，你应该懂得如何往文件读写数据。

写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（2） 【续】

续《写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（2）》

上文中，只举例说明了如何对文件进行读写操作，下面再举一例子讲解如何对文件进行更新和删除：
(1)物理文件 MYPF 结构如下：

```
.....A.....T.Name++++++RLen++TDpB.....Functions++++++++++++++++++++
***************  Beginning of data
```

```
**************************************
     A          R FMYPF
     A            PFFLD1                 4A          TEXT('FIELD1')
     A                                                COLHDG('FIELD1')
     A            PFFLD2                 5P 0        TEXT('FIELD2')
     A                                                COLHDG('FIELD2')
****************** End of data
**************************************
```

(2)逻辑文件 MYLF 结构如下：

```
.....A.....T.Name++++++.Len++TDpB.....Functions++++++++++++++++++
*************** Beginning of data ************************************
     A          R FMYPF                           PFILE(MYPF)
     A          K PFFLD1
****************** End of data ***************************************
```

(3)假如文件有两笔记录如下：

```
PFFLD1          PFFLD2
================
RCD1                1
RCD2                2
================
```

(4)现需要把第一笔记录删除，把第二笔记录更新为"RCD2　　3"，代码如下：

```
.....FFilenameIPEAF........L..I........Device+......KExit++Entry+A....U
*************** Beginning of data ************************************
     FMYLF      UF   E          K       DISK
     F                FMYPF                       KRENAMEFMYLF
     C*
.....CL0N01N02N03Factor1+++OpcdeFactor2+++ResultLenDHHiLoEqComments++++
     C                'RCD1'       SETLLFMYLF
     C                'RCD1'       READEFMYLF                      90
     C                *IN90        IFEQ '0'
     C                             DELETFMYLF
     C                             ENDIF
     C*
     C                'RCD2'       SETLLFMYLF
     C                'RCD2'       READEFMYLF                      90
     C                *IN90        IFEQ '0'
     C                             Z-ADD3            PFFLD2
     C                             UPDATFMYLF
     C                             ENDIF
     C*
     C                             SETON                           LR
****************** End of data ***************************************
```

【注】
(a)第一行中的 MYLF 是逻辑文件名，
U 位于 I 处，表示该文件只作更新操作，
F 处于 P 处，表示该文件是全过程文件，一般都写 F，
E 位于 F 处，表示该文件是个外部文件，
K 位于 I 处，表示该文件有键值，
DISK 位于 Device 处，表示该文件是磁盘文件。
（注：该句声明文件 MYLF 只允许做更新操作，如果同时还要对其进行添加记录，那么只要在同行 A 处写个 A 就可以了，即

```
.....FFilenameIPEAF........L..I.........Device+......KExit++Entry+A....U
*************** Beginning of data
************************************
  FMYLF      UF   E        K      DISK                      A
）；
```

(b)第二行意思是把逻辑文件记录名 FMYPF 改为 FMYLF，这里可改可不改。如果程序里有文件出现同名的记录名，那么必须通过这种方法把其他文件记录名进行重命名；

(c)第三行把指针定位到文件开头处；
(d)第四行中的 READE 表示读等于键值的记录。该句的意思是用值”RCD1“去读逻辑文件，注意，逻辑文件的键值是字段 PFFLD1，所以执行到该句时，程序会拿这个值去找逻辑文件中字段 PFFLD1 值为'RCD1'所在的记录，找到的话指示器置为'0'，反之，置为'1'；

(e)第六行的 DELET 是删除记录，该句意思是删除当前读到的记录；
(f)第十三行的 UPDAT 是更新记录，该句的意思是更新当前的记录。

==========

到此为止，你对物理文件、逻辑文件的操作应该是基本掌握了！

写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（3）

续《写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（2）》
【上文摘要】
RPG 一般用来做什么吧，举例说：
1. 整个 RPG 程序不包含任何外部程序，只是做些简单的数学运算，或者对数据区、DATAQ 等非文件目标进行操作，这属于相对最简单的情况；
2. RPG 程序对物理文件或者逻辑文件进行读、写、更新、删除操作；
3. RPG 程序结合物理文件或逻辑文件、以及显示文件进行操作，这属于比较复杂的情况；
4. RPG 程序结合物理文件或逻辑文件、显示文件、以及 ICF 文件（即通讯文件）进行操作，这种情况也比较复杂。
当然，以上的情况只是平时使用的归纳，你也可以根据自己的实际需要结合各种情况，此处不再深入探讨。

……
上文中讲到了第 2 点，现在讲解第 3 点。
【注】假设以下源码文件存放的路径是：库 MYLIB、源物理文件 MYSRCPF
==========
==========
3. (1)物理文件 MYPF 和逻辑文件 MYLF 使用 2 中建立的就可以了，假如文件物理文件里有以下记录：
==========
PFFLD1 PFFLD2
==========
RCD1 1
RCD2 2
RCD3 3
==========

(2)建立显示文件：
建立步骤和物理文件一样，不过值得注意的是，“Source type”必须为 DSPF，表示该文件是显示文件，这里假设文件名为 MYDSPF，代码如下：

```
.....AAN01N02N03T.Name++++++RLen++TDpBLinPosFunctions++++++++++
++++++++
*************** Beginning of data
*************************************
    A                                                  DSPSIZ(24 80
*DS3)
    A                                                  CA12(12
'CANCEL')
    A                R RCDNM
    A                  FLD1                 4     O    4  26DSPATR(UL)

    A                                                  DSPATR(HI)

    A                                                4 12'PFFLD1:'
    A                                                5 12'PFFLD2:'
    A                  FLD2                 5S 0O    5 26
****************** End of data
*************************************
```

【注】
(a) 第一行指定显示大小，一般都是这个；
(b) 第二行中 CAxx 表示程序里要可用 F12 按键，'CANCEL'表示注释；
(c) 第三行 R 跟物理文件一样，表明后面 Name 对应的 RCDNM 是一个记录名，在显示文件里指一个显示画面；
(d) 第四行 FLD1 在 Name 处，这里指字段名，长度是 4，O 位于“TDpB”的 B 处，表示该字段只允许做输出用，接着的 4 位于 Lin 处，表示屏幕第 4 行，接下来的 26 位于 Pos 处，表示屏幕第 26 列。到此为止，表明在屏幕第 4 行第 26 列的位置输入字段 FLD1。

后面的 DSPATR 是显示文件关键字，是用来指定字段显示属性的，后面的 UL 是单词 UnderLine 的缩写，表明该字段带下划线，HI 是单词 High intensity 的缩写，表明该字段高亮显示；
(e) 第五行意思是在屏幕第 4 行第 12 列输出字符串“PFFLD1”；

保存文件，编译生成目标文件 MYDSPF。

(3)建立对物理文件或者逻辑文件以及显示文件进行简单操作的 RPG 程序：

```
.....FFilenameIPEAF........L..I........Device+......KExit+ +Entry+A....U
*************** Beginning of data
************************************
    FMYLF      IF    E          K       DISK
    F                FMYPF                        KRENAMEFMYLF
    FMYDSPF  CF   E                WORKSTN
    C*
.....CL0N01N02N03Factor1+++OpcdeFactor2+++ResultLenDHHiLoEqComments+
+++
    C                *IN12       DOWEQ'0'
    C                            EXFMTRCDNM
    C     12                     LEAVE
    C                FLD1        IFEQ *BLANKS
    C                            ITER
    C                            ENDIF
    C                FLD1        SETLLFMYLF
    C                FLD1        READEFMYLF                          90
    C                *IN90       IFEQ '0'
    C                            Z-ADDPFFLD2     FLD2
    C                            ENDIF
    C                           ENDDO
    C                           SETON                                LR

****************** End of data
****************************************
```

【注】
(a) 第一行声明逻辑文件 MYLF 为只读的；
(b) 第二行是第一行的延续，把逻辑文件的记录名 FMYPF 重命名为 FMYLF；
(c) 第三行中的 MYDSPF 是我们上面建立的显示文件名，
C 位于 I 处，表明允许对该文件进行写入/读出操作，
F 位于 P 处，表明该文件是全过程文件，一般都这么写，
E 位于 F 处，表明该文件是外部文件，
WORKSTN 位于 Device 处，表明该文件是显示文件或报表文件，
这行和第一行声明很相似。
(d) 第四行中的*IN12 表示指示器 12，这里为什么只用这个指示器，而不用别的呢？看看上面的显示文件 MYDSPF 的第二行吧，还记得吗？我们通过语句“CA12(12 'CANCEL') ”声明了程

序运行时要用到按键 F12，这是提供给用户进行操作用的。该句的意思是当按键没被按下时，进入循环；
(e) 第五行中的 EXFMT 是用来显示一个画面，并且停留着等待用户做出操作，该句意思是显示画面 RCDNM，并且程序停留在该处不继续执行；
(f) 第六行的 LEAVE 表示退出循环，12 位于 N01 的 01 处就是按键 F12 的状态，当用户按下 F12，那么指示器就变为'1'，这时就执行该句，反之，不执行；
(g) 第七行判断画面中的字段 FLD1 是否为空；
(h) 第八行的 ITER 表示跳到循环的开始处；
(i) 第七、八、九行的意思是，如果 FLD1 为空的话，那么跳到循环开始处执行；
(j) 第十行是把指针设置到文件头；
(k) 第十一行是读逻辑文件中字段 PFFLD1 的值等于 FLD1 的记录；
(l) 第十二行表示读到记录；
(m) 第十三行表示把逻辑文件中读到的记录的 PFFLD2 字段值赋给显示画面中的字段 FLD2；

==========

到此为止，你对显示文件应该有了一点点的了解！
由于显示文件操作起来相对复杂，接下来的功夫就靠各位刚入门的朋友多参考别人写的例子，运行运行，这样体会会更深，收获也会更大的！

写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（4）

续《写给刚接触 RPG 的朋友 --- 如何编写 RPG 程序（3）》
【上文摘要】
RPG 一般用来做什么吧，举例说：
1. 整个 RPG 程序不包含任何外部程序，只是做些简单的数学运算，或者对数据区、DATAQ 等非文件目标进行操作，这属于相对最简单的情况；
2. RPG 程序对物理文件或者逻辑文件进行读、写、更新、删除操作；
3. RPG 程序结合物理文件或逻辑文件、以及显示文件进行操作，这属于比较复杂的情况；
4. RPG 程序结合物理文件或逻辑文件、显示文件、以及 ICF 文件（即通讯文件）进行操作，这种情况也比较复杂。
当然，以上的情况只是平时使用的归纳，你也可以根据自己的实际需要结合各种情况，此处不再深入探讨。
.....
上文中讲到了第 3 点，现在讲解第 4 点。

==========

我对 SNA 通讯还不是很熟，平时也用得少，就在此大概说说吧。
首先得建立 ICF 文件，一般都是很简单的，主要含有一个用来传送数据的字段，还有 DETACH、EOS 等关键字。在程序外部需要建立并配置 LIN、CTL、DEV，要求的参数比较多。通讯原理都差不多，这是发送端：
1. 用 DEV 名 ACQ 通讯文件（即 ICF 文件）；

2. WRITE 存放数据的字段对应的记录名，即写数据；
3. 写 DETACH、EOS 结束会话。
接受端也差不多这样，即第二点的 WRITE 改为 READ，即读数据。

IBM 的 SNA 协议通讯现在用得越来越少了，现在主流是用 TCP/IP，用 C 写，SOCKET 方便，也不用配置什么参数，一般找些程序套用一下就行，嘿嘿。
通讯方面，我实在不熟，望熟悉的朋友可以举举例子供大家分享一下！

==========

刚入门的朋友可以先不要急于学这方面的知识，如果真的要学，建议好好学一下 C 吧。
一般同主机通讯就用数据队列(DATAQ)存放数据包就可以了，关于这方面，接下来我会通过举例说明的了。