# ACOS システム 2000 の TSS の使い方

大阪大学 大型計算機センター 研究開発部
馬野 元秀

はじめに

本稿は、大阪大学 大型計算機センターの ACOS システム 2000 のオペレーティング・システム ACOS-6/MVX IIのタイム・シェアリング・システム TSS-AF の使い方についてまとめたものである。

いままで、センターでは「ＴＳＳの手引」を使っていたが、内容が少し古くなった部分があるのと、かなり分厚い(190 ページ)ために、読むのが結構大変というという問題があった(現在、「ＴＳＳの手引」は残部なし。センターの資料室で貸し出しはしている)。そこで、初めてのユーザにとって、本当に必要な部分だけをまとめたのが本稿である。なお、本稿は、センターの講習会「ACOS TSS 入門」で配布していたものをベースとして、最近のＯＳのバージョン・アップに合わせて、修正を行なったものである。

内容的には、Fortran のプログラムを入力し、誤りを修正し、実行して答えを得るまでを中心に、ファイルの管理、ファイルからのデータの読み込み、ファイルへの実行結果の書き出し、簡単な行エディタの使い方、簡単なバッチの使い方について述べる。できるだけ例を用いて説明するようにし、説明は最少限に留めてある(次ページ以降、ユーザの入力は下線を引いて表わし、説明は ― の後ろに入れた。また、少し長い説明は 〈 〉 でくくって、途中に挿入した)。もう少し説明を入れた方がよいとも思ったが、説明を入れると例の流れが中断するし、つい細かいことを書いてしまう(読む方も細かいことを知りたくなってしまう)。そこで、思い切って説明を最少限にとどめた。できれば(必ず？)、実際に端末の前に座って、ディスプレィをみながら、キィーボードをたたき、一度実行させてみていただきたい。詳しい解説やマニュアルなどは、その後に読めば十分である。

なお、使用する端末としては、どこからでも使える無手順のものを想定した。ただし、もうほとんど使われていないプリンタ形式のものは考えずに、ディスプレィ形式で画面の情報が再利用できるものを考えた(PC-9800 シリーズのパソコンと端末エミュレータ ASTER の組み合わせなどはこれに当たる)。

本稿は、まだまだ十分ではないので、初めてセンターを使われる方、初心者の方、中級者の方、ベテランの方などから、誤りの指摘、問題点、感想、意見、文句、コメントなどをいただければと思っています。

## 《システムと接続する》

```
$$$CON,TSS,,ASC                  — 接続のしかたは、数通りある
HANDAI TSS(MVX2 R1.1) ON 04/20/90 AT 12:12:30 CHANNEL 5167   LU=F4E900RX

USER ID -A60000:A
PASSWORD--
パスワード                       — パスワードは見えなくなる

〈〈〈〈〈  ... 1381 YEN RESOURCES USED ( AVAILABLE .. 48619 YEN )  〉〉〉〉〉
〈〈〈〈〈  .... 1620 LLINKS FILE SPACE USED                          〉〉〉〉〉

** 9:04:55**TSS WILL SIGN OFF AT 22:00



SYSTEM ?
```

## 《Ｆｏｒｔｒａｎ ７７ を使う》

```
SYSTEM ?FRT77                    — Fortran77 を使う。下線はユーザの入力
TCMD024 R OLD OR NEW?NEW         — TCMD024 R の部分は無視。新しくプログラムを作成
*0010 PROGRAM EXAMPLE            — カレント・ファイルにはいる
*0020*    - TEST PROGRAM -
*0030 READ(5,*) A,B
*0040 WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B
*0050 ENS                        — わざと END を ENS に間違っている
*LIST                            — カレント・ファイルの中身全部を表示
0010 PROGRAM EXAMPLE
0020*    - TEST PROGRAM -
0030 READ(5,*) A,B
0040 WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B
0050 ENS

*RUN                             — カレント・ファイルを実行；エラーあり
SOURCE LINE       50 AT EXAMPLE
〈F〉 1303 STATEMENT IS UNRECOGNIZABLE OR THE FIRST DELIMITER IS ILLEG
AL
SOURCE LINE       50 AT EXAMPLE
〈W〉 1250 'END' STATEMENT MISSING-SIMULATED
*0050 END                        — 行50を修正
*LIST                            — 確認
0010 PROGRAM EXAMPLE
0020*    - TEST PROGRAM -
0030 READ(5,*) A,B
0040 WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B
0050 END

*
```

```
*RUN
I*?123.45 456.78                    — READ に対する入力
A+B = 580.2300      A-B = -333.3300
*RUN
I*?12 89
A+B = 101.0000      A-B = -77.00000
*$FORM                              — Fortran のプログラムを見やすく編集

STATISTICS      CPU= 0.002 SECONDS
      INPUT RECORDS=      5         OUTPUT RECORDS=     5
      INITIAL LINES=      4         COMMENT LINES=      1

*LIST                               — 見やすく編集できた
0010       PROGRAM EXAMPLE
0020*    - TEST PROGRAM -
0030       READ(5,*) A,B
0040       WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B
0050       END

*SAVE EXAMPLE                       — 新たにファイルを作って、カレント・ファイルを保存
TCMD536 I DATA WAS SAVED TO FILE EXAMPLE        — これをパーマネント・ファイルという
*CATA                               — パーマネント・ファイルの一覧；catalog
    LIST OF CATALOG A60000 ON 04/20/90 AT 12:17:41
    CATALOGS
      FILES

    SHAPEUP                         — カタログ
    I
      F                             — ファイル
      SPUP
       :
       :
       :
      DCG
    UTIGRAPH
      LIBFL
      SYNTAX
      SYNNL
      SYNFL
      TMP
      JJJ
      EXAMPLE                       — EXAMPLE ができた
*
```

## 《ファイルの一覧》

```
*CATA.S                          — アルファベット順に並べ直して
    LIST OF CATALOG A60000 ON 04/20/90 AT 12:17:50
    CATALOGS
      FILES

      AAA
      B.TEXT
    C
      COMP.L
      DCG
      DETAB.F
       :
       :
       :
      TEST2.L
      TEST.L
      TMP
    UTIGRAPH
      YYY
*CATA.A                          — 1 行に 8 個ずつ
    LIST OF CATALOG A60000 ON 04/20/90 AT 12:17:59

*SHAPEUP *I         F         SPUP      SYNDL     KENZEN   *HORIUCHI B.TEXT
*FORTEST  F.TEXT    HG1.TEXT*SXJCL    *SXFORT    IP.TEXT   HT1.TEXT JNL
*C        *EXCOWORK YYY      *S        *M         *K         EXFILE    MOTO1.L
 MOTO2.L  COMP.L    MOTO.L    DETAB.F   FUZZY     OPS5.L    F4        EXPLOD1
 TEST.L   MAIL.BOX JEK1      TEST2.L   FPROLO.L  AAA       FPS.L     FPROLOG
 STRONG   JEF30601 FP.L      DCG       *UTIGRAPH LIBFL     SYNTAX    SYNNL
 SYNFL    TMP      JJJ       EXAMPLE
*CATA.A.S                          — アルファベット順に並べ直して、1 行に 8 個ずつ
    LIST OF CATALOG A60000 ON 04/20/90 AT 12:18:09

 AAA       B.TEXT   *C         COMP.L    DCG       DETAB.F   EXAMPLE *EXCOWORK
 EXFILE    EXPLOD1  F4         F         F.TEXT    *FORTEST  FP.L      FPROLOG
 FPROLO.L FPS.L     FUZZY     HG1.TEXT*HORIUCHI HT1.TEXT*I          IP.TEXT
 JEF30601 JEK1      JJJ       JNL       *K        KENZEN    LIBFL    *M
 MAIL.BOX MOTO1.L   MOTO2.L   MOTO.L    OPS5.L    *S        *SHAPEUP  SPUP
 STRONG   *SXFORT   *SXJCL    SYNDL     SYNFL     SYNNL     SYNTAX    TEST2.L
 TEST.L   TMP       *UTIGRAPH YYY
*
```

## 《簡単な修正》

```
*LIST
0010       PROGRAM EXAMPLE
0020*    - TEST PROGRAM -
0030       READ(5,*) A,B
0040       WRITE(6,*) 'A+B = ',A+B,        A-B = ',A-B
0050       END

*0030    10 READ(5,*,END=99) A,B          — 行30の書き換え
*0050    99 END                           — 行50の書き換え
*45 GOTO 10                               — 行45の挿入
*20                                       — 行20の削除
*LIST
0010       PROGRAM EXAMPLE
0030    10 READ(5,*,END=99) A,B
0040       WRITE(6,*) 'A+B = ',A+B,  '      A-B = ',A-B
45 GOTO 10
0050    99 END

*RESEQ                              — 行番号を付け直す；resequence
*LIST
0010       PROGRAM EXAMPLE
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B,  '      A-B = ',A-B
0040 GOTO 10
0050    99 END

*RUN
II*?12.3 23.4
A+B = 35.70000      A-B = -11.10000
I*?34.5 45.6
A+B = 80.10000      A-B = -11.10000
I*?↵
*$FORM

STATISTICS      CPU= 0.001 SECONDS
      INPUT RECORDS=      5        OUTPUT RECORDS=     5
      INITIAL LINES=      5        COMMENT LINES=      0

*LIST
0010       PROGRAM EXAMPLE
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B,  '      A-B = ',A-B
0040       GOTO 10
0050    99 END

*RESAVE EXAMPLE              — すでに存在しているファイルにカレント・ファイルを保存
TCMD536 I DATA WAS SAVED TO FILE EXAMPLE
*
```

《接続を切る》

```
*BYE                                  — 接続を切る

**USED RESOURCE.....CPU=1SEC CON=10.1MIN T-ID=RX
**COST:  ¥17


$ N1041 12:22:39 DIS - CP DACPRG
```

メ　モ

《もう一度、システムと接続する》

〈前と同じようにする〉

《Fortran 77 を使う》

```
SYSTEM ?FRT77                  — Fortran77 を使う
TCMD024 R OLD OR NEW?OLD       — 前に作ったファイルを使う
TCMD005 R FILE NAME?EXAMPLE
*LIST
0010       PROGRAM EXAMPLE
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B
0040       GOTO 10
0050    99 END

*
```

《一部分を表示》

```
*LIST 20,40
0020    10 READ(5,*,END=99) A,B
0040       GOTO 10

*LIST 20-40
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B
0040       GOTO 10

*LIST 20-
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B
0040       GOTO 10
0050    99 END

*LIST -30
0010       PROGRAM EXAMPLE
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B, '      A-B = ',A-B

*LISTL                         — 最後の行を表示
0050    99 END

*
```

## 《データ・ファイルの作成》

```
*NEW                          — これから新しいプログラムを作る；今までのものは消える
*LIST

*AUTOX                        — 自動的に行番号を生成；automatic
*0010 1 2                     — スペースから入れるのに注意
*0020 3 4
*0030 5 5
*0040↵                        — 終わりは ↵ のみ
*STRIP                        — 行番号を取り去る
*LIST
 1 2
 3 4
 5 5

*RESEQX                       — 無条件に行番号を付ける；最後の X に注意
*LIST                         — RESEQ では、1、3、5 を行番号とみてしまう
0010 1 2
0020 3 4
0030 5 5

*0030 5 6                     — 行30の修正
*STRIP
*LIST
 1 2
 3 4
 5 6

*RESAVE DATA                  — まちがった！
TCMD242 E FILE DATA IS NOT EXISTENT
*SAVE DATA
TCMD536 I DATA WAS SAVED TO FILE DATA
*
```

## 《ファイルからデータを読む》

```
*OLD EXAMPLE                  — EXAMPLE をカレント・ファイルに持ってくる
*LIST                         — 今までのものは消える
0010       PROGRAM EXAMPLE
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B,'      A-B = ',A-B
0040       GOTO 10
0050    99 END

*LIST DATA                    — パーマネント・ファイルの内容を表示
 1 2
 3 4
 5 6

*
```

```
*LIST                              — カレント・ファイルには影響なし
0010      PROGRAM EXAMPLE
0020   10 READ(5,*,END=99) A,B
0030      WRITE(6,*) 'A+B = ',A+B,'      A-B = ',A-B
0040      GOTO 10
0050   99 END

*RUN :F=DATA(05)                   — READ の 5 に対応する装置を DATA というファイルに
A+B = 3.000000     A-B = -1.000000     — 切り換えて実行；DATA(5)は誤り
A+B = 7.000000     A-B = -1.000000
A+B = 11.00000     A-B = -1.000000
*RUN                               — 2 回目からは、省略できる
A+B = 3.000000     A-B = -1.000000
A+B = 7.000000     A-B = -1.000000
A+B = 11.00000     A-B = -1.000000
*LIST DATA                         — なぜか？
TCMD241 E FILE DATA IS BUSY
*FLIST                        — 使用中のファイルを表示；AFT(available file table)
FILE.NAME (FC) MODE  SIZE(LLINK) DEVICE  CATALOG.STRING OR FILE TYPE
 SY**          RAND          12  DKU653  TEMPORARY
 EXAMPLE       LINK           1  DKU653   /EXAMPLE
 05            LINK           1  DKU653   /DATA
 *SRC     (S*) LINK          12  DKU653  TEMPORARY
 *V.      (V*) RAND         168  DKU653  TEMPORARY
```

〈LIST DATA とすると、/DATA というファイルを DATA という名前で、AFT に登録しようとするが、同じファイルを２つ以上の名前で登録できない〉

```
*REMOVE 05                     — AFT から 05 を取り去る
*FLIST
FILE.NAME (FC) MODE  SIZE(LLINK) DEVICE  CATALOG.STRING OR FILE TYPE
 SY**          RAND          12  DKU653  TEMPORARY
 EXAMPLE       LINK           1  DKU653   /EXAMPLE
 *SRC     (S*) LINK          12  DKU653  TEMPORARY
 *V.      (V*) RAND         168  DKU653  TEMPORARY

*LIST DATA
 1 2
 3 4
 5 6

*
```

## 《ファイルへ実行結果を書き出す》

```
*RUN :F=OUT(06)                          — 6 に対応する装置を OUT というファイルに
I*?12 23                                 — 切り換えて実行；OUT(6)は誤り
I*?56 78
I*?
*LIST OUT
TCMD241 E FILE OUT IS BUSY
*FLIST
FILE.NAME (FC) MODE  SIZE(LLINK) DEVICE  CATALOG.STRING OR FILE TYPE
 SY**          RAND          12  DKU653  TEMPORARY
 EXAMPLE       LINK           1  DKU653   /EXAMPLE
 *SRC     (S*) LINK          12  DKU653  TEMPORARY
 *V.      (V*) RAND         168  DKU653  TEMPORARY
 DATA          LINK           1  DKU653   /DATA
 06            LINK           1  DKU653   /OUT

*LIST OUT"06"                            — AFT 上の名前 06 を同時に指定してもよい
 A+B = 35.00000     A-B = -11.00000
 A+B = 134.0000     A-B = -22.00000

*REMOVE 06                               — AFT から 06 を取り去ってもよい
*LIST OUT
 A+B = 35.00000     A-B = -11.00000
 A+B = 134.0000     A-B = -22.00000

*
```

## 《ファイルからデータを読み、ファイルへ実行結果を書き出す》

```
*RUN :F=DATA(05) F=OUT(06)
*FLIST
FILE.NAME (FC) MODE  SIZE(LLINK) DEVICE  CATALOG.STRING OR FILE TYPE
 SY**          RAND          12  DKU653  TEMPORARY
 EXAMPLE       LINK           1  DKU653   /EXAMPLE
 *SRC     (S*) LINK          12  DKU653  TEMPORARY
 *V.      (V*) RAND         168  DKU653  TEMPORARY
 05            LINK           1  DKU653   /DATA
 06            LINK           1  DKU653   /OUT

*REMOVE CLEARFILES                       — すべてのファイルを AFT から取り除く；いくつかは残る
*FLIST
FILE.NAME (FC) MODE  SIZE(LLINK) DEVICE  CATALOG.STRING OR FILE TYPE
 SY**          RAND          12  DKU653  TEMPORARY
 *SRC     (S*) LINK          12  DKU653  TEMPORARY
 *V.      (V*) RAND         168  DKU653  TEMPORARY

*
```

```
*LIST OUT
 A+B = 3.000000      A-B = -1.000000
 A+B = 7.000000      A-B = -1.000000
 A+B = 11.00000      A-B = -1.000000

*
```

## 《ファイルの管理》

```
*CATA,A
    LIST OF CATALOG A60000 ON 04/21/90 AT 13:36:14

*SHAPEUP *I         F         SPUP      SYNDL     KENZEN   *HORIUCHI B.TEXT
*FORTEST  F.TEXT    HG1.TEXT*SXJCL    *SXFORT    IP.TEXT   HT1.TEXT JNL
*C        *EXCOWORK YYY      *S        *M         *K        EXFILE   MOTO1.L
 MOTO2.L  COMP.L    MOTO.L    DETAB.F   FUZZY     OPS5.L    F4       EXPLOD1
 TEST.L   MAIL.BOX  JEK1      TEST2.L   FPROLO.L  AAA       FPS.L    FPROLOG
 STRONG   JEF30601  FP.L      DCG       *UTIGRAPH LIBFL     SYNTAX   SYNNL
 SYNFL    TMP       JJJ       EXAMPLE   DATA      OUT
*CATA EXAMPLE                    ― ファイルの詳しい情報

    FILE NAME-EXAMPLE
    ORIGINATOR-A60000
    DATE CREATED-042090
    DATE CHANGED-042090(12:21:32)
    LAST DATE ACCESSED-042190
    NUMBER OF ACCESSES-3
    MAX FILE SIZE-20 LLINKS
    CURRENT FILE SIZE-1 LLINK
    FILE TYPE-LINKED
    DEVICE-DK1
    GENERAL PERMISSIONS-NONE
    SPECIFIC PERMISSIONS-NONE
  *CATA DATA

    FILE NAME-DATA
    ORIGINATOR-A60000
    DATE CREATED-042190
    DATE CHANGED-042190(13:32:07)
    LAST DATE ACCESSED-042190
    NUMBER OF ACCESSES-4
    MAX FILE SIZE-20 LLINKS
    CURRENT FILE SIZE-1 LLINK
    FILE TYPE-LINKED
    DEVICE-DKC
    GENERAL PERMISSIONS-NONE
    SPECIFIC PERMISSIONS-NONE
*
```

```
*CPY EXAMPLE;EXAM1                    — ファイルのコピー；copy
TCMD540 I COPIED 1 LLINKS
*CPY OUT;OUT1
TCMD540 I COPIED 1 LLINKS
*CATA.A
    LIST OF CATALOG A60000 ON 04/21/90 AT 13:37:27

*SHAPEUP *I         F         SPUP      SYNDL     KENZEN   *HORIUCHI B.TEXT
*FORTEST  F.TEXT    HG1.TEXT*SXJCL    *SXFORT    IP.TEXT   HT1.TEXT JNL
*C        *EXCOWORK YYY      *S        *M         *K        EXFILE   MOTO1.L
 MOTO2.L  COMP.L    MOTO.L    DETAB.F   FUZZY     OPS5.L    F4       EXPLOD1
 TEST.L   MAIL.BOX  JEK1      TEST2.L   FPROLO.L  AAA       FPS.L    FPROLOG
 STRONG   JEF30601  FP.L      DCG       *UTIGRAPH LIBFL     SYNTAX   SYNNL
 SYNFL    TMP       JJJ       EXAMPLE   DATA      OUT       EXAM1    OUT1
*LIST EXAM1;OUT1                      — 2つのファイルを一度に表示
0010       PROGRAM EXAMPLE
0020    10 READ(5,*,END=99) A,B
0030       WRITE(6,*) 'A+B = ',A+B,'      A-B = ',A-B
0040       GOTO 10
0050    99 END

 A+B = 3.000000      A-B = -1.000000
 A+B = 7.000000      A-B = -1.000000
 A+B = 11.00000      A-B = -1.000000

*RELE OUT1                            — ファイル OUT1 を消去；release
ACCE501 I FILE OUT1 WAS RELEASED
*RELE EXAM1;OUT                       — 2つのファイルを消去
ACCE501 I FILE EXAM1 WAS RELEASED
ACCE501 I FILE OUT WAS RELEASED
*CATA.A
    LIST OF CATALOG A60000 ON 04/21/90 AT 13:38:22

*SHAPEUP *I         F         SPUP      SYNDL     KENZEN   *HORIUCHI B.TEXT
*FORTEST  F.TEXT    HG1.TEXT*SXJCL    *SXFORT    IP.TEXT   HT1.TEXT JNL
*C        *EXCOWORK YYY      *S        *M         *K        EXFILE   MOTO1.L
 MOTO2.L  COMP.L    MOTO.L    DETAB.F   FUZZY     OPS5.L    F4       EXPLOD1
 TEST.L   MAIL.BOX  JEK1      TEST2.L   FPROLO.L  AAA       FPS.L    FPROLOG
 STRONG   JEF30601  FP.L      DCG       *UTIGRAPH LIBFL     SYNTAX   SYNNL
 SYNFL    TMP       JJJ       EXAMPLE   DATA
*REMOVE CLEARFILES
*
```

## 《画面を使った修正のしかた》

```
*NEW
*AUTOX
*0010 PROGRAM FACT
*0020 INTEGER N,F
*0030 1 READ(5,*) N
*0040 IF (N.LT.0) STOP
*0050 WRITE(6,*) F(N)
*0060 GOTO 1
*0070 END
*0080*
*0090 INTEGER FUNCTION F(N)
*0100 INTEGER N,J
*0110 F=1
*0120 DO 10 J=1,N
*0130 F=J*F
*0140 10 CONTINUE
*0150 END
*0160↵
*RUN
I*?5
120
I*?10
3628800
I*?13
6227020800
I*?-1
*$FORM

STATISTICS      CPU= 0.002 SECONDS
      INPUT RECORDS=    15       OUTPUT RECORDS=   15
      INITIAL LINES=    14       COMMENT LINES=     1

*
```

〈いままでは、プログラムを修正するのに、いちいち行番号と新しい行の内容全体を入力していた。しかし、画面に行の情報が残っている場合には(例えば、すぐ上のように LIST で表示してある場合には)、カーソルをその上に移動させて、修正を行ない、↵ キィーや実行キィーを押すことにより、その行を修正できる。ただし、詳細については、使用する端末によって異なる。
したがって、必要な行をいかに画面上に表示させるかということが重要になる〉

《エディタの一部を利用》

```
*LIST
0010       PROGRAM FACT
0020       INTEGER N,F
0030     1 READ(5,*) N
0040       IF (N.LT.0) STOP
0050       WRITE(6,*) F(N)
0060       GOTO 1
0070       END
0080*
0090       INTEGER FUNCTION F(N)
0100       INTEGER N,J
0110       F=1
0120       DO 10 J=1,N
0130         F=J*F
0140    10 CONTINUE
0150       END

*-P:/INTEGER/                  - INTEGER と一致する行を表示 (P:print)
                               - 行の先頭のみ探す
END OF FILE - REQUEST EXECUTED      0 TIMES
*-PS:/INTEGER/                 - INTEGER と一致する行を表示
0020       INTEGER N,F         - 行の全体を探す (S:string)

*-PS:/INTEGER/;2               - 2 回実行
0020       INTEGER N,F
0090       INTEGER FUNCTION F(N)

*-PS:/INTEGER/;*               - ファイル全体に対して実行
0020       INTEGER N,F
0090       INTEGER FUNCTION F(N)
0100       INTEGER N,J

END OF FILE - REQUEST EXECUTED      3 TIMES
```

〈このようにして必要な行を表示してから、カーソルを動かしプログラムを修正する〉

```
*-RVS:/INTEGER/:/REAL/         - INTEGER を REAL に置き換える (R:replace)
0020       REAL N,F            - 置き換えた結果を確認する (V:verify)
                               - 行の全体を探す (S:string)
*LIST
0010       PROGRAM FACT
0020       REAL N,F
0030     1 READ(5,*) N
0040       IF (N.LT.0) STOP
0050       WRITE(6,*) F(N)
0060       GOTO 1
0070       END
0080*
```

```
0090       INTEGER FUNCTION F(N)
0100       INTEGER N,J
0110       F=1
0120       DO 10 J=1,N
0130         F=J*F
0140    10 CONTINUE
0150       END

*-RVS:/INTEGER/:/REAL/
0090       REAL FUNCTION F(N)

*-RVS:/INTEGER/:/REAL/
0100       REAL N,J

*LIST
0010       PROGRAM FACT
0020       REAL N,F
0030     1 READ(5,*) N
0040       IF (N.LT.0) STOP
0050       WRITE(6,*) F(N)
0060       GOTO 1
0070       END
0080*
0090       REAL FUNCTION F(N)
0100       REAL N,J
0110       F=1
0120       DO 10 J=1,N
0130         F=J*F
0140    10 CONTINUE
0150       END

*-RVS:/REAL/;*:/INTEGER/        ― ファイル全体の INTEGER を REAL に置き換える
0020       INTEGER N,F

0090       INTEGER FUNCTION F(N)

0100       INTEGER N,J

END OF FILE - REQUEST EXECUTED       3 TIMES
*LIST
0010       PROGRAM FACT
0020       INTEGER N,F
0030     1 READ(5,*) N
0040       IF (N.LT.0) STOP
0050       WRITE(6,*) F(N)
0060       GOTO 1
0070       END
0080*
```

```
0090      INTEGER FUNCTION F(N)
0100      INTEGER N,J
0110      F=1
0120      DO 10 J=1,N
0130        F=J*F
0140   10 CONTINUE
0150      END

*-PS:/INTEGER/+/F/;*          — INTEGER と F の両方を含む行をすべて表示
0020      INTEGER N,F         —「かつ(+)」による接続は、5つまで
0090      INTEGER FUNCTION F(N)

END OF FILE - REQUEST EXECUTED      2 TIMES
*-PS:/INTEGER/-/F/;*          — INTEGER か F の少なくとも一方を含む行をすべて表示
0010      PROGRAM FACT        —「または(-)」による接続は、5つまで
0020      INTEGER N,F
0040      IF (N.LT.0) STOP
0050      WRITE(6,*) F(N)
0090      INTEGER FUNCTION F(N)
0100      INTEGER N,J
0110      F=1
0130        F=J*F

END OF FILE - REQUEST EXECUTED      8 TIMES
*-PS:/PROGRAM/,/END/          — PROGRAM を含む行から、END を含む行までを表示
0010      PROGRAM FACT
0020      INTEGER N,F
0030    1 READ(5,*) N
0040      IF (N.LT.0) STOP
0050      WRITE(6,*) F(N)
0060      GOTO 1
0070      END

*-PS:/FUNC/,/END/             — FUNC を含む行から、END を含む行までを表示
0090      INTEGER FUNCTION F(N)
0100      INTEGER N,J
0110      F=1
0120      DO 10 J=1,N
0130        F=J*F
0140   10 CONTINUE
0150      END

*SAVE FACT.FOR
TCMD536 I DATA WAS SAVED TO FILE FACT.FOR
*DONE
SYSTEM ?
```

## 《会話型リモートバッチ》

```
SYSTEM ?CARDIN                  — 会話型リモートバッチ用サブシステム
TCMD024 R OLD,NEW OR SAME?NEW
*AUTOX 10                       — 以下、会話型リモートバッチ用 JCL の作成
*10##NORM                       — 以下の : をタブとし、行番号を行の最後に移動
*20$:JOB:;A,E                   — A は支払いコード； E はジョブ・クラス
*30$:FRT77:LSTIN,NFORM,LNO      — LISTIN:コンパイル•リストあり；NFORMとLNO:ﾌｧｲﾙの形式
*40$:PRMFL:S*,R,S,A60000/EXAMPLE — S*：FRT77 のソース•ファイル、R：read、S：sequential
*50$:GO                         — 実行；linker と run に展開
*60$:PRMFL:05,R,S,A60000/DATA   — 05：標準入力用装置番号に対応するファイル
*70$:ENDJOB
*80↵
*REMOVE CLEARFILES              — 実行前に、AFT をクリア；EXAMPLE と DATA が AFT にあると、
*RUN                              — JCL を実行                          そこで処理が停止
CRJE350 I   SNUMB # K368T       — ジョブが生成される
*ISTS *                         — ジョブの状況を見る(job status)；* は最も新しいジョブ
CRJE405 I K368T(001) WAIT PERIP — 番号 001 はジョブの進行段階を表わす；activity
*ISTS *
CRJE409 I K368T(003) EXECUTING  — 変化した
*ISTS *
CRJE447 I K368T IN DEMAND FILE IS OUTPUT WAITING
CRJE600 I NORMAL TERMINATION    — ジョブの終了；ID カードによりプリンタに出力可能
*RUN                                — 新しくジョブを生成
CRJE350 I   SNUMB # K369T
*JMON *                             — ジョブの状況を監視（job monitor）
CRJE409 I K369T(002) EXECUTING  @ 13:53:30
CRJE447 I K369T IN DEMAND FILE IS OUTPUT WAITING @ 13:53:40  — 約 10 秒ごと
CRJE600 I NORMAL TERMINATION        — ジョブの終了
```

〈ジョブの実行が長時間に渡るときは、break して、別の仕事ができる。EXAMPLE と DATA は AFT に登録しないこと。ジョブの終了後、ID カードによりプリンタに出力できる〉

```
*JOUT *                         — 端末で実行結果の確認（job output）
CRJE012 R FUNCTION?LIST ALL     — ジョブの全実行状況の表示
ACTI#         RC    LINE   HOLD  CLASS
  --          $$     --     --    I              — JCL と実行レポート
 001(FRT7V )  74      15   YES   I              — FRT77 による翻訳
 002(LINKER)  74      23   YES   I              — LINKER による結合
 003(****RU)  06       4   YES   I              — 実行
CRJE012 R FUNCTION?ACTIVITY 3 — 今後は、ACTIVITY の 3（実行)に注目
CRJE012 R FUNCTION?LIST       — ACTIVITY 3 の実行状況の表示
activity 003(****RU)
report codes                  — 上の RC に対応する
 $$
 06                           — 実行時の標準出力用装置番号；ここに実行結果が入っている
CRJE012 R FUNCTION?PRINT 6
CRJE361 E REPORT 6  IS NOT FOUND
CRJE012 R FUNCTION?
```

```
CRJE012 R FUNCTION?PRINT 06    — 06 を印刷；スペース文字とスペース行の圧縮

A+B = 3.000000 A-B = -1.000000
A+B = 7.000000 A-B = -1.000000
A+B = 11.00000 A-B = -1.000000

CRJE317 I END OF REPORT 06
CRJE012 R FUNCTION?EPRINT 06     — 06 を印刷；ほぼそのまま

A+B = 3.000000      A-B = -1.000000
A+B = 7.000000      A-B = -1.000000
A+B = 11.00000      A-B = -1.000000

CRJE317 I END OF REPORT 06
CRJE012 R FUNCTION?COPY 06;OUT  — 06 をファイル OUT にコピー
TCMD540 I COPIED 1 LLINKS
CRJE012 R FUNCTION?HOLD         — 実行結果をシステムに保持；この後、ID カードにより
*JOUT K368T                     —                                    プリンタに出力可能
CRJE012 R FUNCTION?RELE         — 実行結果をシステムから削除
*LIST
10##N
20$:JOB:;A,E
30$:FRT77:LSTIN,NFORM,LNO
40$:PRMFL:S*,R,S,A60000/EXAMPLE
50$:GO
60$:PRMFL:05,R,S,A60000/DATA
70$:ENDJOB

*SAVE JCL
TCMD536 I DATA WAS SAVED TO FILE JCL
*CATA,A
    LIST OF CATALOG A60000 ON 04/21/90 AT 14:05:40

*SHAPEUP *I         F         SPUP      SYNDL     KENZEN   *HORIUCHI B.TEXT
*FORTEST  F.TEXT    HG1.TEXT*SXJCL    *SXFORT   IP.TEXT   HT1.TEXT JNL
*C        *EXCOWORK YYY      *S        *M        *K         EXFILE   MOTO1.L
 MOTO2.L  COMP.L    MOTO.L    DETAB.F   FUZZY     OPS5.L   F4       EXPLOD1
 TEST.L   MAIL.BOX  JEK1      TEST2.L   FPROLO.L  AAA      FPS.L    FPROLOG
 STRONG   JEF30601  FP.L      DCG       *UTIGRAPH LIBFL    SYNTAX   SYNNL
 SYNFL    TMP       JJJ       EXAMPLE   DATA      FACT.FOR OUT      JCL
*LIST OUT                        — ファイル OUT を確認
A+B = 3.000000      A-B = -1.000000
A+B = 7.000000      A-B = -1.000000
A+B = 11.00000      A-B = -1.000000

*BYE

**USED RESOURCE.....CPU=2SEC CON=40.2MIN T-ID=RX
**COST:  ¥53


$ N1041 14:06:05 DIS - CP DACPRG
```