ACOSのファイル(その3)

研究開発部　多　喜　正　城

Ⅲ　ファイルの利用法

1)　システムファイルへの登録
2)　ソース・プログラム関係
3)　ライブラリィの利用
4)　データ・ファイル/ワーク・ファイルの利用

プログラムやデータを，その処理毎にセンターに運んだり，端末機を操作したりするのは，煩わしいものであり，また時間もかかる。そこで，センターにファイル利用申請をし，予め自分のファイルを確保して，プログラムやデータを好きな形（ソース形式やオブジェクト形式等）にして，自分のファイルに登録しておけば，簡単な操作で使用でき，計算結果を自分のファイルに出力しておけば，いつでも必要な時に見れるし，またそれを，次の処理の為の入力データとして使え，センターのプリンターに出力することも可能である。また，共同研究の場合にも，その共同利用もでき，交換回線端末機を使えば，全国どこからでも，そのファイルにアクセスすることができる。そこで，この章では実際に，プログラムやデータを自分のファイルシステムへ入れて置き，また，取り出したりする為の方法を説明します。*)

初めに，この章の内容を挙げておきます。

1)　ファイル・システムへの登録
　1－1　ファイルシステムへの登録

2)　ソース・プログラム関係
　2－1　ソース・プログラムの登録
　2－2　ファイル内のソース・プログラムの実行
　2－3　ファイル内のソース・プログラムの修正

＊）登録する事を中心に述べますが，ファイルを消す事も大切です。その方法はセンターニュースNo.24「ACOSのファイル」を参照

2－4　ファイル内のソース・プログラムのリストアップ

2－5　ソース・プログラム（カード形式）をオブジェクト形式（C＊）にして登録

2－6　ファイル内のソース・プログラムをオブジェクト形式（C＊）にして登録

2－7　オブジェクト形式（C＊）の実行

2－8　ソース・プログラムを実行形式（H＊）にして登録

2－9　実行形式（H＊）の実行

3）　ライブラリィの利用

3－1　サブルーチン・プログラムの登録

3－2　サブルーチン・プログラムをオブジェクト・シーケンシャル・ライブラリィにして登録

3－3　サブルーチン・プログラムをオブジェクト・ランダム・ライブラリィにして登録

3－4　オブジェクト・シーケンシャル・ライブラリィをオブジェクト・ランダム・ライブラリィにして登録

3－5　ファイルに登録してあるサブルーチン（ソース形式）を用いて実行

3－6　ファイル内のメインプログラム（ソース形式）とサブルーチン（ソース形式）を用いて実行

3－7　ファイル内のサブルーチン（オブジェクト・ランダム・ライブラリィ）を用いて実行

3－8　ファイル内のメインプログラム（ソース形式）とランダム・ライブラリィを使用実行

3－9　ファイル内のオブジェクト形式のメインプログラムとシーケンシャル・ライブラリィを使用実行

3－10　MATH－LIB（メーカ提供）又はセンターライブラリィを使用実行

4）　データファイル／ワークファイルの利用

4－1　データの登録

4－2　ソースプログラム内のREAD/WRITE文にパーマネントファイルを利用

4－3　テンポラリィファイルをワークファイル（シーケンシャル・アクセス・ファイル）として利用

4－4　テンポラリィファイルをランダム・アクセス・ワーク・ファイルとして利用

4－5　ファイル中の計算結果のリストアップ

以上の項目について述べてあります。では，そろそろ本題に入ります。

1）　ファイル・システムへの登録

1－1　ファイル・システムへの登録

ここでは簡単の為，クイック・アクセル・ファイル（UMC：ユーザーズ・マスターカタログのすぐ下のレベルのファイル）として登録する場合を述べます。

バッチ系システムを用いる場合FILSYSディレクティブ[*1)]を利用，TSSではACCESSサブシステム[*2)]又はSAVEコマンドを用いる。

◎　FILSYSによるファイル記述[*2)]

{FC / CF}␣課題番号/ファイル名，〔パスワード/パスワード名/〕〔，パーミッション〕〔，ファイルのサイズ/初期値〔，最大値〕/〕〔，アクセスモード〕

ⅰ）ファイル名；　12文字以内（TSSでは8文字以内）

ⅱ）パスワード名；　12文字以内

ⅲ）パーミッション；*2)

ⅳ）ファイルのサイズの指定方法 {LLINKS / LINKS} /初期値 {UNLIMITED / ，最大値}/

サイズの指定を省略すれば，初期値と最大値が同じとして 1 LINKがとられる。最大値としてUNLIMITEDを指定すればACOS－ファイル利用申請書の申請量の使用可能な範囲まで使用できる。1 LINK=12 LLINK=3840語，1 LLINK=320語

Ⅴ）アクセスモード；SEQ（順アクセス）とRAND（直接アクセス），省略すればSEQ

使用例

カラムNo.	1	8
①	$	FILSYS
②	CF␣6000AB0001/FILENAME, LLINKS/10, 30/	
③	CF␣6000AB0001/FILENUMB, PASSWARD/ABC/, R, LINKS/10/, MODE/RAND/	

＊1）　ファイル・マネージメント・スーパーバイザ説明書を参照

＊2）　センターニュースNo.24　ACOSのファイルを参照

①； FILSYSディレクティブを呼び実行する。

②； CFの後は1個以上のブランクを置く，課題番号は6000AB0001，すなわちUMCがそれで，そのすぐ下にファイル名FILENAME なるファイル記述値を作り，パスワードなし，パーミッションなし．サイズは初期値が10 LLINKSで最大直が 30 LLINKS アクセスモードはSEQ（順アクセス）

③； UMCのすぐ下にファイル名FILENUMBを作り，パスワードがABC，パーミッションはREAD，サイズは初期値と最大値が同じ10 LINKS，アクセスモードはRAND（直接アクセス）

なお，作ったファイル／カタログのリストアップをする場合は次の通りとするとよい。

カラムNo. 1 8

$ FILSYS

LIST␣課題番号

注意） 以下の説明には，課題番号を"□"に，パスワードを"pass"に，ファイル名を"fn"と略します。そして，説明の前に，ローカルバッチ（L.B.S.）は，ジョブ制御言語（J.C.L.）を，TSSの場合はコマンドを，会話型リモートバッチ（C.R.B.S.）の場合はコマンドとJ.C.L.を，はじめに書き並べます。

また，各セクションでファイルを登録する場合は，あらかじめ，FILSYS 又はACCESSでファイル記述を作っておく事です。また，説明では，それが成されている事としておきます。

（特に，バッチ系を用いて作る場合は，必ず作っておいて下さい／）

2） ソース・プログラム関係

2－1 ソース・プログラムの登録

1. L.B.S.

	カラムNo. 1	8	16
①	$	SNUMB	nnnnn
②	$	JOB	□ $pass
③	$	PROGRAM	SCED
④	$	PRMFL	OT,W,S,□/fn
⑤	$	DATA	IN,,COPY
	ソース・プログラム・カード		
⑥	$	ENDCOPY	
⑦	$	ENDJOB	

説　明

①；　JOB受付の為の番号カード

②；　JOBの課題チェック等の制御文（マクロ化されている）
（以後，この$ SNUMB，$ JOBは省いて書かれている）

③；　＊Cエディタというプログラムを使って登録する為，この通りの制御文を使う。

④；　パーマネントファイルへの出力の為の制御文，OTはファイルコードこの通り固定，Wは書き込みを示す，SはSEQ アクセス（順アクセス）

[　　]/fnはUMCの下にfn なるファイルを作ることを示す。この時，あらかじめFILSYS又はACCESSでfnなるファイル記述を作っておく必要がある。

⑤；　＊Cエディタを動かす為の入力ファイルとなる，INはファイルコードこの通り，，COPYもこの通り

⑥；　COPYの終りを示す。

⑦；　JOBの終りを示す。

＊Cエディタを用いて登録されたプログラムはBCDコードで登録され,ライン番号なし，ソース・プログラムがFORM形式であればFORM形式で登録されます。

ファイルへの登録の方法は，他にFILEDIT を使う方法や,CONVERを使う方法がありますが，ここでは初心者向に説明する為＊Cエディタを使う方法を説明しています。以下，各セクションでも＊Cエディタを使います。（FILEDITについてはFORTRANプロミラミング 説明書を参照）

2. T.S.S.

1. 端末入力

① SYSTEM? FORT N
READY

② *AUTO
010 DIMENSION A(10,10)
020 ≀
ソース・プログラム
≀
230 STOP;END

③　240 Cr

④　*SAVE FN

DATA SAVED-FN

端末より入力する場合（以下の説明では端末入力とする）AUTO又はAUTOX コマンドを使う（AUTOはライン番号のすぐ後に1個のブランクがあり，AUTOXはない。FORTRANに限らずソース・プログラムをNFORM形式で入力する場合AUTOの方を，また，制御文を入力する場合はAUTOXを使う方がよい。）

説　明　　下線の部分が入力文字

①；　TSSで使うサブシステムを示す。ここではFORTRANサブシステムを使い新しく作るのでNEW のNを指定（システムの撰択は始めの4文字でよい）

②；　AUTOコマンド入力

③；　Cr はキャリッジリターンを示す。

④；　このままでは，カレントファイルに入ったままなのでSAVEコマンドを使い，クイック・アクセス・ファイルとしてパーマネントファイルに登録。上例では，ファイル名がFN（ファイル名は8文字まで）この場合あらかじめACCESSサブシステムによりファイル記述FNを作らなくてもよい。ただし，ACCESSサブシステムでファイル記述を作っておけば，その作った所に登録される。

2.　センターにカードを持って来て，T.S.S.用にファイルに登録する場合

```
カラムNo.  1        8          16
           $        SNUMB      nnnnn
           $        JOB        [          ] $pass
①          $        PROGRAM    TSCONV
②          $        PRMFL      FT,W,S,    [          ]/fn
③          INPUT,INSERT,1,72,10,10
               ソース・プログラム・カード
           $        ENDJOB
```

説　明

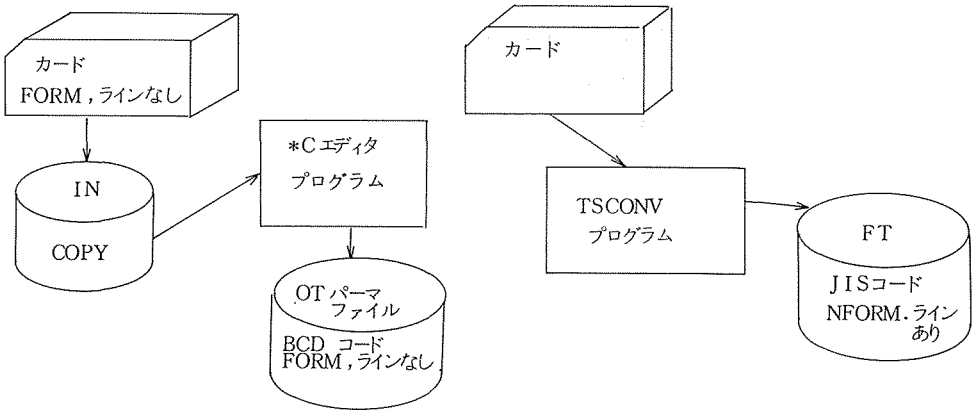
①；　TSCONVプログラムを用いて入力する

②；　L.B.S.の時の説明④と同じ（但し，ファイルコードはFT）

③； TSCONVを使った時ファイルに対する指示する。1カラムから72カラムまでがステートメントでライン番号は10より10おきにつける。

TSCONVにより作られたファイルはTSS標準ファイルとなる。すなわちJIS コードでNFORM 形式, ライン番号付である。（ライン番号を付けなければ，③のINSERT以下をASISにすればよい）

簡単に＊Cエディタと TSCONVの図を示しておく。



バッチ標準ファイル

- BCD コード
- FORM 形式（固定長形式）
- ライン番号なし

TSS標準ファイル

- JIS コード
- NFORM 形式（不定長形式）
- ライン番号付

注意）

1. ＊Cエディタで登録されたソース・プログラムをTSS 端末で見る場合BCDJISコマンドを用いてBCDコードをTSS−JISコードに変換する必要がある。

例）

```
＊BCDJIS BCDFIL①, JISFIL②
LINE NUMBERS? AUTO③
TAB CHARACTER AND SETTINGS? ④(Cr)
＊LIST JISFIL⑤
```

①； BCDコードのファイル名（BCDFILという名）

②； BCDFILという名のファイルをJISコードのJISFILという名にする。

③； ライン番号を付けるか？の質問に対し，10番より10おきに付ける。

④； タブ指定をするか？ なし

⑤； コード変換により出来たファイルをリストアップする。

2. TSS (TSCONVも含む)により作られたファイルをセンター出力する場合。BPRINTコマンドを用いる。

例)

```
        *BPRINT
①       INPUT FILIES? fn
②       $ IDENT? [          ], 識別名
③       LABELS? N
④       SNUMB# 4649T
```

①； センター出力すべきファイル名

②； 課題番号と適当な識別名（９文字以内）

③； NORM (N) を指定することによりMOVE および標準タブセットがなされる。

④； 受け付け番号が4649Tである。

注意1,2,に対してはタイムシェアリング説明書又はタイムシェアリング会話型リモートバッチ説明書を参照

2－2 ファイル内のソース・プログラムの実行

1. L.B.S.

```
カラムNo.  1        8          16
①          $        FORTRAN
②          $        PRMFL      S*,R,S, [          ] /fn
③          $        GO
              データ
           $        ENDJOB
```

説 明

①； FORTRANコンパイラを動かす為の制御文

②； パーマネント・ファイル内のソース・プログラムをFORTRANコンパイラへの入力ファイルとするため，ファイルコードはS*，読み出し指定のR，ファイルの編成はシーケンシャルのS，ファイル名はUMCのすぐ下のfn。

③；　ローダーへの指示と実行の為の制御文（マクロ化されている）

注意）　TSS で作ったファイルを使う場合は$ FORTRANのパラメータにNFORMを指定する。（TSSで作ったファイルはNFORMライン付である）

カラムNo.	1	8	16
	$	FORTRAN	NFORM

2. T.S.S.

```
*RUN␣fn
=データ
```

説　明

- RUNコマンドのパラメータにファイル名を指定するとよい。
- 他人のファイルで読み出し可能なパーミッションがある場合，そのファイルを自分のカレントファイルに引っぱってきて実行させ，又，自分のファイルに入れる方法を述べる。

①
```
SYSTEM?  FORT␣O␣ [          ] /fn,R
READY
```

②
```
*RUN
=データ
^
```

③
```
*SAVE␣fn1
```

①；　SYSTEM選択レベルでFORTRAN サブシステムを使用する。すでに作られているファイルであるからOLD のOを指定，引っぱってくるべきUMC と，そのファイル名fn ，そしてそのパーミッションRを指定する。ここでパーミッションの指定を省略するとW (WRITEパーミッション) となる為，そのファイルがRであればNOT GRANTEDとなり引っぱってこれない。

②；　①によってすでにカレントファイルにプログラムが入っている。

③；　SAVEコマンドで自分のファイルにプログラムfnを入れるがここで元の引っぱってきたファイル名と同じ名前を指定すると，RESAVEせよとシステムの方から帰ってくる。これは，元の他人のファイルへ入れる為である。従ってファイル名を変えてfn1とすれば自分のファイルに入れることが出来る。

◎ アドバイス——自分で作ったファイルを他人に使わせる場合，そのパーミッションは“W”はさけて“R”の方が安全である。なぜなら，自分はそのファイルを他人に使わせた場合，他人がうっかり修正した時，知らない間にファイルが変っていることがある。また他人が使っている時，それが“W”で引っぱった時，自分が同時にそのファイルを使おうと思っても，FILE BUSYのメッセージが帰って来て使えなくなる。今，自分のファイルを使おうと思ってそのファイルを指定したとき，同じくFILE BUSYが帰って来た場合，だれかがそのファイルを使っている為である。そこでOLD␣fn,Rとして OLDコマンドを使えばよい。（ただし,そのファイルのパーミッションは“R”とする。）

○ バッチで作ったプログラムをTSS端末でRUNさせる場合，JISコードに変換させ，ライン番号を付ける必要がある。

例

① *BCDJIS BCDFIL,JISFIL

② LINE NUMBERS? AUTO

③ TAB CHARACTER AND SETTING? (Cr)

④ *RUN JISFIL

=データ

①；②；③はソース・プログラムの登録の項のBCDJISコマンドの使い方と同じ

④； RUNコマンドで変換されたJISコードファイル名を指定する。

3. C.R.B.S. ① SYSTEM? CARD N

READY

② *AUTOX

③ 0010$:JOB:6000AB0001$PASS

④ 0020$:FORTRAN

⑤ 0030$:PRMFL:S*,R,S,6000AB0001/fn

⑥ 0040$:GO

⑦ 0050#123.45

〳

データ

〳

⑧　0090$:ENDJOB

⑨　0100 (Cr)

⑩　*RUN

⑪　SNUMB# 1234T

⑫　CARD FORMAT,DISPOSITION?

⑬　N,J

説　明　これはバッチで作成されファイルに入っているプログラムfnを使う。

①；　SYSTEM選択レベルでCARDINサブシステムを指定，Nは新しく制御文列を作る為のコマンド。

②；　AUTOXコマンドで，ライン番号のすぐ後にステートメントを入れる。以下の制御文はL.B.S.と同じだが“：”によりタブ指定をする。

⑦；　ライン番号とデータの区切りをハッキリさせる為“＃”を使う。

⑩；　RUNコマンドによりバッチ系にプログラムを渡す。

⑪；　バッチ系システムでの受付番号，5ケタ目にTがつく。

⑫；　カードの形式と計算結果の処置はどうするか。

⑬；　Nは標準タブ文字（：）の指定によりタブセットを8，16，32 73カラムに指定編集する。JはJOUTサブシステムを使用し,調べるため各アクティビティの情報をSYSOUTファイルにセーブしておく。

◎　上の例はバッチ系により作成されたプログラムを使用したがTSSで作成されたプログラムを使用する場合，次の$ SELECTA文を使い，$ FORTRAN文にパラメータ，NFORMを用いる。

```
0010$:JOB:6000AB0001$PASS
0020$:FORTRAN:NFORM
0030$:SELEDTA:fn
0040$:GO
0050#1234.5
    ↓ データ
0080$:ENDJOB
0090 (Cr)
*
```

注意)　データ入力のときライン番号の後“＃”を入れなければ，例えば0050␣123.5，430.21とし，データを“123.5，430.21”のつもりで入れたが，システムの方では次のように見る。

ライン番号“0050123”

データは“.5，430.21”となる。

2－3　ファイル内のソース・プログラムの修正

1. L.B.S.　　1　　8　　16

```
①    $    PROGRAM    SCED
②    $    PRMFL      IN,R,S, [      ] /fn1
③    $    PRMFL      OT,W,S, [      ] /fn2
④    $    DATA       A*,,COPY
⑤    $    CHANGE     2,2
⑥         READ (1,5)  A,B
       5  FORMAT (V)
⑦    $    CHANGE     10,15
⑧         WRITE (6,7) C
       7  FORMAT (E20.9)
     $    ENDCOPY
     $    ENDJOB
```

説　明

①；　＊Cエディタを用いる。

②；　修正すべきファイルを＊Cエディタの入力とする。ファイルコードは“IN”。　ファイル名は“fn1”

③；　修正されたファイルを登録する為の＊Cエディタの出力ファイルとなるファイルコード“OT”　ファイル名は“fn2”（fn1と違う名を付ける）

④；　修正する為の中間ファイル，ファイルコードは“A＊”を指定

⑤；　この制御文により元の2枚目のカードは消去される。

⑥；　元の2枚目と3枚目の間に入るカード

⑦；　この制御文により元の10枚目から15枚目まで消去される。

⑧；　元の10枚目と11枚目の間に入るカード

- 修正したいステートメントの前に $ CHANGE文を入れる。
- $ PRMFL で入力ファイルと出力ファイルの名前を別にする必要が

ある。

- ステートメントの削除（m行目からn行目まで）

```
1       8           16
$       CHANGE      m,n
```

- ステートメントの追加（m行目から）

```
1       8           16
$       CHANGE      m
｛ステートメント
```

2. T.S.S. ①

```
*OLD␣fn
READY
```

②

```
*LIST
010   ソース・プログラムの内容
      ≀
200 STOP;END
```

③

```
*030
```

④

```
*050 10 FORMAT (V)
      ≀
```

説　明

①；　パーマネントファイルにあるプログラムをOLD コマンドを用いて，カレントファイルに呼び出す。

②；　カレントファイルの内容をリストアップする為LISTコマンドを用いる。

③；　ソース・プログラムの30ライン目を消去

④；　ソース・プログラムの50ライン目を10␣FORMAT (V) と入れかえる。

- すなわち，カレントファイルに呼び出し，ライン番号を指定し，ステートメントを打てば，そのライン番号の処には新しいステートメントが入り，元のステートメントは消える。ステートメントとステートメントの間に新しいステートメントを追加したい場合は，そのライン番号と次のライン番号の間のライン番号を入れ，新しいステートメントを打てばよい。例えば次の様に

```
      ≀
040␣1␣CONTINUE
050␣DO␣2␣I＝1，N
060␣A（I）＝AA＋B（I）＊I
      ≀
```

このプログラムの040と050の間にAA=0 を入れたいとき

＊045␣AA＝0　とすればよい。

○　その他EDITORサブシステムを用いると便利な方法が多くある。

（TSSテキストエディタ／ランオフ説明書　参照）

注意）　修正したプログラムはカレントファイル内にあるのでRESAVEコマンド又はSAVEコマンドでパーマネントファイルに入れることを忘れなく！

2－4　ファイル内のソース・プログラムのリストアップ

1. L.B.S.

```
      1       8          16
$     PROGRAM  SCED
$     PRMFL    IN,R,S,  [          ] /fn
$     FILE     OT,NULL
$     ENDJOB
```

＊Cエディタを用い＄ FILE 文でファイルOT，第2パラメータNULLを指定する。

2. T.S.S.

＊LIST␣fn

2－5　ソース・プログラム（カード）をオブジェクト形式ファイル（C＊）にして登録

1. L.B.S.

```
      1       8          16
①     $       FORTRAN   DECK
②     $       PRMFL     C＊,W,S,  [          ] /fn
          ソース・プログラム
③   ( $       GO                  )
    (   デ　ー　タ                 )
      $       ENDJOB
```

説　明

①；　FORTRAN コンパイラを動しバイナリデックファイルを作る為のパラメータDECKを指定する。

②；　バイナリデックファイルのファイルコードは“Ｃ＊”，書込みだから“Ｗ”，以下前述と同じ

③；　実行までする場合

○　オブジェクト形式ファイル（Ｃ＊）はコンパイルされ，ローダーへの入力となるファイルであり，順編成ファイルである。

2. Ｔ.Ｓ.Ｓ.

```
SYSTEM? FORT N
READY
①  *AUTO
②  010 @C :::TEST PROGRAM::::
   020 READ (5,10) A
   030 10 FORMAT (U)
   040
                ≀
③  *RUN=;CSTR1 (NOGO)
```

説　明

①；　AUTOコマンドによりソース・プログラムの入力

②；　ライン番号のすぐ後にブランクが入っているので@でそのブランクを消す。そしてコメントのCを入れる。

③；　RUNコマンドで例の様に指定する。CSTR1という名のバイナリィデックファイルである。（NOGO）はコンパイルまで，実行をしない指定したファイル記述がなければ３～２０LLINKSの大きさで利用者のパーマネントファイルに作られる。（クイック・アクセスとして）

２－６　ファイル内のソース・プログラムをオブジェクト形式（Ｃ＊）にして登録

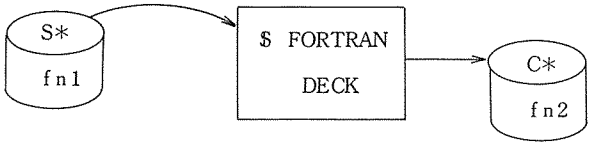
1. Ｌ.Ｂ.Ｓ.

```
        1       8          16
        $       FORTRAN     DECK
①       $       PRMFL       S*,R,S, [      ] /fn1
②       $       PRMFL       C*,W,S, [      ] /fn2
        $       ENDJOB
```

①； Ｓ＊でコンパイラーへの入力を示し

②； Ｃ＊でコンパイラーからの出力結果を入れる

○ Ｓ＊で使うファイル名とＣ＊に入れるファイルは違えること。



2. Ｔ.Ｓ.Ｓ.

①　＊RUN　OLDPRO=;CSTR2

②　＝　デ　ー　タ

説　明

①； RUNコマンドによりファイル内のソースプログラム を呼び（例ではファイル名がOLDPRO) Ｃ＊ファイルに入れる （その時のファイル名をCSTR2と名を付けた）そして実行している。

②； 実行のためのデータを入れる。

3. C.R.B.S.

```
SYSTEM? CARD N
READY
*AUTOX
0010$:JOB:[        ] $PASS
0020$:FORTRAN:DECK,NFORM
0030$:SELECTA:fn1
0040$:PRMFL:C*,W,S,[        ]/fn2
0050$:ENDJOB
0060 (Cr)
*RUN
SNUMB# 3259T
CARD FORMAT,DISPOSITION?
N,J
```

○ この例はＴＳＳで作ったファイル（ファイル名fn1）をＣ＊にする

$ FORTRAN で DECKパラメータとNFORMパラメータが必要

2－7　オブジェクト形式（C＊）の実行

1. L.B.S.

```
          1        8              16
①         $        SELECT         [          ] /fn
          $        GO
                   デ　－　タ
          $        ENDJOB
```

説　明

①；　C＊につくられたファイルを引っぱる為の制御文

◎　アドバイス――C＊につくられたファイルはバッチ系，ＴＳＳを問わず実行できる。すなわちバッチで作られたC＊をＴＳＳでもなんの変換なしに使える。

2. T.S.S.

```
＊RUN  CSTR1      又は      ＊OLD  CSTR1
＝ デ － タ                 ＊RUN
                            ＝ デ － タ
```

○　C＊ファイル名を呼びRUNさせる。

2－8　ソース・プログラムを実行形式（H＊）にして登録

1. L.B.S.

```
          1        8              16
①         $        OPTION   SAVE/fn1
          $        FORTRAN
                   ソース・プログラム
          $        GO
②         $        PRMFL    H＊,W,R,  [          ] /fn
③                  デ　－　タ
          $        ENDJOB
```

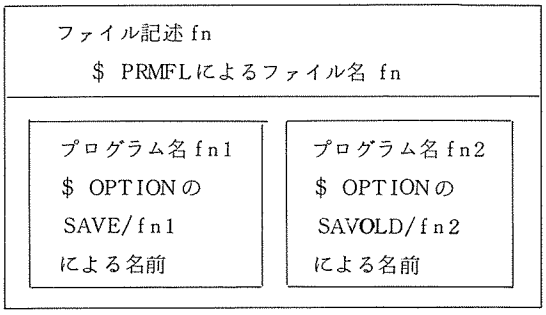
説　明

①；　H＊ファイルにプログラムをセーブする，この時，そのH＊にされたプログラムに名前を付ける必要がある。これは，その名前を使ってプログラムを検索される為に必要。

注意――名前は6文字以内

②；　H＊を格納する為の制御文。ファイルコードは“H＊”書き込みに“W”，直接アクセスだから“R”ファイル名は“fn”，ここでファイル名は①のプログラム名と必ずしも一致しなくてもよ

い。―― というのは，このファイル記述にさらに別のＨ＊にしたファイルを追加登録が出来る（但し，ファイルに余裕があればの話）それには＄ OPTIONでSAVOLD/ｆｎ２ を使う。ここでｆｎ２は始めのSAVE/ fn1 の fn1 と別の名前を用い＄ PRMFLではファイル名ｆｎと同じ



③； Ｈ＊ファイルにプログラムを登録し，さらに実行する場合である。もしＨ＊への登録だけで実行しない場合は＄ OPTION NOGO，SAVE/fn1とする。すなわちOPTION指定にNOGOを追加しておく。

2. T.S.S.

① ＊RUN FN=HSTR

説 明

①； ＦＮはファイル内のソース・プログラム名。それを実行形式プログラムにしファイルへ登録する（この例ではファイル名はHSTR）ここで指定したファイル記述（例ではHSTR）がない場合，システムは自動的に利用者のカタログに３６LLINKS の直接編成ファイルを作り，クイック・アクセス・ファイルとして登録する。（但し，コンパイラーによる致命的エラーがない場合に限る。）

3. C.R.B.S.

```
SYSTEM? CARD N
READY
*AUTOX
0010$:JOB:[          ]$PASS
①  0020$:OPTION:SAVE/FILNAME
   0030$:FORTRAN
②  0040$:PRMFL:S*,R,S,[          ]/fn1
   0050$:GO
```

③　0060$:PRMFL:H*,W,R,[　　]/fn

0070#　データ

　　≀

0100$:ENDJOB

0110 (Cr)

*

説　明

①；　H*に登録するプログラム名を指定（例ではプログラム名FILNAME)

②；　ファイル内のソース・プログラムを引っぱってくる。ファイルコードS*，ファイル名fn1（コンパイラへの入力）

③；　実行形式プログラムを入れるためのファイル，ファイルコードH*，ファイル名fn（ローダからの出力ファイルである）

2－9　実行形式（H*）の実行

1. L.B.S

	1	8	16
①	$	PROGRAM	RLHS,NAME/fn1
②	$	LIMITS	ttt,mmK
③	$	PRMFL	H*,R,R,[　　]/fn
	データ		
	$	ENDJOB	

説　明

①；　H*を実行するためのプログラム。この通り制御文を作る。fn1は③のパーマネント・ファイル（ファイル名fn)の中にある実行すべきプログラムの名前

②；　H*を実行する時の制限値を指定する。tttはCPU時間（1/100時間単位），mmKはメモリの大きさ，単位はKW。

③；　①のプログラムの入っているパーマネント・ファイル。ファイルコードは"H*"，読み出しの"R"，直接編成の"R"，ファイル名fn

2. T.S.S.　①　*RUN HSTR

＝　データ

説　明

①；　ファイル内にあるH*ファイルを呼び出し実行する。

3. C.R.B.S. ① *AUTOX 10

0010$:JOB: [　　] $PASS

② 0020$:PROGRAM:RLHS,NAME/FILNAME

0030$:LIMITS:10,50K

③ 0040$:PRMFL:H*,R,R, [　　] /fn

0050# データ

≀

0080$:ENDJOB

説　明

①； AUTOXコマンドによりライン番号を付ける。パラメータの"10"はライン番号を"10"より付けることを意味する。

AUTOX␣m,n　　mは最初のライン番号

nはステップ巾

②； L.B.S. の説明①と同じ。例ではプログラム名FILNAME これは③のfnの内にある。

③； L.B.S. の説明③と同じ

注意) 1. L.B.S. で作ったH*はC.R.B.S. で実行することが可能であるが，T.S.S. では不可能である。

2. また，T.S.S. で作ったH*はT.S.S. のみで可能である。すなわち，L.B.S. 又はC.R.B.S.で実行不可能。これは，バッチ系及びT.S.S. の入出力を管理するOSが異なる為による。

3) ライブラリィの利用

○ シーケンシャル・ライブラリィとランダム・ライブラリィ

ユーザの登録するライブラリィには，上述の2種類のライブラリィがある。それぞれ，使われ方に特徴があり，次にその説明をする。

・ シーケンシャル・ライブラリィ

これが使われる場合，その使われる順番を意識する必要がある。用途としてはライブラリィの数が少ない場合や，T.S.S.などの場合，RUNコマンドのパラメーターが簡単で，複数個可能，すなわちメモリのT.S.S. 利用領域の範囲まで可能である。

・ ランダム・ライブラリィ

これは，使われる順番を意識せずに使える。たとえばMATH−LIBは，これで

ある。T.S.Sの場合，RUNコマンドのパラメータでULIBを指定する必要があり，9個まで可能。

- オブジェクト形式にすれば，どちらの系で作られたかによらずT.S.S.,バッチ系の両方で使用可能である。

◎ アドバイス——どちらかと言えば，ランダム・ライブラリィにしておく方が使いよいと思う。

3－1　サブルーチン・プログラムの登録

1. L.B.S.

```
        1       8           16
$       PROGRAM SCED
$       PRMFL   OT,W,S, [        ] /fn
$       DATA    IN,,COPY
        サブルーチン・ソース・プログラム
$       ENDCOPY
$       ENDJOB
```

説　明

- ソース・プログラムの登録と同じ

2. T.S.S.

```
SYSTEM? FORT N
READY
*AUTO
010 SUBROUTINE SUB (A,B,C,X,Y)
020
          ≀
      ソース・プログラム
          ≀
150 RETURN
160 END
170 (Cr)
①   *SAVE SUB
```

説　明

①；　SAVEコマンドを使用（例ではファイル名 SUB）

- ソース・プログラムの登録と同じ

3－2　サブルーチンプログラムをオブジェクト・シーケンシャル・ライブラリィにして登録

1. L.B.S.

```
          1        8          16
          $        FORTRAN    DECK
          $        PRMFL      C*,W,S, [          ] /fn
                 サブルーチン・ソース・プログラム
          $        ENDJOB
```

説　明

- オブジェクト・プログラムの登録と同じ

2. T.S.S.

```
①    *RUN SUB=;OBJSUB
```

説　明

①；　C＊ファイルの作成と同じ。例ではファイル内のソース・プログラム（サブルーチン）ファイル名SUB,それをRUN コマンドを使いOBJSUBという名のC＊ファイルを作成（順編成ファイル）

- オブジェクト・プログラムの登録と同じ

3－3　サブルーチン・プログラムをオブジェクト・ランダム・ライブラリィにして登録

1. L.B.S.

```
          1        8          16
①         $        FORTRAN    DECK
②         $        FILE       C*,AIS,30L
                 サブルーチン・ソース・プログラム
③         $        PROGRAM    RANLIB
④         $        FILE       R*,A1
⑤         $        PRMFL      A4,W,R, [          ] /fn
          $        ENDJOB
```

説　明

①；　オブジェクトファイルの作成と同じ，まずC＊ファイルを作成する。

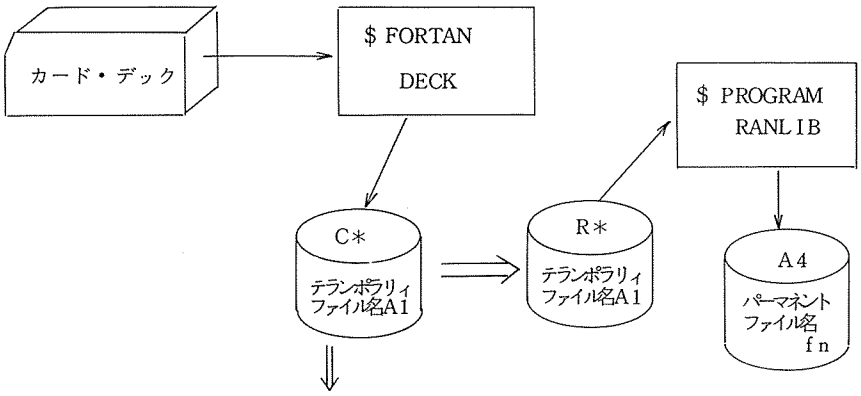
②；　$ FILE文でテンポラリィファイルを確保し，それを$FORTRANによるコンパイラからの出力をパラメーター，"DECK"によりC＊へ出力する。従って，第1パラメータはファイルコード，"C＊"，第2パラメータの"A1S"は，次のアクビティ（$

PROGRAM）への為にそのファイルを“A1”という名でセーブすることを意味し，第3パラメータの“30L”はそのテンポラリィファイルの大きさを30リンク確保し，“L”は順編成ファイルであることを意味する。

③； この制御文でランダムライブラリィを作る。この通りの制御文を使用する。

④； ②と同じテンポラリィファイル使用の制御文で$ PROGRAMへの入力ファイルは“R*”ファイルである。実は②でのテンポラリィファイルである。それは④の第2パラメータでファイル名を“A1”とし，②での第2パラメータと一致させている。第1パラメータは前述の通りであるのでファイルコード“R*”である。

⑤； $PROGRAM RANLIB よりの出力ファイル，ファイルコード“A4”，書き込み指定の“W”，ランダムアクセス指定の“R”



シーケンシャル・ライブラリィが出来ている。

2. T.S.S.————できない。

3－4 オブジェクト・シーケンシャル・ライブラリィをオブジェクト・ランダム・ライブラリィにして登録

1. L.B.S.

	1	8	16
①	$	PROGRAM	RANLIB
②	$	PRMFL	R*,R,S, [　　] /fn1
③	$	PRMFL	A4,W,R, [　　] /fn2
	$	ENDJOB	

説　明

①；　RANLIBプログラムを使いシーケンシャルをランダムに変換する。

②；　①の入力ファイルコードは"R＊",fn1 はオブジェクト・シーケンシャルライブラリィ名

③；　①からの出力ファイルコード"A4",fn2 はオブジェクト・ランダム・ライブラリィ名

○　3－3の説明と同じ，ただ入力ファイルがすでに作られていただけ。

2. T.S.S.————できない。

3. C.R.B.S.　　＊AUTOX

0010$:JOB:［　　］PASS

0020$:PROGRAM:RANLIB

0030$:PRMFL:R＊,R,S,［　　］/fn1

0040$:PRMFL:A4,W,R,［　　］/fn2

0050$:ENDJOB

0060 (Cr)

＊RUN

SNUMB#7654T

◎　アドバイス——T.S.S.で作ったシーケンシャル・ライブラリィをC.R.B.S.を使ってランダム・ライブラリィに変換することが出来る。すなわちシーケンシャル・ライブラリィ（C＊)は，T.S.S.でもバッチ系でも使用可能。

3－5　ファイルに登録してあるサブルーチン（ソース形式）を用いて実行

1. L.B.S.　　1　　8　　16

①　$　FORTRAN

②　メインプログラム　（CALLサブルーチン）

③　$　FORTRAN

④　$　PRMFL　S＊,R,S,［　　］/fn

⑤　$　GO

データ

$　ENDJOB

説　明

①；　メインプログラム（CALL文によりサブルーチンを呼ぶ）コン

パイルする。

②； CALL文によるサブルーチンがファイルに登録してある（ソース形式）

③； ファイル内のソース形式のサブルーチンのコンパイルをする。

④； メインプログラムより呼ばれるサブルーチンのファイル。ファイル名 fn

⑤； 結合し実行する。

2. T.S.S.

*RUN *;SUB

説 明

RUNコマンドのパラメータの"*"はカレントファイルの内容（メインプログラムがカレントファイル内にあり，サブルーチンコールをしている）SUBはメインプログラムで呼ばれるサブルーチンを登録してあるファイルの名

3－6 ファイル内のメインプログラム（ソース形式）とサブルーチン（ソース形式）を用いて実行

1. L.B.S.

	1	8	16
①	$	FORTRAN	
②	$	PRMFL	S*,R,S, [　　]/fm
③	$	FORTRAN	
④	$	PRMFL	S*,R,S, [　　]/fn
	$	GO	
	データ		
	$	ENDJOB	

説 明

①； メインプログラムのコンパイルを行う。

②； コンパイラの入力ファイル メインプログラムを含む ファイル名fm

③； サブプログラムのコンパイルを行う

④； コンパイラの入力ファイル サブルーチンのファイル名fn

2. T.S.S.

*RUN␣fm;fn1;fn2

＝ データ

説　明

fmはメインプログラム fn1，fn2はソース形式のサブルーチンいずれも，T.S.S.で作られたプログラムである。もし，これがバッチ入力の場合，コード変換をし，T.S.S.－JISコードにしておく必要がある。ファイルをつづける場合“；”でつづける。

3. C.R.B.S

```
*AUTOX
0010$:JOB:[　　] $PASS
①  0020$:FORTRAN:NFORM
②  0030$:SELECTA:fm
③  0040$:FORTRAN:NFORM
④  0050$:SELECTA:fn
⑤  0060$:GO
0070# データ
        ≀
0120$:ENDJOB
0130 (Cr)
*
```

説　明

- この例はT.S.S.で作ったメインプログラムfmと同じくT.S.S.で作ったサブルーチンfnいずれもソース形式のプログラムを①，③でコンパイルし，⑤で結合実行する。

3－7　ファイル内のサブルーチン（オブジェクト・ランダム・ライブラリィ）を用いて実行

1. L.B.S.

```
          1       8          16
①    $    LIBRARY  S1
     $    FORTRAN
       メイン・プログラム　(CALLサブルーチンを含む)
     $    GO
②    $    PRMFL     S1,R,R,[　　]/fn
       データ
     $    ENDJOB
```

説　明

①；　この制御文でユーザー・ライブラリィを含むJOB であること

をシステムに知らす，$ LIBRARY 文のファイルコード"S1"にそのライブラリィがある事を示す。システムは始めにユーザーライブラリィを捜す。$ LIBRARY のファイルコードは2文字の英数字で書く。これは②の$ PRMFL文のファイルコードと一致させる必要がある。なぜなら，そのファイルコードを持つファイルの内にそのライブラリィがあるから，そのファイル名 fn

②； $ LIBRARYで定義したファイルコードと同じファイルコードを用いる。

○ ライブラリィがいくつもある場合，ライブラリィの数だけの $ LIBRARY と $ PRMFL のファイルコードが同じペアが必要。

注意 ——— 次のファイルコードは使用禁止（システムで予約されている）

LC，LM，L*，*L，

2. T.S.S. ① *AUTO

② 010 ⓐ ******MAIN PROGRAM******

020

≀

③ 030 CALL SUBPRO(

≀

④ *RUN = (ULIB) SUB1

= デ ー タ

説 明

①； AUTOコマンドによりメインプログラムをカレントファイルに入力する。

②； NFORM形式のプログラムではコメントには"C"又は"*"を用いる。また継続行の指定では"&"が用いられる。

③； サブルーチンの要求

④； RUNコマンドでカレントファイルの内容をコンパイルし,オプション指定"=(ULIB)"によりローダにユーザーライブラリィ，そのファイル名 SUB1を結合させ，そして実行することを指示する。

○ 呼ばれるサブルーチンがオブジェクト・ランダム・ライブラリィであれば，必ず"=(ULIB)"をオプション指定する。最大個9個で指定可能，ファイルは";"で続ける。

3－8　ファイル内のメインプログラム（ソース形式）とオブジェクト・ランダム・ライブラリィを用いて実行

1. L.B.S.

```
          1        8          16
$         LIBRARY  L1
$         FORTRAN
$         PRMFL    S*,R,S,  [          ] /fm
$         GO
$         PRMFL    L1,R,R,  [          ] /fn
    デ　－　タ
$         ENDJOB
```

説　明　　3－7と同じ。

2. T.S.S.　　*RUN␣fm= (ULIB) fn1;fn2

＝　デ　－　タ

説　明

メインプログラム名fm，ランダム・ライブラリィfn1，fn2の2個を使う。

3－9　ファイル内のオブジェクト形式のメインプログラムとオブジェクトシーケンシャル・ライブラリィを用いて実行

1. L.B.S.

```
          1        8          16
①         LIBRARY  S2
②         SELECT   [          ] /fm
          GO
③         PRMFL    S2,R,S,  [          ] /fn
    デ　－　タ
          ENDJOB
```

説　明

①；　ライブラリィ使用をシステムに知らす。ファイルコードは③のファイルコードと一致させる。2文字の英数字（例では"S2"）

②；　C*としてあるメインプログラムを引っぱってくる。

③；　①と同じファイルコード，読み出の為の"R",シーケンシャルアクセスの為の"S"，ファイル名fn

2. T.S.S.　　*RUN␣fm;fn

= デ ー タ

説 明

fmはC*のメインプログラム，fnはC*のサブルーチン，このそれぞれのC*はT.S.S成作のものでもバッチ系成作のものでもどちらでも，T.S.SのRUNコマンドで実行できる。

3－10 MATHLIB（メーカー提供のライブラリィ）又は，センターライブラリィを使用

1. L.B.S.

```
          1        8              16
①    $'       LIBRARY   LM
     $        FORTRAN
         ソース・プログラム・デック
②    $        GO
         デ ー タ
     $        ENDJOB
```

説 明

①； この制御文でMATHLIB使用をシステムに知らせる。

ファイルコード“LM”は固定

②； この制御文はマクロ制御文であり，この中に，

$ PRMFL LM,R,…が含まれている。

（センターライブラリィはまだ登録されていません。77−4−1現在）

2. T.S.S.　　*RUN = (ULIB) LIB/MLIB,R

= デ ー タ

説 明

この例ではカレントファイル内にメインプログラムがあり，そのプログラムでMATHLIBを呼ぶ。

パラメータの“(ULIB)”はランダム・ライブラリィを使うこと意味し，“LIB/MLIB,R”はLIBというUMCの内のMLIBというファイル名で，読み出し指定Rであるということを意味する。

注意 —— 最後のパラメータ“,R”を忘れるとそれはR/W指定と同じ事なので，次のメッセージが出て使えない。

<50>FILE MLIB--PERMISSION NOT GRANTED

3. C.R.B.S

```
*AUTOX
0010$:JOB: [          ] $PASS
0020$:LIBRARY:LM
0030$:FORTRAN
0040$:PRMFL:S*,R,S, [          ] /fn
0050$:GO
0060# デ ー タ
      ↓
0100$:ENDJOB
0110 (Cr)
*
```

説　明　この例はメインプログラム（ソース形式）がファイル内にある。

4）　データファイル／ワークファイルの利用

○　データを自分のパーマネントファイルに入れておき，それを実行時に読み出し実行させたり，また出力結果をパーマネントファイルに入れておき必要なとき見たり，使ったりする事はパーマネントファイルを持つ利点の一つであろう。また，パーマネントファイルをワークファイル的に使い，中間結果を後日調べることにより，プログラムのデバッグにも活用できるし，次のＪＯＢ への入力データとしても使える。ここでは，その様なパーマネントファイルの活用と，システムが持つテンポラリティファイルの使用方法を述べてみる。

◎　アドバイス ――入出力文READ/WRITEでFORMAT を指定する場合，とかくＩ型，Ｆ型，Ｅ型等，忘れがちでしかも桁数に至っては特にわずらわしい。そこで，特にＴ．Ｓ．Ｓではデータ入力の前にまず WRITE 文を使い，何のデータかを端末に知らせてやる事を勧める。そしてデータのFORMATをやたらうるさく言わない人には"FORMAT (V)"，またはＬＩＳＴ型READ/WRITE文をドウゾ！（暗黙書式です。詳しくはFORTRAN 文法説明書7－ 5－１３を参照）。 また，データの区切はなるべく","でするのが確実です。（追伸） プログラムの最後ＳＴＯＰ文にコメントが書けるのでこれも利用すると便利。

例 STOP ▼▼ GOOD BYE▼▼

4－1　データの登録

1. L.B.S.

```
1       8           16
$       PROGRAM     SCED
$       PRMFL       OT,W,S
$       DATA        IN,,COPY
        データ
$       ENDJOB
```

説　明

○　これはソース・プログラムの登録と同じ

2. T.S.S.

1. 端末入力

```
*AUTOX
①  0010#12.345
   0020#0.234
        ≀
   *SAVE DATA1
```

説　明

○　AUTO又はAUTOXコマンドにより入力する。

①；　ライン番号の後に"＃"を入れ，ライン番号とデータの区切りをハッキリさせる。

注意）　このまま，データとして使う場合，一般にライン番号もデータと見なされるため，ライン番号を取る必要がある。（但し，C.R.B.Sで$ SELECTAを使えば別），そこで次の様にJISJISコマンドを用いライン消去を行う。

```
①  *JISJIS DATA1,DATA2
   FILE-DATA1
②  LABELS? S
③  TAB CHARACTERS AND SETTINGS? (Cr)
④  *JISJIS DATA2,DATA3
⑤  LINE NUMBERS? (Cr)
   FILE-DATA3
⑥  TAB CHARACTERS AND SETTINGS? (Cr)
   *
```

説　明

①；　はじめのファイル名（例ではDATA1）がTSS－JISファイ

ル，これはライン番号付，次のファイル名（例ではDATA２）はこれから作られるBATCH−ＪＩＳ ファイル

②； Ｓを指定することにより，ライン番号除去

③； タブ不用

ここまでで，BATCH−ＪＩＳファイルのライン番号なしのデータがファイル名DATA２でコピーされた。

④； 次にこのコマンドでBATCH−ＪＩＳファイルをTSS−ＪＩＳファイルに変換する。DATA３ が目的のライン番号なしのデータをコピーするTSS−ＪＩＳファイル名である。

⑤； キャリッジリターンでライン番号を付けない。

⑥； タブ不用

これで，ライン番号のないデータファイルができた。

2. カードでT.S.S. 用にセンターから登録する。

```
          1        8          16
          $        PROGRAM    TSCONV
          $        PRMFL      OT,W,S, [          ] /fn
①        INPUT,ASIS
              データー・カード
          $        ENDJOB
```

説 明

○ ソース・プログラムの登録と同じ

①； ライン番号を付けない為パラメータ“ＡＳＩＳ”を使う。

◎ アドバイス—— 一般にデータはT.S.S. 又はバッチ系で使う場合，ライン番号なしで用いられる。プログラムの場合はライン番号付の方が便利である。T.S.S. はライン番号付，バッチ系で使う場合$ FORTRANのオプションとしてNFORMを指定すればよい。

４－２ ソース・プログラム内のREAD/WRITEにパーマネントファイルを使う

1. L.B.S.

```
          1        8          16
          $        FORTRAN
              ソース・プログラム
                  ≀
①             READ (5,10) A
```

```
                ≀
②               WRITE (1,20) AB
                ≀
③               REWIND␣1
④               READ (1,30) B
                ≀
⑤               WRITE (6,40) C
                ≀
        $       GO
⑥       $       PRMFL    05,R,S, [          ] /fn1
⑦       $       PRMFL    01,W,S, [          ] /fn2
⑧       $       PRMFL    01,R,S, [          ] /fn2
        $       ENDJOB
```

説　明

①；　READ文のファイルコード５は一般にカードリーダーであるがこれを⑥で指定したパーマネントファイルから読み込む。

②；　WRITE文のファイルコードを１にし⑦のパーマネントファイルに書き込んでおく，後④でこの結果を読み出す。ワークファイル的な使用をする。

③；　ファイルを一度閉じる。ファイルコード１のファイルを閉じる。

④；　ファイルコード１のファイルより読み出す。

⑤；　通常 WRITE 文のファイルコード６はラインプリンター

⑥；　①の読み出しファイル，ファイル名 fn1,ファイルコードは必ず２桁，だから０５を指定

⑦；　②の書き込み用ファイル　ファイル名 fn2 ファイルコード０１

⑧；　読み出し用ファイル　ファイルコード０１ だから⑦のファイル名と同じファイル名 fn2

注意）　いずれのファイルも，あらかじめＡＣＣＥＳＳ又はＦＩＬＳＹＳによりファイルのエリアを必ず確保しておく。

2. T.S.S.

```
*AUTO
010   ソース・プログラム
          ≀
050 READ (1,10) A
          ≀
```

```
100 WRITE (2,10) B
110 10 FORMAT (V)
                ↓
*RUN #DATA1"01";DATA2"02"
```

説　明

- ○　カレントファイル内のプログラムを実行するが，READ文でパーマネントファイルより読み込み，WRITE文でパーマネントファイルに書き込む。RUNコマンドのパラメータ　ファイル名DATA1がREADのファイルコード1と一致させるため01と2桁で指定，ファイル名DATA2がWRITEのファイルコード2と一致させるため02と指定する。すなわち01，02はそれぞれファイルの代替名として，AFTに登録される。
- ◎　これはRUN コマンドのパラメータにパーマネントファイルを指定したやり方であるが，ソースプログラム中にパーマネントファイルを指定するシステムサブルーチンATTACHがある。（詳しくはFORTRANサブルーチンライブラリィ説明書2．16を参照）

3. C.R.B.S.

```
*AUTOX
0010$:JOB:[          ]$PASS
①  0020$:FORTRAN:NFORM
②  0030$:SELECTA:fn
   0040$:GO
③  0050$:SELECTA:fn1
④  0060$:PRMFL:01,W,S,[          ]/fn2
⑤  0070$:PRMFL:02,R,S,[          ]/fn3
   0080$:ENDJOB
```

説　明　○　T.S.S.で作ったプログラムを使う。

①；　T.S.S.で作成されたプログラムはNFORM，ライン付だからオプション指定

②；　T.S.S.ファイルを捜す。ファイル名fn

③；　T.S.S.で作成された入力データ，フアイル名fn1（READ文でファイルコード5を使用）

④；　出力ファイルとしてファイル名fn2を使いファイルコード01

⑤；　READ文でファイルコード2を使ったファイルより読み出す。ファイル名fn3，そのときファイルコード02

4－3　テンポラリィファイルをワークファイル（シーケンシャルアクセス）として使用

1. L.B.S.　　1　　8　　16

```
$      FORTRAN
  ソース・プログラム
        ≀
①       READ (5,10) IN
        ≀
②       WRITE (1,10) A
        ≀
③       READ (1,10) B
        ≀
④       WRITE (2,10) C
        ≀
⑤       READ (2,10) D
        ≀
  10  FORMAT (V)
        ≀
$      GO
⑥      $      FILE      01,,10L
⑦      $      FILE      02,,20L
⑧        デ　ー　タ
$      ENDJOB
```

説　明

①；　このREAD文の入力データは⑧のデータである。

②；　ファイルコード1で中間結界をワークファイルに書き出す。

③；　②の中間結果を読み込むファイルコードは同じ1

④；　ファイルコード1で中間結果をワークファイルに書き出す。

⑤；　④の中間結果を読み込む。ファイルコードは2

⑥；　②，③のワークファイルを$FILE文で確保するファイル コードは②，③と同じであるが2桁表示であるから“01”，“，，”はこの通り“10L”はワークファイルの大きさを10 リンクとりアスセス指定はL（LINKED：順編成）

⑦；　④，⑤のワークファイル，ファイルコード“02” ④，⑤と一致させる。“，，” の後“20L”はワークファイルの大きさを20

リンクアクセス指定はL

- $ FILE 文はテンポラリィファイルの使用を示す制御文である。
- これはメモリー使用を小さくするかわり入出力回数が増すが多量の中間データを使うプログラム（シミュレーションなどに）適する。

2. T.S.S.

```
①  *RUN #01;02
   = デ ー タ
②  *BYE
③      2 TEMPORARY FILES CREATED
④  01        ? LIST 01
⑤  ERR-RESPONSE MUST BE "NONE","SAVE",
   OR CR
⑥  01        ? SAVE D1
   02        ? SAVE D2
⑦  **COST:          $2 TO DATE:     $2= 0%
⑧  **ON AT 12.207-OFF AT 12.251 ON 04/01/77
```

説　明

①；RUNコマンドによりカレントファイルの内容を実行する。プログラムはL.B.Sと同じ形，ただしTSSファイルとする。"#01;02"でファイルコード01，02のテンポラリィ・ワークファイルが割り当てられる。

②；BYEコマンドでシステムと会話を打ち切る。

③；AFTに残っていたテンポラリィファイルの数を出力してくる。会話の途中でREMOVEコマンドを使って01と02をAFTから除くことも出来るし，STATFコマンドによりAFTに登録されているファイルを見ることができる。

④；01のファイルはどうするか？　ここで01の内容を見ようと思いそのLISTコマンドを使うと，⑤のメッセージが帰ってくる。したがって見たければ，まず01をセーブし，後にLISTコマンドで見る。

⑥；01をD1というファイル名でセーブする。

⑦；02をD2というファイル名でセーブする。

⑧；システムからのメッセージ

注意　会話終了時BYEコマンドのわかりに簡易デスコネクト（コントロール・キーと文字盤のCの同時打鍵）をすると③のメッセージは返さ

れてこない。

4－4　テンポラリィ・ファイルをランダム・アクセス・ワークファイルとして使用

1. L.B.S.　　1　　8　　16

case 1

```
$       FORTRAN
    ソース・プログラム
        ≀
①       CALL␣CREATE (1,38400,1,ISTAT)
②       CALL␣RANSIZE (1,38400)
③       WRITE (1,10) A
        ≀
④       READ (1,10) B
        ≀
$       GO
    デ ー タ
$       ENDJOB
```

説　明　○　プログラム中にランダムアクセスファイルを指定するシステム・サブルーチンCREATEとRANSIZEをペアにして使う。
例は③のWRITE文でランダム出力し，後の④のREAD文でランダム入力する。このときそれぞれのステートメントの前に①，②が必要で，しかもファイルコードを一致させる（例ではファイルコードで，ファイルの大きさ，語単位①の第3パラメータは直接編成ファイルを示す。第4パラメータはステータスリタンワード）

case 2

```
        1       8       16
$       FORTRAN
    ソース・プログラム
        ≀
①       CALL␣CREATE (2,76800,1,ISTAT)
②       WRITE (2,10) A
        ≀
③       READ (2,10) AB
        ≀
$       GO
④   $   FILE      02,,20R
```

```
データ
$       ENDJOB
```

説　明　・　case 1と同じであるが，case2 ではCREATEと$ FILE とのペアである。

④；　①，②，③に対するテンポラリイファイル・ファイルコードはすべて一致させる。"20R"はファイルの大きさが20リンクでRはランダムアクセスを指定する。

◎　CREATE，RANSIZEの詳しい説明はFORTRAN サブルーチン説明書の2・14，2・12を参照

2. T.S.S.　○　プログラム中にCREATEとRANSIZE を用いて使う方法だけである。その時RUNコマンドでランダムファイルと同じファイルコードをテンポラリィ指定するとエラーになる。

4－5　ファイル中の計算結果のリストアップ

1. L.B.S.

```
1       8           16
$       PROGRAM     SCED
$       PRMFL       IN,R,S, [          ] /fn
$       FILE        OT,NULL
$       ENDJOB
```

説　明　2－4のファイル内のソース・プログラムのリストアップと同じ

2. T.S.S.

```
*LIST␣fn
```

○　センターへ出力する場合BPRINTコマンドを使う。

```
*BPRINT
INPUT FILES? fn
$IDENT? [          ],識別名
LABELS? N
SNUMB# 2365T
```

以上，ざっと25項目挙げましたが，この他に色々の組み合せや，使い方，また，別のサービス・プログラムを用いる方法もありますが，それらは，各々のマニュアルを御覧いただきたいと思います。

次回で，「ACOSのファイル」も最終回となりますが，次回はファイルに関する項目で述べ足りなかった事柄について，少し補足いたします。