

Title	ACOSのファイル (その3)
Author(s)	多喜, 正城
Citation	大阪大学大型計算機センターニュース. 26 P.79- P.116
Issue Date	1977-08
Text Version	publisher
URL	<a href="http://hdl.handle.net/11094/65355">http://hdl.handle.net/11094/65355</a>
DOI	
rights	

*Osaka University Knowledge Archive : OUKA*

<https://ir.library.osaka-u.ac.jp/repo/ouka/all/>

## 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認して、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。<sup>\*)</sup>

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・アクセラ・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<sup>\*1)</sup> を利用, TSS では ACCESS サブシステム<sup>\*2)</sup> 又は SAVE コマンドを用いる。

#### ◎ FILSYS によるファイル記述<sup>\*2)</sup>

{ FC } 課題番号 / ファイル名, [パスワード / パスワード名 / ] [, パーミ  
{ CF } 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 } / 初期値 { UNLIMITED } /  
{ LINKS } { , 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 LINK = 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, PASSWORD/ABC/, R, LINKS/10/  
, MODE/RAND/

---

\* 1 ) ファイル・マネージメント・スーパーバイサ説明書を参照

\* 2 ) センターニュースNo24 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個以上の空白を置く, 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      n n n n n
            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FILEEDIT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。(FILEEDIT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blank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	<input type="text"/> \$pass
①	\$	PROGRAM	TSCONV
②	\$	PRMFL	FT, W, S, <input type="text"/> /fn
③		INPUT, INSERT,	1, 72, 10, 10
		ソース・プログラム・カード	
	\$	ENDJOB	

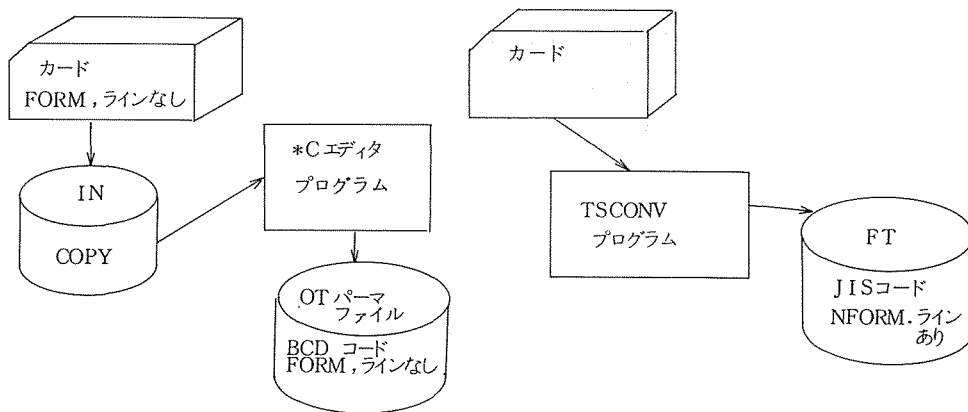
説明

- 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? ④Cr
*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   -   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_ f n
=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ばってきて実行させ、又、自分のファイルに入れる方法を述べる。

① SYSTEM? FORT\_ O\_  / f n , R  
 READY

② \*RUN  
 =データ  
 ↓

③ \*SAVE\_ f n 1

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、引っばってくるべき UMC と、そのファイル名 f n , 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 となり引っばってこれない。

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 f n を入れるがここで元の引っば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 f n 1 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<sub>fn</sub>,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  
 }  
 デ - タ  
 }

- 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撰択レベルで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, 3273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$:FORTTRAN:NFORM
0030$:SELEDTA:fn
0040$:GO
0050#1234.5
      データ
0080$:ENDJOB
0090 (Cr)
*
```

注意) データ入力する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, <input type="text"/> /fn1
	③	\$	PRMFL OT,W,S, <input type="text"/> /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コマンドでパーマメ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   -   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

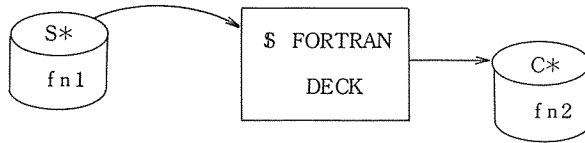
- ①; AUTO コマンドによりソース・プログラムの入力
- ②; ライン番号のすぐ後にブランクが入っているので@でそのブランクを消す。そしてコメントのCを入れる。
- ③; RUN コマンドで例の様に指定する。CSTR1 という名のバイナリデックファイルである。(NOGO) はコンパイルまで, 実行をしない指定したファイル記述がなければ 3 ~ 20 LLINKS の大きさで利用者のパーマネントファイルに作られる。(クイック・アクセスとして)

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\*でコンパイラへの入力を示し
- ②; C\*でコンパイラ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
  
```

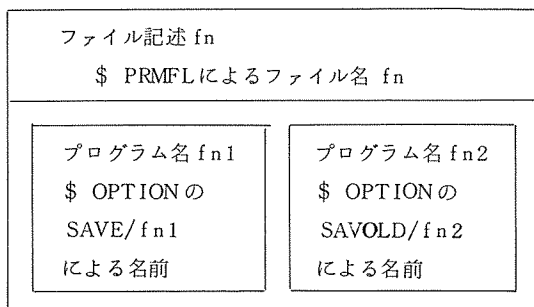
### 説 明

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な。

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/f n2を使う。ここでf n2は始めのSAVE/f n1のf n1と別の名前を用い\$ PRMFLではファイル名f n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/f n1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,  /f n1

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・プログラムを引っばってくる。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   t t t , 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) 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。t t t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る、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

```

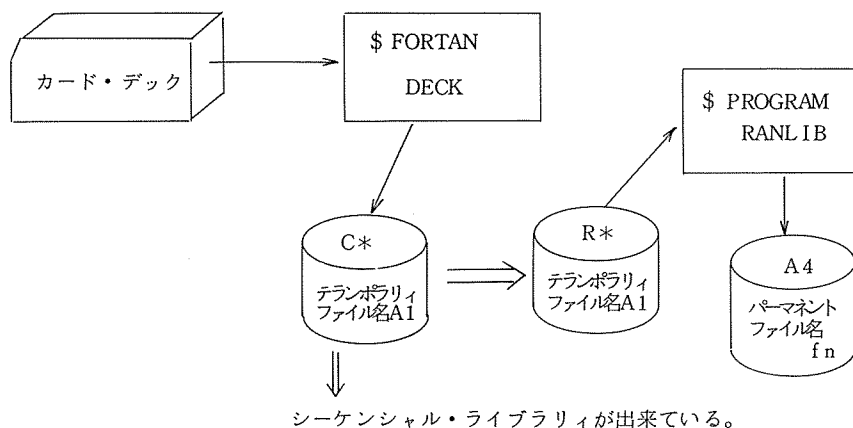
1. L.B.S.      1      8      16
    ①      $      FORTRAN  DECK
    ②      $      FILE      C*,AIS,30L
                サブルーチン・ソース・プログラム
    ③      $      PROGRAM  RANLIB
    ④      $      FILE      R*,A1
    ⑤      $      PRMFL    A4,W,R, [ ] /fn
                $      ENDJOB
    
```

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“AIS”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”



2. T.S.S. ———— できない。

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

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・ライブラリをC.R.B.S.を使ってランダム・ライブラリ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(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 - 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, <input type="text"/> / fm
③	\$	FORTRAN	
④	\$	PRMFL	S*,R,S, <input type="text"/> / fn
	\$	GO	
		デ - 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 - 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)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・ライブラリを含むJOBであること

をシステムに知らず、\$ LIBRARY 文のファイルコード“S1”にそのライブラリがある事を示す。システムは始めにユーザーライブラリを捜す。\$ 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。

- ライブラリがいくつもある場合、ライブラリの数だけの \$ 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 ② \*\*\*\*\*MAIN PROGRAM\*\*\*\*\*  
020  
}
- ③ 030 CALL SUBPRO(  
}
- ④ \*RUN = (ULIB) SUB1  
= デ - 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、  
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
                   データ
                   $      ENDJOB
    
```

説明 3-7と同じ。

```

2. T.S.S.          *RUN fm = (ULIB) fn1 ; fn2
                   = データ
    
```

説明

メインプログラム名 fm, ランダム・ライブラリ 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
                   データ
                   $      ENDJOB
    
```

説明

- ①; ライブラリ使用をシステムに知らせず。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ばってくる。
- 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    f m ; f n  
                          = デ - タ

説明

f m は C \* のメインプログラム, f n は C \* のサブルーチン, このそれぞれの C \* は T.S.S 成作のものでバッチ系成作のものでどちらでも, 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) 又は, センターライブラリを使用

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 - 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 - 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 であるということの意味する。

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はまず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

④； 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・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説明書2.16を参照)

3. C.R.B.S.

```

*AUTOX
0010$ : JOB: [ ] $PASS
① 0020$ : FORTRAN: NFORM
② 0030$ : SELECTA: f n
0040$ : GO
③ 0050$ : SELECTA: f n1
④ 0060$ : PRMFL: 01, W, S, [ ] / f n2
⑤ 0070$ : PRMFL: 02, R, S, [ ] / f n3
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 f n
  - ③; T.S.S. で作成された入力データ, ファイル名 f n1 (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 f n2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 f n3, 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で中間結果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リンクとりアクセス指定は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えばその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コマンドで見ると見る。
- 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
              $      ENJOB
```

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わせや、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、今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。