

45-3

電 子 計 算

技 術 管 理

土 木 研 究 所 資 料

第 564 号

電 算 速 報

第 11 号

昭和 45 年 3 月

建設省土木研究所

電 算 速 報 1 1 号 目 次

§ 1. 情報化社会における情報伝達用語	1
§ 2. 自動図化とプログラム例	2 6
(電算速報 1 0 号の訂正)	
§ 3. 電算室よりのお知らせ.....	3 5
〔 1 〕 図化機について.....	3 5
〔 2 〕 JOB割り込みについて	4 0
〔 3 〕 プログラマ名簿	4 4

§ 1. 情報化社会における情報伝達用語

中 村 慶 一

これは土木研究所のALGOL 2 日間講習会の第1日の午前の導入部と、第2日最後のまとめの部の速記録から編集したもので、実際のコースではこの間に導入部の知識による各自のプログラムによる実習、文法詳論、手続きも利用した各自のプログラムによる実習、文法マニュアルによる復習などが入る。各機関で行なわれる講習会での参考として御利用いただければ幸いである。

(1) 情報化社会への移行

木の実の採集や狩猟によって当座の食物を調達していた狩猟時代から定着農業、そして都市化が始まり、第1次産業革命までのいわゆる農業化社会では、人間が主役を演じ、その社会での価値の根元は物をいかに多く持っているかに存する時代であった。これに対して約200年前イギリスに起った第1次産業革命の嵐は逐次アジアにも拡がり、日本にもその波が押寄せて来たのは100年前であった。

日本はその持ち前の新規な物に対するこだわりのない興味、均質化し、しかも平均して高い教育水準、勤勉さ、理解の早さなどを十分に発揮して他の諸国を倍の速度でかけぬけて、一足先にこの工業化社会からさらに次の情報化社会へ移行しようとする勢いを見せているが、来るべき情報化社会の主役は現在の工業化社会の主役が人間で脇役が機械であるのに対して、人間と機械が一体となったマンマシンシステムであると云われている。また工業化社会の価値の根元がエネルギーに置かれていたのに対し情報化社会の価値の根元は情報そのものであり、この20年来静かに進行しつつある第2次産業革命によってちょうど物そのものを売るより、いかにエネルギーを費して加工したかの程度に価値を認めて貰った時代から今までは正当に評価されることのない情報そのものに価値が認められる時代に入るわけである。

社 会	価値の根元	主 役	
農 業 化	物 質	人 間	> 第 1 次産業革命
工 業 化	エ ネ ル ギ	人 間 と 機 械	> 第 2 次産業革命
情 報 化	情 報	人 間 機 械 系	

ある集団が円滑に仕事を進めて行く為には、その集団の中で意志の伝達が十分に行なわれることが必要なのはバベルの塔の故事をひくまでもないことであるが、人間の筋肉的能力の拡大をねらった工業化社会における人間と機械の関係と異なり、情報化社会の主役マンマシンシステムは人間の神格的能力の拡大をねらって、しかも云わば一体となって働かなければならぬ宿命からも、このシステム内の情報伝達用語はよほど人間語に近いものでなければならぬ。

(2) 人間機械系の用語

(1) 機 械 語

電子計算機にはその設計当初にそれぞれ設定された機械語がある。たとえば 3 4 5 番地の内容を取り出してそろばんに加えよという命令が

0 3 / 3 4 5

という形のものであったとしよう。昔のプログラマがそうであったようにある A という種類の計算機の機械語を一生懸命に覚えたプログラマが、それと異なる B という種類の計算機に変更になると、今まで知識は御破算になって、今度は 2 1 という命令が加算ということになっているかも知れない。まだどの番地までが何に使われているかをクロスワードパズルを解くように一心に考えたりするような問題の本質とは何等かわりのない部分で、これらプログラマのエネルギーの大半をすり減らさざるを得なかったものである。(もっとも何でもあれその努力感だけで満足している人なら別であるが、効率を考えると、全くむだな努力を強いられていることになる)

(2) アセンブラ

当然誰しも考えることであるが、加算なら ADD とか、減算なら SUB といったどの機械にも共通なシンボリックな命令語が使用でき、300番地をA番地としたら、345番地はA+45番地というように絶対番地でなくて相対番地が利用できるならば

ADD / A + 45

と機種が変更になろうが命令語は変わらず、Aに実行の時に実際の値を入れてやればAを起点として考える範囲は手を加える必要がないといった、いわゆるアセンブラは機械語に比して大いに人間側に歩み寄ったものであるが、依然として機械語と1対1の対応をしており、横丁のポストに手紙を入れに行けと云いたいのに左足を前に出せ、右足との中間に重心が移動したならば、次は右足を前に出せという面倒さは変らない言語である。

(3) コンパイラ

1. FORTRAN

1956年IBM704用にユーザ団体の手によって開発されたFORTRANは、今までのプログラマ人口を一挙に20倍にしたと云われている。人間語に近い表現で書かれたプログラムを文法チェックした上誤りがなければ機械語に翻訳編集してそれに従って実行するもので例として

(A) $4 \times B$ の結果をCに代入せよ

(B) もし $A = B$ なら5, そうでなければ4をPに代入せよ。

(C) IAについて-1から2.5きざみでAが3.5をこえない範囲の値の総和を求めよ。

に対し、FORTRANプログラムは

(A) $C = 4.0 * B$

(B) $P = 4$

IF (A.EQ.B) P = 3

(INTEGER P の処置の後)

```

(C)      S = 0.0
          I A = - 3.5
10  I A = I A + 2.5
          I F ( I A . G T . 3 5.0 ) G O   T O   20
          S = S + I A
          G O   T O   10
20 .....      ( R E A L   I A   の処置の後 )

```

2. ALGOL

これに対し1962年制定の改訂ALGOL60はカードの欄にしばられない、変数に限る所も式を使用できる。ブロック構造を持つその他のより人間語に近い多様な表現を持つ言語で上の例について云えば、

```

(A) C := 4 * B;      ( 4.0 とする必要はない real c,B; の後 )
(B) P := if A=B then 5 else 4;  ( 条件式がある )
                                   ( real A,B,P; の後 )
(C) S := 0;
    for IA := -1 step 2.5 until 3.5 do
        S := S + I A;      ( real S,I,A; の後 )

```

と人間語の表現はほとんどそのままよい。

3. COBOL

アメリカ国防省が事務処理用言語として、COBOLを制定したが、各変数は途中一時使用のものに至るまでPICTURE指定を必要とするほか特定の記号の前後に空白がなくてはならぬとかあってはならぬとかいった制限が多い。上の例をCOBOLで書いてみると、

```

(A) COMPUTE C = 4 * B  ( CのPICTURE 指定
                        FILE RECORDの定義後 )

(B) MOVE 4 TO P
    IF A EQUAL TO B  ( P,A,B のPICTURE,
                      FILE RECORDの定義後 )
    MOVE 5 TO P.

```

```

(C)  MOVE 0 TO S
      PERFORM SUM VARYING A FROM -1
      BY 2.5 UNTIL A GREATER 35.
      SUM.
      COMPUTE S=S+A.  (S,AのPICTURE
                       FILE,RECORDの定義後)

```

4. PL/I

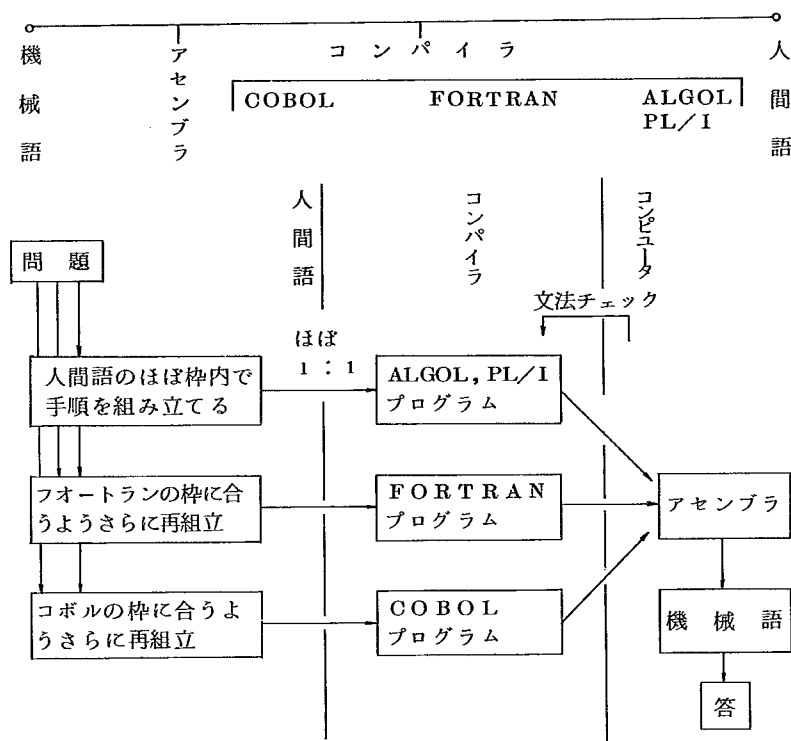
IBMはFORTRAN COBOL は前の世代の言葉であるとして、カードの欄にしばられない変数に限る所も式を使用できる。ブロック構造を持つその他の人間語に近い多様な表現を持つ言語として、PL/Iを開発した上の例をPL/Iで書いてみると、

```

(A) C = 4 * B;      (DECLARE (C,B) FLOAT; の後)
(B) IF A = B THEN P = 5;
      ELSE P = 4;    (DCL (A,B,P) FLOAT; の後)
(C) S = 0;
      DO IA = -1 BY 2.5 TO 35;
      |   S = S + IA;
      END;           (DCL (S,IA) FLOAT; の後)

```

これらの言語の状況は下図のようになるう。



(4) コンピュータの活躍範囲

よく設計計算から設計図作成まで計算機が実行した話をきくと人間の仕事は無くなってしまふのでないかと考える人が多い。マンマシンシステムにおいては人間は各ケースの帰納的処理によって一般原則化することができたものは、プログラム化し計算機側に持ち込むが、この原則化されたものは完全にカバーできないので例外がなくなることはあり得ない。人間はこの例外事項の処理を行なうのであって、一般原則的処理は計算機にまかせて何等さしつかえない。

前例だけにたよって行なっているような仕事は、一体人間が行なうべき

仕事であるかどうか問題である。

この意味で計算機の処理する仕事は次のように大別できよう。

1. 部分的事務処理

給料，償却，受払計算等

2. プロセス制御

定常的判断業務の自動化

3. 情報検索

文献検索，座席予約

4. 総合的事務処理

生産，資金，在庫管理

5. 技術計算

設計，計画

6. 経営科学計算

OR，シミュレーション，スケジューリング

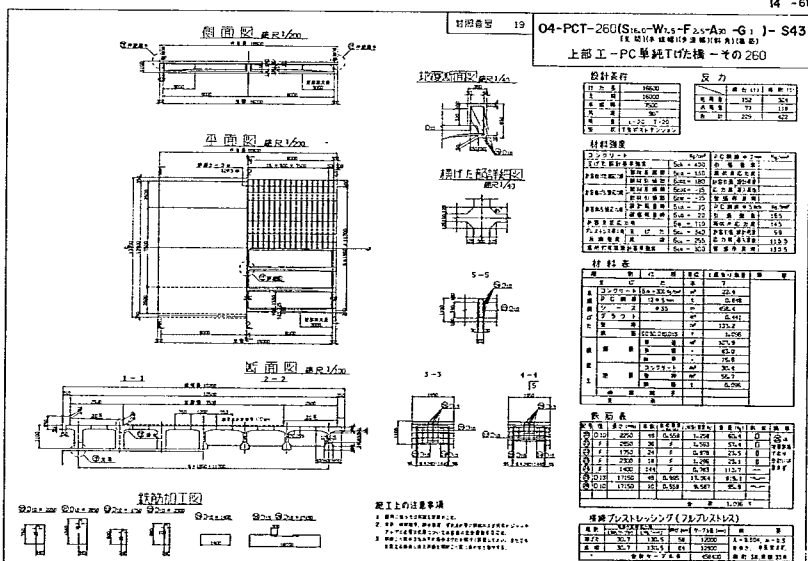
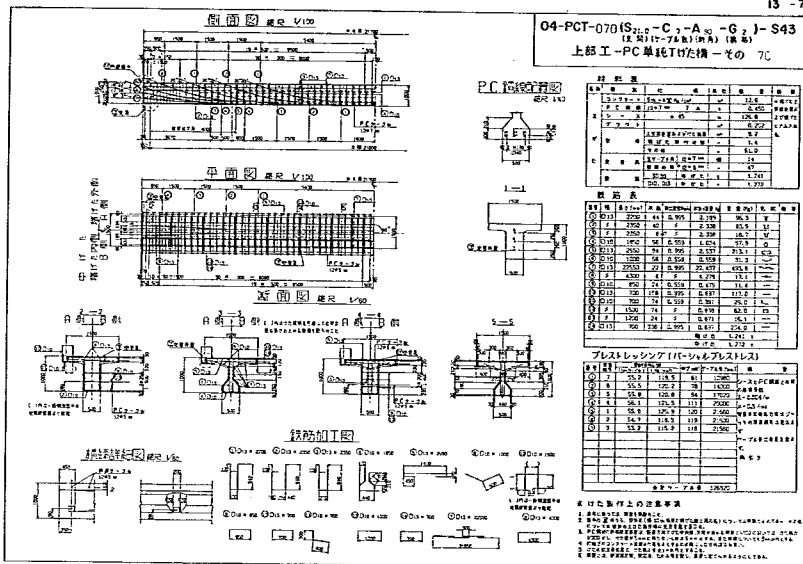
たとえば自動設計の例としてPC橋のスパン，全巾，歩道巾，斜角，橋格を与えて最適設計計算製図を行なった結果を図-1に示す。

この中の図形は最適設計計算結果によってあらかじめ示された作図方法に従って描き出したものであり，図中の数字は自から計算した結果を指定された場所，方向，大きさに描き出したもの，漢字も計算機の造り出したものである。

漢字については，へんづくり等に分解して，たとえば木ハムを1, 2, 3となづけ，ある領域を縦に2等分し，左に1を右に2を入れる動作をP(1, 2)とし，上下に2等分して上に1を下に2を入れる動作をQ(1, 2)としたとき

P(1, Q(2, 3))

によって松という字を造り出したという証明のために，すべての組み合わせを自動的に造り出させた結果を図-2に示す。



[3] プログラム例1 (ヨーロッパ向け書状)

ヨーロッパ向け書状送量F円は重量Wグラム(0.1グラム読みとする)により

10グラムまで 110円

10グラムを越える10グラムまでごとに 100円増
として計算される。

今N番, Wグラムの順に次々とデータを読み料金F円を計算し, N, W, Fの値を1行に印刷するプログラムを作れ, ただしN, Wが共に0のときデータの終了とみなす。

データは今の場合2番8, 4番6.2, 5番45.8とする。

この問題は極く普通にどこにも見られる種類の問題である。この問題を新入社員に処理させるためには, これをもう少し手順化し厳密な表現に変える必要がある。(これは日本語の範囲内の問題でプログラミングの難しさとは関係ない。よくプログラミングが難しいと云う人の頭の中に問題自身の難しさが描かれていることが多いのは注意を要する。)

① (定義) 番号をN, 重量をWグラム(0.1グラム読み)料金をF円とする。

② (報告書) 新しい紙の表頭に

(空白2)N (空白5)W (空白5)F

と印字し2行おくり以下1組づつ1行に印字したい。

③ (データ読み込み) データをN, Wの順に読む。

④ (データの終り?) もしN=0, W=0なら終り⑧へ。

- ⑤ (料金計算) そうでなければ F は, $W \leq 10$ なら 110 円, それ以上は $W/10$ の小数部分に 100 円を掛けたものが加算される。
 ($W \leq 10$ の場合も含めて) ちょうど 10 の倍数になった時に下のクラスに入れるためには最小単位 0.1 を W から引いたものにすればよい。
 整数部分をとる割算を $\div$ と書けば F は
- $$110 + (W - 0.1) \div 10 * 100$$
- とすればよい。
- ⑥ N, W, F の値を 1 行に印字
- ⑦ ③へ飛び越す
- ⑧ 終り

この問題を ALGOL で書けば次のようになる。

```
begin
  real N, W, F ;
  NEWPAGE;
  PRINTS ( '## W ##### W ##### F' );
  CRLF ( 2 );
  START: READ ( N, W );
  if N=0 W=0 ^ then go to END;
  F:=110 + (W-0.1) ÷ 10 * 100 ;
  PRINT I ( 3, N ); PRINT F ( 4.1, W );
  PRINT I ( 5, F ); CRLF ( 1 );
  go to START;
  END:
end;

2, 8, 4, 6.2, 5, 45.8, 0, 0,
```

ほとんど説明を要しないと思うが 2, 3 補足すると,

a ALGOL プログラムは全体を文の `begin` と `end` でつつんだも

- のであり、縦線は対応関係をはっきりさせるためにつけたにすぎない。どこで行を折ってもかまわないので人間が見やすい様にならねばよい。
- b `begin` に続く宣言部で対応する `end` までの間で使用する変数名等を実数なら `real` として宣言する。整数は `integer` であるが、3.0 と 3 を区別する必要もないので一般にはすべて `real` で差支えない。
 - c `NEWPAGE` によって改頁される。1つの文が終るとセミコロンをつける。
 - d `PRINTS` はその後のかっこ中の ' ' の中にはさまれた文字をそのまま印字準備する。空白は # と約束する。
 - e `CRLF` は以上の印字を実行し () 内の回数改行する。
 - f すべての文には : を介して名札をつけることができる。 `START` という物理的内容とは無関係だが、イメージの合ったものにしておく と便利である。
`READ` はデータテープ上の情報をかっこ内の名前に順に格納することを命じている。どのような形であっても宣言されたような形に自動変換するので先頭の 2 というデータは、`real` の `N` 次の 8 は `real` の `W` に対応してそれぞれ 2.0 , 8.0 と変換し格納される。
 - g `^` は共に成り立つ (アンド) の意で、これが成立すると `END` という名札の所へ飛び越し、そうでなければ次に行くことを命じている。
 - h 前にあげた手順の式通りで、この右辺の式の値を左辺の名前 `F` に代入 (`:=`) することを命じている。
 - i `PRINT I` は 2 番目のパラメタ `N` の値を 1 番目のパラメタの値有効 3 けたの整数タイプとして (4 捨 5 入して) 印刷準備を、`PRINT F` は 1 番目のパラメタ 4.1 により小数点上 4 けた `F1` けたの小数点タイプとして `W` の値を、それぞれ印字準備するよう命じている。値は次々と右へ並べて用意される。
 - j 先頭の `begin` に対応する `end` が来てプログラム終り。
 - k データはどのような形でもよくカンマで区切ってあれば良い。

このプログラムをPL/I に直すことは極めて容易である。

```
EUR: PROCEDURE OPTIONS (MAIN);  
    DECLARE (N,W,F) FLOAT;  
    PUT PAGE;  
    PUT LIST('    N        W        F');  
    PUT SKIP(2);  
    START: GET LIST(N,W);  
    IF N=0 & W=0 THEN GO TO END;  
    F=110+FIX((W-0.1)/10)*100;  
    PUT EDIT(N,W,F)  
        (F(3),F(8,1),F(10,1),SKIP);  
    GO TO START;  
    END:  
END;
```

2, 8, 4, 6.2, 5, 45.8, 0, 0

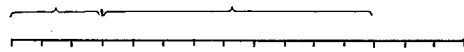
ほとんどALGOLと同じなので特に説明を要しないが、ALGOLと違う所を補足すると、

- a プログラムはPROCEDUREブロック(アルゴルのprocedoreと同じで、違うのは名前が先頭に来てコロンの入ることである。プログラム全体にEURと名がつけられる。また主プログラムに限りOPTIONS(MAIN)と書かれ、これによって自動的にこのプログラムが1回実行される。
- b 宣言は、DECLARE (DCLと略してもよい)の後に名前をカンマでつなぐ、そのすべてにFLOAT(アルゴルのrealにほぼ相当)という属性をつけるときはかっこでくくる。整数はFIXED
- c アルゴルの整数割算÷がないので、結果をFIXする。代入と比較の両方に=を用いる。
- d 出力は、はじめのかっこ内に出力したいもののリスト後のかっこは、フ

フォーマットでF(3)は3けた、F(8,1)は全部で8けた分のうち小数以下1けたSKIPは改行。

e データはアルゴル同様、カンマ又は空白で区切ってありさえすればよい。FORTRANで書くと次のようになる。

```
      INTEGER F
      WRITE (6,10)
10    FORMAT (1H1,2X,1HN,5X,1HW,5X,1HF/)
20    READ (5,30) N,W
30    FORMAT (I3,F10.1)
      IF (N.EQ.0.AND.W.EQ.0.0) STOP
      F=110+FIX((W-0.1)/10)*100
      WRITE (6,40) N,W,F
40    FORMAT (1H1,I3,F10.1,I5)
      GO TO 20
      END
```



2	8.0
4	6.2
5	45.8
0	0

規則が複雑だが、これを読むに必要なものの中から重要なものをピックアップすると次のようになる。

- a 名前の先頭の文字がI~Nで始まるものは整数、それ以外の文字で始まるものは実数扱いとなる。ここではFは整数扱いにしたいので特に宣言する。(名前を暗黙の宣言で扱うことは一見便利のようであるが、ブロックの構造をつくることができなくなる)
- b 出力文WRITEの最初のパラメタは、機番でいまラインプリンタを6

カードリーダーを5としておく、後のパラメタはフォーマットを示す文番号。

- c カードの第1欄にCを書くと、そのカードはコメント扱い、2～5欄に入るのは文番号で、今は20第6欄に何か入ると、そのカードは前のカードからの継続であることを示し、文は第7欄から記入する。また、1つの文が終わったら必ず行を変えなくてはならない。
- d フォーマットは入出力様式を示すもので出力の場合、先頭は必ずキャラクタ読み(文字としての読み込み)を行ない、それが1なら改頁、0なら2行改行、ブランクなら改行、+なら行送りをしない。
- e フォーマット中Hはその次のものを文字として扱い、その通り出力することを指示し、2HA=だとA=という2文字分である。したがって次に書く文字数を正確に数えないと、6HA43B=、によってA43B=、と考えるので区切りのカンマがなくなってしまうことになる。また、48HABO……と書いている途中でカードが一杯になって改行すると、大変なことになるので、区切りのカンマが来てからカードを変えなければならない。
- f 5Xはスペース5個、1はレコードの終りを示し、この後はまたフォーマットの先頭同様1なら改頁、0なら2行改行、ブランクなら改行……となる。この後にたとえばF1 0.1が来て、たまたま4683.2という値なら頭はブランクとなるので、これによって改行される。終りかっこが次に来れば改行実行
- g 第6欄に何か入っていれば継続行
- h READは第1パラメタが入力機番、第2パラメタがフォーマット文番号で入力すべき対象はかっこ後に書く。
- i フォーマットI3は3けたの情報
- j IFの後のかっこの中が真ならば、後の文を実行、そうでなければ次の行へ、.EQ. は等しいかどうか、.AND. は共に成り立つかどうかを問うている。これらの両辺は同じ型でないと誤りになるので、Wは実数であ

るから0でなく, 0.0としなければならない。

STOPによってプログラムの実行完了 ENDへ

k 代入は=によって行なう。

l ENDで終り

m データは入力フォーマット30の示す通り先頭の3けた分にNの値を,
4~12けた目までの間にWの値を入れなければならない。特にNについて最初のデータ2を1けた目に入れると, 200と読まれてしまうので注意を要する。

COBOLでは大変なことになってしまうので読み方の説明はあきらめて,
プログラムのみを示す。

IDENTIFICATION DIVISION.

PROGRAM-ID. MAILI.

ENVIRONMENT DIVISION.

INPUT-OUTPUT SECTION.

FILE-CONTROL

SELECT CARD-FILE ASSIGN TO IN.

SELECT PRNT-FILE ASSIGN TO OUT.

DATA DIVISION.

FILE SECTION.

FD CARD-FILE

DATA RECORD IS CARD-REC.

01 CARD-REC.

02 W PICTURE 9(4).9.

FD PRNT-FILE

DATA RECORD IS PRNT-REC.

01 PRNT-REC.

02 FILLER PICTURE X(120).

WORKING-STORAGE SECTION.

```

7 7   WORK   PICTURE   9 ( 5 ) .
0 1   PRNT-LINE
      0 2 LP-1   PICTURE   ZZZZ . Z
      0 2 FILLER PICTURE X(10)VALUE SPACE.
      0 2 LP-2   PICTURE   Z ( 5 ) .
PROCEDURE DIVISION.
      OPEN   INPUT   CARD-FILE,
           DUTPUT CARD-FILE.
YOMIKOMI .
      READ CARD-FILE AT END GO TO OWARI .
      COMPUTE WORK=(W-0.01)/10 .
      COMPUTE WORK=110+WORK*100
      MOVE WORK TO LP-2 .
      MOVE WAIT TO LP-1 .
      WRITE PRNT-REC FROM PRNT-LINE
           AFTER ADVANCING 2 LINES .
      GO TO YOMIKOMI .
OWARI .
      CLOSE CARD-FILE, PRNT-FILE .
      STOP RUN .

```

4 8.6

6 6.3

PROCEDURE DIVISION の始まるまでの23行はALGOLやPL/Iの2行目の1行分に相当すると云えば、その複雑さは想像できよう。これは欠くべからざる部分のみを示したので、一般のCOBOLのように、AUTHORなどいろいろ書けばその部分はまだまだ長くなってしまふ。後

の15行と合計38行という数字は他の12~13行の3倍である。一般にはALGOL PL/Iの4倍の長さで、しかも(の前は空白がなくてはならない。後はあってはならないといったつまらぬ規則が多い上に、END STOPをはじめ、約320の普通の英語の単語はほとんど特定の目的にしか使用できないので、名前は英語以外から選ばないと危険である。

[4] プログラム例2

データテープよりPを読み、Aが-15からもしP=0なら3.5きざみ、そうでなければ4.5きざみで125まで、次にA=150と280について、その後は前のAを倍々としながら、 $A < 1350$ である間Aと A^2 の値を1行にならべて印刷するプログラムを作れ。
ただし、今のデータとして2.5を入れておくものとする。

ALGOLではこの通りに書き下すことができる。

```
begin
  real A, P;
  READ(P);
  for A := -15 step (if P=0 then 3.5 else 4.5)
    until 125, 150, 280, A*2 while A < 1350 do
    begin
      PRINTF(4.1, A); PRINTF(7.2, A↑2);
      CRLF(1);
    end;
end;

2.5,
```

ALGOLでは変数の書ける所にはほとんど式を書くことができる。しかも式の中には条件式も含むのでstepとしてはPの値によってどちらかを選ぶ条件式を書いた。

for A:=.....do とは制御変数Aの値を次々と変えて, do の後の1つの文をくり返せといっており, ここでは3つの文をくり返したかったので文の
かって begin と end でつつんで1つの文としてある。

$A \uparrow 2$ は A^2 の意である。

if を()に包んであるのは分りやすくしただけで, かっては対応していれば余分にあっても差し支えない。

PL/IはALGOLとはほとんど同じである。たゞ条件式がないためにこの部分だけちょっと複雑になる。

```
PL:PROC OPTIONS (MAIN);  
  DCL (A,B,P) FLOAT;  
  GET LIST (P);  
  IF P=0 THEN B=3.5;  
    ELSE B=4.5;  
  DO A=-15 BY B TO 125, 150, 280,  
    A*2 WHILE (A < 1350);  
    PUT EDIT (A,A**2)  
      (F(7.1),X(2),F(10.2),SKIP);  
  END;  
END;
```

2.5

PROCEDUREはPROCと略してもよい。ELSEの前はALGOLとちがって;がなくてはならない。

ALGOLのfor をDOに, :=を=に, step をBYに, until をTOに, while の後をかつことつつみ, do を;におきかえ, DOに対するENDを考慮すれば, PL/Iにかわってしまう。

A^2 は $A ** 2$ と書く。

FORTRANはfor に当るものとしてDOがあるが, 制御変数は正整数に限り単純な使い方しかできないので大変なことになる。またWRITEに

式が書けないので一旦 $A ** 2$ を C か何かに受けなくてはならない。

```
A = 1 5.0
READ ( 5 , 1 0 ) P
1 0  FORMAT ( F 1 0.2 )
      A I = 4.5
      IF ( P .EQ. 0.0 ) A I = 3.5
4 0  IF ( A - 1 2 5.0 ) 5 0 , 5 0 , 6 0
5 0  B = A * * 2
      WRITE ( 6 , 7 0 ) A , B
7 0  FORMAT ( I H X , F 7.1 , 2 X , F 1 0.2 )
      A = A + A I
      GO TO 4 0
6 0  A = 1 5 0.0
      B = A * * 2
      WRITE ( 6 , 7 0 ) A , B
      A = 2 8 0.0
100  B = A * * 2
      WRITE ( 6 , 7 0 ) A , B
      IF ( A - 1 3 5 0.0 ) 8 0 , 8 0 , 9 0
8 0  A = A * 2.0
      GO TO 1 0 0
9 0  STOP
      END
```

2.5

条件式がないので P によって B の値を 3.5 あるいは 4.5 とするが、PL/I よりさらに複雑になる。

また for や DO の機能が多様でないので新らしく代入したり，あるきざみを加えては終りかどうかを判定し，終りでなければ頭に帰る形をとらざるを得ないで 26 行を要している。

COBOL は単に結果のみを示す。

IDENTIFICATION DIVISION.

PROGRAM-ID.

ENVIRONMENT DIVISION.

INPUT-OUTPUT SECTION.

FILE-CONTROL.

SELECT CARD-FILE ASSIGN TO IN.

SELECT PRNT-FILE ASSIGN TO OOT.

DATA DIVISION.

FILE SECTION.

FD CARD-FILE

DATA RECORD IS CARD-REC.

01 CARD-REC.

02 P PICTURE 9(4).99.

FD PRNT-FILE

DATA RECORD IS PRNT-REC.

01 PRNT-REC.

02 FILLER PICTURE X(120).

WORKING-STORAGE SECTION.

77 A PICTURE 9(4).9.

01 PRNT-LINE.

02 A PICTURE 9(4).9.

02 FILLER PICTURE X(2) VALUE SPACE.

02 C PICTURE 9(6).99.

PROCEDURE DIVISION

OPEN INPUT CARD-FILE, OUTPUT
PRNT-FILE.

START.

READ CARD-FILE

MOVE 4.5 TO B

IF P IS ZERO MOVE 3.5 TO B.

MOVE -1.5 TO A.

L1. COMPUTE $C = A * A$.

PERFORM PRINT.

COMPUTE $A = A + B$.

IF A NOT GREATER THAN 125

GO TO L1.

MOVE 150 TO A.

COMPUTE $C = A * A$.

PERFORM PRINT.

MOVE 280 TO A.

COMPUTE $C = A * A$.

PERFORM PRINT.

L2. COMPUTE $A = 2 * A$.

COMPUTE $C = A * A$.

PRINT. FROM PRNT-REC FROM PRNT-LINE.

L3. IF A NOT GREATER THAN 1350 GO TO 22.

OWARI. CLOSE CARD-FILE, PRNT-FILE.

STOP RUN.

(5) 情報化社会における情報伝達用語

昨年末日刊工業新聞の記者が突然面会を求めて来て、プログラム用語の得失を聞きたいと云い出した。何の為かと思って聞いてみると、アメリカでIBMがCOBOL FORTRANは前の世代の言葉であるから最早バックアップを中止し、新しい世代の言葉PL/Iに集中すると宣言したことに対し、ある会社が「お前が良いと云うから一生懸命社員のFORTRAN COBOL教育を推進して来たのに今さら何事か、このマンパワーの損失をどうしてくれる」という訴訟が起っていて記事にしたいから、各種言語の比較を聞きたいということであった。

そこで私はいろいろな比較論議を聞くよりも、一度身体で感じた方が早かろうと考えて(3)にあげたヨーロッパ向け書状の日本語の問題、ついで手順を示し、これを遅次計算機に分る言葉に書き直す方法として、まづALGOLをとりあげ、aからkまでの書き方に関する一般的注意を5分ばかり話してすぐその記者に、まったく独力でbegin から書き始めて貰った。

ほとんど余分の注意なしに、手順の日本語を逐条ALGOLに訳し終った記者にALGOLからPL/Iに移す原則a～eを話して、PL/Iに直して貰った。いづれも5分足らずであったが、次にFORTRANの番で、a～mの注意を30分位話して書き始めても細かい制約にひっかかったり、フォーマットがなかなかのみ込めなかったり、それよりも日本語の手順通りいなくて、フォートランの枠向きにこの手順を組み立て直すのに時間がかかって、1時間たってもなかなかENDまで来なかった。

最後にCOBOLは結果を見て貰うに止めたが、記者はこれらの相違が身にしてみた様子なので、次の表を渡してその日はお引きとり願った。

表とはIBMが研修に使用している「FORTRAN ユーザーのためのPL/Iの手引書」の中でPL/Iの特長としてあげている項目の一覧表である。

PL/Iの特長

1. カード欄にしばられない。
2. レーベルには数字でなく名前を用い：をつける。
3. 文は；で区切り続けて書いてよい。
4. 異なる種類のデータを式中で混合して使ってよい。
5. 単一変数に限られた所に式を使用してよい。
6. 添字式は（定数）＊（変数）±（定数）などという制限はない。
7. DOの制御変数は正整数でなくてよい。
8. ダイナミックな割りあてができる。
9. フォーマットはなくてよい。
10. COMMON不用。
11. DO END, BEGIN END, PROCEDURE ENDでくくったり，後2者ではブロックが作れる。
12. データはコンマかブランクで分ければよい。
13. DOの使い方がWHILEなど多様。
14. データの構造，配列の構造がある。

実はこのうち1～13はALGOLでは始めから当り前として使っていたことである。14のデータの構造はCOBOLから来たもの。配列の構造はPL/I独自のものであるが，配列の構造の一部や初期値設定ができるといった点は，建設省土木規格（KDK）ALGOLでは実現して現に使用しており，要するにIBMではFORTRANの欠点に気付きながらも面子にこだわって，ALGOLが良いとは云い出しかねてALGOLの特長に独自の特長を追加し，（たゞしALGOLの条件式はとり入れられてない）これをPL/Iと名づけたものと受け取ることができる。

マンマシンシステムで仕事を進めて行く上には人間側に荷重がかからない言葉である方が有利で，昔は計算機的能力が不足していたために，人間側が一生懸命4の代りに4.0とか，くり返しの制御を正整数に限るとか，添字式で3*Iはよいが，I*3はいけないとか，カードのどの欄に書かねばなら

ぬとか、入力フォーマットが F 1 0. 2 だからそのような位置にデータを書かねばならぬといった注意を払う必要があったが、今のように計算機的能力が上ってくると計算機の方もあまり有難がらぬどころか、わづらわしく思っているのではないかと考えられるふしさえ見受けられる。

我々人間側は問題の側にこそ全エネルギーを投入すべきであって、組み立てられた手順を計算機の分る言葉に直す部分に無駄な努力を払ってみても仕方のないことである。

また ALGOL にはアセンブラ等を procedure として組み込んだり、すでに中間語（リロケータブルバイナリな言語）にまで翻訳がすんで、システムに格納してある procedure を external としてその名前だけを指定すれば使用できる機能もあり、土木研究所ではさらにマイクロ回線によって遠隔地からプログラムテープを電送し万能入出力装置によって逐次実行中のプログラムに、文法チェックや短時間（10分以内）のプログラムが割り込んでその終了（10分以上のときは自動的に排除）と同時にバックグラウンドの仕事を実行する機能や本番計算と平行して図化機などを制御する機能を付加して、今プログラムテープを持ち込んで割り込みボタンを押そうとする者にとっては他の仕事を実行中の計算機があたかもその新しいプログラムを待っていたかのような感覚で使用できる所まで来ている。

昭和39年以来2日間ALGOL講習をくり返し、昭和44年末ですでに4,300人のプログラマが計算機と組んで、新しい観点から自分の問題を見つめ直している現状で、今後さらに多くの人達がこのALGOLあるいはPL/Iを習得して運動に参加されることを期待している。

参 考 文 献

1. 建設省土木研究所 電算速報 1～10
2. 中村 慶一 電子計算用語ALGOL 土木研究所資料
3. 中村 慶一 アルゴル・プログラミング入門 森北出版（1968）
4. 小岩・伊藤 やさしいプログラミング言語PL/I 日刊工業新聞社（1969）

§ 2. 自動図化とプログラム例 (電算速報10号の訂正)

中 村 慶 一

電算速報10号35頁のFORTRANプログラムは誤っているので、

D(7, N, X, Y, XE, YE) 径上の点XYよりXE, YEに向って

N1なら時計方向, N2なら反時計方向に円を描く

と訂正して書いたプログラムは次のようになる。

```
SUBROUTINE DRAW(DIA, CL3, S)
DIMENSION X(17), Y(17), XA(5), YA(5)
IF (DIA-24.) 50, 24, 27
24 X(1) = 0.
   X(2) = 0.
   X(3) = 20.
   X(4) = 20.
   X(5) = 30.
   X(6) = 30.
   X(7) = 150.
   X(8) = 150.
   X(9) = X(5)
   X(10) = 160.
   X(11) = 160.
   X(12) = 180.
   X(13) = 180.
   X(14) = 107.5
   X(15) = 107.5
   X(16) = 107.5
   X(17) = 180.
```

$Y(1) = 0.$
 $Y(2) = CL\ 3$
 $Y(3) = CL\ 3$
 $Y(4) = 0.$
 $Y(5) = CL\ 3 / 2. - 5\ 0.$
 $Y(6) = CL\ 3 / 2. + 7\ 0.$
 $Y(7) = Y(6)$
 $Y(8) = Y(5)$
 $Y(9) = Y(5)$
 $Y(10) = 0.$
 $Y(11) = CL\ 3$
 $Y(12) = CL\ 3$
 $Y(13) = 0.$
 $Y(14) = CL\ 3 / 2. - 1\ 0.$
 $Y(15) = Y(14) + 3\ 5. * 2.$
 $Y(16) = Y(14)$
 $Y(17) = 0.$
GO TO 100
27 $X(1) = 0.$
 $X(2) = 0.$
 $X(3) = 2\ 0.$
 $X(4) = 2\ 0.$
 $X(5) = 3\ 0.$
 $X(6) = 3\ 0.$
 $X(7) = 1\ 6\ 5.$
 $X(8) = 1\ 6\ 5.$
 $X(9) = X(5)$
 $X(10) = 1\ 7\ 5.$

```

X( 1 1 ) = 1 7 5.
X( 1 2 ) = 1 9 5.
X( 1 3 ) = 1 9 5.
X( 1 4 ) = 1 1 5.
X( 1 5 ) = 1 1 5.
X( 1 6 ) = 1 1 5.
X( 1 7 ) = 1 9 5.
Y( 1 ) = 0
Y( 2 ) = CL 3
Y( 3 ) = CL 3
Y( 4 ) = 0
Y( 5 ) = CL 3 / 2. - 7 7.5
Y( 6 ) = CL 3 / 2. + 5 7.5
Y( 7 ) = Y( 6 )
Y( 8 ) = Y( 5 )
Y( 9 ) = Y( 5 )
Y( 1 0 ) = 0.
Y( 1 1 ) = CL 3
Y( 1 2 ) = CL 3
Y( 1 3 ) = 0.
Y( 1 4 ) = CL 3 / 2. - 1 0.
Y( 1 5 ) = Y( 1 4 ) + 3 5. * 2.
Y( 1 6 ) = Y( 1 4 )
Y( 1 7 ) = 0.
100 SS = 0.0 1 * S
CALL D( 5 , SS , SS )
DO 15 I = 1 , 1 7
X( I ) = X( I ) * SS

```

```

1 5  Y( I ) = Y( I ) * SS
      CALL D( 3 , 2 )
      CALL D( 6 , 2 , X , Y , 2 )
      CALL LINE( 2 , 1 , 1 )
      CALL LINE( 2 , 2 , 2 )
      CALL LINE( 2 , 3 , 1 )
      CALL LINE( 5 , 4 , 2 )
      CALL LINE( 2 , 8 , 1 )
      CALL LINE( 2 , 9 , 2 )
      CALL LINE( 2 , 10 , 1 )
      CALL LINE( 2 , 11 , 2 )
      CALL LINE( 2 , 12 , 1 )
      CALL D( 7 , 1 , X( 14 ) , Y( 14 ) , X( 15 ) , Y( 15 ) )
      CALL D( 7 , 1 , X( 15 ) , Y( 15 ) , X( 16 ) , Y( 16 ) )
      CALL LINE( 2 , 16 , 1 )
5 0  CALL D( 1 )
      RETURN
      END

      SUBROUTINE LINE( M , N , P )
      COMMON X( 17 ) , Y( 17 ) , XA( 5 ) , YA( 5 )
      DO 110 I = 1 , M
      XA( I ) = X( I + N )
110  YA( I ) = Y( I + N )
      CALL D( 6 , M , XA , YA , P )
      RETURN
      END

```

このプログラムは土木技術 2 4 巻 3 号 1 1 8 頁にあるプログラムを完全にこの図を描くように直したものである。

なお 3 4 頁の ALGOL プログラム ももっと短いものにすることができる。
(2 号ペン使用の PEN 2 , 折線 PDRAW , 移動 PMOVE , 円を書く
PENX は external で使用できるものとする。ただし PDRAW は X , Y
配列から順に座標値をとり出すが値が 5,000 より大きいときはそこへペンを
上げて移動するものとする。

```
procedure DRAW ( DIA , CL 3 , S );
  real DIA , CL 3 , S ;
begin
  external PEN 2 , PDRAW , PMOVE , PENX ;
  real I , X 1 , X 2 , X 3 , X 4 ;
  real procedure DR ( X ) ; real X ;
    DR := X / S ;
  real procedure B ( X 1 , X 2 ) ; real X 1 , X 2 ;
    B := if DIA = 2 4 then X 1 else X 2 ;
  X 4 := DR ( B ( 6 0 , 6 7.5 ) ) ; X 3 := DR ( B ( 7 0 , 7 7.5 ) ) ;
  X 2 := DR ( B ( 9 0 , 9 7.5 ) ) ; X 1 := DR ( CL 3 / 2 ) ;
  Y := vec 0 , -X 4 , -X 4 , X 4 , X 4 , 10000 - X 1 , X 1 , 10000 - X 1 , X 1
    tor ;
  PEN 2 ;
  for I := 1 , -1 do
    begin real P , Q , R ; P := I * C ( X 4 ) ; Q := I * C ( X 3 ) ; R := I * C ( X 2 ) ;
      X := vec 0 , 0 , P , P , 0 , 10000 + Q , Q , 10000 + R , R tor ;
      PDRAW ( X , Y , 9 ) ;
      PMOVE ( 0 , D ) ;
      PENX ( 0 , -D - 1 0 , I * D ) ;
    end ;
end ;
```

ベクトル式のない一般のアルゴルの場合は，配列Yへの代入は，integer Jおよびreal AEの宣言を追加してから

```
J := 1 ;  
for AE:=0,-X4,-X4,X4,X4,10000-X1,X1,10000-X1,X1 do  
begin  
  Y(J) := AE;  
  J := J + 1 ;  
end ;
```

とすれば良いし，配列Xに対しては

```
J := 1 ;  
for AE:=0,0,P,P,0,10000+Q,Q,10000+R,R do  
begin  
  X(J) := AE;  
  J := J + 1 ;  
end ;
```

で良い。

これをPL/Iで書けば

```
DRAW: PROCEDURE(DIA, CL3, S);  
  DECLARE(DIA, CL3, I, X1, X2, X3, X4, S) FLOAT,  
    (PEN2, PDRAW, PMOVE, PENX) ENTRY EXTERNAL;  
  DR: PROC(X) FLOAT;  
    DCL X FLOAT;  
    RETURN(X/S);  
  END;  
  B: PROC(X1, X2) FLOAT;  
    DCL (X1, X2) FLOAT;  
    IF DIA=24 THEN RETURN(X1);  
    ELSE RETURN(X2);  
  END;
```

```

X4=DR(B(60,67.5)); X3=DR(B(70,77.5));
X2=DR(B(90,97.5)); X1=DR(CL3/2);
-BEGIN;
  DCL Y(9) INITIAL(0,-X4,-X4,X4,X4,
    10000-X1,X1,10000-X1,X1);
  CALL PEN2;
  DO I=1,-1;
    BEGIN
      DCL X(9) INITIAL(0,0,I*C(X4),0,1
        10000+I*C(X3),I*C(X3),10000+I*C(X2),
        I*C(X2));
      CALL PPRAW(X,Y,9);
      CALL PMOVE(0,D);
      PENX (0,-D-10,I*D);
    END;
  END;
END;

```

このALGOLプログラムをFORTRANで書くと、

```

SUBROUTINE DRAW(DIA,CL3,S)
COMMON FI,DIA,S
DIMENSION X(9),Y(9)
X4=DR(B(60.0,67.5))
X3=DR(B(70.0,77.5))
X2=DR(B(90.0,97.5))
X1=DR(CL3/2)
Y(1)=0.0
Y(2)=-X4
Y(3)=-X4

```

```

Y( 4 )=X4
Y( 5 )=X4
Y( 6 )=1 0 0 0 0.0-X1
Y( 7 )=X1
Y( 8 )=1 0 0 0 0.0-X1
Y( 9 )=X1
CALL PEN2
DO 10 I=1,2
IF (I.EQ.1) GO TO 20
FI=1.0
GO TO 30
20 FI=-1.0
30 X( 1 )=0.0
X( 2 )=0.0
X( 3 )=C( X4 )*FI
X( 4 )=X( 3 )
X( 5 )=0.0
X( 6 )=1 0 0 0.0+C( X3 )*FI
X( 7 )=C( X3 )*FI
X( 8 )=X( 6 )
X( 9 )=X( 7 )
CALL PDRAW(X,Y,9)
CALL PMOVE( 0.0 ,D)
10 CALL PENX( 0.0 , -D-1 0.0 ,FI*D)
RETURN
END

```

```
FUNCTION DR(X)
COMMON FI,DIA,S
DR=X/S
RETURN
END
```

```
FUNCTION B(X1,X2)
COMMON FI,DIA,S
IF(DIA.EQ.240) GO TO 200
B=X2
GO TO 210
200 B=X1
210 RETURN
END
```

§ 3. 電算室よりのお知らせ

(1) 図化機について

1. 図化機の概要

型	岩崎通信機製	渡辺測器製
	ロールタイプ	フラットベッドタイプ
プロット面 X軸	(ロールペーパーの長さ)	1100 mm
Y軸	250 mm	800 mm
速 度	400ステップ/秒	800ステップ/秒
	ステップ巾 0.1 mm	ステップ巾 0.1 mm
精 度	0.2 mm	0.15 mm
記録ペン	ボールペン	インクペン
		0.1 mm S型
		0.3 mm D型
		0.5 mm G型の3種

2. 図化機の基本命令語

ペン上げ	= U	= U
ペン下げ	= D	= S (S型を下げる)
	= S (=Dと同じ)	= D (D ")
	= G (")	= G (G ")
直 線	= F	= F
右廻り円弧	= R	= R
左廻り円弧	= L	= L
停 止	= H, (TM)	= H, (TM)
原点復帰	= Oは無視する	= O

これらの各種命令語はハード的に決められたもので出力されたデータは両プロッタに互換性を持たせた。

3. 基本手続

基本命令語に座標点として8進数を与えるので8進数に変換を行ったり，ペンの座標をおいかけたり，各種アルゴル手続きにて共通にもちいるもので，一般プログラマは4番のアルゴル手続を用いる。

3.1 PUP

命令語 = U を出力する

3.2 PFP

命令語 = F を出力する

3.3 PRP

命令語 = R を出力する

3.4 PLP

命令語 = L を出力する

3.5 PSP(I)

I : integer

命令語を I = 1 の時 = S , I = 2 の時 = D , さもない場合は = G として出力する。

3.6 PNP(X, Y)

X, Y : real

X, Y の値を 10 倍し 8 進整数に変換し出力する。

3.7 XYM(N, X, Y, UD, S, AX, AY)

N, UD, S : integer

X, Y, AX, AY : real

各種アルゴル手続きにペン先の状態を受け渡す。

X, Y はペン先の座標，UD は = 1 の時ペン先は下げの状態 = 0 はペン先は上げの状態，S は = 1 の時 S 型，= 2 の時 D 型 = 3 の時 G 型をそれぞれ使用している。

AX, AY は誤差修正用にもちいる。

N = 1 で読み込み，N = 0 で書き込む。

3.8 STRING(S, A)

S: string

A: integer array

長さNのstring Sをその内部コードに変換し, integer array A(1:N)に入れる。POUTSにて使用する。

4. アルゴリズム手続

4.1 PREADY(X, Y)

X, Y: real

現在のペン先のある位置を(X, Y)とする座標係を設定せよ。

従って現在ペン先のある位置を原点にとりたい場合はPREADY(0.0)とする。

4.2 PMOVE(X, Y)

X, Y: real

現在ペン先のある位置から点(X, Y)までペンを上げて移動せよ。

4.3 PSEN(X1, Y1, X2, Y2)

X1, Y1, X2, Y2: real

もし現在ペン先のある位置が(X1, Y1)ならただちに, さもなければその点までペン先を上げ移動した後に点(X2, Y2)まで直線を引け。

4.4 PSENX(X, Y)

X, Y: real

現在ペン先のある位置から点(X, Y)まで直線を引け。

4.5 PSENA(A, L)

A, L: real

現在ペン先のある位置からA度の方向角で長さL^{mm}の直線を引け。

4.6 PSENCN(N, X, Y)

N: integer

$X, Y: \text{array}(1:N)$

もし現在ペン先のある位置が $(X(1), Y(1))$ ならただちに、さもなければその点までペン先を上げて移動した後に N 個の座標点を逐次直線で結ぶ。

4.7 PSENVN(N, A, L)

$N: \text{integer}$

$A, L: \text{array}$

N 個の線分の方角とその長さを与え、これを逐次直線で結ぶ。

4.8 PEN($X1, Y1, X2, Y2, R$)

$X1, Y1, X2, Y2, R: \text{real}$

もし現在のペン先のある座標が $(X1, Y1)$ ならただちに、さもなければその点までペン先を上げ移動した後に点 $(X2, Y2)$ まで半径 R の円を R が正なら右廻り、負なら左廻りで画け。

4.9 PENX(X, Y, R)

$X, Y, R: \text{real}$

現在ペン先のある位置から点 (X, Y) まで半径 R の円を R が正なら右廻り負なら左廻りで画け。

4.10 POUTI(A, S, N, E)

$A: \text{real}$

$S, N, E: \text{integer}$

式 E の値を有効 N 桁で方向角 A 、大きさ S *mm* で整数タイプで画け。

4.11 POUTF(A, S, N, E)

$S: \text{integer}$

$A, N, E: \text{real}$

$N = a1 . a2$ の形である。

式 E の値を小数点以上 $a1$ 桁、小数点以下 $a2$ 桁で方向角 A 、大きさ S で画け。

4.12 POUTS (A , S , N , ST)

A : real

S , N : integer

ST : string

長さNの記号列STを方向角A , 大きさS mmで画け。

記号列中アルファベット以外はスペースに置き換える。

4.13 PEN 1

これ以後S型 (0.2 mm) のペン先を使用する。

4.14 PEN 2

これ以後D型 (0.3 mm) のペン先を使用する。

4.15 PEN 3

これ以後G型 (0.5 mm) のペン先を使用する。

4.16 PMARU (X , Y)

X , Y : real

中心の座標 (X , Y) を与えると , 現在ペン先のある位置から全円
を画く。

4.17 PDRAW (N , X , Y)

N : integer

X , Y : array

N個の座標点を逐次直線で結ぶ。ただしX〔1〕又はY〔1〕に

1 0 0 0 0を加えておくとその点まではペン先を上げて移動する。

現在の座標係と独立な座標係を設定することが出来る。

4.18 PEND

ペン先を原点復帰させ (TM) をパンチシテープをフィードする。

4.19 HALT

命令語 = Hを前後にFeedを入れて出力する。

4.20 FEED

紙テープを150字分フィードする。

5. そ の 他

- 5.1 XYMで最期値として $X=0$, $Y=0$, $S=2$, $UD=0$ を与えているのでPREADY(0.0)とPEN2を最初に発動しなくてもよい。
- 5.2 PSENA, PSENVN, POUTI, POUTF, POUTS で用いている方向角Aは反時計廻りを正とし単位は度である。
- 5.3 座標点はすべてmm単位とする。

(2) JOB割り込みについて

0. は じ め に

JOB割り込みの機能について東芝側と種々の折衝を重ねて来ましたが、この2月より我々が意図する機能をほぼ満足する状態にて作動始めましたので概要を紹介します。

1. JOB割り込みの概要

すでにデバックずみの実行時間が10分以上用するプログラムは一括してMT(unit 3)にスタックさせる。これらのJOBは逐次コアへ送られ結果はMT(unit 4)へ書き込まれる。(バックグランドJOB)このバックグランドJOBの実行中にデバックや10分以内のプログラムを割り込ませる。割り込みが発生するとバックグランドJOBはDISKに待避し割り込みJOB終了(割り込みJOBが10分たつとモニタが自動的にバックグランドJOBに戻す)と同時にバックグランドJOBを再開する。

2. タイプライタへのメッセージ

モニタが今どのJOBを実行中であることをオペレータに知らせるために種々のメッセージがタイプライタに印字される。

2.1 □□□NEW□DACK□ MOUNT

job name

割り込みボタンを押し割り込みが受け付けられると、タイプライタ

にその job name と共に印字される。

2.2 W 0 0 1

割り込みが受け付けられなかった時に、このメッセージが印字される。この時もう一度割り込みボタンを押して下さい。

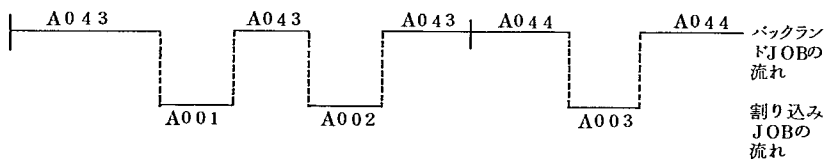
2.3 □□OLD□FILE□MOUNT

このメッセージが印字された時、割り込みJOBは終了し、バックグラウンドJOBを再開したことを意味する。

例

A043	バックグラウンドJOB開始
NEW DACK MOUNT	割り込み発生
A001	割り込みJOB名
OLD FILE MOUNT	バックグラウンドJOBへ戻る
NEW DACK MOUNT	
A002	
m 260	割り込みJOB A002にて
er-3	ERRORが起きた。
OLD FILE MOUNT	
A044	バックグラウンドJOBがA043からA044へ
NEW DACK MOUNT	
A003	
OLD FILE MOUNT	

JOBの流れ



3. 割り込み操作

割り込みボタンはPTRの右下についている白いボタンで、PTRにプログラムテープをセットし、コンソール板のENTRY KEY 2番がON(OFFの時割り込みは受け付けない)になってるか確認して、ボタンを押して下さい。タイプライタにメッセージが出るまで2秒ぐらいかかりますので、その間にボタンを何回も押さぬこと。テープの最後に/STOPコードを必ず入れて下さい。このコードを読み込まないと、OLD FILE MOUNTと印字しません。すなわちバックグラウンドJOBにもどりませんので注意して下さい。

4. 割り込みJOBの打ち切り

割り込みJOBは十分たつとモニタが自動的に打ち切りますが、(この時LPにM100を印字する)。それ以前に打ち切りたい時はコンソール板のENTRY KEY 1番をONにして割り込みボタンを押して下さい。この時タイプライタに次のような印字される。

```
JOB TERMINATION jobname
er-5
```

OLD FILE MOUNT

使用後ENTRY KEY 1番のresetを必ずして下さい。
バックグラウンドJOBが打ち切られることがありますので充分注意のこと。

5. JOB割り込みのためユーティリティルーチン

5.1 ISTK

MT(unit 3)にJOBをstackするルーチン

5.2 OSTK

MT(unit 4)に書かれたバックグラウンドJOBの結果をLPに打ち出すルーチン

5.3 AGIN

割り込みJOB中ハード・エラー, オペレーション・エラーが出たとき
割り込み時点のバックグラウンドJOBから再スタートさせるルーチン

5.4 SELJ

ハード・エラー, オペレーション・エラー等によりバックグラウンドJOB
の続行が不可能になった時, stack したJOBを希望のところまで
ポジショニングし, 再スタートさせるルーチン

6. 注 意 事 項

駒込にテープを送る方は, 操作指図に予想時間, 打切時間, デバッグか
本計算かを必ず記入すること。記入なきものはバックグラウンドJOBとし
て扱います。

今後時間外に特定の研究室に時間割りすることはいたしません。

(3) プログラム名簿

プログラム登録人員総括表

所 属	登録番号別	登録人員 (A)	登録人員 (B)	計
土木研究所本所	A	1 2 1		1 2 1
千葉支所	B	2 1 3		2 1 3
赤羽支所	C	1 0 2		1 0 2
篠崎試験所	D	2 4		2 4
鹿島 "	E	3		3
新潟 "	F	7		7
土木研究所計		4 7 0		
本 省	G	2 4 7	6 4	3 1 1
東北地建	K	1 0 3	2 5 8	3 6 1
北陸地建	L	1 3 8	5 9 2	7 3 0
関東地建	M	2 9 4	9 5	3 8 9
中部地建	W	1 4 0	2 4 5	3 8 5
近畿地建	P	1 1 4	2 5 1	3 6 5
中国地建	Q	1 1 3	1 6 7	2 8 0
四国地建	R	8 8	1 6 4	2 5 2
九州地建	S	9 4	1 5 5	2 4 9
地 建 計				
都府県関係	T	4 1 9		4 1 9
公団公社	U	6 7		6 7
そ の 他	V	2 1 3		2 1 3
合 計		2 5 0 0	1 9 9 1	4 4 9 1

A：土木研究所にて研修

1970，2月末

B：各地建等にて研修