

[04_03]九州大学大型計算機センター広報 : 4(3)

<https://doi.org/10.15017/1467976>

出版情報 : 九州大学大型計算機センター広報. 4 (3), pp.1-54, 1971-06-26. 九州大学大型計算機センター
バージョン :
権利関係 :

制御文のいろいろ

3月下旬に、ファイル装置を主とした機器構成の変更があり、それと共にソフトウェアの面でも大幅な変更がありました。センターでは制御プログラムの大幅なレベルアップに伴い、制御文もそれに合わせ能率の良いものと考え、一部変更いたしました。詳細については、すでにセンターニュース(№14)でお知らせしましたが、ここではいくつかの例をあげながら、制御文の使用法について説明いたします。

1. FORTRANプログラムの実行 (SSLを使用しない場合)
2. FORTRANプログラムの実行 (SSLを使用する場合)
3. ALGOLプログラムの実行
4. FORTRANプログラムとFASPプログラムを結合して実行
5. コンパイルだけしたいとき
6. カード出力があるとき
7. LP出力量を少なくしたいとき
 - (イ) FORTRANのマップを出さないとき
 - (ロ) プログラムのリストを出さないとき
 - (ハ) LIEDのマップを出さないとき
8. プログラムデバッグ中の制御文とデバッグ終了後の制御文

2. FORTRANプログラムの実行

科学用サブルーチンライブラリ (SSL) やセンターで作成したライブラリ等をプログラム中で使用しているとき。

3. ALGOLプログラムの実行

例2

```

1 | 1 2 | 5 6 7 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50
  |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
1 | $NO | } 受付で準備している
2 |     | XXXXXX | }
   |     | 受付番号 |
3 | $JOB | XXXXXXXXXXXX, XXXXXXXXXXXX, X, L=XXXXX
   | (1個以上のスペース) | 課題番号 | 登録名 | X: A, B, C, D の | 使用言語 |
   | (手入力) | | | (手入力) | (FORTRAN 手入力) |
   | | | | | (ALGOL 手入力) |
4 | $FORTRAN | MAP, NOOPT
   | (手入力) |
   | ($ALGOL) |
   |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
   | FORTRAN 原プログラム |
   | (手入力 ALGOL 原プログラム) |
5 | $LIEDRUN | LMAP
   |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
   | データ |
6 | $JEND
   |-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
1 | 1 2 | 5 6 7 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50

```

4. FORTRANプログラムとFASPプログラムを結合して実行する場合

科学用サブルーチンライブラリやセンターで作成したライブラリ等も使用できます。

例3

```

1  $NO
2  XXXXXX } 受付に準備されている
   受付番号
3  $JOB=XXXXXXXXXX,XXXXXXXXXX,X,L=FORTRAN
   (1個以上のスペース) 課題番号 登録名 プログラ種別 使用言語
   (X,A,B,C,D)
4  $FORTRAN_LMAP,NOOPT
   FORTRAN原プログラム
5  $FASP_LIST
   FASP原プログラム
6  $LIEDRUN_LMAP
   データ
7  $JEND

```

5. コンパイルだけしたいとき

例4

| | | | | | | | | | | | | | |
|---|--|---|---|----|-----|----|----|----|--|----|----|----|----|
| | 1 | 2 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 |
| 1 | \$No | | | | | | | | | | | | |
| 2 | XXXXXXXX | | | | | | | | | | | | |
| | 受け番号 | | | | | | | | | | | | |
| 3 | \$QJIBXXXXXXXXXXXX,XXXXXXXXXX,X,L=XXXX | | | | | | | | | | | | |
| | (140420) 課題番号 | | | | 表録名 | | | | ジョブ番号 使用装置 (A,B,C,D) A...I...R...X | | | | |
| 4 | \$FORTRAN | | | | | | | | | | | | |
| | (FORTRAN) | | | | | | | | | | | | |
| | (\$ALGOL) | | | | | | | | | | | | |
| | FORTRAN 原プログラム | | | | | | | | | | | | |
| | (FORTRAN, ALGOL 原プログラム) | | | | | | | | | | | | |
| | \$JEND | | | | | | | | | | | | |
| | 1 | 2 | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 | 55 |

6. カード出力があるとき

- (イ) QJOBカードにCP=nのパラメータを入れる。
- (ロ) データカードの最後に \$SYSPCH の制御文を入れる。

例5

```

1 2      5 6 7      10      15      20      25      30      35      40      45      50
1 $NO
2      XXXXXX
      受付番号
$QJOB_XXXXXXXXXX,XXXXXXXXXX,X,L=XXXX,CP=1000
      (1個址のスペース)登録番号      登録名      ジョブ種別 (X,A,B,C,D)が可能な入力      使用言語      カード打ち切回数
$FORTRAN_GO,MAP,LMAP,NOOPT
FORTRAN系プログラム
$GO
データ
$SYSPCH
$JEND

```

7. LP出力量を少なくしたいとき

出力量打ち切りなどのため、出力量を少なくしたいときがあります。その場合は計算結果の出力量を少なくすること以外に次のような方法があります。

- (イ) FORTRANのマップを出さないとき。

\$FORTRANのパラメータMAPを除く。

例6 \$FORTRAN GO (例1のとき)

または

\$FORTRAN (例2のとき)

(ロ) プログラムリストを出さないとき

\$FORTRAN, \$ALGOLの制御文にNOLISTというパラメータを追加する。

例7. 1 \$FORTRAN┘NOLIST

または

\$FORTRAN┘GO, NOLIST (※) (例1のとき)

例7. 2 \$ALGOL┘NOLIST (例2のとき)

(ハ) LIEDのマップを出さないとき

\$FORTRANまたは\$LIEDRUNのパラメータLMAPを除く。

例8. 1 \$FORTRAN┘GO, MAP (例1のとき)

例8. 2 \$LIEDRUN (例2のとき)

※ この場合はLIEDのリストも出力されない。

LIEDのリストを出さない場合として「\$LIEDRUN NOLIST」もありますが、これらは頁数の節約には全然なりません。

注：実行時のエラーが出たときで、FORTRANのマップ、LIEDのマップがないときはプログラム相談を受けに行っても門前払いになることが少なくないので、できる限りマップはとった方が良いでしょう。少なくともLIEDのマップは必ずとるようにしておいた方が良いでしょう。

8. プログラムデバッグ中の制御文とデバッグ終了後の制御文

実際にプログラムを書いて、計算機で実行させるとき、最初にコンパイルエラーがあり、それを修正し、次にテストデータをもって実行テストを行なうという順になるはずですが、普通よほど単純な論理でできたプログラムでない限り、期待する結果が出ないことの方が多い、つまり、論理ミスがあるわけです。従って、何回かのテストランをし、論理ミスを修正した後、期待している計算ができるわけです。この間使用する制御文は次のようにすると良いでしょう。

- デバッグ中

例1 または 2の制御文

- デバッグ終了後

例1 または 2 の制御文で \$FORTRAN のパラメータ NOOPT をとる。

ALGOL の場合はこの制御文のままです。

FORTRAN の場合 NOOPT を取る (OPT にする) と実行プログラムが最適化されるという利点があります。

最適化というのは実行速度を早くするために、配列の扱い等を効率良くコンパイルすることです。しかしこの場合の欠点として、FORTRAN の DEBUG 文が使用できないこと、実行時にエラーが起きたとき、発生個所がわかりにくくなります。