（第4回）

フォートラン・プログラミングにおける バッグ（誤り）とデバッグ（修正）

単純GO TO 文
ASSIGN 文
割り当て形GO TO文
計算形GO TO 文

研究開発部 磯 本 征 雄

1．はじめに

FORTRANプログラムの文は，1行ごとに上から下に順次並べられる。しかし，これらの文の実行順序は必ずしも物理的順序通りではない。この，文の実行順序を制御する文の1つがGO TO文である。GO TO文の使われたプログラムでは，実行の流れが分岐をもち，あるいは飛び越しがあり処理手順が複雑である。しかもプログラムの構造に直接関わるものなので，GO TO文の役割りは重要である。

ところが，GO TO文の誤りは他の文の誤りの場合とその症状において類似している。このために，テスト段階での誤り発見の時に，誤りの原因に対する推測を間違う場合も多い。しかし，第4節で示されるようにGO TO 文の誤りの症状は意外に単純なので（あるいはその病根は浅いので），誤りの発見は推察さえ確かであれば比較的容易であろう。

今回は，このようなGO TO 文に関する文法，及び誤りの症状について解説する。文法は，JIS-FORTRAN（水準7000）[1]にもとづき，誤りの症状は（NEAC）FORTRAN-700 [2] にもとづいて説明する。解説の方法・手順は前回までと同じである。

2．GO TO 文に関する文法

GO TO文は，実行の順序を制御する文の1つである。GO TO 文実行の後，このGO TO文で示された番号のついた文に実行の制御が移る。このGO TO 文による実行の制御の移る範囲は，1つのプログラム単位内である。したがって実行の制御のために使われる文の番号については，次のように定められる。

制御文に使われている番号は，その制御文を含むプログラム単位内の実行文の番号でなければならない

GO TO 文によって実行の制御が移る方法により，GO TO 文は次のように分類されている。

GO TO文（GO TO statement）は，

単純GO TO 文

割当て形GO TO文

計算形GO TO文

の３種類とする。

2.1. 単純GO TO文

単純 GO TO 文は，分岐をもたない，単なる実行制御の飛び越しのための文である。単純GO TO文の文法は次のように定められる。

単純GO TO文（unconditional GO TO statement）は，つぎの形とする。

GO TO k

ここで，kは文の番号とする。この文が実行されると，番号kを持つ文が，つぎに実行されるものとする。

例：単純GO TO文
GO TO 3

参 考 単純GO TO文は，無条件GO TO文ともいわれる。

2.2. 割当て形GO TO文とASSIGN文

割当て形GO TO文とASSIGN文は常に対にして用いられる。割当て形GO TO文実行後に実行の制御の移る文の番号は，予めASSIGN文で指定されていなければならない。例えば図2.1. のように使われる。

図2.1. 割当てGO TO文とASSIGN文の使用例

```
      ASSIGN 200  TO  I
        ⋮
      GO TO I, (100, 200, 300)
100   X=Y+Z
        ⋮
200   X=Y+2.0*Z
        ⋮
300   X=Y+3.0*Z
        ⋮
```

ASSIGN文は，割当て形GO TO文実行後に実行の制御が移るべき文の番号を指定する文である。ASSIGN文の文法は次のように定められている。

ASSIGN文（assigned GO TO statement）は，つぎの形とする。

ASSIGN k TO i

ここで，kは文の番号とし，iは整数型の変数名とする。

例：ASSIGN文
ASSIGN 2 TO I
ASSIGN 10 TO M

この文が実行されたのち，iの再定義がないならば，このASSIGN文を含むプログラム単位内にあって，変数名iを使う割当形GO TO文が実行されれば，その次ぎに実行される文は番号kを持つ文になるものとする。kは，このASSIGN文を含むプログラム単位内の実行文の番号でなければならない。

備 考 ここでいう再定義は，第１階での再定義とASSIGN文による再定義との両方を含む。

ASSIGN文が実行されると，その文に現われる変数名は，再定義されないかぎり割当て形GO TO文以外の文では引用してはならない。

一方，割当て形GO TO文はASSIGN文によって定義された変数を受けて，実行順序の制御をおこなう文である。実行の飛び越し先は2つ以上であり，常に実行の流れの分岐点になる。割当て形GO TO文の文法は次のとおりである。

割当形GO TO文（assigned GO TO statement）は，つぎの形とする。

GO TO i, (k_1, k_2…, k_n)

ここで，iは整数型の変数の引用とし，k_1, k_2, …, k_nはいずれも文の番号とする。

この文が実行されるときには，iの値にはすでに実行されたASSIGN文によってk_1, k_2, …, k_nのうちの一つが割当てられていなければならない。これによってiに割当てられた番号を持つ文が，つぎに実行されるものとする。

例：割当て形 GO TO文

文　例：

GO TO K, (3, 2, 5)

文の効果：

ASSIGN 12 TO I

GO TO I, (10, 11, 12, 13)

この二つの文はつぎの文と同じ効果を持つ：

GO TO 12

誤　例：

ASSIGN 10 TO M

M＝M＋1

2.3. 計算形GO TO文

計算形GO TO文は，整数型変数の値に従って2つ以上の分岐の内のいずれかに実行の流れの飛び越しをおこなう文である。計算形GO TO文の文法は次のとうりである。

計算形GO TO文（computed GO TO statement）は，つぎの形とする。

GO TO (k_1, k_2…, k_n), i

ここで，k_1, k_2, …, k_nはいずれも文の番号とし，iは整数型の変数の引用とする。

この文の実行によって，番号k_jを持つ文が，つぎに実行されるものとする。

ここで，jはそのときのiの値で，

$1 \leqq j \leqq n$でなければならない。

例：計算形GO TO文

文例例：

GO TO (11, 12, 13), j

GO TO (7, 82, 4), k

文の効果：

K＝2

GO TO (7, 8, 2, 4), K

この二つの文はつぎの文と同じ効果を持つ。

GO TO 8

上の例で計算形GO TO文が実行されるとき$K > 4$となるようなプログラムを書いてはならない。

2.4. 文法上の補足事項

文法規則は，具体的計算機システムとは別に定められたプログラミングのための一般的約束である。したがって，具体的処置の段階でその文の取扱いに計算機ごとに異なった独特の方法がある。（NEAC）FORTRAN-700では，GO TO文に関して次の補足事項がある。

文法上の補足事項について

以下に文法説明書に対する補足事項を述べる。

(1) 割当て形GO TO文において，整数型変数名にそのGO TO文中の文の番号の並びにない

文の番号が割当てられても，その割当てられた文の番号をもつ文に実行の制御が渡る。

エラー・メッセージは出されない。

(2) 計算形GO TO文において，整数型変数に範囲外の値が入れられていると，実行時にエラー メッセージが出されＵＥＰになる。

これらは，(NEAC) FORTRAN-700を使う場合に必ず知っておかなければならないことである。特に(1)についてはエラー・メッセージの出力がないので，もしこれがプログラムのバッグに絡む場合には，デバッグに手間取るかもしれない。

3. GO TO文に関するコンパイル時・エラー・メッセージ

本節では，(NEAC) FORTRAN-700におけるGO TO文に関するコンパイル時エラー・メッセージを示し，それらについての補足説明をおこなう。[2)]

3.1. WARNING

WARNINGは，文法規則としては必ずしも致命的誤りではないが，しかし実行時に到ってプログラムのアルゴリズムの上から致命的誤りであることが確認される場合が多い。誤り見逃しのまま，なまじリンクロード，実行へと処理手順が進行するだけに，プログラム作成時には注意を怠り勝ちである。したがって，WARNINGの意味を充分理解した上で，正しい処理を施しておかなければならない。以下に具体的メッセージとその例を示す。

メッセージ番号	説明
024	WARNING：THIS STATEMENT WILL NEVER BE EXECUTED ○ STOP, RETURN, GO TOまたは術IF文に続く最初の実行文文の番号が付けられていない。この実行文はそのままコンパイルされる。 備考： 文の番号のない実行文が２つ以上続く場合，最初の実行文にだけこのエラー表示がつけられる。

(例)

内部文番号	フォートラン文
	⋮
0004	GO TO 20
0005	I=I+2
	|

実行の流れは，内部文番号0004の所で文番号20へ飛ぶ。したがって内部文番号0005が実行されることはない。この時，論理的誤りの場合が多いので注意を要する。

061	WARNING：VARIABLE "name" IN ASSIGN NOT USED IN GO TO ○ ASSIGN文に現われた変数名が割り当て形GO TO 文で引用されていない。そのままコンパイルされる。

（例）　内部文番号　　フォートラン文

⋮

0007　　ASSIGN 30 TO J

⋮

この例において，Jを使った割り当て形GO TO 文がない。この時，ASSIGN文が不要であるか又は割り当て形GO TO 文が落ちているかのいずれかである。

3.2. FATAL ERROR

GO TO文に関する致命的エラーのメッセージは，主にGO TO文自体についての表現形式上の文法違反について出力される。ところが，GO TO 文は実行の流れを制御する文であるためにその効果・影響は必ず他の文に直接及ぶ。したがって，GO TO 文の文法違反はほとんどの場合他の文についてのエラー・メッセージの出力を並発する。

図3.1. GO TO文のFATAL ERRORとその副作用
GO TO文のみが誤りであるにもかかわらず，他のメッセージも出力される例。

```
                ⋮
0004             GO TO 100,
                ⋮
0009    100      X=Y+2.0
                ⋮

   ISN     DIAGNOSTIC MESSAGE
   0004    217  ILLEGAL UNCONDITIONAL GO TO STATEMENT SYNTAX
   0009    022  WARNING：UNREFERENCED STATEMENT LABEL "100"
```

図3.1. には，単純GO TO 文の誤りがWARNINGを並発した場合の例である。他に，GOTO文の誤りが他の文と重大なかかわりをもつ場合あるいは，逆に他の文の誤りがGO TO 文の誤りに影響する場合があるので，例えコンパイル時であっても実行の流れに注目しながらデバッグ（修正）すべきである。

以下，具体的なFATAL ERRORについてのメッセージの説明をおこなう。

195	ILLEGAL UNCONDITONAL GO TO STATEMENT SYNTAX ○ 無条件GO TO 文に文法違反がある。この文は削除されてコンパイルされる。プログラムの実行時に，この文に実行の制御が渡るとＵＥＰになる。

（例）　GO TO 100,

無条件GO TO 文の後に不要なコンマがついている。

196	ILLEGAL GO TO STATEMENT SYNTAX ○ 割当て形GOTO文または計算形GO TO文において，かっこ内の区切り記号に誤りがあるか，かっこのつけ方に誤りがある。この文は削除されてコンパイルされる。プログラムの実行時にこの文に実行の制御が渡るとＵＥＰになる。

（例）GO TO　（10, 11＊12), I

GO TO I, (13, 14

第１の例では，11＊12となりおそらく“, ”とすべき所を“＊”と誤っている。

第２の例ではGO TO 文最後の右カッコが欠けている。

197	COMMA MISSING IN ASSIGNED OR COMPUTED GO TO ○ 計算形GO TO文か割当て形GO TO文にコンマがない。 この文な削除されてコンパイルされる。プログラムの実行時にこの文に実行の制御が渡るとＵＥＰになる。

（例）GO TO (10, 11, 12) I

GO TO J (1, 2, 3)

第１の例では，GO TO 文最後のＩの前にコンマが欠けている。

第２の例では，Ｊの次のコンマが欠けている。

198	NON-INTEGER VARIABLE “name” IN COMPUTED OR ASSIGNED GO TO ○計算形GO TO文あるいは割当て形GO TO文に現われた変数名が整数型ではない。この文は削除されてコンパイルされる。プログラムの実行時にこの文に実行の制御が渡るとＵＥＰになる。

（例）REAL A

GO TO A, (10, 11, 12, 13)

計算形及び割り当て形のいずれのGO TO文においてもGO TO文にあらわれる変数は整数型でなければならない。上の例では実数形となったために誤りである。

215	SYNTAX ERROR IN ASSIGN STATEMENT ○ ASSIGN文において，整数型変数がないが，あるいは文法違反がある。この文は削除されてコンパイルされる。プログラムの実行時に文に実行の制御が渡るとＵＥＰになる。

（例）REAL A

ASSIGN 150 TO A

ASSIGN 125 TB J

例1ではASSIGN文中の変数が実数型であるために誤りである。これは整数型でなければならない。

第2の例ではTOをTBとパンチ・ミスしている。

216	UNDEFINED VARIABLE "name" IN ASSIGED GO TO ○ 割当て形GO TO文で引用された変数名がASSIGN文に現われていない。そのままコンパイルされるが，この文を含むプログラム単位はGOファイル上に出力されない。

（例）ASSIGN 125 TO I,

例ではASSIGN文の最後に不要なコンマがついたため誤りとなる。この時，この文は無視されコンパイルが続けられるためIがASSIGN文にあらわれなかったことになって後の割り当て形GO TO文でFATAL ERROR 251となる。

217	ILLEGAL REFERENCE TO FORMAT ○ ASSIGN文，GO TO文あるいは算術I文でFORMAT文の文の番号が参照されている。この実行文は削除されてコンパイルされるが，プログラムの実行時にこの文に実行の制御が渡るとUEPになる。

（例）

内部文番号	フォートラン文
0009	100 FORMAT (IH,: ……:)
0010	GO TO 100
	⋮

FORMAT文の文の番号はGO TO文で引用してはならない。

251	UNDEFINED STATEMENT LABEL REFERENCE TO "label" ○ 入出力文を除く実行文かデバッグ文で参照されている文の番号が実行文で定義されていない。 実行文の場合はその文が削除されてコンパイルされる。プログラムの実行時にその文に実行の制御が渡るとUEPになる。デバッグ文の場合はそのデバッグ文を含むプログラム単位はGOファイル上に出力されない。

（例）

内部文番号	フォートラン文
⋮	⋮
0004	ASSIGN 30 TO J
	⋮

この例において文の番号30は全くあらわれていない。

257	ILLEGAL VARIABLE NAME “name” ○　名前のくるべきところに英字名以外のものがある。 実行文の場合にはその文が削除されてコンパイルされる。プログラムの実行時にその文に実行の制御が渡るとＵＥＰになる。非実行文の場合にはその文が削除されてコンパイルされるが，その文を含むプログラム単位はＧＯ　ファイル上に出力されない。デバッグ文の場合にはそのデバッグ文は削除されてコンパイルされ，プログラムの実行時にそのデバッグ文により指定された範囲内の文に実行の制御が渡るとＵＥＰになる。

（例）　ASSIGN 20 TO ＊

ここでは，整数型変数のあるべき所にそれ以外のものがきている。

4．GO TO文の実行時に発見される誤り

GO TO文の誤りは，すなわちプログラム実行順序の誤りである。したがってGO TO文に関する誤りが発見されるのは，１つにはTRACE文等のデバッグ文や実行途中でのWRITE文による出力により，プログラム作成者が予定していたこと以外の実行順序で処理が進んでいることを直接発見する場合である。他の場合として，CUU-TIMEが予想と大きく違っていて又は（そして）出力結果の変数値が誤りであることから関接的に発見されることもある。

これらのGO TO文の誤りの結果発生する現象（症状）は，代入文，IF文及びDO文等の誤りの結果発生する現象（症状）と同じでか又は非常に似ている。このために，テスト段階で単に誤りの状態を推察する場合には予測が非常に困難である。ところが，実行順序を調べることを目的に実行途中の分岐点にWRITE文を挿入し，実行中の状態・条件と実行の流れとの相互の関係を順を追って調べさえすれば，GO TO文の誤りは容易に発見される。これらはデバッグ技法の基本として心得ておくべきだと思う。

以下に３種類のGO TO文について個々に誤りの状態と症状を調べてみよう。

4.1. 単純GO TO文の誤り

単純GO TO文は，その機能が単純でありしかも表現形式も簡単である。このために実行時の誤りも単純である。しかし，誤りの発見が簡単であるとは限らない。

図4.1には実行可能なプログラムの実例を示した。GO TO文は２つ以上の文の間の実行順序を結びつける。したがって，これら結び付きが誤まる場合として次の２つの場合が考えられる。

(1)　単純GO TO文において，行き先きの文の番号が誤りである。

(2)　単純GO TO文実行後に実行される文につけられた文の番号が誤りである。

これら(1)及び(2)の誤りの可能性は，例えばプログラム図4.1において，“GO TO 20” と “20 WRITE（6，110)” がプログラム作成者の意図どおりではなかった場合に相当する。特に，

図4.1. 単純GO TO文使用例

```
      CCC     UNCONDITIONAL GO TO STATATEMENT.
       C                 :
       C                 :
                      I = 0
                      GO TO 20
       C                 :
                         :
            20        WRITE ( 6, 110)
       C                 :
       C                 :
                      GO TO 1000
            30        WRITE ( 6, 120)
       C                 :
       C                 :
           110        FORMAT ( 1H3,▼UNCONDITIONAL GO TO (STEP  1)▼)
           120        FORMAT ( 1H3,▼UNCONOITIONAL GO TO (STEP  2)▼)
          1000        STOP
                      END
ISN       DIAGNOSTIC MESSAGE
          022 : WARNING : UNREFERENCED STATEMENT LABEL "30"
    実行時出力結果
        UNCONDITIONAL GO TO (STEP  1)
```

(3) 022 WARNING : UNREFERENCED STATEMENT LABEL "30" が出力されている,

ことから, 場合によってはGO TO 20はGO TO 30の誤りであるかも知れない。誤り(3)は誤り(1)の結果生ずることが多い。誤り(1), (2)及び(3)のいずれにおいても, その結果として

(4) 単純GO TO文実行後の実行の流れが異常であることになる。実行の流れの異常は, 幸運な場合には

(5) WRITE文による高速製表印字装置への出力順序が異常である。

ことにより知り得るであろう。ところが, 多くの場合

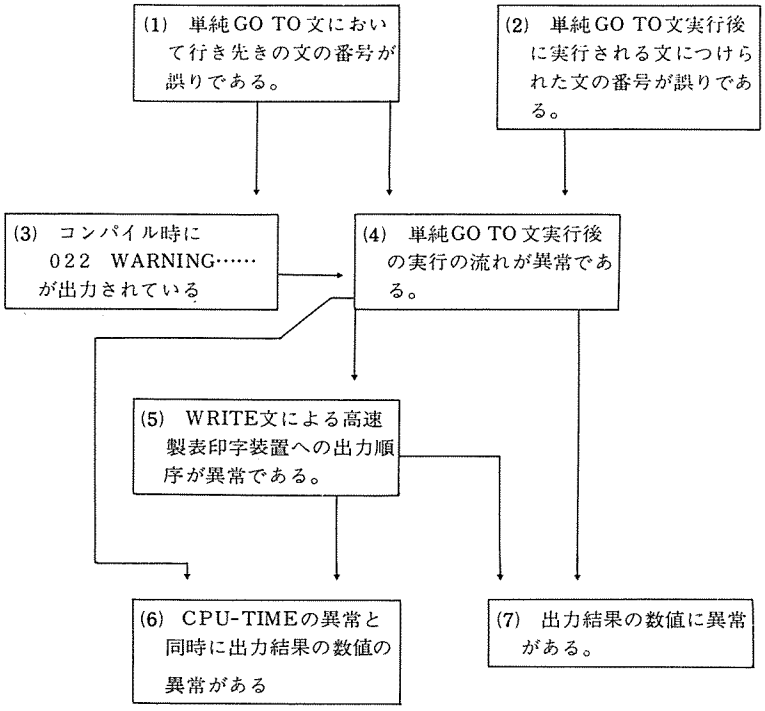
(6) CPU-TIMEの異常と同時に出力結果の数値の異常がある。

としてのみ発見される。あるいは

(7) 出力結果の数値に異常がある

としてのみ発見される場合も多い。以上を整理すれば, 図4.2のようにまとめられる。

図4.2. 単純GO TO文実行時の誤動作



4.2. 割当てGO TO文とASSIGN文の実行時に発見される誤り

割当てGO TO文とASSIGN文は常に対にして用いられるものである。したがってここでは割当てGO TO文とASSIGN文との実行時における振舞いについて同時に調べる。

ASSIGN文では，割当てGO TO文で引用される整数型変数の値を定義する。ところが，ASSIGN文で定義された整数型定数が割当てGO TO文で正しく引用されなければ，実行時に誤りとなる。この，ASSIGN文と割当てGO TO文間の変数の受渡しの誤りの原因として考えられることに2つある。第1に

(8) ASSIGN文で定義された整数型変数が割当てGO TO文実行前に再定義された，場合である。この場合には，実行時にエラー・メッセージとして次のものが出力される。

(9) 実行時・出力エラー・メッセージ

＊＊＊ MRM001I UEP 00000000 OP-CODE ERR=04

もちろん，実行は中断される。図4.3には，この誤り(6)及び(7)に関連した具体例を示した。図4.3の実例を見ることにより，上記説明の意味は充分わかるであろう。

次にASSIGN文で考え得る誤りとしては，

(10) ASSIGN文で定義された整数型変数の値に誤りがある。

場合である。以下，図4.4のプログラムを実例にあげながら考えることにする。この例では，内部文番号0001ではIは20に割当てられている。上記エラー(10)で述べたことは，この内部文番

図4.3. ASSIGN文及び割当てGO TO文の使用例
実行時にエラー・メッセージが出力されて，UEP（実行中断）
となる場合の例である。

```
CCC   ASSIGNED GO TO STATEMENT
      内部文番号              フォートラン文
                                :
      0001                ASSIGN 20 TO I
                C             :
      0002                I=30
                C             :
      0003                GO TO 1, (10, 20, 30)
      0004              10 WRITE (6, 110)
                C             :
      0005                GO TO 10000
      0006              20 WRITE (6, 120)
                C             :
      0007                GO TO 1000
      0008              30 WRITE (6, 130)
                C             :
      0009             110 FORMAT (1H3,▼TEST OF ASSIGNED GO TO(STEP 1)▼)
      0010             120 FORMAT (1H3,▼TEST OF ASSIGNED GO TO(STEP 2)▼)
      0011             130 FORMAT (1H3,▼TEST OF ASSIGNED GO TO (STEP 3)▼)
      0012            1000 STOP
      0013                 END

   実行時・出力エラー・メッセージ
     *** MRM00I UEP 00000000 OP-COPE ERR=04
```

号0001でのIへの割当てが，誤りである場合もあり得るということである。すなわち内部文番号0001はASSIGN 30 TO I かも知れない。

エラー番号(10)は，割当てGO TO文の側から見れば，

(11) 割当てGO TO文で引用される整数型変数の値に誤りがある。

と言いかえることができる。エラー番号(11)の結果として当然，

(14) 割当てGO TO文実行後の実行の流れの異常という状況をひきおこす。

さて，エラー番号(14)の原因となるものは，単にエラー番号(11)のみではない。

(12) 割当てGO TO文実行後に実行される文の文番号が誤りである。

場合においても，また

(13) 割当てGO TO文の中の行き先き指定の文番号が誤りである。

場合でも，誤りの現象としてはエラー番号(14)になる。但し，エラー番号(13)に対しては時にはエラー番号(3)

(3) 022 W ARING：UNREFERENCED STATEMENT LABEL “n n”

がコンパイル時メッセージ中に出力されることがある。エラー番号(14)は，残念ながら直接発見

図4.4. ASSIGN文及び割当てGO TO文の使用例
実行時にエラー・メッセージの出力のない場合の例。

```
CCC   EXAMPLE OF  ASSIGNED GO TO STATEMENT
      内部文番号              フォートラン文
                                 :
      0001                ASSIGN 20 TO I
                C
      0002                GO TO I, (10, 20, 30)
      0003          10    WRITE (6, 110)
                C               :
      0004                GO TO 1000
      0005          20    WRITE (6, 120)
                C               :
      0006                GO TO 1000
      0007          30    WRITE (6, 130)
                C               :
      0008                GO TO 1000
      0009          40    WRITE (6, 140)
                C               :
      0010         110    FORMAT (1H3,▼TEST OF ASSIGNED GO TO (STEP 1)▼)
      0012         120    FORMAT (1H3,▼TEST OF ASSIGNED GO TO (STEP 2)▼)
      0013         130    FORMAT (1H3,▼TEST OF ASSIGNED GO TO (STEP 3)▼)
      0014        1000    STOP
      0015                END
```

コンパイル時・出力の PIAGNOSTIC MESSAGE

```
   022  WARNING : UNREFERENCED STATEMENT LABEL "40"
```

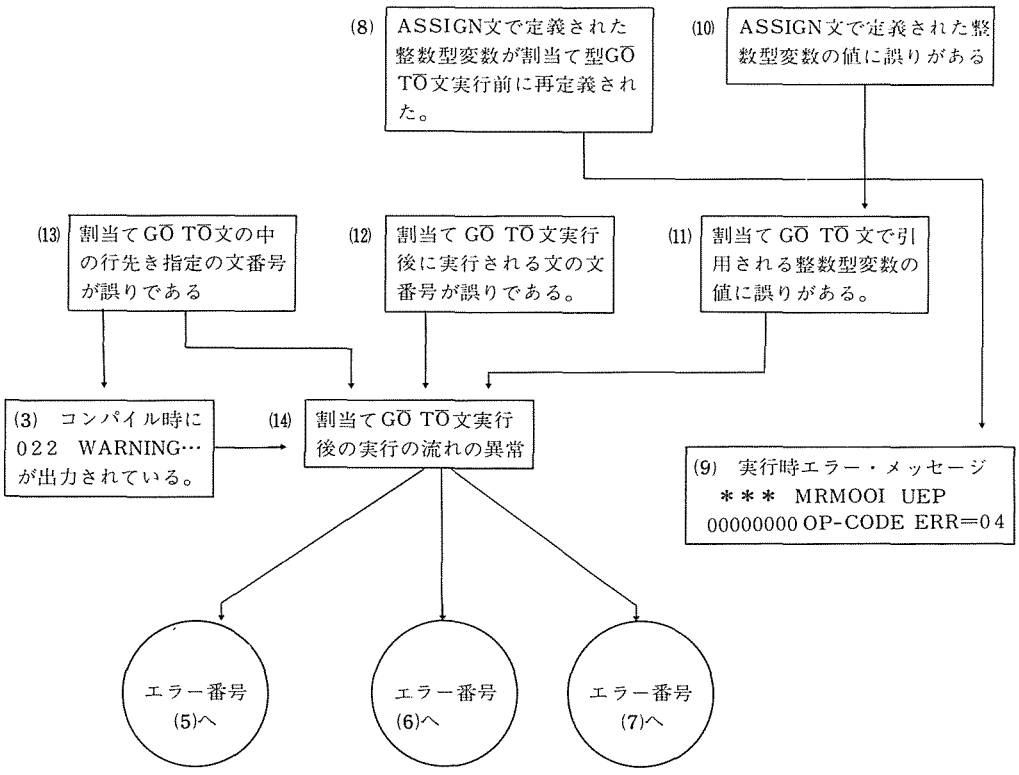
実行時・WRITE文による出力結果

```
   TEST OF ASSIGNED GO TO (STEP 2)
```

されることは少い。これらは，単純GO TO文の場合と同じく，エラー番号(5), (6)及び(7)に結びつき，一般に“プログラムは誤動作をおこしている”という言い方でしか表現できないような場合が多い。

以上の結果を流れ図にまとめたものが図4.5である。

図4.5. 割当て形GO TO文の実行時の誤り



4.3. 計算形GO TO文の実行時に発見される誤り

算術形GO TO文は，実行の行き先き文番号と整数型変数より成っている。図4.6には，計算形GO TO文の使用例を示した。以下これを見ながら説明する。このプログラム例は，一応実行はされている。しかし，もし図4.6が誤りであると仮定するならば，第1に

(15) 計算形GO TO文で引用される整数型変数Iの値を定義する文において，定義されたIの値が誤りである。

ために

(16) 計算形GO TO文で引用される整数型変数Iの値が誤りである。

結果となる。このために

(19) 計算形GO TO文実行後の実行の流れの異常を引きおこす。

しかし，図4.7に見るように

(20) 変数Iが計算形GO TO文で指定されている文番号の数より大きい

場合には，実行時に

図4.6. 計算形GO TOO文の使用例
実行時にUEP（実行中断）のおこらない場合の例である。

```
ISN      LABEL FORTRAN STATEMENT
         CCCC   COMPUTED GO TO STATEMENT.
0001            I=2
0002            GO TO (10, 20, 30), I
0003         10 WRITE (6, 110)
         C             :
         C             :
0004            GO 10 1000
0005         20 WRITE (6, 120)
         C             :
         C             :
0006            GO TO 1000
0007         10 WRITE (6, 130)
         C             :
         C             :
0008            GO TO 1000
0009         40 WRITE (6, 140)
         C             :
         C             :
0010        110 FORMAT(1H3,▼TEST OF COMPUTED GO TU(STEP 1)▼)
0011        120 FORMAT(1H3,▼TEST OF COMPUTED GŌ TŌ(STEP 2)▼)
0012        130 FORMAT(1H3,▼TEST OF COMPUTED GO TO(STEP 3)▼)
0013        140 FORMAT(1H3,▼TEST OF COMPUTED GO TO(STEP 4)▼)
0014       1000 STOP
                END
```

コンパイル時のエラー・メッセージ

```
022 WARNING：UNREFERENCED STATEMENT LABEL "40"
    NUMBER OF WARNINGS：00001
```

実行時のWRITE文による出力結果

```
TEST OF COMPUTED GO TO (STEP 2)
**405F OUT OF RANGE VALUE "4" IN COMPUTEED GO TO
       ERROR DETECTED IN MAINPG      0002   00003660
*****MRMOOI UEP 0007242 DEPENDENT
```

(21) OUT OF RANGE VALUE "nn…n" INCOMPUTED GO TO が出力される。

一方，誤り番号(19)は，

(17) 計算形GO TO文実行後に実行の制御が移る文の文番号が誤りである，

場合においても，また

(18) 計算形GO TO文の中の行き先き指定の文番号が誤りである。

場合も同様に生ずることである。但し，誤り番号(18)に対しては同時にエラー番号(3)が生じていることもある。

また，誤り(19)は単純GO TO文及び割当てGO TO文の場合と同様にエラー番号(5)，(6)及び

(7)として発見される。

以上を流れ図にまとめたものが，図4.8である。

図4.7. 計算形GO TO文の使用例
実行時にＵＥＰ（実行中断）をおこしてしまう場合の例である。

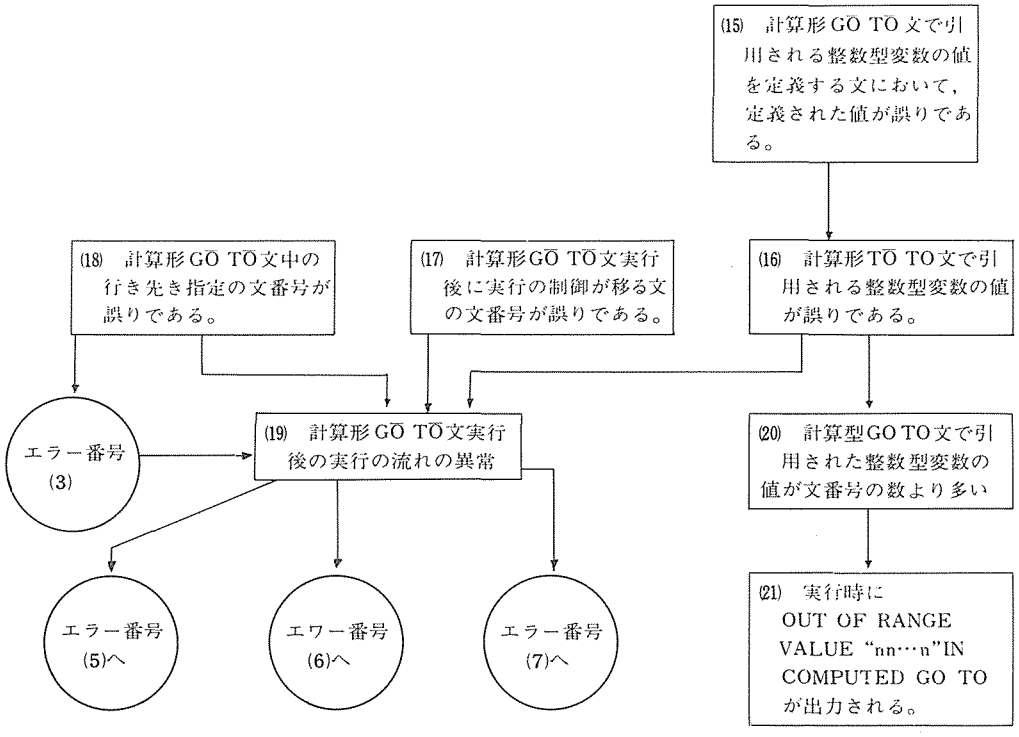
```
 ISN     LABEL FORIRAN SIAIEMENI
         CCCC    COMPUTED GO TO STATEMENT.
0001             I =4
0002             GO TO  (10, 20. 30), I
0003          10 WRITE ( 6, 110)
         C              :
         C              :
0004             GO TO 1000
0005          20 WRITE ( 6, 120)
         C              :
         C              :
0006             GO TO 1000
0007          30 WRITE ( 6, 130)
         C              :
         C              :
0008             GO TO 1000
0009          40 WRITE ( 6, 140)
         C              :
         C              :
0010         110 FORMAT (1H3, ▼TEST OF COMPUTED GO TO (STEP 1)▼)
0011         120 FORMAT (1H3,▼ TEST OF COMPUTED GO TO (STEP 2)▼)
0012         130 FORMAT (1H3, ▼ TEST OF COMPUTED GO TO (STEP 3)▼)
0013         140 FORMAT (1H3, ▼ TEST OF COMPUTED GO TO (STEP 4)▼)
0014        1000 STOP
0015             END
```

コンパイル時のエラー・メッセージ

```
      022  WARNING : UNREFERENCED STATEMENT LABEL "40"
           NUMBER OF WARNINGS : 00001
```

実行時のエラー・メッセージ

図4.8. 計算形GO TO 文の実行時の誤り



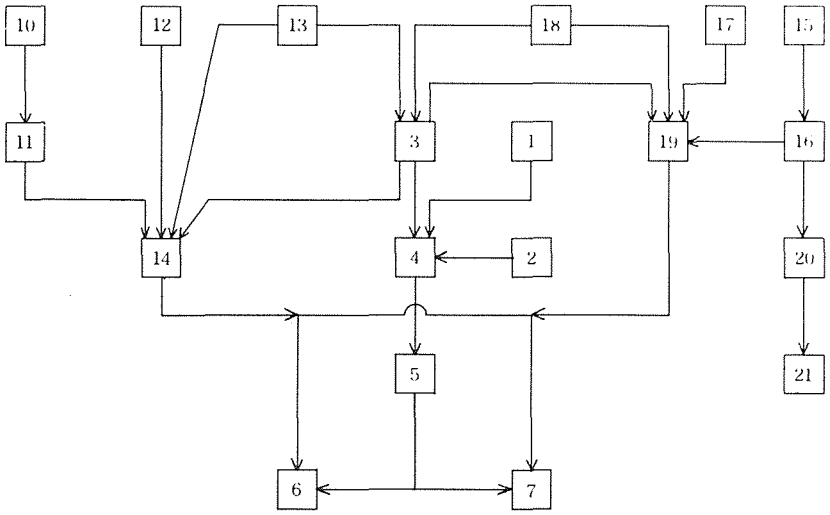
5. まとめ

GO TO文は，実行結果を直接に見ることができない。これらは，WRITE文による出力結果を見ることにより，正しい事を確認するほかない。したがって，GO TO文に関わるバッグを調べる上からは，GO TO 文実行の前後に必ず高速製表印字装置へ，その時の内部の状況を示す内容と共に出力するのが良い。ただこれを実行するだけで，GO TO 文に関わるデバッグは非常に楽になる。

これだけの事さえしておれば，あとはここに示された流れ図に従って，バッグの所存を追えば良い。このことは，具体的状況においては多くの場合，煩わしいために省略されがちである。しかし，現象だけを外から見るだけしていたのでは，算術式等の誤りと区別できないことが多いために，結局は時間の無駄になることが多い。

プログラミングの際には，実行の流れをも確認するためのデバッグ用の文を適宜押入することを推奨します。

付録　GO TO文実行時の誤りの流れ図



GO TO文実行時の誤り

単純GO TO文

(1) 単純GO TO文において，行き先きの文の番号が誤りである。

(2) 単純GO TO 文実行後に実行される文につけられた文の番号が誤りである。

(4) 単純GO TO 文実行後の実行の流れが異常である。

割り当てGO TO 文

(8) ASSIGN文で定義された整数型変数が割当て型GO TO 文実行前に再定義されている。

(10) ASSIGN文で定義された整数型変数の値に誤りがある。

(11) 割当てGO TO 文で引用される整数型変数の値に誤りがある。

(12) 割当てGO TO 文実行後に実行される文の文番号が誤りである。

(13) 割当てGO TO 文の中の行先き指定の文番号が誤りである。

(14) 割当てGO TO 文実行後の実行の流れの異常

計算形GO TO文

(15) 計算形GO TO 文で引用される整数型変数の値を定義する文において，定義された値が誤りである。

(16) 計算形GO TO 文で引用される整数形変数の値が誤りである。

(17) 計算形GO TO 文実行後に実行の制御が移る文の文番号が誤りである。

(18) 計算型GO TO 文中の行き先き指定の文番号が誤りである

(19) 計算形GO TO 文実行後の実行の流れの異常

(20) 計算形GO TO 文で引用された整数形変数の値が文番号の数より多い。

(21) 実行時に

OUT OF RANGE VALUE "nn…n"

が出力されている。

共通部分

(3) コンパイル時に

002 WARNING：UNREFERENCED STATDMENT LABEL "nnn"

が出力されている。

(5) WRITE文による高速製表印字装置への出力順序が異常である。

(6) GPU-TIMEの異常と同時に出力結果の数値の異常がある，

(7) 出力結果の数値に異常がある。

(9) 実行時エラー・メッセージ

** MRM001I UEP 00000000 OP-CODE ERR=04

が出力される。