The University of Osaka
Institutional Knowledge Archive

Title	ACOSシステム1000Fortran77(V)　エラーメッセージ入門
Author(s)	後藤, 米子
Citation	大阪大学大型計算機センターニュース. 1985, 59, p. 47-60
Version Type	VoR
URL	https://hdl.handle.net/11094/65670
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

https://ir.library.osaka-u.ac.jp/

The University of Osaka

資　料

ACOS システム1000 Fortran 77(V) エラーメッセージ入門

大阪大学大型計算機センター研究開発部　後　藤　米　子

1. はじめに

当センターでFortran 77のVモード（Fortran 77(V)と略す）を運用の標準として以来、この11月で1年余りが経過しました。現在では、Fortran 77(V)のジョブ件数は以前の標準であったFortranのジョブ件数を大幅に越え、Fortran言語の標準として利用者のあいだに確実に定着してきています。

本稿では、ACOSシステム1000（以下、S1000と略す）においてFortran 77(V)で記述されたプログラムを実行する場合に、初心者が経験するであろうと予想される典型的なエラーメッセージのうち、わかりにくいものをいくつか取り上げ、メッセージごとにその意味、原因、処置について述べています。原因としては典型的な場合を示していますから、解決できないケースもありうることをお断りしておきます。特に断らない限り、内容はバッチ処理モードとTSS処理モードに共通です。エラーメッセージは現在は英語で表示されており、日本語による表示機能はありません。

第2節では、計算機がジョブを処理する段階と検出されるエラーとの関係を概観します。第3節以降で、エラーの種類をジョブ処理の段階に対応して“ジョブの投入時”（第3節）、“翻訳処理時”（第4節）、“結合編集時”（第5節）、“プログラムの実行時”（第6節）に分類して説明し、第7節では、初心者が悩まされがちな“配列破壊”について別途、説明を補足しました。第8節ではデバッグのためのコンパイラオプションの概要と注意事項を述べていますので、あわせて参考にして下さい。

2. ジョブ処理の段階と検出されるエラー

計算機によるジョブの処理は、

(1) ジョブの投入時（バッチ処理）またはRUNコマンドの入力時（TSS処理）の処理

(2) 翻訳処理

(3) 結合編集処理

(4) 実行

の4段階を経る。S1000のバッチ処理では(2)、(3)、(4)のそれぞれをアクティビティ(activity)と呼び、例えば、(4)を“実行のアクティビティ”と呼ぶが、一方、TSS処理ではこの用語はもちいない。“ジョブ”とは、通常はバッチ処理における用語であるが、本稿ではTSS処理にお

いても使用することにする。

ジョブの結果を入手したあと、望ましい結果に至っていないと判断した場合には、上記の(1)から(4)までのどの段階にトラブルがあるのかを、まず、知る必要がある。表1に(1)から(4)の各段階における主な検査項目とエラーメッセージの表示箇所（バッチ処理の場合）およびエラーコードについてまとめている。エラーコードはメッセージの先頭に付加されているもので、エラーのレベルを示している。エラーのレベルには、(a)警告と(b)致命的の二つがある。(a)はその名のとおり注意を喚起することを、(b)は致命的なものであることを意味しており、(a)だけが検出された場合は次のジョブ処理の段階にすすむが、一つでも(b)が検出された場合は、その段階でジョブ処理を終了し、次の段階以降は無視される。エラーの検出においてバッチ処理とTSS処理の差はなく、万一、バッチ処理でエラーが起きTSS処理では起きないような場合（またはこの逆の場合）、あるいは、異なったエラーメッセージが出力されている場合は、後述の配列破壊が起きている可能性が大きい。

ここで、次のことを指摘して、読者の注意をうながしておきたい。もしも、エラーメッセージが全くただの一つも出力されていず、一見ジョブが正常に終了しているようにみえる場合でも、このことがすなわち“文法の仕様を満たした正しいプログラムであること”や“結果が正しいこと”を意味しておらず、単に“ジョブが異常終了しなかったこと”を示しているにすぎない、ということである。この理由については、第7節で述べる。

表1　ジョブ処理の段階と検出されるエラー

ジョブ処理の段階	検査項目	エラーの表示箇所	エラーコード	
			警告	致命的
ジョブの投入	JCLの構文 ファイルの存在の有無 $LIMITS文の制限値 RUNの構文(TSS処理)	JCLリスト内 （図1②参照）	—	—
翻訳	ソースプログラムの構文	ソースプログラム内 （図1④参照）	W	F
結合編集	プログラム単位間相互の参照関係	リンカレポート内 （図1⑤参照）	WR、他	ER、他
実行	プログラム・サイズ オーバーフロー、零除算 入出力関係 制限値（CPU時間、出力記録数）	アボートコードは図1③ これ以外は実行結果内注）	C	A

注）装置番号6または42の出力をファイルに指定した場合は、エラーメッセージもファイル内に出力される。
通常はこれらの出力先はプリンタである。

計算機から出力されるメッセージは、すべて大文字で表示されるが、本稿ではみやすさを考慮して、先頭の１文字のみを大文字で、残りを小文字で表記した。メッセージテキスト内の“英字名”は、実際にはプログラムで使用している変数名や配列名が表示されることを意味する。個別のエラーメッセージを書けない場合は、その現象を示す説明文で代用した。以下では、原因の(a)に対応する処置を(a)で示し、意味や原因を読むだけで処置が自明なものは省略した。

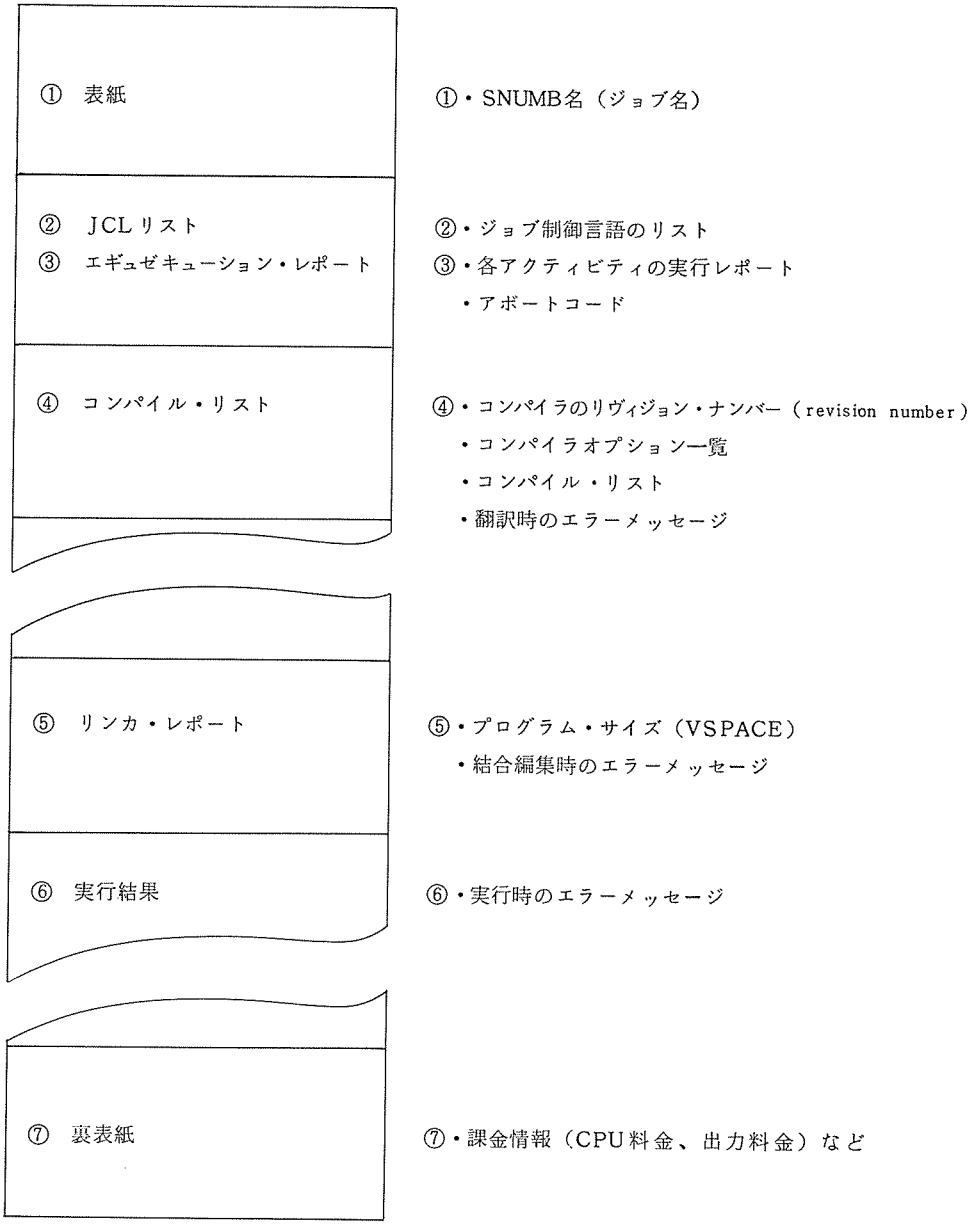

図１　計算機から出力されるリスト（バッチ処理）

3. ジョブの投入時

本節ではバッチ処理の場合のみを説明する。

Sysin job delete

【意味】システムに登録されずに、ジョブが削除された。

【原因】(a) ジョブ制御言語（Job Control Language ; JCLと略す）の構文に誤りがある（つづりの誤り、桁ずれ）。(b) 課題番号またはパスワードに誤りがある（(a)とおなじ）。(c) パーマネントファイルが存在しない（ファイル名のつづりの誤り、桁ずれ）。(d) パーミッションが、そのファイルでは許されていない。 (e) $LIMITS文で、そのジョブクラスの制限値を越える値を指定している。

【処置】(d) $JOB文で指定している課題番号とは異なった課題番号の下のパーマネントファイルを指定している場合に限りおこる。ACCESSサブシステムでパーミッションを変更する。

(e) $LIMITS文の第2, 第3パラメータに数値を指定していないかどうかを調べる。

Fortran 77(V)ではこの指定は不要である。

（誤）$ LIMITS 10,4000K,−2K,6000 （正）$ LIMITS 10,,,6000

課題番号に誤りがある場合以外では、通常の方法でリストを取出すことができるので、より詳細なメッセージを知ることができる。

sysinとは、ACOSシステムの用語で、system inputの略である。

Line too long（会話型リモートバッチ処理）

【意味】CARDINサブシステムにおいて、タブ（標準はコロン）を展開すると1行の長さが最大許容長の80桁を越えている。RUNコマンド投入時に“TAB CHARACTER SETTING？”の問いに対する応答が“タブ文字使用”（例えばN）のときに起きる。

【原因】(a) タブ文字の設定箇所の誤り。(b) ソースプログラムを$SELECTA文で指定している場合で、ソースプログラム中のコロン（例えば文字列、FORMAT文やDIMENSION文の中）もタブ文字とみなされている。

【処置】(b) ソースプログラムで使用していない文字をタブ文字として使用する。または、タブ文字は標準のままで、ソースプログラムの指定を$PRMFL文に変更し、タブの操作を受けないようにする。

4. 翻訳処理時

W 412 “英字名” is not defined

【意味】英字名の値が未定義である。

【原因】(a) 値を定義することを忘れている。例えば、A＝Bという代入文があるがBの値が与えられていない。またはDATA文の並びにBを入れることを忘れている。(b) 英字名のミスタイプがあるために、値が未定義となっている。

【処置】値を定義する。処置をしないと実行結果が異常になったり、ジョブがアボートする。バッチ処理、TSS処理ともに、データ領域は零で初期化されていないので、ごみがその値とみなされてしまうからである。

【注意】値が未定義の変数でも、COMMON文や引数の並びにあるときなどでは、このメッセージは出力されない。

W 242 A dummy word is inserted before "英字名" in common block "英字名"

【意味】COMMON文において、この英字名の前に意味のない1語を強制的に挿入した。倍精度、4倍精度の実数型および複素数型関係のデータは主記憶の偶数番地から格納しなければならないためである。

【原因】COMMON文中の並びが語長の長いものの順になっていない。

【処置】各プログラム単位の対応するCOMMON文の並びの型が一致している場合は、何の影響もないので必ずしも上の順序に並べかえなくてもよい。この対応が万一、不一致の場合は、値が異常になるなどの影響があるので、一致するように修正するか、順序を並べかえなければならない。

W 1470 Equality or non-equality comparison may not be meaningful in logical if expressions

---

【意味】論理IF文のなかで、浮動小数点表現をとるデータを対象に、等しいか否かの判定を行っている。(例) IF (A. EQ. B) GO TO 100 ただし、AとBは実数型の変数。

【原因】上記参照。

【影響】有限桁計算に起因する誤差により、制御の流れが変わることがある。

W 1457 "英字名" may not be redefined in call or abnormal function

---

【意味】引用しているサブルーチンまたは関数内で、この引数を再定義してはならない。

【原因】実引数の並びに、DO変数または整合寸法を示す変数がある。

【処置】手続き内において再定義している場合は文法違反であるので、仮引数を他の英字名に置き換えて、それを再定義するようにする。再定義していなければ無視してもよい。無視できる場

合であればメッセージを抑止するために、この変数を（　）でくくり式とみなされるようにする。

多数のコンパイル・エラーが出力される。

【原因】(a) ソースプログラムの形式に対する指定を誤っている。例えば、行番号付き、自由形式のソースプログラムを＄PRMFL文で指定している場合で、コンパイラオプションLNO、NFORMの指定を忘れている。または固定形式のソースプログラムをTSS処理で使用する場合に、コンパイラオプションFORMを忘れている。(b) ソースプログラムが自由形式であるにもかかわらず継続行の先頭の文字が＆となっていない。

Y 1

【意味】翻訳処理時のメモリ・サイズが不足しているために翻訳処理を続行できない。

【原因】ソースプログラムのステップ数が多いために、翻訳処理に要するメモリ・サイズが既定値で不足である。

【処置】バッチ処理では　＄FRT 77文の直後に＄LIMITS文で120Kないし150Kを指定する。CPU時間は既定値で十分なため、指定は不要である。

例　　＄　　LIMITS　　,120K

I 8	Run time exhauted
F 0	Memory address fault
F 7	Undefined op code
F 421	Internal compiler error
F 429	Compiler error --T code is illegal
T 14	Compilation terminated

【意味】コンパイラのエラーで、これ以上、翻訳処理を行えない状態となった。

【処置】発生したリストとソースプログラムをセンターに提出する。ソースプログラムがファイルに登録されている場合には、リードパーミッションを指定しておくこと。

5. 結合編集処理時

WR 13002 Reference “英字名” in “英字名” still unresolved

【意味】この英字名の手続き名がみつからない。

【原因】(a) サブルーチン副プログラムまたは関数副プログラムの定義を忘れている。

(b) 配列名につづりの誤りがあるために配列とみなされず、外部手続き名とみなされた。

(c) サブルーチンまたは関数を引用している名前につづりの誤りがある。(d) ライブラリを使用するプログラムで、$GO文にその指定を忘れている。例えば日本電気提供のMATHLIBを使用するときには、$GO文のオプション欄（16桁以降）にMLIBが必須である。

【処置】つづりの誤りや$GO文のパラメータを修正する。この手続きを引用している文（CALL文または代入文）を実行しないときは無視しても何の影響もないが、実行するときはこの文でアボート（第6節参照）する。

ER 13043 Definitions for “英字名” in modules “英字名” … assumed to be different one's

---

【意味】同じ名前をもつ手続きが定義されている。

【原因】(a) 同一の名前のサブルーチンまたは関数を定義している。(b) 手続き名が、6文字以内で一意になっていない。Fortran 77(V)では英字名は8文字まで許されるが、大域的要素を表すものは6文字までで一意でなければならない。(c) 手続き名と共通ブロック名とに同じ英字名を使用している。(d) 特に主プログラム名の重複を指摘されている場合は、ソースプログラムの最後の行がコメントになっている可能性がある。

【処置】(b) サブルーチン名を一意になるように変更する。(c) 手続き名と共通ブロック名が一意になるように変更する。(d) コメント行を除去する。

6. プログラムの実行時(1)

以下のエラーの大半は第7節で説明する配列破壊に起因している場合にも起こりうるが、この節ではほかの原因について述べる。入出力に関連した使用法については以前にセンターニュース[8]に解説しており、その使用法を守ればエラーをおこすこともないので今回は除外した。

Out of bound（バッチ処理）

【意味】プログラム・サイズが、ジョブクラスで許されている範囲外である。

【原因】プログラム単位を結合してみると、指定したジョブクラスの範囲を越えている。

【処置】ジョブクラスを変更する。プログラム・サイズはリンカ・レポート内のキーワードVSPACEの次に10進数（単位はK語）で示されている。

I8 Run time exhausted

【意味】CPU時間の制限値に達した。

【原因】(a) ジョブクラスの指定が妥当でない。(b) $GO文の直後に$LIMITS文の指定を忘

れている。(c) $LIMITS文またはRUNコマンドのTIMEオプションによるCPU時間の値が妥当でない。(d) プログラムの誤りで、無限ループに陥っている。

【処置】TSS処理での既定値は900秒と十分に長く、これで不足する場合には、システム全体の効率を良くするためにバッチ処理で実行させることが望ましい。

(b) $LIMITS文を指定する。$LIMITS文がない場合は、ジョブクラスにかかわらず、実行時のCPU時間の制限値は36秒である。

0> Output limit exceed (バッチ処理)

【意味】出力記録数の制限値に達した。

【原因】上のI8の説明で、"CPU時間"を"出力記録数"に読みかえたものに同じ。

【処置】(b) $LIMITS文のない場合は、ジョブクラスにかかわらず、実行時の出力記録数の制限値は5000記録である。

Q6 Abort end of file reading file code *nn*

【意味】ファイルコード*nn*による入力文を実行中に、データが終了した。

【原因】データ数が不足である。入力文の並びの数よりデータの数が少ない。

【処置】入力並びの数とデータの数をチェックする。データ数が不足していてもさしつかえないときは、READ(…, END = *m*) として、*m*で分岐先の実行文の文番号を指定して実行が継続されるようにする。

Fc *nn* does not exist

【意味】ファイルコード*nn*に対応する(入力)ファイルが接続されていない。

【原因】(a) バッチ処理ではファイル定義文を、TSS処理では、ファイルを指定するためのFILEオプションを忘れている。(b) $PRMFL文において、ファイルコードを1桁の10進の整数で指定している。正しくは、2桁で指定しなければならない。

(誤) $ PRMFL 9, R, S, … (正) $ PRMFL 09, R, S, …

【注意】Fortran 77(V)では、ファイルが接続されていない場合でも、プログラムの最初の入出力文がWRITE文であれば、テンポラリファイルを作成してジョブの実行を続行し、エラーメッセージを出力しない。テンポラリファイルであるからジョブの実行終了後には、利用できないことに注意すること。

Block serial error (TSS処理)

【意味】ファイルのブロック通し番号が乱れているために、OLDコマンドでカレント・ファイルにコピーできない。

【原因】ブレーク・キーを押してプログラムの実行を強制的に中断したために、この出力用のファイルにファイル終了記録が書かれていない。

【処置】最初から実行をやりなおす。

【注意】この場合でもLISTコマンドでファイルの内容を端末に表示できる。

F7　Undefined op code

【意味】計算機の命令として定義されていないものである。

【原因】プログラム単位間の情報の受け渡しの誤りによることが多い。(a) 手続きの実引数と仮引数の数が不一致である。(b) 手続きの実引数と仮引数の型が不一致である。(c) 関数手続きの型が、定義と引用で不一致である。(d) 未定義変数がある。(e) 未定義の手続きがある。(f) COMMON文の並びの型がプログラム単位間で不一致である。(g) IAPオプション指定時に、作業域が不足している。(h) 仮配列に対応する実引数が変数になっている（引用しているプログラム単位内で配列宣言を忘れているような場合）。(i) スイッチ変数の値が未定義である。(j) ENTRY文を含むプログラム単位内で仮引数となっている英字名の使用箇所に誤りがある。次の例においては入口がF1とF2の2つである。それぞれの入口からこの手続きに入ったときに、その実行範囲に現れてよい仮引数と同一の英字名は、表に示すとおりである。

例

```
 ・
 ・
 ・
X＝F1(XX,U)
Y＝F2(YY,V)
 ・
 ・
```

```
FUNCTION F1(A,B)
      ・
C＝exp
      ・
ENTRY F2(B,C)
A＝exp
      ・
      ・
      ・
```

入口＼英字名	A	B	C
F1	可	可	不可
F2	不可	可	可

【処置】(a), (b) オプションARGCHKを指定するとよい。(d) コンパイルリストにW 412が出力されていないかを調べる。(e) 結合編集時にWR 13002が、出力されていないかを調べる。(g) IAP(n)のnを大きくする。nの単位は1024回で既定値は5。

【付記】上記の原因の(d), (f)は、次節に述べる配列破壊を引き起こす原因ともなる。(g), (h), (i), (j)の原因で“F0　Memory address fault”が発生することがある。

7. プログラムの実行時(2)……配列破壊のある場合

利用者が経験する実行時のエラーのうち、配列破壊によるものがその大半を占めると言っても過言ではない。配列破壊とは、本稿ではプログラムの実行時に配列要素の添字の値が、宣言され

た配列の範囲を越えてしまうことを言うことにする。例えば、DIMENSION A(3)と宣言されているプログラム中の代入文 A(I)＝Bが実行されるとき、添字式 Iの値が0以下、または4以上となる場合である。言うまでもなく、このことは文法で禁止されており、文法には“添字式の値は、その配列宣言子の対応する下限以上かつ上限以下でなければならない”とある。

配列破壊は、代入文、READ文、WRITE文などの配列要素を引用している箇所で発生する。配列破壊が起きる直接の原因は、プログラムの論理ミスがほとんどであり枚挙にいとまがないが、以下では配列破壊が何を引き起こすかということを述べる。

Fortran 77 (V) では、一つの実行可能プログラムのデータ領域は、表2に示すように複数のセグメントに分割されており、各セグメントはその構成要素と個数が異なる。これらの個々のセグメントに対し、セグメントの範囲を越えたアクセス（書き込みおよび参照）を行おうとすると、ハードウェアのチェック機構が作用し、フォルトが発生するような記憶保護機構がはたらく。

表2　データセグメントの種類

種　　類	構　成　要　素	個　　数
①局所データ	局所変数、定数	プログラム全体で　1
②局所配列	局所配列	プログラム単位で　1
③無名共通ブロック	無名共通ブロック	プログラム全体で　1
④名前付き共通ブロック	名前付き共通ブロック	名前ごとに　1
⑤ＩＡＰ作業域	ＩＡＰ作業域	

配列破壊が発生するとすれば、そのセグメントは②、③または④である。これらのセグメントで、(a)不正な添字の値に対応する番地がこの配列の属するセグメント外となった場合には、セグメント外に対するアクセスを行うことになるので、前述の記憶保護機構によって直ちに“F0 memory address fault”が検出されプログラムは異常終了する。一方、(b)その番地がセグメント内となる場合には、同一セグメント内の他のデータを書きかえるか、不正な値を参照する。その値によって、幸いにも計算の途中で致命的と認識できるエラーが起きてアボートする場合もあれば、不幸にも致命的エラーとして検出してもらえず、誤った計算を行ったまま“normal termination”を出力して計算を終了する場合もあるので、くれぐれも注意を要する。

ついでながら、補足しておけば、Fortran(R)では、一つの実行可能プログラム全体が一つの“セグメント”を構成しており、前述の記憶保護機構は、プログラム全体の領域外をアクセスした場合にのみ適用されることになる。この差は、言語仕様の差によるのではなく、RモードとVモードという、オペレーティングシステムのアーキテクチャの差によるのである。

【配列破壊の実際】

図2に、入力データによっては配列破壊の起き得るプログラムの例を示す。このプログラムは、配列に予め初期値を設定しておき、添字の値を入力して、それに対応する配列要素の値をサブルーチンによって書き変えるものである。この例で問題になるのは、ラベル付き共通ブロックセグメントZであり、これは主記憶内でa番地からa＋5番地までの6語を占める。

```
①  COMMON /Z/L(3), I, J, K
②  L(1)=1 ; L(2)=2 ; L(3)=3
③  READ(5,*) I, J, K
④  CALL SUB
⑤  WRITE(6,*) I, J, K, L
⑥  STOP ; END
⑦  SUBROUTINE SUB
⑧  COMMON /Z/LL(3), II, JJ, KK
⑨  LL(II)=-1
⑩  LL(JJ)=-2
⑪  LL(KK)=-3
⑫  RETURN ; END
```

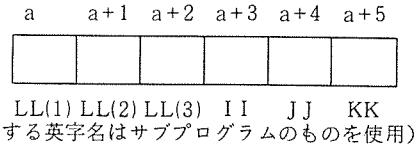

図2　プログラムとデータセグメント（一部）の関係

(1) 入力データが、0、2、3のとき（(a)の場合）

⑨の実行時にLL（II）の添字の値が0になって、（a－1）番地を示すので、セグメント外となり、“F0 Memory address fault”が発生し、プログラムの実行は直ちに終了する。

(2) 入力データが、1、2、4のとき（(b)の場合）

⑪の実行時にLL（KK）の添字の値が4になって、（a＋3）番地を示すので、この番地に格納されている変数IIの値を－3に書き変える。LL(3)の値はそのままで変化せず、プログラムは“normal termination”で終了する。

【配列破壊によるメッセージ】

配列破壊による典型的なアボートメッセージには次のものがある。

F0　Memory address fault

Class 2 security fault

これら以外にも、実行結果が異常である；バッチ処理では正しいようであるが、TSS処理ではエラーが発生する（またはこの逆）；オーバーフロー、零除算、SQRTの引数が負など、起こるはずがない値の異常；FORMAT文にエラーがある（FORMAT文を調べると正しい）；など予想できないようなエラーが起きる。

配列破壊を引き起こす直接の原因については、前節の“F7 Undefined op code”と重複するので、それを参照していただきたい。配列破壊の起きた配列要素とその不正な添字の値は、コンパイラオプションのSUBCHK（第8節参照）を指定することによって、知ることができる。

なお、文字位置式の値が不正の場合にも配列破壊と同様のエラーが起きる。

8. デバッグのためのオプション

デバッグのためのコンパイラオプションには表3に示すものがあり、それぞれの目的に応じて指定するとよい。表に示すもの以外に、FDSおよびFTIMERがあるが省略した。これらは通常は休止状態となっているため、使用時には利用者が指定しなければならない。

必要であれば、表中のものを同時に指定してもよい。

表3　デバッグ用のコンパイラオプションとその機能

オプション名	機　　能
SUBCHK	配列要素の添字の値を検査する。
ARGCHK	実引数と対応する仮引数との、数および型を検査する。
FLTCHK	零除算、指数部の桁あふれ、アボート(F0、F7)注1)などの発生箇所をソースプログラムのオルタ番号で表示する。
MAP	英字名の一覧（型や配列の大きさ）を出力する。
XREF	プログラム単位ごとの以下の一覧を出力する。 ・実行文の文番号によるプログラムの制御の流れ。 ・英字名および文番号とその参照されている箇所のオルタ番号。注2)

注1)　FLTCHKオプション下では、アボートコードはQ6に変更される。
注2)　ソースプログラムの各行に付加される通し番号で、行番号なしの場合は、プログラム単位ごとにその先頭の行を1、増分を1とし、行番号付きの場合は、行番号をそのまま採用する。

使用上の注意事項を以下に示す。

- 最適化オプション（OPT＝2またはOPT＝3）とSUBCHKまたはARGCHKを同時に指定した場合は、最適化オプションは無効となり、OPT＝1が指定されたものとみなされる。この旨、リストにメッセージが表示される。
- SUBCHK、ARGCHK、FLTCHKオプションは、実行可能プログラム全体に指定されていなければ、検査は完全ではない。
- SUBCHKオプションは、プログラム内における配列要素の引用・参照のすべてにわたり、添字の値を検査するので、実行時のCPU時間は10倍近くに増加することも珍しくない。デバッグが終了ししだい、忘れずに除去すべきである。
- 最適化オプション指定下でFLTCHKオプションを使用すると、エラーの発生に対応したソースプログラムの位置が正しく出力されない場合がある。
- SUBCHKオプションは次のものについては検査を行わない。

擬文字長指定の文字型の配列	CHARACTER C(5)*(*)
擬寸法仮配列	DIMENSION A(10, 2, *)
文字位置式の値	

・ 下記のプログラム[注]は、慣習上許されているがSUBCHKオプションからのメッセージが出力されてしまう。SUBCHKオプションの視界が1つのプログラム単位内であり、プログラム単位の相互の連絡の情報を参照していないからである。

```
DIMENSION A (10)          SUBROUTINE SUB (AA)
...                       DIMENSION AA (1)
CALL SUB (A)              ...
...                       I = 8 ; AA (I) = Z
END                       END
```

9. おわりに

本稿では、初心者の誰もが経験するような典型的なエラーをとりあげ、その対処法について解説しました。初心者のかたのなかには、文法上の誤りや操作上での誤りはすべて計算機側で検出されるものという勝手な推測をしていたり、“ normal termination ”で終了していればプログラムは正しいもの、と勝手に決めこんでいるかたが少なからずみうけられます。しかし、本稿の第7節を読まれた方は、もはや“エラーがあったら、計算機が教えてくれるさ”と言う安易な考えは捨てられたことでしょう。プログラムにエラーのある場合には、何が起きるかはまったく予期できません。計算機が格段に大型化、高速化した現在でも、文法を忠実に守って正しいプログラムをかくこと、正しい操作法に従って使用することが、大切な心得であることには変わりありません。S1000から出力されるエラーメッセージはすべてマニュアル[2]の付録またはマニュアル[3]に掲載されていますから、それらを参照して下さい。

本稿は従来の講習会で使用していた資料をもとに、その一部を抜粋し加筆したものです。これまでに有益なコメントを寄せていただきました、現・元プログラム相談員のかたがたに感謝いたします。

注) このようなプログラム作成技法を勧めているわけではない。

参考文献

1) AGB01－4　FORTRAN77　言語説明書，日本電気（1984）。

2) FGB27－1　FORTRAN77(V)　プログラミング手引書，日本電気（1985）。

3) FCB23－1　エラーメッセージ／アボートコード説明書（プログラム管理編），日本電気（1985）。

4) 藤井：新しいオペレーティングシステムの仮想記憶機能（第2章および第3章），大阪大学大型計算機センターニュース，No.39（1980）。

5) 大中，後藤：FORTRAN77　概説(1)，大阪大学大型計算機センターニュース，Vol.11，No.3（1981）。

6) 大中，後藤：FORTRAN77　概説(2)，大阪大学大型計算機センターニュース，Vol.11，No.4（1982）。

7) 大中，後藤：FORTRAN77　概説(3)，大阪大学大型計算機センターニュース，Vol.12，No.1（1982）。

8) 後藤，大中：FORTRAN　入出力文からみた記録の長さ，大阪大学大型計算機センターニュース，Vol.13，No.2（1983）。

9) FORTRAN77とFORTRANのバージョンアップ，大阪大学大型計算機センター速報，No.114（1984）。

10) ジョブ制御言語の手引（第4版），大阪大学大型計算機センター（1985）。

11) TSSの手引，大阪大学大型計算機センター（1985）。

12) 泉谷，他：統合アレイプロセッサについて，大阪大学大型計算機センターニュース，第45号（1982）。