インタプリタ言語処理機能を有する アプリケーションプログラム

西松建設株式会社　正員　根本隆栄

1．はじめに

当社においては，構造解析プログラムに代表される汎用型の大型プログラムは，年々レベルアップが行われ多くの機能が付加されてきた．反面，プログラムの規模が大きくなり，簡単な修正要求に対しても迅速な対応ができにくいプログラム構造になってきている．一方，設計業務の省力化の要請は増々強くなり，例えば，得られた解析結果を更に設計者固有の設計業務にすぐに役立つように加工して出力したいという要望も少なくない．これらの中には汎用的なものもあるが，多くは当該の設計業務だけに使用するものである．したがって，これらの要望に対して逐一サブルーチンを付加する形で対応していては，プログラマの負担も膨大であり，コンピュータ資源も無駄となる．

筆者は，数年前ＢＡＳＩＣ言語に類似したプログラム文（以下言語と記す）を解釈・実行するプログラム（以下インタプリタと記す）を開発していた．この度，このインタプリタと既存のアプリケーションプログラムとを結合し，両者で相互にデータの転送を可能にすることにより，利用者が入力データや解析結果を自由に加工できるような機能の実現化を試みた．

本報告では，このインタプリタの構造及び機能，システム開発者及びユーザ側からみた結合の方法，使用例，この機能の付加による利点及び今後の課題等について論述する．

2．言語の概要

2－1　言語の特長及び機能

この言語は次のような特長を有している．

①ソースプログラムはフリーフォーマットで記述できる．

②一つの文は継続マーク＇@＇をつけることにより３行まで継続できる（１行72文字）．

③sin,cos,log 等２３種の外部関数が使用できる．

④配列は３次元まで定義できる．

⑤配列同士の演算が一行でできるような配列計算文が用意されている（表－１参照）．

⑥計算結果は出力時にフォーマットを指定して出力することができる．

⑦表題や項目名に漢字文字を出力することもできる．

⑧配列データのソート，最大・最小値の抽出，Cardano法，Regula-Falsi法等を関数の形で引用することができる．

表－１にこの言語の文の種類と機能の概要を示す．

表－１　プログラム文とその機能

文の種類	機　　能	例
代入文	算術式を定義する．	Y=2*X+Z Y=(COS(X)+COS(-X))/2
READ文	変数および配列要素の値を読み込む．	READ X Y A(1 1)
PRINT 文	変数および配列要素の値をプリントする．	PRINT X @F6.1 Y @F7.1 A(1 1) @12.3

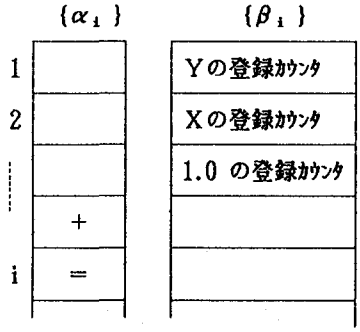
IF文 JUMP文	条件判断を行う． 指定された文番号へとぶ．	IF X >= Y JUMP 10 JUMP 20
LOOP文 NEXT文	LOOP～NEXT区間を制御変数の値が満足されるまで繰り返す	LOOP 10 X=1 20 1.5 ⋮ NEXT 10
ARREY 文	整列を定義する．	ARRAY A(10 10) B(10)
MREAD 文	配列全体の読み込みを行う．	MREAD A(* *) MREAD B(I J) J=1 5 I=1 5
MPRINT文	配列全体のプリントを行う．	MPRINT A(* *) MPRINT B(I J) J=1 5 I=1 5
配列計算文	マトリックスの加，減，乗算および各種のマトリックス操作を行う．	A(* *)=B(* *)+C(*) ：和 A(*)=B(* *)+C(*) ：積 A(* *)=INVERS(A(* *),N) ：逆ﾏﾄﾘｯｸｽ
関数文	データの並べ替え，３次方程式の解 他	X(*)=CARDANO(a_1 a_2 a_3 a_4)
AGET文	アプリケーションプログラム内のDIMENSION データを使用者がARRAY 文で定義した配列に取り込む．	AGET COORD-X X 節点のＸ座標を配列〔X〕に取り込む AGET SFOR-MOM 1 MOM 荷重ｹｰｽ1のﾓｰﾒﾝﾄ の値を配列〔MOM〕に転送する
APUT文	使用者の定義した配列の値をアプリケーションプログラムのDIMENSION に送り込む．	APUT SPJ SPJ-NK 配列〔SPJ〕の値を節点ﾊﾞﾈﾃﾞｰﾀ として所定のDIMENSION に転送する

2－2　ソースプログラムの解析手順

ソースプログラムは字句解析→構文解析の手順を経てスタックへ登録される．字区解析では，まず，プログラム文を意味ある最小の単位のトークンに分解する．そして，トークンが変数の場合は変数名表にその名前を，定数であればその値を定数表に登録する．ARRAY 文の場合は，配列の名前，次元数，全体のサイズ等を登録する．トークンに分けられたプログラム文は，本言語の文法に従って構文解析される．例えば，代入文は逆ポーランド記法に変換される．

2－3　スタックのデータ構造

構文解析されたプログラム文は図－１に示すようなスタックに登録される．



{α_i} ：命令コード

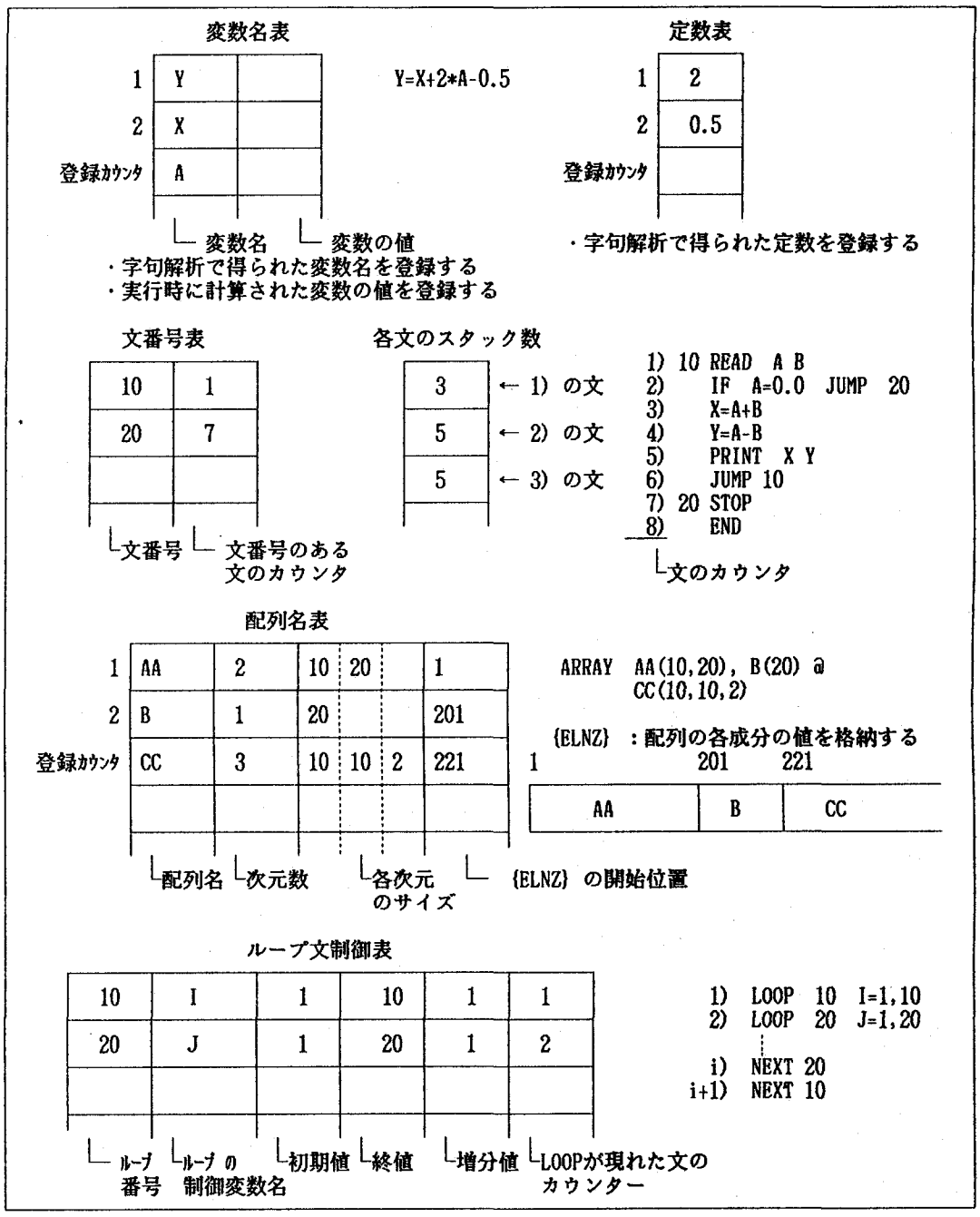
{β_i} ：・変数名，定数，配列の登録番号
・JUNP文やNEXT文の場合には文番号

i ：スタックのカウンタ

（左図は，Y＝X＋1.0 を逆ポーランド記法に変換した後のスタックの例である）

図－１　スタックの構造

このスタックに関連している他の基本的なデータ構造を図－２に示す．

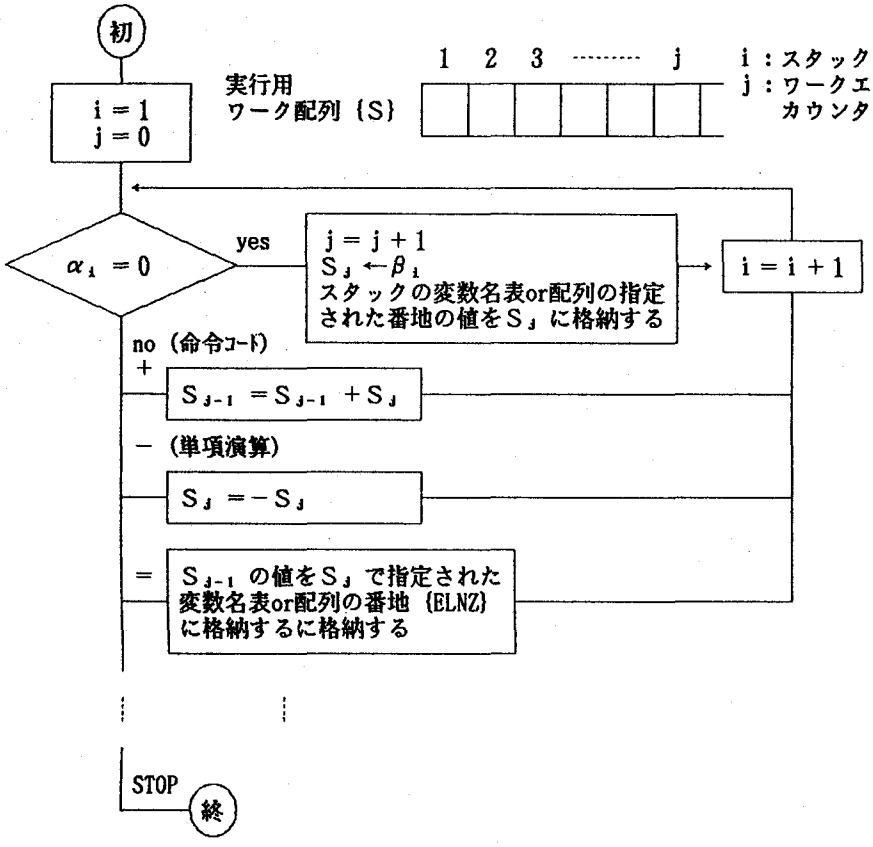


図－2　インタプリタの基本データ構造

2－3　インタプリタの実行手順

スタックに登録されたプログラムはインタプリタにより命令コードの解読→実行がなされる．実行に先だち，スタック（β_1）のJUMP文の飛先きやLOOP文の戻り先は，対応するスタックのカウンタとなるように調整する．実行は次のような手順による．まず，スタックから命令コード（α_1）とデータの登

録カウンタ（β_i）をペアで取り出す．命令コードがない場合はβ_iで示された番地の変数や配列の値をワークエリア｛S｝に格納する．命令コードがあれば｛S｝内で演算操作を行う．また，命令コードが＝であれば演算結果を変数名表や配列エリアに登録する．インタプリタの実行の概念を図－3に示す．



図－3　インタプリタの実行の概念図

2－4　インタプリタの主要機能

インタプリタの実行時における主要機能は次の通りである．

（1）本言語によるプログラムは次の2通りの方法で実行できる．

①ソースリストを一行ずつ読み込み，直ちに解釈・実行する（ただしLOOP文は不可）．

②ソースリストを全てスタックに登録した後，表－2に示すコマンドで実行する．

（2）表－2に示すCONST コマンドにより実行時に変数の値をREAD文によらずに定義できる．

（3）表－2に示す DO コマンドや LOOP コマンドにより，プログラム全体を指定した変数の値を少しずつ変更しながら実行することができる．

（4）表－1に示すAGET，APUT文により，アプリケーションプログラム内の DIMENSIONやファイルのデータとARRAY 文で定義した配列とでデータの相互転送が可能である．即ち，ユーザは ARRAY文で定義した配列に，AGET文によりアプリケーションプログラム内のDIMENSION の値やファイルデータを取り込み，自由に加工することができる．また，APUT文により，ARRAY 内のデータをアプリ

ケーションプログラムの DIMENSIONに転送することができる（ファイルへの転送は不可）．

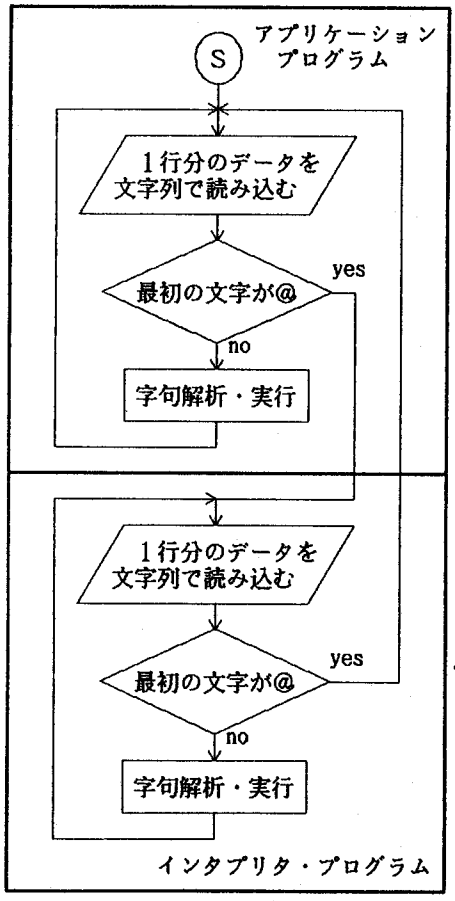
表－2　実行コマンドとその機能

コマンドの種類	機　　能
RUN	定義したプログラムを実行する．
LOOP X=m_1 m_2 m_3	プログラム全体をX がm_1からm_2まで増分値m_3により繰り返し実行する．
DO X=X_1 X_2 X_3 ……	プログラム全体をX がX_1 X_2 X_3…の値に対して実行する．
CONST X=1.5,B=10…	プログラム内の未定義の変数に対して値を定義する．

3．アプリケーションプログラムとインタプリタの結合

3－1　インタプリタの結合

インタプリタは全てFORTRAN で記述されている．このプログラムを既存のアプリケーションプログラムに組み込むためのハードウェア上の制限は特には存在しない．ただし，インタプリタのサイズが大きい（約10000 ステップ）ため，結合したプログラムを実行するためには仮想記憶の機能を有するコンピュータが望ましい．ソフトウェア上の組み込み条件は次の通りである．



図－4　プログラムの結合

①両者のプログラム間で同一の変数名，配列名，サブルーチン名があってはならない（このインタプリタでは使用している変数名，配列名，サブルーチン名には全て¥文字をつけ，混用を起こり難くしている．また，GOTO文の文番号も同一番号台にグループ化して変更を容易にしている）．

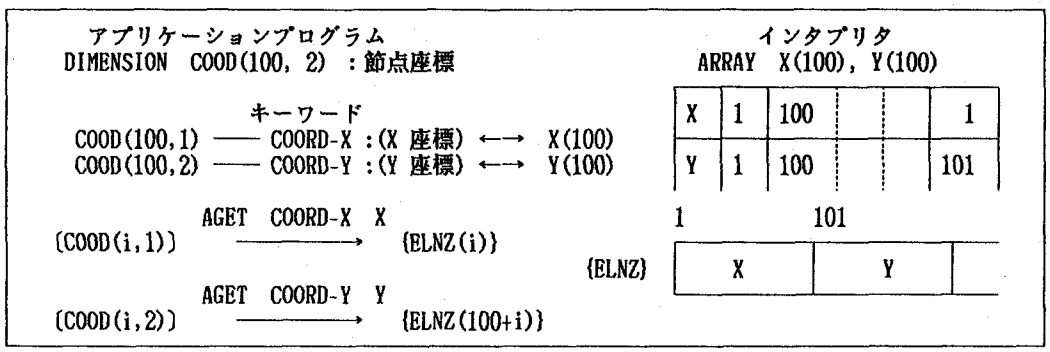
②図－4に示すように，アプリケーションプログラムもデータは行単位に文字列で読み込み，プログラム内で字句解析（コマンドの解読）→実行する形式でなければならない．この形式により，図－6に示すようにアプリケーションプログラムのデータと本言語との混在が可能となる．

③アプリケーションプログラム内のDIMENSION やファイル内のデータと本言語のARRAY 文で定義した配列とのデータ転送をするためには，アプリケーション側に図－5に示すような機能を行うサブルーチンを組み込んでおかなければならない．

3－2データと言語の記述形式

プログラムの実行は，アプリケーションプログラムの実行とインタプリタ言語の解読・実行に大別される．言語機能は，ユーザ側からみた場合，アプリケーションプログラム用に作成されたデータの後に単純に追加すればよいように工夫した．これにより，この言語の使用の有無にかかわらず利用環境は変化せず，作成済のデータとの互換性が保たれる．骨組構造物の解析において，構造データを修正する場合を想定した組

合せの１例を図－６に示す．



図－５　データ転送の概念

```
TYPE   PLANE  FRAME                    ┐
INPUT  STRUCTURE                       │
C  TEST                                │
/COORD                                 │
 1  0  10                              │
 2  3  10                              │  骨組解析のための基本データ
 3  6  10                              │    (従来の入力形式)
 4  9  10                              │
  ┆                                    │
/MEMBER                                │
 1  1  2  1                           │
 2  2  3  1                           │
 3  3  4  1                           │
  ┆                                    │
/SPRING  JOINT  K=100  DIR=1           │
 S  5  9                               │
/;                                     ┘
ARRAY  X(100)  Y(100) &                —  アプリケーションプログラム内のDIMENSION やファイ
       SPJ(20)                            ルの値を格納するための配列を定義する

 AGET  COORD-X   X                     ┐  アプリケーションプログラム内の節点座標
 AGET  COORD-Y   Y                     │  節点バネデータを，それぞれ {X }  {Y } および
 AGET  SPR-NK    SPJ                   ┘  {SPJ } に転送する

@ FL1000                               —  モードの切換
 X(*)=2*X(*)                           ┐
 Y(*)=2*Y(*)                           │  データの変更
 SPJ(*)=10*SPJ(*)                      ┘

 APUT  SPJ  SPR-NK                     ┐  変更したデータをアプリケーションプログラム内の
 APUT  X    COORD-X                    │  DIMENSION に戻す
 APUT  Y    COORD-Y                    ┘

@ FB4000                               —  モード切換
 LIST STRUCTINE                        ┐
 ANALYZE                               │  アプリケーションプログラムによる解析の実行
 LIST  SFORCE  ETC                     │    (従来の入力形式)
 LCASE  ALL                            │
 PLOT  DIAGRAM  SFORCE  ALL            ┘
```

図－６　アプリケーションプログラムのデータとインタプリタ言語との組合せ例

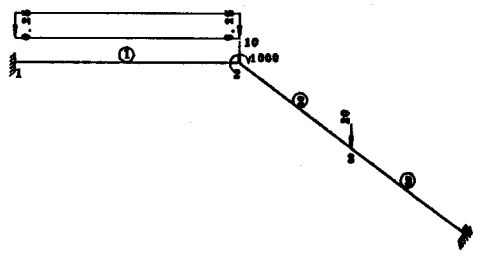
４．使用例

このインタプリタを骨組構造物の解析プログラムと結合して，構造解析を行った後，曲げモーメントが0となる位置でのせん断力，軸力を求める例を示す．このプログラムでは，曲げモーメントは各部材を10等分した各点で計算されている．この例では，M＝0となる位置はモーメントの符号が反転する2点間を線形補間して求めることにする．データリストを図－7(a)に，出力結果を図－7(b)に示す．

```
TYPE PLANE FRAME
INPUT STRUCTURE
C PLANE FRAME -- STANDARD TYPE
/PROPERTY MEMBER
1 100000 1.E-4 1.E-2
/COORDINATE
1 0    75
2 100  75
3 150  37.5
4 200  0
/MEMBER
1 1 1 2 1
2 1 2 3 1
3 1 3 4 1
/BOUNDARY
1 1 1 1
4 1 1 1
C LOAD DATA
/JLOAD
2 2 -10
2 6 -1000
3 2 -20
/MLOAD
1 1 2 0 100 -0.25 -0.25
/;
LIST STRUCTURE  ;  構造データの出力
LIST LOAD       ;  荷重データの出力
ANALYZE         ;  解析
LIST SFORCE ETC ;  解析結果の出力
*
ARRAY LENG(3)
ARRAY MOM(3 11) SHR(3 11) AXI(3 11)
AGET  MEM-LENG      LENG
AGET  SFOR-MOM   1  MOM
AGET  SFOR-SHE   1  SHE
AGET  SFOR-AXI   1  AXI
*
@ FL1000
PROGRAM
    FL=LENG(NEL)
    IFLG=0
    LOOP 100 I=1 10
    X1=(I-1)*FL/10
    X2=I*FL/10
    I2=I+1
    M1=MOM(NEL I)
    M2=MOM(NEL I2)
    M12=M1+M2
    IF M12 > 0  JUMP 10
    IFLG=1
    M11=ABS(M1)
    M22=ABS(M2)
    XI=(X2-X1)*M11/(M11+M22)
    XLO=X1+XI
```

構造データ

荷重データ

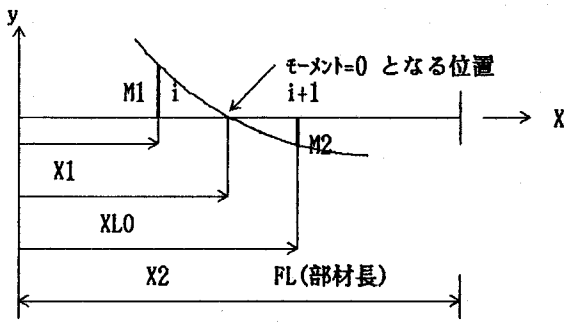


解析モデル

解析結果から荷重ケース1の断面力を指定した配列にGET する

荷重ケース

MEM-LENG：部材長のｷｰﾜｰﾄﾞ
SFOR-MOM：モーメントのｷｰﾜｰﾄﾞ
SFOR-SHE：せん断力のｷｰﾜｰﾄﾞ
SFOR-AXI：軸力のｷｰﾜｰﾄﾞ



XLOの位置の計算については上図を参照

```
   S1=SHR(NEL I)
   S2=SHR(NEL I2)
   A1=AXI(NEL I)
   A2=AXI(NEL I2)
   SX=S1+((S2-S1)/(FL/10))*XI
   AX=A1+((A2-A1)/(FL/10))*XI
   PRINT ' 部材ＮＯ＝'  NEL @I2 '      ＸＬ０＝'  XLO @F6.2 &
         '      Ｓ＝'  SX @F8.2  '      Ｎ＝'  AX @F8.2
10 NEXT 100
   IF IFLG = 1 JUMP 20
   PRINT ' 部材ＮＯ＝' NEL @I2'  モーメント＝０となる位置はなし
20 END
/
   DO    NEL=1 2 3    ―― 上記のプログラムを部材番号1,2,3 に対して実行する
```

図－7(a)　モーメントが０となる位置とその位置でのせん断力と軸力の計算

```
   部材ＮＯ＝  1  モーメント＝０となる位置はなし
   部材ＮＯ＝  2  モーメント＝０となる位置はなし
   部材ＮＯ＝  3    ＸＬ０＝    19.11    Ｓ＝     －20.58    Ｎ＝     －41.16
```

図－7(b)　出力結果

5．プログラムの効果

本プログラムの利点は次のように要約される．

①プログラムの開発担当者は，ユーザ固有の要望に煩わされることなく，汎用性のある機能の開発に専念できる．他方，ユーザは自分の固有の設計業務に対して入・出力データの加工が自分で自由に行えるようになり，双方で省力化が可能となる．

②インタプリタの結合に伴うユーザのデータの直接的な変更は全くない．そのため，この機能を付加しても過去のデータとの互換性が保たれる．従って，ユーザにとってこの機能の使用の有無にかかわず使用環境は変化しない．

③ＢＡＳＩＣ言語の知識があればこの言語と表－２に示すコマンドを単独で使用して設計上の簡単な問題を解くこともできる．

6．おわりに

設計業務におけるコンピュータの利用が増大するなかで，省力化へのニーズも高まる一方である．設計サイドの多種多様な省力化要求に対して，本方法は一つの解決手段であろう．反面，現状では如何にも言語機能が弱い．例えば，柔軟な表出力機能や図形出力機能は設計計算書の作成には不可欠である．しかしながら，そのような機能の実現化には，いくつかのソフトウェア上の技術開発が必要になると共にプログラムサイズも巨大となるため開発は容易ではない．コンピュータの容量・処理能力が年々拡大してゆく中で，今後共ユーザに受け入れ易く様々なニーズに柔軟に対応できるシステムの開発に取り組んでゆきたい．

参考文献

1）　田中育男　「コンパイラの技法」　竹内書店新社

2）　根本隆栄　「設計補助計算のための簡易インタプリタ言語の開発」土木学会，第４回電算機利用に関するシンポジウム講演集　1979-11