リスト FORTRAN とそのプリプロセッサ†

松山公一†† 中村良三†††

構造化された多量の情報の蓄積，検索，処理を含む科学技術計算を行うとき，FORTRAN などの算法言語の中にリスト処理機能を取り入れれば，多様な情報の処理ができるであろうという発想から，この種の記号処理言語として，SLIP, FLIPS などが作られている．

本稿で報告するリスト FORTRAN もこの類の記号処理言語であるが，リストの入出力，リストに対する演算および帰納的関数の定義において，LISP 的記述法を併用しうるように，FORTRAN を拡張した言語である．

このリスト FORTRAN で記述されたプログラムはプリプロセッサで処理した後，FORTRAN コンパイラを介し，すでに組み込んでいるリスト処理手続きなどと結合して実行する．

この方式によって，上述のような処理に対するプログラミングおよび計算時間に大きなメリットが期待できる．

また，このシステムではプリプロセッサおよびリスト処理に関する手続きもすべて FORTRAN で記述しているので，新しい機能の追加も比較的容易で，かつ FORTRAN コンパイラを持つ計算機であれば種別を問わず使用できるはん用性がある．

1. はじめに

多量の構造化された情報の蓄積，検索，処理を含む科学技術計算を行うとき，FORTRAN などの算法言語のなかにリスト処理機能を取り入れれば，プログラム，計算時間のいずれにも大きなメリットが期待できるであろうという発想から，JIS 7000 FORTRAN に基本的なリスト処理文を書くことができるように FORTRAN を拡張した言語（これを仮りにリスト FORTRAN と呼ぶ．）を設定し，この言語で記述したプログラムをプリプロセッサで処理した後 FORTRAN コンパイラを介し，すでに組込んでいるリスト処理用の手続きや算術計算用の手続きなどと結合して実行する処理方式を開発したので報告する．

2. リスト FORTRAN の概要

リスト FORTRAN は JIS 7000 FORTRAN の機能にリストの入出力，リストに対する演算，帰納的関数の定義などを付加するために LISP[1),2)] 的記述を併用しうるようにしたものである．

したがって FORTRAN の全機能を十分利用しながら基本的なリスト処理ができる．

このリスト FORTRAN システムでは，FORTRAN における科学計算用の組込関数と同じようにリスト処理用の基本的な手続きを組み込んでいるので，この言語でプログラミングをするとき，あたかもシステムがリスト処理機能を持っているかのように，リストの内部表現には一切触れることなく容易にアルゴリズムを記述することができる．

またプリプロセッサおよびリストに関する手続き等もすべて FORTRAN で作成しているので，新しい機能を追加することは比較的容易であり，かつ FORTRAN コンパイラを持っている計算機であれば機種を問わず使用できるはん用性がある．

次に JIS 7000 FORTRAN に付加されるリスト処理文の様式を示す．

（1） 主譜ですべてのリストは次の形式で明示する．

＊LIST LT 1, LT 2

＊LIST はリスト変数の宣言子で LT 1, LT 2 はリスト変数名である．

（2） リスト評価文* で使用する変数名，配列名はすべて次の形式で明示宣言する．

＊INTEGER TKG, SKG, IXYZ

＊LOGICAL LW 1, LOG 1

＊DIMENSION NAME 1

＊COMMON TKG, LW 1

＊INTEGER, ＊LOGICAL, ＊DIMENSION, ＊COMMON はそれぞれ整数型変数，論理型変数，配列ならびに COMMON 変数の宣言子である．TKG, SKG, IXYZ は整数型変数名，LW 1, LOG 1 は論理

† List-FORTRAN and It's Preprocessor by KIMIKAZU MATSUYAMA (Faculty of Engineering, Kumamoto University) and RYOZO NAKAMURA (Processing Center, Kumamoto University).

†† 熊本大学工学部電気工学科

††† 熊本大学電子計算機室

* (7) 参照

型変数名，NAME 1 は次元 1，寸法 20 の配列名，TKG, LW 1 は COMMON 変数名となる．

（3） リスト評価文で用いる関数でリスト組込関数* 以外の関数は次の形式の外部関数宣言文で明示する

＊FUNCTIONI ADD, REVL

＊FUNCTIONL MEMBER

＊FUNCTIONI, ＊FUNCTIONL はそれぞれ整数型外部関数，論理型外部関数の宣言子であり，ADD, REVL は整数型外部関数名，MEMBER は論理型外部関数名となる．

（4） 主譜において実行文の先頭にリストの内部構造の初期化を行う次のリスト初期化文を許す．

＊INITIALIZE LIST

（5） リストの入力を行う次の形式の入力文を許す．

＊READLIST LT 1, LT 2

LT 1, LT 2 はリスト変数名である．

（6） リストの出力を行う次の形式の出力文を許す．

＊WRITELIST LT 1, LT 2

LT 1, LT 2 はリスト変数名である．

（7） リストに対する演算を行う次の形式のリスト評価文を許す．

＊EVAL LT 3=(CDR (CAR LT))

右辺のリスト記法で記述されたS式が評価され，評価値を左辺の変数 LT 3 に代入する．

（8） 副プログラム文に LISP の関数定義文に準じ，次の形式の外部関数定義文を許す．

＊DEFINE (ADD (LAMBDA (L)

(COND ((NULL L) 0)(T (PLUS ((CAR L)(ADD (CDR L)))))))

（9） リスト処理文を含むFORTRAN 副プログラムでは次のリスト使用文を明示する．

＊USELIST

＊USELIST はリスト使用の宣言子で，これによってリストの内部構造を格納する領域の共通使用を宣言する．

(10) リスト処理文を含むFORTRAN 副プログラムの出口には次の RETURN 文を明示する．

＊RETURN

(11) 外部データ形式として次のようなリスト記法を許す．

(PLUS Y (TIMES Z 4))

* 表1参照

3. リスト FORTRAN の構文

リスト FORTRAN 構文のうち FORTRAN を拡張した部分を BNF 類似の記法で示す．なお現在開発を終っているこのシステムは中型の計算機を用いたので，若干の容量上の制約を付している．これらの制約は注記の形で付記した．

3.1 宣 言 文

〈リスト変数宣言文〉::=＊LIST 〈リスト変数名〉{,〈リスト変数名〉}注 1

〈リスト変数名〉::=〈変数名〉注 2

〈整数型変数宣言文〉::=＊INTEGER 〈変数名〉{, 〈変数名〉}注 2

〈論理型変数宣言文〉::=＊LOGICAL 〈変数名〉{, 〈変数名〉}注 2

〈配列宣言文〉::=＊DIMENSION 〈配列名〉{, 〈配 列名〉}注 2

〈COMMON 文〉::=＊COMMON 〈COMMON 変数名〉{, 〈COMMON 変数名〉}

〈COMMON 変数名〉::=〈変数名〉|〈配列名〉

〈外部関数宣言文〉::=＊FUNCTIONI 〈整数型外部関数名〉{, 〈整数型外部関数名〉}|
＊FUNCTIONL 〈論理型外部関数名〉{, 〈論理型外部関数名〉}注 3

〈リスト使用宣言文〉::=＊USELIST

〈整数型外部関数名〉::=〈変数名〉

〈論理型外部関数名〉::=〈変数名〉

注 1. リスト変数の個数は最大 99 までとする．

注 2. 変数名，配列名はFORTRAN に準じ，個数は各々最大 50 までとする．

注 3. 整数型外部関数名，論理型外部関数名の個数はそれぞれ最大 20 までとする．

3.2 制 御 文

〈リスト初期化文〉::=＊INITIALIZE LIST

〈RETURN 文〉::=＊RETURN

〈FIN 文〉::=＊FIN

3.3 入 出 力 文

〈リスト入力文〉::=＊READLIST 〈リスト変数名〉{, 〈リスト変数名〉}

〈リスト出力文〉::=＊WRITELIST 〈リスト変数名〉{, 〈リスト変数名〉}

3.4 代 入 文

〈リスト評価文〉::=＊EVAL 〈名前〉=〈形式 1〉

〈名前〉::=〈リスト変数名〉|〈変数名〉|〈配列名〉

〈形式1〉::=(〈関数名〉 〈引数〉…〈引数〉)注5
〈関数名〉::=〈整数型関数名〉|〈論理型関数名〉
〈整数型関数名〉::=〈整数型組込関数名〉|〈整数型外部関数名〉
〈整数型組込関数名〉::=CAR|CDR|CONS|PLUS|TIMES|QUOTI|REMAIN|DIFFER|MINUS|ADD1|SUB1
〈論理型関数名〉::=〈論理型組込関数名〉|〈論理型外部関数名〉
〈論理型組込関数名〉::=EQ|EQUAL|ATOM|NULL|GREATP|LESSP|MINUSP|ZEROP|ONEP|OR|AND|NOT
〈引数〉::=〈アトム〉|〈名前〉|〈形式1〉
〈数字〉::=0|1|2|3|4|5|6|7|8|9
〈英大文字〉::=A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z
〈アトム〉::=〈数値アトム〉|〈文字アトム〉
〈数値アトム〉::=〈数字〉|+〈数字〉|−〈数字〉|〈数値アトム〉〈数字〉注6
〈文字アトム〉::=〈英大文字〉|〈文字アトム〉〈英大文字〉|〈文字アトム〉〈数字〉注7

注5. 〈引数〉, 〈入引数〉, 〈COND 対〉の並びの最大個数は5までとする.
注6. 〈数値アトム〉の最大けた数は8までとする.
注7. 〈文字アトム〉の最大文字数は20までとする.

3.5 副プログラム文

〈外部関数定義文〉::=＊DEFINE (〈外部関数名〉 (LAMBDA (〈入引数〉…〈入引数〉注5) 〈形式〉))注8
〈外部関数名〉::=〈整数型外部関数名〉|〈論理型外部関数名〉
〈形式〉::=〈形式1〉|〈条件式〉
〈条件式〉::=(COND 〈COND 対〉…〈COND 対〉)注5
〈COND 対〉::=(〈条件部〉〈値部〉)
〈条件部〉::=〈形式2〉|T注9
〈形式2〉::=(〈論理型関数名〉〈引数〉…〈引数〉)注5
〈値部〉::=〈形式1〉|〈アトム〉注9
〈入引数〉::=〈英大文字〉注5

注8. 定義文の総文字長は最大400字までとする, また値部に帰納的定義を許し, 帰納的関数の位置は深さ3のレベルまでとする.

表1 リスト組込関数
Table 1 Built-in list function.

関数名	関数の型	引数の数	引数の型	機能
CAR	I	1	C	left branch of binary tree.
CDR	I	1	C	right branch of binary tree.
CONS	I	2	C, I	build list from atom or list.
ATOM	L	1	C, I	if x_1=ATOM then T else F.
EQ	L	2	C, I	if $x_1=x_2$=ATOM then T else F.
NULL	L	1	C, I	if x_1=NIL(=0) then T else F.
EQUAL	L	2	C, I	if $x_1=x_2$ then T else F.
ZEROP	L	1	I	if $x_1=0$ then T else F.
LESSP	L	2	I	if $x_1<x_2$ then T else F.
GREATP	L	2	I	if $x_1>x_2$ then T else F.
ONEP	L	1	I	if $x_1=1$ then T else F.
MINUSP	L	1	I	if $x_1<0$ then T else F.
AND	L	2	C, I	$x_1 \cap x_2$.
OR	L	2	C, I	$x_1 \cup x_2$.
NOT	L	1	L	$\bar{x}_1$.
PLUS	I	2	I	x_1+x_2.
MINUS	I	1	I	$-x_1$.
TIMES	I	2	I	$x_1 \times x_2$.
DIFFER	I	2	I	x_1-x_2.
QUOTI	I	2	I	x_1/x_2.
REMAIN	I	2	I	mod (x_1, x_2).
ADD1	I	1	I	x_1+1.
SUB1	I	1	I	x_1-1.

I: integer type L: logical type C: character type (contain list name) x_1: first argument x_2: second argument

注9. 文字アトムの T, F, NIL はそれぞれ論理値真, 偽, 偽とみなす.

3.1 から 3.5 を総称してリスト処理文と呼び, これらの文では空白は区切り記号の意味をもっている.

3.6 リスト組込関数

4. リスト FORTRAN のデータ構造

この言語処理系では JIS 7000 FORTRAN で許される外部および内部データの他に, リストに関する次のようなデータ構造を付加している.

4.1 データの外部表現

リストの外部表現は括弧, コンマ, 空白で区切られたリスト, いわゆるリスト記法で表わす.

4.2 データの内部表現

(1) セルの構造

セルは FORTRAN の3語を基本単位として構成し, 1語ずつ3つの欄に分け, それぞれ LLINK, RLINK, FLINK と名づける. LLINK は番地およびアトムの格納, RLINK は番地の格納, FLINK は自由リストの番地の格納とごみ集め時のフラッグに用いる. なお番地は正の整数, アトムは負の整数で表わす.

(2) アトムの表現

1語 32 ビットの計算機を使用しているので, 1語

表 2 文字コード
Table 2 Character code.

文字	文字コード	文字	文字コード
A	01	Z	26
B	02	0	27
C	03	1	28
D	04	2	29
E	05	3	30
F	06	4	31
G	07	5	32
H	08	6	33
I	09	7	34
J	10	8	35
K	11	9	36
L	12	空白	37
M	13	(	38
N	14	)	39
O	15	.	40
P	16	,	41
Q	17	'	42
R	18	−	43
S	19	+	44
T	20	*	45
U	21	/	46
V	22	=	47
W	23	¥	48
X	24	$	49
Y	25		

表 3 関数コード
Table 3 Function code.

関数名	関数コード
CAR	−900000101
CDR	−900000102
ATOM	−900000103
CONS	−900000104
EQ	−900000105
NULL	−900000201
EQUAL	−900000202
ZEROP	−900000203
LESSP	−900000204
GREATP	−900000205
ONEP	−900000206
MINUSP	−900000207
AND	−900000208
OR	−900000209
NOT	−900000210
PLUS	−900000301
DIFFER	−900000302
MINUS	−900000303
TIMES	−900000304
QUOTI	−900000305
REMAIN	−900000306
ADD 1	−900000307
SUB 1	−900000308
整数型外部関数名	−900000501 ～
論理型外部関数名	−900000601 ～

で整数を表わすと 10 進9桁の数が格納される．そこで9桁目をアトムの分類に用いる．

（a） 数値アトム

・正の数値アトム　9桁目 0.

・負の数値アトム　9桁目 1.

（b） 文字アトム

1文字ずつ文字コード（**表2**）に変換し，4文字単位で区切り（10 進8桁の数），9桁目に8を付け格納する．4文字以上のときにはRLINK でつなぎリストで表わす．なお関数名は関数コード（**表3**）に変換して格納する．

（3） リストの表現

例1

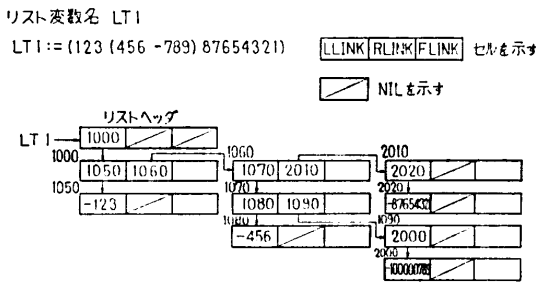

図 1 リスト LT1 の内部表現
Fig. 1 Internal representation of LT 1.

リストはリスト変数名ごとにリストヘッダをもち，本体は二進木リストで，次例のように表現する．

例2

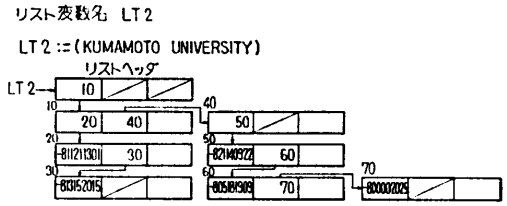

図 2 リスト LT2 の内部表現
Fig. 2 Internal representation of LT 2.

5. プリプロセッサの構成と機能

プリプロセッサ自体すべてFORTRAN で記述している．その大きさはソースプログラムカードで約3000枚，ロードモジュールで約 60 KB である．

（補足，リスト処理手続き群もすべてFORTRAN で記述し，大きさはソースカード約 1400 枚，リロケータブル・オブジェクトで約 40 KB である．）

このプリプロセッサの大きさの約半分が外部関数定義文の解釈と処理のためのプログラムで占められている．それは本来帰納的機能を備えてない FORTRAN で帰納的に定義された外部関数と等価な機能をもつ無駄のないFORTRAN プログラムを自動的に生成するためである．

プリプロセッサは次の3つの部分から構成される．

1. リスト FORTRAN 読込部.
2. 文分類部.
3. 構文解析と FORTRAN 文生成部.

次に各構成部の機能について述べる．

5.1 リスト FORTRAN 読込部

ソースプログラムカードを文字型式で1枚ずつ読み込む．

5.2 文分類部

読込んだ記号列を1文字ずつ調べ，区切り記号で最初の単語を取り出す．それがリスト処理文の識別子(＊LIST, ＊INTEGER, ＊LOGICAL, ＊DIMENSION, ＊COMMON, ＊FUNCTIONI, ＊FUNCTIONL, ＊INITIALIZE LIST, ＊READLIST, ＊WRITELIST, ＊EVAL, ＊DEFINE, ＊USELIST, ＊RETURN, ＊FIN) に該当すれば，リスト処理文とみなし残りの記号列を各文の構文解析部に引き渡し，該当しなければ通常の FORTRAN 文として処理する．

5.3 構文解析と FORTRAN 文生成部

文分類部から引き渡された記号列を次のような各ルーチンで解析し，いくつかの FORTRAN 文に展開する．構文上の誤りがあれば，指定したエラーメッセージを出力する．

(1) ＊LIST 文

リスト変数名を識別し，リスト名表に順次登録する．

(2) ＊INTEGER 文

変数名を識別し整数型変数名表に登録したのち，型宣言文で整数型として生成する．

(3) ＊LOGICAL 文

整数型が論理型に変る以外(2)に同じ．

(4) ＊DIMENSION 文

配列名を識別し配列名表に登録したのち，DIMENSION 文で次元 1，寸法 20 の配列を生成する．

(5) ＊COMMON 文

COMMON 変数名を識別し COMMON 変数名表に登録したのち，名前付共通ブロック（ブロック名，EXPLIC）で生成する．

(6) ＊FUNCTIONI 文

関数名を識別し整数型関数名表に登録する．

(7) ＊FUNCTIONL 文

整数型が論理型に変わる以外(6)に同じ．

(8) ＊INITIALIZE LIST 文

リストの内部構造を格納する領域を確保したり，自由リストを形成する手続きをよぶ引用文を生成する．

(9) ＊READLIST 文

リスト変数名がリスト名表に登録されていれば，各リスト変数名ごとリスト入力サブルーチン副プログラムの引用文を生成する．

(10) ＊WRITELIST 文

リスト出力サブルーチン副プログラムを引用する以外は(9)に同じ．

(11) ＊USELIST 文

副プログラムの宣言文にリスト格納領域を共通ブロックで生成する．

(12) ＊EVAL 文

リスト記法で書かれた形式を括弧対ごと逐次評価し関数副プログラムを引用した代入文を生成する．評価の中間段階ではそれぞれの関数の型に従ったワーキング変数に代入し，最終段階で名前に評価値を代入する処理関数の引用文を生成する．

(13) ＊DEFINE 文

外部関数定義文を 2 パス方式で解釈処理する．第 1 パスでは条件文，帰納的定義の有無など定義文の骨子となる構造を調べ，帰納的定義を含んでいれば演算に必要なスタック類の配列を構造に応じて生成する．

第 2 パスでは第 1 パスの情報をもとに，リスト記法で書かれた形式を ＊EVAL 文の処理と類似した方式で展開するが，帰納的演算に対してはさらにいくつかの FORTRAN 文に展開して，関数副プログラムを生成する．

(14) ＊RETURN 文

FORTRAN 副プログラムの出口での処理を行う．

(15) ＊FIN 文

すべてのリスト FORTRAN ソースプログラムの終了を示す．

6. リスト FORTRAN プログラム

リスト FORTRAN プログラム例として簡単な例題を示し，具体的にリスト FORTRAN プログラムの作成ならびにプリプロセッサが生成する FORTRAN プログラムを示す．

6.1 リスト FORTRAN プログラムの作成

[例題]

「2 個のリストを読み込み，それぞれのリストに対して次のような演算を行う．

第 1 番目のリストに対してアトム WHALES がメンバーにあるかどうかを調べその結果を出力する，またリストの要素の順序を逆にした新しいリストを作りその結果およびそのリストの先頭の要素を出力する．

次に各人の体重を要素とする第 2 番目のリストに対して，その集団の平均値ならびに標準偏差を求め，(MEANWEIGHT 平均体重 SDWEIGHT 標準偏差値) なるリストを作り結果を出力する．」

例題をリスト FORTRAN プログラムで書くと次のようになる．

6.1.1 主　　譜

```
01 C      LIST-FORTRAN MAIN-PROGRAM
02        *LIST LT1, LT2, LT3, LT4, LT5
03        *INTEGER TKG, SKG, X1, X2, X3
04        *LOGICAL LOG1
05        *DIMENSION NAME1
06        *FUNCTIONI ADD, REVL
07        *FUNCTIONL MEMBER
08        *INITIALIZE LIST
09        *READLIST LT1, LT2
10        *WRITELIST LT1, LT2
11        *EVAL LOG1=(MEMBER WHALES LT1)
12        WRITE (6, 100) LOG1
13        *EVAL LT3=(REVL LT1 NIL)
14        *WRITELIST LT3
15        *EVAL NAME1=(CAR LT3)
```

```
16      *WRITE (6, 110) NAME 1
17      *EVAL TKG=(ADD LT 2)
18      TKG=TKG/3
19      *EVAL X 1=(CAR LT 2)
20      *EVAL X 2=(CADR LT 2)
21      *EVAL X 3=(CADDR LT 2)
22      XX=X 1 * * 2 +X 2 * * 2 +X 3 * * 2
23      SKG=SQRT(XX/3-TKG * * 2)
24      *EVAL LT 4=(CONS MEANWEIGHT (CONS TKG
25     /                (CONS SDWEIGHT (CONS SKG
                        NIL))))
26      *WRITELIST LT 4
27    100 FORMAT (1 H, 16 X, L 5)
28    110 FORMAT (1 H, 20 X, 20 A 1)
29      STOP
30      END
```

図 3 リスト FORTRAN の主譜

Fig. 3 Main-program of List-FORTRAN.

説明の都合上最左側列に行番号を記している．なおリスト FORTRAN のコーディング様式はすべて FORTRAN に準じる．

行を追ってリスト FORTRAN をみてゆくと，01 行は注釈文である，02 から 07 行までが宣言文で 08 行から実行文が始まる．まず宣言文を具体的にみてゆくと，02 行はリスト変数宣言文でリスト変数名 LT 1, LT 2, LT 3, LT 4, LT 5 を宣言する．これによってプログラムで使用するリスト変数名が 5 個確保される．03, 04 行はリスト評価文で用いる変数の型宣言文で整数型変数 TKG, SKG, X 1, X 2, X 3 および論理型変数 LOG 1 を宣言する．

FORTRAN では変数の第 1 文字で暗黙の型宣言を許しているが，リスト処理文で用いる変数名はすべて明確に型宣言をしなければならない．

05 行は配列宣言文で配列名 NAME 1 (次元 1, 寸法 20) を宣言する．

06, 07 行は外部関数宣言文である．

06 行は整数型関数名 ADD, REVL, 07 行は論理型関数名 MEMBER をそれぞれ宣言する．

08 行はリスト初期化文で，この文によってリストを格納する内部構造が作られる．

09, 10 行はリスト入出力文である．

09 行は外部データ表現のリストを 2 個読み込み，それぞれリスト変数名 LT 1, LT 2 としてリストの内部構造を作る．10 行は確認のため読み込んだリストの内部構造をリスト記法の型で出力する．

11 行はリストに対する演算を行うリスト評価文である．リスト LT 1 に関数 MEMBER を作用させ，文字アトム WHALES がメンバであれば論理値真なければ偽を論理型変数 LOG 1 に代入する演算である．

12 行は純 FORTRAN の出力文である．

13 行はリスト評価文である．この評価文はリスト LT 1 に関数 REVL を用いてリスト LT 1 の要素の順序を逆にしたリストを作り，そのリストを LT 3 と名付ける演算を行う．14 行はリスト出力文である．

15 行はリスト LT 3 の第 1 要素を取り出し，配列 NAME 1 に格納するリスト評価文である．

17 行もリスト評価文で，リスト LT 2 に関数 ADD を作用させて，数値アトムの総和を求め，その値を TKG に代入する演算である．

19 行から 21 行もすべてリスト評価文であり，それぞれの評価値を整数型変数 X 1, X 2, X 3 に代入する．

22 行，23 行は FORTRAN の算術代入文に他ならないが，23 行では FORTRAN の組込関数 SQRT を用いている．

24, 25 行はリスト組込関数 CONS を用いて，新しいリスト LT 4 を作るリスト評価文である．

6.1.2 副　譜

副譜はすべて外部関数定義文で次のように定義する．

```
 *DEFINE (ADD (LAMBDA (L) (COND ((NULL L) 0)
/          (T (PLUS (CAR L) (ADD (CDR L)))))))
 END
 *DEFINE (REVL (LAMBDA (A B) (COND ((NULL A) B)
/          (T (REVL (CDR A) (CONS (CAR A) B))))))
 END
 *DEFINE (MEMBER (LAMBDA (M L) (COND ((NULL L) F)
/          ((EQ M (CAR L)) T)
/          (T (MEMBER M (CDR L))))))
 END
```

図 4 リスト FORTRAN の副譜

Fig. 4 Sub-program of List-FORTRAN.

6.1.3 入力データ

```
(GIANT TIGERS DRAGONS WHALES CARP)
(67  43  58)
```

6.1.4 結　果

```
****LIST 1****
          (GIANTS TIGERS DRAGONS WHALES CARP)
****LIST 2****
          (67 43 58)
           T
****LIST 3****
          (CARP WHALES DRAGONS TIGERS GIANTS)
          CARP
****LIST 4****
          (MEANWEIGHT 56 SDWEIGHT 9)
```

図 5 実行結果

Fig. 5 Output of List-FORTRAN.

6.2 生成された FORTRAN プログラム

6.1 のリスト FORTRAN プログラムはプリプロセッサによって次のように生成される.

6.2.1 主 譜

```
C       LIST-FORTRAN MAIN-PROGRAM
      COMMON/INPLIC/IAL (5000,3), IBFSB, IBFS, NOL,
     *          IBL (1000, 2), IBLP, IEC
       INTEGER CAR, CDR, CONS
       INTEGER PLUS, DIFFER, MINUS, TIMES, QUOTI,
     *          REMAIN, ADD1, SUB1
       LOGICAL LW1, LW2, LW3, LW4, LW5 , LW6,
     *          LW7, LW8, LW9, LW10
       LOGICAL ATOM, EQ, NULL, EQUAL, ZEROP,
     *          LESSP, GREATP, MINUSP, ONEP,
     *          LEVEND, AND, OR, NOT
          INTEGER TKG, SKG, X1, X2, X3
          LOGICAL LOG1
       INTEGER NAME1 (20)
       INTEGER ADD, REVL
       LOGICAL MEMBER
       COMMON/INPLC1/NDEFFC (20, 6), IMAXFC (20),
     *                 NARGFC (20), NDEFLG (20, 6),
     *                 IMAXLG (20), NARGLG (20)
       CALL INITIL (5)
       CALL RLIST (1)
       CALL RLIST (2)
       CALL WLIST (1)
       CALL WLIST (2)
       IW1=ITN (2, -823080112, -800000519,  0,  0,  0)
       LW1=MEMBER  ( IW1,      1)
       LOG1=LEVEND (     0, LW1)
          WRITE (6,100) LOG1
       IW1=REVL    (    1,  -200140912)
       LT3=IEVEND (    3,        IW1)
       CALL WLIST (        3)
       IW1=CAR     (        3)
       MOGI=MEVEND (  NAME1,   IW1)
          WRITE (6,110) NAME1
       IW1=ADD     (           2)
       TKG=IEVEND (         -1,     IW1)
          TKG=TKG/3
       IW1=CAR     (            2)
       X1=IEVEND (          -1,     IW1)
       IW1=CDR     (            2)
       IW1=CAR     (         IW1)
       X2=IEVEND (           -1,     IW1)
       IW1=CDR (            2)
       IW1=CDR (         IW1)
       IW1=CAR (         IW1)
       X3=IEVEND (           -1,     IW1)
          XX=X1**2+X2**2+X3**2
          SKG=SQRT (XX/3-TKG**2)
       IW1=IEVCNG (      TKG     )
       IW2=IEVCNG (      SKG     )
       IW2=CONS    (        IW 2,   -200I40912)
       IW3=INT (2, -819042305, -809070820,  0,  0,  0)
       IW2=CONS (      IW3,     IW2)
       IW1=CONS (      IW1,     IW2)
       IW2=ITN (3, -813050114, -823050907, -800000820, 0, 0)
       IW1=CONS (      IW2,     IW1)
       LT4=IEVEND (      4,     IW1)
       CALL WLIST (      4)
  100 FORMAT (1H, 16X, L5)
  110 FORMAT (1H, 20X, 20A1)
       STOP
       END
```

図 6 生成されたメインプログラム

Fig. 6 Main-program of FORTRAN to be produced by preprocessor.

6.2.2 副 譜

```
      INTEGER FUNCTION ADD (L)
      COMMON/INPLIC/IAL(5000,3), IBFSB, IBFS, NOL,
     IBL(1000,2), IBLP, IEC
      INTEGER PLUS, CAR, CDR
      LOGICAL NULL
      INTEGER ISTCON(500), ISTK(1000), KRT(1,500)
      INTEGER S1(1,500)
      LOGICAL LW1
      KP=0
      KCON=0
      K1=0
      IL=L
 1000 LW1=NULL (    IL)
      IF (. NOT. LW2) GO TO 2000
      NN=           0
      IF (K1. GE. 1) GO TO 1010
      ADD=NN
      GO TO 6000
 1010 JJ=KRT (1, K1)
      K1=K1-1
      ISTK (JJ)=NN
      IF (KCON. LT. 1) GO TO 7000
      ICON=ISTCON (KCON)
      KCON=KCON-1
      GO TO (2510), ICON
 2000 IW1=CAR (        IL)
      IW2=CDR (        IL)
      K1=K1+1
      S1 (1, K1)=IW2
      KP=KP+1
      ISTK (KP)=IW1
      KP=KP+1
      KRT (1, K1)=KP
      IL=S1 (1, K1)
      KCON=KCON+1
      ISTCON (KCON)=1
      IF (KCON. GT. 490) GO TO 7010
      IF (K1. GT. 490) GO TO 7020
      IF (KP . GT. 970) GO TO 7030
      GO TO 1000
 2500 CONTINUE
 2510 IF (K1. LT. 1) GO TO 2520
      JJ=KRT (1, K1)
      K1=K1-1
      ISTK (JJ)=PLUS (ISTK (KP-1), ISTK (KP))
      KP=KP-2
      IF (KCON. LT. 1) GO TO 7000
      ICON=ISTCON (KCON)
      KCON=KCON-1
      GO TO (2510), ICON
 2520 IF (KP. LT. 2) GO TO 2530
      ISTK (1)=PLUS (ISTK (KP-1), ISTK (KP))
 2530 ADD=ISTK (1)
 6000 CONTINUE
      RETURN
 7000 WRITE (6,7005)
      RETURN
 7010 WRITE (6,7015)
      RETURN
 7020 WRITE (6,7025)
      RETURN
 7030 WRITE (6,7035)
      RETURN
 7005 FORMAT (1H, 50X,'·····THE STACK (ISTCON) IS
      UNDERFLOW. ')
 7015 FORMAT (1H, 50X,'·····THE STACK (ISTCON) IS
      OVERFLOW. ')
 7025 FORMAT (1H, 50X,'·····THE STACK ( KRT ) IS
      OVERFLOW. ')
 7035 FORMAT (1H, 50X,'·····THE STACK ( ISTK ) IS
      OVERFLOW. ')
      END
```

図 7 生成された関数 ADD

Fig. 7 Sub-program (ADD) of FORTRAN to be produced by preprocessor.

```
      INTEGER FUNCTION REVL (A, B)
      COMMON/INPLIC/IAL (5000, 3), IBFSB, IBFS,
      NOL, IBL (1000,2), IBLP, IEC
      INTEGER CDR, CONS, CAR
      LOGICAL NULL
      INTEGER A, B
      LOGICAL LW 1
      IA=A
      IB=B
 1000 LW 1=NULL (    IA)
      IF (. NOT. LW 1) GO TO 2000
      NN=IB
      REVL=NN
      GO TO 6000
 2000 IW 1=CDR  (     IA)
      IW 2=CAR  (     IA)
      IW 3=CONS (    IW 2,     IB)
      IA=IW 1
      IB=IW 3
      GO TO 1000
 6000 CONTINUE
      RETURN
      END
```

図 8 生成された関数 REVL

Fig. 8 Sub-program (REVL) of FORTRAN to be produced by preprocessor.

```
      LOGICAL FUNCTION MEMBER (M, L)
      COMMON/INPLIC/IAL (5000,3), IBFSB, IBFS, NOL,
      IBL (1000,2), IBLP, IEC
      INTEGER CAR, CDR
      LOGICAL NULL, EQ
      LOGICAL LW 1
      LOGICAL LL
      IM=M
      IL=L
 1000 LW 1=NULL (      IL)
      IF (.NOT.LW 1) GO TO 2000
      LL=.FALSE.
      MEMAER=LL
      GO TO 6000
 2000 IW 1=CAR    (       IL)
      LW 1=EQ     (      IM,      IW 1)
      IF (.NOT.LW 1) GO TO 3000
      LL=.TRUE.
      MEMBER=LL
      GO TO 6000
 3000 IW 1=CDR  (      IL)
      IM=IM
      IL=IW 1
      GO TO 1000
 6000 CONTINUE
      RETURN
      END
```

図 9 生成された関数 MEMBER

Fig. 9 Sub-program (MEMBER) of FORTRAN to be produced by preprocessor.

6.3 リスト FORTRAN の実行過程

リスト FORTRAN プログラムはプリプロセッサ（すでにロードモジュールで格納されている）で FORTRAN 文に展開され，FORTRAN コンパイラで翻訳されたのち，すでにリロケータブル・オブジェクトの形で格納されているリスト処理用手続きなどのライブラリと結合編集し実行する．（**図 10** 参照）

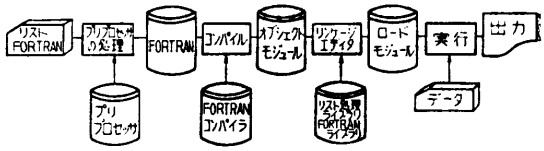

図 10 リスト FORTRAN の実行過程

Fig. 10 Processing List-FORTRAN.

7. おわりに

ゲームプレイングプログラムなどを計算言語のひとつである FORTRAN で記述すると，データを木構造で表現したり，あるいは帰納的に処理したりするプログラムの作成には多くの労力と時間を要する．

他方，これらの処理に LISP のようなリスト処理言語を用いれば，数値計算の部分の記述には括弧が多くなり見難く，さらに計算時間が多くかかる．（残念ながら当学の中型計算機システムには LISP は備わっていない．）

このような観点から，筆者等は比較的大衆化されている計算言語 FORTRAN に基本的なリスト処理機能（リストの入出力，リストに対する演算，帰納的関数の定義）を付加しえるように FORTRAN を拡張し，数値処理ならびにリスト処理を効率よく処理できる言語（仮りにリスト FORTRAN と呼ぶ）を設定し作成した．

この言語は FORTRAN を理解している者には，LISP の基本的な機能と簡単な記述法を学ぶだけでリストの内部構造には一切触れることなく，比較的容易に数値計算を含むリスト演算を行うプログラムを作成することができる．

さらに，比較的小型のシステムでも FORTRAN コンパイラを持つシステムでは機種を問わず使用することができる．

筆者等はこのリスト FORTRAN の現在の機能で当面の問題には十分対処することができると考えている．

しかし，帰納的機能を備えていない FORTRAN にリスト処理機能を埋め込んだ形でリスト FORTRAN を形成しているので，プリプロセッサの大半が帰納的演算を許した外部関数定義文の解釈と処理のために占められる．

この点は文字処理機能あるいは帰納的処理機能を備

えた言語のひとつである PL/I にリスト処理機能を埋め込めば，プリプロセッサおよびリスト処理手続きなどの作成において労力と時間が減少できる．

これらを検討した結果，PL/I コンパイラを持つシステムを対象に，新たに PL/I にリスト処理機能を埋め込んだ言語（仮りにリスト PL/I と呼ぶ．）を設定し，検討を行っている．

最後に，当研究室の大学院生，畑周二（現在，富士通），佐藤仁勇（現在，NHK）両君の協力に対して感謝の意を表わす．

参 考 文 献

1) The M. I. T. Press: LISP 1.5 Programmer's Manual (1975).
2) Maurer W. D.: The Programmer's Introduction to LISP, Macdnald/American Else Vier Inc. (1971).
3) 浅井清ほか：記号処理言語，総合図書 (1971).
4) 松山，中村：リスト FORTRAN，電気学会情報処理研究会資料，IP-78-73 (1978).

(昭和 54 年 2 月 26 日受付)
(昭和 54 年 8 月 23 日採録)