JAPLA研究会資料 2018/10/30

# Jのランクとは－名詞と動詞とから考える

西川　利男

## はじまり

先月のJAPLAの例会で、鳥邊錬太郎氏より次のような質問を受けた。
Jの動詞を定義して、数値のときはふつうに実行できたのに、数値を名前をつけた名詞にして実行しようとするとエラーになる。区切りをスペースでなく、コンマにすればOKである。

```
   f =: 3 : 0
+/ *: y
)
   f 1.2 0.3 _4
17.53
```

ところが

```
   a =: 1.2
   b =: 0.3
   c =: _4
```

として、実行してみる。 エラーになる。

```
   f a b c
|syntax error
|       f a b c
```

ところが、間にコンマをいれれば、エラーなしに実行できる。

```
   f a, b, c
17.53
```

言われてみるとふしぎである。このふしぎを解くカギはJのランクにある。これを機会に私自身、このテーマをゆっくり検討してみた。

なお、手元にJ Quick Referenceをおいて、参照されることをおすすめする。

## １．J では名詞と動詞(品詞)を考えプログラムする

従来の言語では、データ、値　および　関数　処理　手続きなどと呼ばれるものを
J では自然言語の品詞にならって、名詞と動詞、さらに副詞、接続詞がある。

**名詞(noun) = データ、値**

1.2, 0.3, _4(負の数)

a =: 1.2　これを代名詞(pronoun) と呼ぶこともある。

なお、J ではほかの言語(C など)で用いる int, float など型宣言を必要としない。

**動詞(verb) ＝ 関数、処理、手続き、メインルーチン、サブスブルーチン**

+ - * % *: %: i. #　…　プリミティブ(primitive) J に常備されているもの

ユーザ定義動詞　先の f など、名前を付けて定義する、proverb とも呼ばれる。

動詞の働きを拡張、変化させるものに、副詞と接続詞がある。

**副詞(adverb) ＝** 動詞を左引数にとり、動詞を拡張する。たとえば +/　としての使用。

**接続詞(conjugation) ＝** 動詞を左引数として、さらに右引数をとり、動詞の機能を変化させる。本稿のテーマであるランク(")が接続詞である。

## ２．J は関数型言語である

J の処理は、「**入力した値に、処理を行い、出力を返す**」を基本として操作される。
プログラミング言語では、引数(argument)、動詞、返り値(return)に対応する。

入力 → 処理 →　出力

Input → Function → Output

arguments → verb　→　returns

上の方式で、処理する言語を関数型言語(Applicative Language)という。これ以外の通常の言語は手続き型言語(Procedure Language)ある。

関数型言語には Lisp, APL, J などがある。一方、FORTRAN, COBOL, Pascal, Basic などはすべて手続き型言語である。また、C は main()で始めるように、一見関数型言語のように見えるが、関数の定義もあるが、必ずしも純粋ではなく、全体は手続き型言語である。

関数型言語の引数(argument)としては、一つに限らず複数とることができる。同様に返り値(return)も複数の値を返すことも可能である。

Lisp の関数では、その右引数として一つだけでなく、複数とることが可能である。一方 APL, J の関数では、右引数だけでなく、右引数と左引数と２つとることができる。

J ではこの考え方で、四則演算の操作も関数と考える。例えば

12.3 + 4.5 は　左引数 12.3　右引数 4.5 として、これに 関数 + を作用させて、その結果として、返り値 16.8 が得られると、考えるのである。

## ３．J におけるランク(rank)とは

J では、名詞である引数と返り値、および操作する動詞の両方について、ランクの考えの上に処理される。

### ３．１　名詞に対するランク

J では、いろいろな大きさ、形の値＝名詞が存在する。これらをきめるのがランクである。

・スカラーまたはアトム…単一の値　3.14　　=> ランク　0

・ベクトルまたはリスト… 複数の値が一方向だけに並んだもの 1, 2, 3 => ランク　1

・アレーまたは配列　… 複数の値がタテヨコ方向に並んだもの => ランク　2

```
A =: i. 2 3
   A
0 1 2
3 4 5
```

・3 次元以上のアレーまたは配列　=> ランクはその次元数になる

### ３．２　動詞に対するランク

J では、動詞に対してもランクが定められていて、作用する名詞のランクに対応していなくてはならない。

数学計算の動詞では、ランクは 0, 1, 2 のいずれでもよい。

たとえば、２乗の動詞 *: ではつぎのようになる。

```
  *: 1.2
1.44
  *: 2, 3
4 9
```

先の配列 A に対して

```
  *: A
0  1  4
9 16 25
```

### ３．３　名詞と動詞とのランクの係わり合い

合計する動詞 sum を定義して、さらにランクをいろいろ指定してやってみよう。

```
sum =: 3 : 0
+/ y
)
```

この定義で、/　は副詞と呼ばれ、演算＋値の間にあるものとして計算されるので合計が求められる。

ランクを指定するには " を用いる。これは接続詞であり、動詞を左にランク数を右とした書式をとり、このような機能を持った動詞として操作される。

先の配列 A を用いて示してみる。

```
  A
0 1 2
3 4 5
```

配列のヨコ方向（行にそって）の和はランク 1 を指定して求める。これは配列をランク 1 のベクトルの集合として作用する、と考えることができる。

```
  sum"(1) A
3  12
```

となる。

一方、タテ方向（列にそって）の和はランク 2 を指定して求める。これは配列をそのまま、ランク 2 として作用すると、考える。

```
  sum"(2) A
3 5 7
```

なお、ランクを指定しないときは、配列そのもののランクが指定されたとして、タテ方向の合計が得られる。　ふつうは、このように操作される。

```
  sum A
3 5 7
```

なお、念のためランク 0 を指定すると、配列のアトムだけに作用するので、それぞれの値が返される。

```
  sum"(0) A
0 1 2
3 4 5
```

このように、ランクとは配列言語において、動詞の処理の対象を、配列の形に応じて、きめこまかに操作する極めて重要な役割を持つ。

また、配列関数型言語 J では、動詞の処理により、名詞はその形＝ランクが変わりうるものであることにも注意が必要である。

まず、プリミティブ動詞 i. では、スカラーの値(ランク 0)を入力して、ベクトル(ランク 1)の値が出力される。

```
   i. 5    =>  0 1 2 3 4
```

四則演算の動詞では左１つと右１つの引数をとって、１つの値が返される。

```
  2 + 3  =>  5
```

さらに、次のようなことが可能である。動詞+の左引数はランク 0 のスカラーだが、右引数の名詞はランク 1 のベクトルであり、結果はベクトルである。

```
   2 + 3,4  =>  5 6
```

また、合計の定義動詞 sum では、ベクトル(ランク 1)の入力値にたいして、スカラーの値(ランク 0)の合計結果が返される。

```
  sum 2, 3, 4  =>  9
```

プリミティブ動詞のランクを知るにはどうするか。　[Help]で見られるプリミティブ項目の説明には、たとえば、プリミティブ i. では次のようになっている。

```
   i.  1 _ _
```

これは、単項動詞としては、ランク 1 つまりベクトルの値をとる。一方、２項動詞の左、右の引数のランクは無限大(_)、0 でも 1 でも、つまりなんでもよい、ことを示す。

## ４．tacit 形式プログラミングの場合のランクの注意

J の tacit 形式のプログラム定義とは動詞の引数をあらわに示さない形式で、頻繁に使う定義などで、かってはよく使われたが、初心者にはあまり勧められない。

２乗和を求めるのに、本稿の最初に示したとおりでなく tacit 形式では次のようにする。

```
  g =: (+/)@:(*:)
  g 1.2 0.3 _4
17.53
```

のように接続詞@:では、うまく計算するが、接続詞@ではうまくいかない。

```
  g1 =: (+/)@(*:)
  g1 1.2 0.3 _4
1.44 0.09 16
```

その理由は動詞２乗(*:)と和(+/)とでは、引数としてとる名詞のランクが 0 と 1 と異なるからである。接続詞@:と@をそのために使い分けている。