

Jによる Personal & Recreational Programming のすすめ

西川 利男

パソコン = personal computer, PC なることばが使われるようになってから、どの位経つだろうか。

かつてはコンピュータと言えば、大学、大会社、銀行などで、ガラス張りの空調のきいた一種の聖域であるコンピュータセンターに鎮座した神様、仏様であった。ここでは磁気テープがめまぐるしく回り、ラインプリンターからは出力の紙が滝のようにほとぼしり、中央にはたくさんのランプが点滅したコンソールなる祭壇があった。そして、そこにはたらくプログラマなる祭司たちは憧れのまとだった。

それと同規模の性能のパソコンが、私の目の前の机の上にある。パソコンとは personal (個人) のコンピュータである。決して、インターネットなるマス・プロダクションの社会に連なり、奉仕するターミナルなどではない。

Jによる Personal な Programming を、今あらためて強調したいと思う。数年前、やっとの思いで、DOS マシンの上で BASIC のプログラムを動かしたときの感激を忘れてしまったのだろうか。その BASIC は Visual BASIC などと進化はしたが、かえって手に負えぬブラックボックスになってしまった。

J のプログラミング環境こそ、今や最も使い勝手の良い Personal な楽園といえよう。ときには、「ボケ予防」と言われることもあるが、「ボケ予防」しごく結構ではないか。Personal さらには Recreational な Programming は「ものづくり」としての生きがいになる。市販の工業製品では味わえない Art としてのプログラミングの楽しみがある。

ご参考までに、お恥ずかしいが、私の J の Personal Programming の接し方を内緒で白状しよう。

・プログラミングはもっぱら explicit でコーディングする。引数をあらわに指示しただけで、ずっと安全なプログラムが作れる。そして、後での利用、メンテナンスにもこの方がよい。

・ループ構造、if-then 構造はどしどし使う。

・fork, hook など tacit プログラムは特別なとき以外は使わない。

・キー入力 rd =: 1!:1 や出力 wr =: 1!:2&2 を適宜用いて、プログラム途中をトレースしながら作成する。

要は安全に動かすのが、第一であり、長くてみっともないときもあるが、それはしかたがない。時間があつたら、エレガントなしゃれたコーディングにしたい気持ちもないことはないが…。

ここでは、Jによる手作りプログラミングのいくつかの例を紹介しよう。

・primes - shieve エラトステネスのふるいにより素数を求める。

Jではふつう $p: \wedge(1) 10 \Rightarrow 2\ 3\ 5\ 7$ として求められる。

・cross

タテ・ヨコの文字配列で、文字の一致、不一致をテストして表示

・hanoi's tower

再帰呼び出しのプログラム

NB. Primes by Sieve

```
wr =: 1!:(2&2
```

```
primes =: 3 : 0
```

```
N =. 2 + i. <: y.
```

```
PR =. ''
```

```
i =. 0
```

```
while. i < y.
```

```
do.
```

```
    P =. {. N
```

```
wr    PR =. PR, P
```

```
wr    N =. (-. 0 = P|N) # N
```

```
wr    '-----'
```

```
    if. 0 = #N do. goto_fin. end.
```

```
    i =. i + 1
```

```
end.
```

```
label_fin.
```

```
'primes = ', ": PR
```

```
)
```

```
primes 20
```

```
2
```

```
3 5 7 9 11 13 15 17 19
```

```
-----
```

```
2 3
```

```
5 7 11 13 17 19
```

```
-----
```

```
2 3 5
```

```
7 11 13 17 19
```

```
-----
```

```
2 3 5 7
```

```
11 13 17 19
```

```
-----
```

```
2 3 5 7 11
```

```
13 17 19
```

```
-----
```

```
2 3 5 7 11 13
```

```
17 19
```

```
-----
```

```
2 3 5 7 11 13 17
```

```
19
```

```
-----
```

```
2 3 5 7 11 13 17 19
```

```
-----
```

```
primes = 2 3 5 7 11 13 17 19
```

NB. cross.ijs

```
NB. cross 'JAPLA';'APL'  
cross =: 3 : 0  
'P Q' =: y.  
PQ0 =: P ="(1 0) Q  
IPQ0 =: { (i. #Q);(i. #P)  
IPQ =: (,PQ0) # (,IPQ0)  
(P ; Q) dis"(1) > IPQ  
)
```

```
NB. ('JAPLA';'APL') dis 1 2  
dis =: 3 : 0  
:  
'I J' =: y.  
'R S' =: x.  
CRO =: ((#S), (#R))$' '  
CR1 =: S (<(i.#S);J) } CRO  
CR2 =: R I } CR1  
)
```

```
cross 'JAPLA';'APL'  
JAPLA  
P  
L  
  
JAPLA  
P  
L  
  
A  
JAPLA  
L  
  
A  
P  
JAPLA
```

途中経過

PQ0

0 1 0 0 1

0 0 1 0 0

0 0 0 1 0

IPQ0

+---+---+---+---+---+

|0 0|0 1|0 2|0 3|0 4|

+---+---+---+---+---+

|1 0|1 1|1 2|1 3|1 4|

+---+---+---+---+---+

|2 0|2 1|2 2|2 3|2 4|

+---+---+---+---+---+

IPQ

+---+---+---+---+

|0 1|0 4|1 2|2 3|

+---+---+---+---+

('JAPLA' ; 'APL') dis 0 4

JAPLA

P

L

('JAPLA' ; 'APL') dis 1 2

A

JAPLA

L

NB. Hanoi's Tower

```
wr =: 1!:2&2
```

```
hanoi =: 3 : 0
```

```
:
```

```
'X Y Z' =. x.
```

```
if. 0 = y.
```

```
do. '*** end ***' return.
```

```
else.
```

```
(X, Z, Y) hanoi y. - 1
```

```
wr 'move ', (': y.), ' from ', X, ' to ', Z
```

```
(Y, X, Z) hanoi y. - 1
```

```
end.
```

```
)
```

```
'ABC' hanoi 3
```

```
move 1 from A to C
```

```
move 2 from A to B
```

```
move 1 from C to B
```

```
move 3 from A to C
```

```
move 1 from B to A
```

```
move 2 from B to C
```

```
move 1 from A to C
```

```
*** end ***
```

```
'ABC' hanoi 4
```

```
move 1 from A to B
```

```
move 2 from A to C
```

```
move 1 from B to C
```

```
move 3 from A to B
```

```
move 1 from C to A
```

```
move 2 from C to B
```

```
move 1 from A to B
```

```
move 4 from A to C
```

```
move 1 from B to C
```

```
move 2 from B to A
```

```
move 1 from C to A
```

```
move 3 from B to C
```

```
move 1 from A to B
```

```
move 2 from A to C
```

```
move 1 from B to C
```

```
*** end ***
```

