J 言語でエクササイズ (2)
(ボックスを用いた配列演算とループ演算)
デュドニーの数学パズルから

SHIMURA Masato
jcd02773@nifty.ne.jp

2012 年 2 月 26 日

目次

概要

1917 年に出されたデュドニーの問題は組み合わせに関するプログラムの練習に格好の課題である

1 デュドニーの問題

アメリカンサイエンスに永く連載された「マーチン · ガードナーの数学ゲーム」のうち初期に連載された赤攝也 · 冬子訳の 22 編が 2011 年末に「II(新装版)] として日経サイエンス社から出版された。いずれの章も水準の高い力作である。その幾つかを配列演算言語 J で料理してみよう。

このデュドニーの問題は「第 3 話 ビール缶 どこまで飲めば安定に」の第 7 問である [*1]

[*1] この問題は *Henry Ernst Dudeney*) が [Amusements in Mathematics](1917) に発表したものの No81 に載っているとのことである

次のように 1 から 9 までの 9 個の数字が次のように 2 のグループに分かれている。左に 3 桁が入る。各数を 1 回ずつ使用して掛け算を行い、左と右の答えが同じになる最大の組み合わせを求めよ。	各数は次のように引用する $\begin{array}{rcl} & abc & \\ \times & de & \times \quad fg \\ \hline \end{array}$

例題 .

$$\begin{array}{rcl} & 158 & \\ \times & 23 & \times \quad 79 \\ \hline 3634 & = & 3634 \end{array}$$

デュドニー デュドニーは「ある判断力と忍耐力とを発揮しなければ見つかりっこない」と述べて次の解を与えた

$$\begin{array}{rcl} & 174 & \\ \times & 32 & \times \quad 96 \\ \hline 5568 & = & 5568 \end{array}$$

ミーリー アイルランド・ダブリンのミーリーがその後出した解

$$\begin{array}{rcl} & 584 & \\ \times & 12 & \times \quad 73 \\ \hline 7008 & = & 7008 \end{array}$$

藤村の友人 日本のパズルの権威 藤村幸三郎の友人が求めた解 (1971)。

ガードナーは次のように述べてこれを課題としている

[*2]

- まだ証明はされていないが最大解であろう
- この解を求め、さらに大きい解が有るか否かを探索せよ
- 「読者はコンピューターの助けを借りずにこの解を見つけられるであろうか」

2 J のトレーニング

2.1 階乗と組み合わせ

階乗 9 の階乗 (9!)

J では ! 9

```
   ! 9
362880
```

組み合わせの J のイディオム *tap Table of Pertations*

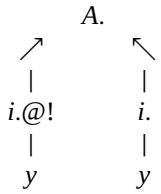
[*2] 藤村幸三郎 数学パズルの古典的著書多数 デュドニーの訳書もある

```
tap=:i.@! A. i.
```

```
   tap 3
0 1 2
0 2 1
1 0 2
1 2 0
2 0 1
2 1 0
```

次の *folk* を構成する。

```
i.@! -> 0 1 2 3 4 5
```



辞書式順序　*Anagram A.*

```
 7{.  tap 4
0 1 2 3
0 1 3 2
0 2 1 3
0 2 3 1
0 3 1 2
0 3 2 1
1 0 2 3
```

```
   A. 0 3 2 1
5
```

NB. **0 オリジンで 5 番目**

1 から 9 までの tap　階乗の個数と一致する

```
   $  tap 9
362880 9
```

```
   5{. tap 9  NB. 最初 5 個
0 1 2 3 4 5 6 7 8
0 1 2 3 4 5 6 8 7
0 1 2 3 4 5 7 6 8
0 1 2 3 4 5 7 8 6
0 1 2 3 4 5 8 6 7
```

2.2　ボックスは第 3 の型

第 3 の型　J は数値、文字列に続く第 3 の型としてボックスをサポートしている。

混合配列　ボックスは文字列と数値の混合配列とすることができる。

ネスティッド · アレー　配列の要素にベクトルや配列やを用いるネステッド · アレーとして用いることができる。

APL の混合配列やネスティッド · アレーはシームレスになっている

ボックスに構造を持たせる　.

- 3 次元マトリクスをボックスで囲むと取り扱いやすい、
- 同時処理にも活用できる。ボックスの中の各要素に作用させるには、(L:0) を用いる
- 2 重ボックスなど工夫次第で便利に使える

Each Box　ループを使わない同時処理を行うため、配列の各要素をボックス化する

```
 {@>  tap 3                          3, L:0{@>  tap 3
+-+-+-+                             +---+---+---+
|0|1|2|                             |3 0|3 1|3 2|
+-+-+-+                             +---+---+---+
|0|2|1|                             |3 0|3 2|3 1|
+-+-+-+                             +---+---+---+
|1|0|2|                             |3 1|3 0|3 2|
+-+-+-+                             +---+---+---+
|1|2|0|                             |3 1|3 2|3 0|
+-+-+-+                             +---+---+---+
|2|0|1|                             |3 2|3 0|3 1|
+-+-+-+                             +---+---+---+
|2|1|0|                             |3 2|3 1|3 0|
+-+-+-+                             +---+---+---+
```

ボックス化　ボックスを作る動詞など

box 化	>	
Link	;	両項
Catalogue	{	
each – box	{@>	

```
   1 2;3 4;5 6
+---+---+---+
|1 2|3 4|5 6|
+---+---+---+
```

ボックスを開く　.

open	<	
raze	;	単項
	;(”2)	*each – box* を形を残して開く

ボックスに関連する動詞など

Level – of	*L.*	多重ボックスの深度を測る
Level – at	Ł : 0	各ボックスの中で計算
Ravel	, *.and*, ./	ボックスを縦に並べる

3　配列のみで計算する

36 万の組み合わせがあるが、最も力技は配列を用いて、ループを全く使用しない方法で、スクリプトはスマートになり、オーバーラップが少ない分計算時間も短いが、組合せの個数が増えると急激にコンピューターの資源を消費する。

3.1　組み合わせのスクリプト

3.1.1　経過と解説

tap　.

```
tap=: i.@! A. i.           NB. Table of all permutations
```

tab_all　.

- tap 9 を打ち出し
- 3,2,2,2 の区切りでボックスに入れ
- 各行もボックス化して
- 3 桁 (a,b,c) と 2 桁 (d,e)(f,g)(h,i) の数値に直す

```
tab_all=: 3 : 0
tmp=. |: > { L:0  |:(L:0) 1 0 0 1 0 1 0 1 0 <;.1 |: >: tap 9
(3 numerical_sub {."1 tmp),. 2 numerical_sub }."1 tmp
)
```

リストを 1 つの数に改める　例えばリスト (1 2 3) を (123) に、(7 8) を (78) に変える。

- 基底を用いる

```
 10 #.  3 2 1
321
```

この汎用の方法はメモリーを大量に消費しこの例題では *out of memory* になるので次を作成した

- 簡易な基底変換のスクリプト。*3* 個と *2* 個に限定

```
 numerical_sub=: 4 : 0
NB. x is 3 or 2
NB. y is  table_abc_sub ''
if. 3 = x do.  +/ (L:0) 100 10 1 *  L:0  y    NB. 3 桁用
 else.  +/ (L:0) 10 1 * L:0 y                 NB. 2 桁用
end.
)
```

calc_ans_sub　.

3×2 と 2×2 の組の乗算を行うサブプログラム。36 万行を一度に行う

```
calc_ans_sub=: 3 : '(*/"1 ;("1) 0 1{"1 y),.*/"1 ;("1) 2 3{"1 y'
```

メインプログラム　(パン焼き釜=バッチ)

- 乗算の結果を=で照査し、0/1 のインデックスを作成し、1 を取り出す
- =の 24 ケースの乗算結果を表示する

```
calc_tab=: 3 : 0
tmp0=. tab_all ''
tmp1=. calc_ans_sub tmp0
ind=. =/"1 tmp1
tmp2=.ind #  tmp0
tmp2 ,. {calc_ans_sub tmp2
)
```

3.1.2　解

解は次の 22 組で 11 通りである。

```
   calc_tab ''
+---+--+--+--+---------+
|134|29|58|67|3886 3886|
+---+--+--+--+---------+
|134|29|67|58|3886 3886|
+---+--+--+--+---------+
|138|27|54|69|3726 3726|
+---+--+--+--+---------+
|138|27|69|54|3726 3726|
+---+--+--+--+---------+
|146|29|58|73|4234 4234|
+---+--+--+--+---------+
|146|29|73|58|4234 4234|
+---+--+--+--+---------+
|158|23|46|79|3634 3634|
+---+--+--+--+---------+
```

```
|158|23|79|46|3634 3634|
+---+--+--+--+---------+
|158|32|64|79|5056 5056|
+---+--+--+--+---------+
|158|32|79|64|5056 5056|
+---+--+--+--+---------+
|174|23|58|69|4002 4002|
+---+--+--+--+---------+
|174|23|69|58|4002 4002|
+---+--+--+--+---------+
|174|32|58|96|5568 5568|
+---+--+--+--+---------+
|174|32|96|58|5568 5568|
+---+--+--+--+---------+
```

```
|186|27|54|93|5022 5022|
+---+--+--+--+---------+
|186|27|93|54|5022 5022|
+---+--+--+--+---------+
|259|18|63|74|4662 4662|
+---+--+--+--+---------+
|259|18|74|63|4662 4662|
+---+--+--+--+---------+
|532|14|76|98|7448 7448|
+---+--+--+--+---------+
|532|14|98|76|7448 7448|
+---+--+--+--+---------+
|584|12|73|96|7008 7008|
+---+--+--+--+---------+
|584|12|96|73|7008 7008|
+---+--+--+--+---------+
```

3.1.3 **最大解**

最大解はやはり藤村の友人であった。

$$\begin{array}{rrrr} & 5 & 3 & 2 \\ \times & & 1 & 4 \\ \hline 7 & 4 & 4 & 8 \end{array} \quad = \quad \begin{array}{rrrr} & & 7 & 6 \\ \times & & 9 & 8 \\ \hline 7 & 4 & 4 & 8 \end{array}$$

4 Loop で計算する

4.1 Script そして経過と解説

最初の 3 桁 (a,b,c)　1 から 9 の異なる数で 3 桁の数 (a,b,c) の組み合わせを作る。$9 \times 8 \times 7 = 504$ 通りの組み合わせ

```
NB. Loop method
table_123_sub=: 3 : 0
NB. u ''
NB. n=9*8*7=504
NR9=. >:i.9
tab_0=. > { L:0({@> NR9),. L:0 NR9 -. (L:0){@> NR9
tmp0=. NR9 -. L:0 tab_0
tmp1=. |: (L:0) 7# "0 (L:0) tab_0
,.> { L:0 tmp1,. L:0 tmp0
)
```

```
  5{.  table_123_sub ''
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|1 2 3|1 2 4|1 2 5|1 2 6|1 2 7|1 2 8|1 2 9|1 3 2|1 3 4|1 3 5|1 3 6|1 3 7|1 3 8|1 3 9|1 4 2|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|2 1 3|2 1 4|2 1 5|2 1 6|2 1 7|2 1 8|2 1 9|2 3 1|2 3 4|2 3 5|2 3 6|2 3 7|2 3 8|2 3 9|2 4 1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|3 1 2|3 1 4|3 1 5|3 1 6|3 1 7|3 1 8|3 1 9|3 2 1|3 2 4|3 2 5|3 2 6|3 2 7|3 2 8|3 2 9|3 4 1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|4 1 2|4 1 3|4 1 5|4 1 6|4 1 7|4 1 8|4 1 9|4 2 1|4 2 3|4 2 5|4 2 6|4 2 7|4 2 8|4 2 9|4 3 1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
|5 1 2|5 1 3|5 1 4|5 1 6|5 1 7|5 1 8|5 1 9|5 2 1|5 2 3|5 2 4|5 2 6|5 2 7|5 2 8|5 2 9|5 3 1|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

9×56 である

2 桁の組み合わせ　table_45_sub

(a,b,c) を除いた (d,e)(f,g)(h,i) の候補の数値

```
table_de_sub=: 3 : ' (>: i.9) -. L:0 y'
table_de_all=: table_de_sub
 &table_abc_sub
```

```
3{."1] 5{.  table_de_all ''
+-----------+-----------+-----------+
|4 5 6 7 8 9|3 5 6 7 8 9|3 4 6 7 8 9|
+-----------+-----------+-----------+
|4 5 6 7 8 9|3 5 6 7 8 9|3 4 6 7 8 9|
+-----------+-----------+-----------+
|4 5 6 7 8 9|2 5 6 7 8 9|2 4 6 7 8 9|
+-----------+-----------+-----------+
|3 5 6 7 8 9|2 5 6 7 8 9|2 3 6 7 8 9|
+-----------+-----------+-----------+
|3 4 6 7 8 9|2 4 6 7 8 9|2 3 6 7 8 9|
+-----------+-----------+-----------+
```

tap 6　6 個の組み合わせの数は各々 720 ある。全てを作る

```
  $ tap 6
720 6

 table_d2i_sub=: 3 : '|: >{ L:0 (tap 6) { L:0 y'
 NB. y is table_de_sub ''
```

リストを 1 つの数に改める　例えばリスト (1 2 3) を (123) に、(7 8) を (78) に変える。

こちらでは基底変換を用いる

メインスクリプト　2 個のループを用いている

```
calc_loop_main=: 3 : 0
NB. u ''
Mabc =., table_abc_sub ''              NB. 9 * 56 (a,b,c) を求める
Mde =. ,table_de_sub&table_abc_sub '' NB. 9 * 56 (d,e)(f,g)(h,i) の候補
Mabc=.3 numerical_sub  Mabc            NB. 3 dights 3 桁の数に
ANS=.<''                               NB. 解の受け皿
for_ctr0. i. # Mabc do.                NB.ctr0= 504 回のループ
 TMabc=. ctr0{Mabc                     NB.1/720 ずつ取り出し
 TMd2i=. table_d2i_sub ctr0{Mde        NB. 上に対する (d,e)(f,g)(h,i) を 720 個作成
 for_ctr1. i.# TMd2i  do.              NB. ctr1=720 回のループ
tmp=. 2 numerical_sub  1 0 1 0 1 0 <;.1 > ctr1{TM4to9   NB.(d,e)(f,g)(h,i) に区分
して数値化
tmp=. TMabc,tmp                        NB. (a,b,c) を上の先頭に付ける
ind=. (tmp3=. */ > 0 1 { tmp) =  tmp4=. */ >2 3 { tmp   NB. 掛け算して左右の一致を検
査する
  if. 1=ind do. ANS=. ANS, <tmp,(<tmp3),<tmp4 end.      NB. 一致したもののみ取り出す
 end.
end.
,.>,.}.ANS                             NB. 解の整形
)
```

4.2 解

解は等しい

```
   calc_loop_main ''
+---+--+--+--+----+----+
|134|29|58|67|3886|3886|
+---+--+--+--+----+----+
|134|29|67|58|3886|3886|
+---+--+--+--+----+----+
|138|27|54|69|3726|3726|
+---+--+--+--+----+----+
|138|27|69|54|3726|3726|
+---+--+--+--+----+----+
|146|29|58|73|4234|4234|
+---+--+--+--+----+----+
|146|29|73|58|4234|4234|
+---+--+--+--+----+----+
```

```
|158|23|46|79|3634|3634|
+---+--+--+--+----+----+
|158|23|79|46|3634|3634|
+---+--+--+--+----+----+
|158|32|64|79|5056|5056|
+---+--+--+--+----+----+
|158|32|79|64|5056|5056|
+---+--+--+--+----+----+
|174|23|58|69|4002|4002|
+---+--+--+--+----+----+
|174|23|69|58|4002|4002|
+---+--+--+--+----+----+
|174|32|58|96|5568|5568|
+---+--+--+--+----+----+
|174|32|96|58|5568|5568|
+---+--+--+--+----+----+
```

```
|186|27|54|93|5022|5022|
+---+--+--+--+----+----+
|186|27|93|54|5022|5022|
+---+--+--+--+----+----+
|259|18|63|74|4662|4662|
+---+--+--+--+----+----+
|259|18|74|63|4662|4662|
+---+--+--+--+----+----+
|532|14|76|98|7448|7448|
+---+--+--+--+----+----+
|532|14|98|76|7448|7448|
+---+--+--+--+----+----+
|584|12|73|96|7008|7008|
+---+--+--+--+----+----+
|584|12|96|73|7008|7008|
+---+--+--+--+----+----+
```

まとめ

処理時間時間と使用スペースを計測する外部接続詞

`ts=:6!:2 , 7!:2@]` NB. time and space

配列	`ts 'calc_tab 1'` `11.7984 2.79181e8`	時間は 11.8sec 使用 CPU はかっての名作 Pentium-M 1.8GHZ
ループ	`ts 'calc_loop_main 1'` `10.486 619968`	スピードも若干速く、使用スペースは $\frac{1}{45}$。*numerical_sub* を用いれば更にスペースは減る

References

マーチン・ガードナー/赤攝也・冬子訳「マーチン・ガードナーの数学ゲーム II(新装版)」日経サイエンス社 2011

J 言語の入手

`http://www.jsoftware.com` から DL する。

Script の入手

`http://japla.sakura.ne.jp` の workshop 01/2012