教育資料

Lotus 1—2—3 マクロ・プログラミング入門 (1)

藤田 芳夫*

ヴイジカルクからはじまった表計算ソフトはロータス1—2—3やその同類に至って、実用性が広く認識され、大いに普及するようになった。

ところが、表計算、グラフ、データベースさらに言えばワープロの機能も合せ持っているロータス1—2—3その他も、ある程度以上のことをプログラムして実行させようとすると案外難しい。このロータス1—2—3でプログラムを組むこと、即ちロータス1—2—3のマクロ命令の入門をできるだけ容易しく、しかもプログラミング技法（アルゴリズム）の一般的方法にそったものとして展開し、ロータス1—2—3やエクセルと言ったこの種の高度な「簡易」言語の入門を手助けしようとするのが本稿の目的である。

1．列巾変更マクロ

マクロ・プログラミングの最も簡単なもので、その効果が視覚的に明白なものとして、列巾変更マクロはマクロ・プログラミングの中でも白眉とでも言うべきものである。そこで、列巾変更マクロから入ることにする。

今、20行10列の計算表を作成することとする。A列からJ列までを一度に画面に表示するには各列の巾を調節しなければならない。

① まず、列巾変更の状態がよくわかるように、20行10列の計算表に、縦横の罫線を引いておく。このため次のコマンドを実行する。

/ WXLA0A1..J20 ↵ Q ↵

/……メイン・メニュー

W……ワーク・シート

X……罫線

L……線引き

A……格子

0……ゼロ（線の種類）

A1..J20……線引きの範囲

Q……終了

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4										
5										
⋮										
19										
20										
21										

* 東京情報大学教授

1993年1月11日受理

② A列からJ列まで、一度に画面に入れるには、すべて7バイト巾にすればよい。このためセルポインターをA列に位置づけ、

/ WCCSA1..J1 ↵ 7 ↵
/……メイン・メニュー
W……ワーク・シート
C……列巾表示
C……複数列
S……列巾変更
A1..J1……変更範囲
7……7バイトの列巾　　を実行する。

これによって、A列からJ列まで、すべて7バイト巾に変更できる。しかし、巾を変更すべき列が一ケ所にまとまっていない場合とか、各列によって、変更すべき列巾がちがう場合には、セルポインターを変更したい列に位置づけた上で、

③ / WCS 5 ↵　を実行する。
/……メイン・メニュー
W……ワーク・シート
C……列表示
S……列巾変更
5……5バイトの場合

この場合、A列、C列、E列を5バイトに変更するためには、三回同じ操作をしなければならない。そこで、これをマクロ命令として作成する。

④ 列巾変更マクロ命令の作成方法と使用方法は次の通りである。

④—① セルポインターを任意のセル（例えばB23）に置き、
④—② ' / WCS5 ~　を入力する
④—③セルB23にマクロ名¥C　をつける。
/ RNC¥C ↵ B23..B23 ↵
/……メイン・メニュー
R……範囲
N……範囲名
C……設定
¥C……列巾変更マクロ名
B23..B23……マクロ名の所在範囲
④—④セルA23にマクロ名¥Cのメモ書きを入力する（省略可能）
④—⑤列巾変更の実行
㋑任意の列へセルポインターを移す
㋺CTRL+C ↵　列巾を5バイトに縮少する。
④—⑥注意　マクロ命令¥Cの最後の～を省略すると、CTRL+C ↵の他にもう一回CRキーを押す。
④—⑦注意　マクロ命令¥Cでは、列巾は5バイトに固定している。これを任意の列巾にするには、
/ WCS
とする。最後のSで「列巾を指定して下さい」と言うプロンプトが出るので、列巾を入力して、CRを押す。
この場合、
/ WCS {?}~　又は
/ WCS {?} としてもよい。{?}~の場合は、たとえば「15」と入力するだけでよいが、{?}の場合には、「15」と入力した後で、CRを押さねばならない。

2．文字列のマクロ

住所録を作成するとき、東京都、埼玉県、千葉県など、都道府県名は頻繁に現われる。これを一々仮名入力し、漢字変換していては大変面倒である。そこで、下図のように、E列にマクロ名、F列に都道府県名を入れておき、住所録記入時に、東京都が必要になればCTRL+T、千葉県が必要になればCTRL+Cを押すのである。

	A	B	C	D	E	F
1	氏　名	住所	電話			
2	Xさん	東京都…	……		¥T	東京都
3	Yさん	千葉県…	……			
4	Zさん	埼玉県…	……		¥S	千葉県
5						
6					¥C	千葉県
7						

このため、

①セルポインターをE2へ置き、

②'¥Tを記入し、

③/ RNLR E2..E2 ↵　とする。これで東京都を記入する場所F2に¥Tと言うマクロ名をつけたことになる。

/……メイン・メニュー
R……範囲
N……範囲名
L……文字列
R……右側（マクロ名の設定位置）
E2..E2……文字列の場所（マクロ名の存在場所）

④セルポインターをF2へ移し、「東京都」を入力する。

ここで注意すべきは、東京都の後に、~をつけない方がよいと言う点である。何故なら、東京都の後に、市区町名を続けるのが普通であるからである。職員録の職名欄のように、部長、課長、係長と言う職名だけが単独で記入されるときには「部長~」とCRキーを入れる方がよい。

⑤　東京都、埼玉県、千葉県の間に少くとも1行以上の空白行を入れておかなければならない。

3．通貨記号と3桁カンマ・マクロ

前節では「東京都」、「埼玉県」と言った文字列に名前をつけ、任意のセルにその文字列を入力するマクロ命令を考えた。本節では任意のセルに数値を入力し、それに通貨記号「¥」をつけ、3桁毎にカンマを入れることを考えてみよう。

3－1．数値入力マクロ

まず、任意のセルに特定の数値たとえば1235を入力するマクロを考えよう。

1．任意のセル（たとえばB5）に'1235~を入力し、
2．B5のマクロ'1235~に¥Bと言う名前をつける。
　/ RNC¥Bb5..b5 ↵
3．¥Bのメモを A5に記入する（省略可能）。

以上により、任意のセル（たとえばE10）に数値1235を記入することができる。

3－2．通貨記号、3桁カンマ・マクロ

このE10に記入されている数値1235に通貨記号をつけ、3桁毎にカンマで区切り、小数1桁の表示をつけるとすれば、次の操作を行わねばならない。

/……メイン・メニュー
R……範囲
F……表示形式
C……通貨表示
1……小数桁1桁
～……小数桁の確定
～……表示内容を上記のように変更する

いまセルB6に上記のマクロを記入する。そして、¥Bと言うマクロ名をつける。

4．B6……/ RFC 1~~
5．B6にマクロ名¥Bをつける。
　/ RNC¥BB6..B6~
6．セルポインターをE10に移し、

CTRL＋B ↵を実行する。
7．E10の１２３５が¥1,235.0に変わる。

	A	B	C	D	E
1					
2					
3				¥9,876.0	
4					
5	¥A	'1235～			
6	¥B	/RFC1～			
7					
8					
9					
10					¥1,235.0

上記の作業でマクロ命令¥Aと¥Bは上図のように、B列の5行目と6行目に記入されている。

ロータス１２３のマクロは同一列に連続して記入してあると、連続して実行する。したがって、この場合、CTRL＋Aを実行すると、任意のセルに１２３５を記入し、続いて¥マークをつけ、3桁毎にカンマを挿入し、小数1桁の処理を行うことになる。

また、任意のセル（たとえばD3）に９８７６と言う数値が記入してあると、そのセルにセルポインターを合せ、CTRL＋Bで、それを¥9,876.0に変更することが出来る。

3－3．任意の数値を入力し、¥マークと3桁カンマを付けるマクロ

上例では入力される数値が１２３５に限定される。そこで、任意の数値が入力できるようにするには、

'/２３５~　を　{?}~

に変えるとよい。同様に小数桁数についても1を{?}~で置き替えるとよい。

	A	B
1		
2	¥C	{?}~
3		/RFC {?}~
4		
5		¥1,235

上図に示したマクロ¥Cでは、任意のセルに任意の数値を入力し、欲する小数桁数を入力すれば、直ちに、¥マーク付きで3桁カンマを付け、指示した小数桁数をつけて表示することができる。

たとえば、セルポインターをB5に置き、CTRL＋Cを実行し、1234.56と入力し、小数桁数ゼロを指定すると¥1,235が表示される。(小数部は四捨五入される点に注意)。

3－4．文字列と数値を入力し、¥マーク3桁カンマを付けるマクロ

上例では数値を入力し、それに通貨記号をつけ、3桁カンマで区切り、小数部をつけることを実行したが、文字列と数値を上下、又は左右に並べるように入力してもよい。たとえば下図のようにすると、セルポインターをC1に置き、CTRL＋Dを実行し、まず文字列「１９９３年予想売上高」を入力する。これが済むと、C列を２０バイト巾に拡げ、次にセルポインターが1行下がり、１２３４５６７を入力すると、¥マークを付け3桁毎にカンマで区切り、指示された小数部分をつけて表示してくれる。

	A	B	C
1			1993年予想売上高
2	¥D	{?}~	¥1,234,567
3		/WCS20~	
4		{d}~	
5		{?}~	
6		/rfc {?}~	
7			

マクロ命令¥Dでは文字列を入力し、その下の行に数値を入力したが、下図に示すように、文字列を入力し、その右側に数値を入力することも出来る。

	A	B	C	D
1			1993年度予想売上高	¥1,234,567
2	¥E	{?}~		
3		/WCS20~		
4		{r}~		
5		{?}~		
6		/rfc{?}~		
7				

4．入力プロンプト付きマクロ

前例では文字列や数値を入力するとき、入力プロンプトが現われないので、不便である。ロータス1－2－3には操作パネルに入力プロンプトが現われるマクロがある。それは

{getLabel 入力プロンプト，入力セル}　　と

{getNumber 入力プロンプト，入力セル} である。

getLabelは文字列の入力用に、getNumberは数値の入力に用いる。両者ともに、第1アーギュメントは入力プロンプトで、第2アーギュメントは入力セル又は入力すべき変数(範囲名)である。

したがって、入力セルに変数名(範囲名)を使用するときは、あらかじめ変数名を宣言しておかねばならない。また、セルポインタが変数の位置と異る場合には、たとえば列巾変更を行う場合には、{goto}文を使用しなければならない点に注意しなければならない。

	A	B
1	¥C	{getLabel モジ→,Title}
2		{goto} Title~
3		/WCS30~
4		{getNumber キンガク→,Amount}
5		{goto}Amount~
6		/rfc {?}~
7		
8		
9	Title	
10	Amount	

上の例では、

①まず、¥C, Title, Amount に範囲名を与えておく。

②セルポインターは任意のセルにあってよい。

③「モジ→」と言うプロンプトで「1993年度予想売上高」を記入し、

④{goto} Title ~ でセルポインターはB列に行き、列巾を30バイトに広げる。

⑤「キンガク→」と言うプロンプトで、１２３４５６７８９０と言う数値を入力すると、小数桁を要求するので、たとえば1桁と入力すると、¥１，２３４，５６７，８９０．０と言う結果がB10、すなわちAmountセルに現われる。

この例では{goto} Title ~、{goto} Amount ~によってセルポインターを必要な所へ移動している点がポイントである。

5．四則演算とマクロ(1)

本節ではロータス1－2－3で、四則演算を行う場合、マクロ・プログラミングの立場から見ると、どうなるかと言う点を検討してみる。

	A	B
1	A=	
2	B=	
3		
4	A+B=	
5	A−B=	
6	A*B=	
7	A / B	
8		

二つの変数AとBの和、差、積、商を求める場合、上図に示すように、B1にAを入れ、B2にBを入れ、和A+BをB4に、差A−BをB5に、積A＊BをB6に、そして商A/BをB7に計算する。このとき、説明として、

A1に　"A=
A2に　"B=
A4に　"A+B=
A5に　"A−B=
A6に　"A＊B=
A7に　"A / B=

を入れ、次に変数A、Bの計算式をB4、B5、B6、B7にそれぞれ

B4……+B1+B2
B5……+B1−B2
B6……+B1＊B2
B7……@if(B2=0,"",+B1/B2)

を入れておく。B7の式は変数Bがゼロの時、エラーにならないようにするためである。

この準備の後、セルポインターをB1に移し、変数Aを入力し、次にセルポインターをB2に移し、変数Bを入力すればよい。

したがって、この四則演算の実行をマクロプログラムで実行するためには、変数AとBをマクロで入力すればよい。

	A	B	C	D	E
1	A=	55		¥a	{goto} b1~ {?}~
2	B=	45			{goto} b2~ {?}~
3					
4	A+B=	100			
5	A−B=	10		¥b	{getNumber A=,b1}
6	A*B=	2475			{getNumber B=,b2}~
7	A / B=	1.22222			

このため、上図に示すように、

①セルポインターをD1に位置づけ、'¥aを入力する。
②/RNLRD1.D1でE2にマクロ名¥aをつける。
③セルポインターをE1に移す。
④ {goto} B1~ {?}~を入力する。
⑤セルポインターをE2に移す。
⑥ {goto} b2~ {?}~を入力する。
⑦以上により、CTRL+aを実行すればよい。

上記のマクロ・プログラムではデータを入力するとき、プロンプトが現れないので、何を入力したらよいのか分かり難く、不親切である。そこで {getNamber} を用いるとよい。

この入力プロンプトを表示するマクロ命令{getNamber}を使用した場合を¥bに示す。この¥bプログラムで注意すべきは、最後の {getNamber X, Y }~に、CR記号をつけている点である。これがないと、変数A、Bの入力だけで止まり、A+BからA/Bの計算に移るため、手動でCRキーを押さなければならない。

6．反覆計算と前データの消去

この簡単な四則演算プログラムを何回か繰り返し実行しようとすると、新データを入力する前に、古いデータを消去しておく方が混乱がなくて便利である。このためには、マクロプログラム¥Aの先頭にデータ消去命令

/ REb1..b2 ~ を入れるとよい。

/……メイン・メニュー
R……範囲
E……消去
b1..b2……消去範囲

また、¥Aの先頭にデータ消去命令を追加するのではなく、新しくマクロ・プログラム¥Bをつくり、その先頭に不要データ消去部分を置き、次にマクロ・プログラム¥Aを {¥A} ~で実行してもよい。これは言わばマクロ・プログラム¥Aをサブルーチン化する方法である。

	A	B	C	D	E
1	A=	123			
2	B=	45			
3					
4	A+B=	168		¥a	{getNumber A →,b1}
5	A−B=	78			{getNumber B →,b2}
6	A*B=	5535			
7	A / B=	2.73333		¥b	/reb1..b2 ~
8					{¥a} ~
9					
10					
11					
12					

注意　このデータ消去は

/ WDA { V B1..B2 / U B1..B2

を用いてもよいし、

{Blank b1..b2} としてもよい。

7．四則演算と Let 文

	A	B	C	D	E
1	A=	123		¥a	{getNumber A →,B1}
2	B=	0			{getNumber B →,B2}
3					{let B4, +b1+b2}
4	A+B=	123			{let B5, +b1−b2}
5	A−B=	123			{let B6, +b1 * b2}
6	A*B=	0			{let B7, @if(b2=0,"",+b/b2)}
7	A / B=				
8					

前節の四則演算プログラムでは変数A、Bの演算をそれぞれのシエルに書き込んだ。しかし、出力セルには演算結果さえあればよく、式はマクロ・プログラムの方にだけあればよい場合も多い。

こうした場合には、上掲のように、2変数A、Bの和、差、積、商はマクロ・プログラム¥Aのなかで、

{let b4, +b1+b2}
{let b5, +b1−b2}
{let b6, +b1 * b2}
{let b7, +b1 / b2}

のように、Let文の中に式をとり入れることによって実行することができる。

ただし、最後の演算については、ゼロで割ることがないように、

{let b7, @if(b2=0,"",+b1/b2)}

とする方がよい。

8．　Let 文と文字列とサブルーチン

Let文は式だけでなく、文字列を扱うこともできる。Let文に文字列としての式を扱わせ、それをサブルーチンとして使用したのが次例である。

	A	B	C	D	E
1	A=	444		¥a	{getNumber A→,b1}
2	B=	555			{getNumber B→,b2}
3					{let sub,+b1+b2 ~:string}
4	A+B=	999			{goto} b4 ~ {sub}
5	A−B=	−111			{let sub,+b1−b2 ~:string}
6	A＊B=	246420			{goto}b5 ~{sub}
7	A / B=	0.8			{let sub, b1 ＊ b2:string}
8					{goto} b6 ~ {sub}
9					{let sub, @if(b2=0,"",+b1/b2) ~:string}
10					{goto} b7 ~ {sub}
11					
12				sub	@if(b2=0,""+b1/b2) ~{return}
13					

上掲プログラムに、更にデータ消去部分を追加した四則演算プログラムを下に示す。このマクロ・プログラム¥b は言はば二つのサブルーチン¥A と sub を使用するマクロ・プログラムである。即ち¥b はその内でマクロ・プログラム¥a をサブルーチンとして使い、¥a はそれ自体の内部でサブルーチン sub を、内容を変えながら4回使うという構造になっている。

9．if 文

ロータス１２３では判断を行うため if 文が用意されている。この if 文には、通常のセル計算を行うための@if 文とマクロ用の {if} 文の二つがある。

セル計算を行うための@if 文は、前節の割算の場合に示したように、条件が真の場合の引数と、偽の場合の引数の二つを持っている。

@if(条件・真の場合の引数・偽の場合の引数)

これに対し、マクロ用の {if} 文は、条件が真のときは同一行のマクロ文を実行し、偽のときは次行のマクロ文を実行する。

	A	B	C	D	E	F	G	H
1	A=	333		¥a	{getNumber "A=>",b1}			
2	B=	444			{getNumber "B=>",b2} ~			
3					{let sub,+b1+b2 ~:string}			
4	A+B=	777			{get to} b4 ~{sub}			
5	A=B=	−111			{let sub,+b1−b2 ~:string}			
6	A ＊ B=	147852			{go to} b5 ~{sub}			
7	A/B=	0.75			{let sub,+b1 ＊ b2 ~:string}			
8					{go to} b6 ~{sub}			
9					{let sub,@if(b2=0,"",+b1/b2) ~:string}			
10					{go to} b7 ~{sub}			
11								
12				sub	@if(b2=0,"",+b1/b2) ~			
13					{return}			
14								
15	SUB	E12		¥b	/reb1. .b7 ~			
16	¥A	E1			{¥A} ~			
17	¥B	E15						
18								

{if 条件}{真のとき実行}
{真および偽のとき実行}

このマクロ if 文の使用法を、1から10までの和を求めるプログラムを例にとって説明しよう。

	A	B
1	SUM	55
2	i	11
3		
4	¥a	{blank b1..b2}
5		{let i, 1}
6	f	{if i> 10}{Quit}
7		{let sum, sum+i}
8		{let i, i+1}
9		{branch f}

まず、求める和を出力するための変数 sum と1から10までのカウントをとるための変数 i をそれぞれ b1 と b2 に宣言し、{blank b1..b2} でこの二つの変数をクリヤーする。次に {let i,1} で i を初期値1にセットする。

i が10になるまで、i を1つづう増やしながら sum に i を足す作業を繰り返すのが

f {if i> 10} {Quit}
{let sum, sum+i}
{let i, i+1}
{branch f} である。

もちろん、このプログラムは次のように書いても同じであり、この方が for ループに似ている。しかし、これでは見苦しい。

	A	B
1	SUM	55
2	i	11
3		
4	¥a	{blank b1..b2}
5		{let i, 1}
6	f	{if i <=10}{let sum, sum+i}{let i, i+1}{branch f}
7		{Quit}

10. for ループ

ほとんどすべてのプログラミング言語にループ文がある。PASCAL では for ループ、while ループ、repeat ループがあり、Fortran にも BASIC にもある。ロータス1-2-3にも for ループ文がある。

前節でとりあげた1から10までの和を求めるプログラムを For ループ文を用いて書くと次のようになる。

	A	B
1	SUM	55
2	i	11
3	WA	{let sum,sum+i}
4		{return}
5		
6	¥a	/re b1..b2~
7		{for i, 1, 10, 1, WA}
8		

11. for ループ・パラメータの変数化

{for i, st, ed, eten,シゴト}

シゴト [.....]

ロータス1-2-3の for ループはカウンター i を初期値 st(1) から終値 ed (10) まで間隔 step(1) 刻みで特定のシゴト(サブルーチン)を繰返すのである。したがって、1から10まで1刻みで変化し、和を求めると言う場合には、カウンター i とシゴト (WA) 及び初期値 st、終値 ed、刻み step は {for i, 1, 10, 1, WA} のなかで自動的に定義してあることになる。

これに対し、刻みが1で、初項と終項を変数とする場合にはプログラムは次のようになる。

	A	B
1	SUM	207
2	i	21
3	st	3
4	ed	20
5		
6	WA	{let sum, sum+i}
7		{return}
8		
9	¥a	/re b1..b4/ ~
10		{get Number st →, st}
11		{get Number ed →,ed}
12		{for i, st, ed, 1, WA}

上の例は3から20までの和を求めたものである。

なお、初項、終項の他、刻みstepも変数とする場合には、刻みstepを変数として加えればよい。下のプログラムでは1から10まで2刻みで和を求めたものである。

12. 金種計算とforループ

forループの応用問題として、本節では金種計算をforループで解く場合を考えてみることにする。

まず、金種計算の問題を見ることにしよう。金種計算の問題は、任意の金額をまづ、一番大きな貨幣金額で割り、残りを次に大きな貨幣金額で割って行く。

たとえば¥12,345と言う金額は

1．まず1万円で割って、商(1)が一万円札の枚数を示し、

¥12,345÷¥10,000＝1　残金¥2,345

2．残り¥2,345を¥5,000札で割る。この場合商はゼロで残金は¥2,345である。

¥2,345÷¥5,000＝0　残金¥2,345

3．次に残り¥2,345を¥1,000札で割り、商2残金¥345をえる。

¥2,345÷¥1,000＝2　残金¥345

4．残り¥345を¥500円で割ると商はゼロ、残金¥345となる。

	A	B	C	D	E	
1	sum	25			ハンイメイ	イチラン
2	i	11			ED	B4
3	st	1			I	B2
4	ed	10			ST	B3
5	step	2			STEP	B5
6					SUM	B1
7	work	{let num,+sum+i}			WORK	B7
8		{return}			¥A	B10
9						
10	¥a	/reb1. .b5 ~				
11		{getnumber st=>,st}				
12		{getNumber "ed=>",ed}				
13		{getNumber "step=>"step}				
14		{for i, st, ed, step, work}				
15						
16						

¥345÷¥500＝0　残¥345

以下同様にして、貨幣金額で割って行く。

5．¥345÷¥100＝3　残¥45

6．¥45÷¥50＝0　残¥45

7．¥45÷¥10＝4　残¥5

8．¥5÷¥5＝1　残0

9．¥0÷¥1＝0　残0

この場合、残金がゼロになれば中止してよい。ただし、中止せず最小貨幣金額である1円で割るところで繰り返してもかまわない。

以上の計算をロータス123の計算表を使って実行する場合、もっともわかり易い方法の一つは、次のようにすることであろう。

まづ、次図に示すように、C1に金額たとえば12345円を入力する。A2からA10までに貨幣金額を大きいものから小さいものへと入力しておく。

	A	B	C
1		金額＝	x
2	10000	@Int(C1/A2)	@Mod(C1,A2)
3	5000	@Int(C2/A3)	@Mod(C2,A3)
4	1000	@Int(C3/A4)	@Mod(C3,A4)
5	500	⋮	⋮
6	100		
7	50		
8	10		
9	5		
10	1	@Int(C9/A10)	@Mod(C9,A10)

B2の計算式@Int (C1/A2) で金額12345円中に1万円札1枚が含まれることがわかる。そこで残金をC2の計算式@Mod (C1,A2) で計算して2345円を得る。このようにして、金額12,345円については下図のように金種計算が完了することになる。

	A	B	C
1		金額＝	12345
2	10000	1	2345
3	5000	0	2345
4	1000	2	345
5	500	0	345
6	100	3	45
7	50	0	45
8	10	4	5
9	5	1	0
10	1	0	0

この計算表に適当な見出しをつけ、わかり易いものにすることは読者の演習にまかせることにしよう。

ところで、この問題をforループを用いて解くためには、どのように考えればよいであろうか。12345円の例をもう一度検討してみよう。

	0列	1列
0行	10000	1
1行	5000	0
2行	1000	2
3行	500	0
4行	100	3
5行	50	0
6行	10	4
7行	5	1
8行	1	0
9行		
10行		

金種計算表のA列とB列をとり出して考えると、金種計算の解答はセルA2からB10の範囲に示されている。

即ち、この解答部分だけを取り出して考えると、各貨幣の金額とその枚数は9行2列の配列で、1万円から1円までの貨幣金額が先頭列すなわちゼロ列に並び、各貨幣の枚数が2番目の列すなわち1列目に並んでいる。

この配列（すなわちA2からB10）にC (Currency, 通貨）と言う名前をつけると、

1万円札の金額10000は@Index (C,0,0)) で取り出すことができる。

ここに、@Index 関数は行列を示す範囲名について、ゼロからはじまる列と行を指定して、行列の要素を取り出す関数である。

@Index（範囲名 (配列名)、列位置、行位置）

したがって、

5000円札の金額5000は@Index (C,0,1)

1000円札の金額1000は@Index (C,0,2)

となる。

したがって、金額 x に含まれる１万円札の枚数は

@Int (X / @Index(C,0,0))

で計算でき、１万円札で換えた残金は

@Mod(X,@Index(C,0,0))

で計算できることになる。

１万円札の枚数を計算したら、次は5000円札の枚数を計算する。このために、１万円札を計算した残りを、x に入れ替えておけば、@Index 関数の第３引数を０から１に変えるだけで計算できる。すなわち、

１万円札の計算式

{put C,1, 0 ,@Int(X/@Index(C,0, 0))}

{let x,@Mod(X,@Index(C,0, 0))}

が、5000円札の場合

{put C,1, 1 ,@Int(X/@Index(C,0, 1))}

{let x,@Mod(X,@Index(c,0, 1))}

と言うように下線を施した部分だけが変わればよい。そこで、次図のマクロ・プログラムを作ればよいことになる。

	A	B	C	D	E	F
1	¥a	{get Number =>, X}				
2		{let bll, x}				
3		{for i, 0, 8, 1, chg}				
4						
5	i	9				
6	x	0				
7						
8	chg	{put c, 1, i, @int(x/@index(c, 0, i))}				
9		{let x,@index(c, 0, i))}				
10						
11	金額=	567890		範囲名一覧		
12	10000	56		C	A12..B20	
13	5000	1		CHG	B8	
14	1000	2		I	B5	
15	500	1		X	B6	
16	100	3		¥A	B1	
17	50	1				
18	10	1				
19	5	0				
20	1	0				