第４部：　『魔方陣の最新研究』
第４章：　解説『魔方陣研究の諸技法』II：　摂田　寛二
第11節：　三，五，七進法による『新オイラー法』で各種の「オイラー三，五，七方陣」を構成する（改訂版）

０．この節の目的

2012年に『オイラー方陣』・『完全オイラー方陣』に関する私の考え方が少し変わった。

それに応じて今節のこの記事を改訂する。その際，元の文章を極力残す努力をしよう。

ここでの主要目的は，『**新オイラー法**』を奇数次の三，五そして七次の魔方陣に適用して，試験することだ。一番最初にやるべき適用試験を，三次の魔方陣に対してまだ試していなかった上に，五次と七次に対しても，まだ『桂馬とび構成法』しか試していなかった。

今回あらためて**新オイラー法**を適用することによって，その能力を検証したい。また五次・七次では，**桂馬とび構成法**の結果を再現し，両方の構成法の正当性を実証したい。

１．三進法による『新オイラー法』で標準型三方陣を構成する

新オイラー法の基礎を成すのは，三進法分解図と構成図の間に見られる下図のような対称的な手順だ。分解図を作る手順を関数として捉えるならば，三進法ユニットから元の方陣を再現構成する手順はその「逆関数」ということになる。我々は，その双方向で**新オイラー法**を考える。その手順は，どちらも簡単な算数によって説明することができる。

新オイラー法の概念　　/三進法分解図　　チェック・サム|対1\対2/行|列|

[古典的表記] |15\15/	[数学的表記]|12\12/	/D3i 高|3\3/	低|3\3/
2 9 4|15|15|	1 8 3|12|12|	0 2 1|3|3|	1 2 0|3|3|
7 5 3|15|15|	6 4 2|12|12|	2 1 0|3|3|	0 1 2|3|3|
6 1 8|15|15|	5 0 7|12|12|	1 0 2|3|3|	2 0 1|3|3|

/三進法構成図：オイラー・ユニット（または単位方陣）をもとに元の方陣を再構成する

/EU 高|3\3/	低|3\3/	[数学的表記]|12\12/	[古典的表記]|15\15/
0 2 1|3|3|	1 2 0|3|3|	1 8 3|12|12|	2 9 4|15|15|
2 1 0|3|3|	0 1 2|3|3|	6 4 2|12|12|	7 5 3|15|15|
1 0 2|3|3|	2 0 1|3|3|	5 0 7|12|12|	6 1 8|15|15|

上の段の仕組みはもうおわかりだろう。解の実例の**古典的表記**は，１，２，３，...，８，９の自然数を用いている。**数学的表記**は，０，１，２，...，７，８の整数を用いる。だから，古典的表記よりどれも１だけ小さい数字になっている。

三進法分解図は，数学的表記に基づき，３で割った商の整数部分が高位に，低位には３で割った時の剰余が入る。高位と低位を別の面に表わすのが，昔からの約束だ。

左端の古典的表記と右端２面の間には，次のような関係式が成り立つ。

１＝0×3＋0＋1；　２＝0×3＋1＋1；　３＝0×3＋2＋1；　４＝1×3＋0＋1；　５＝1×3＋1＋1；
６＝1×3＋2＋1；　７＝2×3＋0＋1；　８＝2×3＋1＋1；　９＝2×3＋2＋1；

$$Vn = An \times 3 + Bn + 1 \quad (n = 1, 2, 3, \ldots, 8, 9)$$

我々がこれから試みたいのは，上段を逆方向に見て，つまり下段のように，先ずは三進法ユニットを何か別の方法で作り，出来た中から２面ずつを選んで組み合わせ，上の式で解を直に計算・合成・構成することである。どうしたら，そんなことができるか？

この工夫のしどころが，**新オイラー法**の特徴になっている。

上段の三進法分解図 /D3i と下段の三進法ユニット /EU は，上の例では同じ内容にしてあるが，先ずそれらをじっくりと観察してほしい。

各面とも，行列は全て｛0，1，2｝で出来ている。並び方に違いはあるが，内容は違わない。故に，各行列は 0＋1＋2＝3 と定和に等しく，元の方陣でも (0＋1＋2)×３＋(0＋1＋2)＝12(+) と，定和に必ず等しくなる。こういう構造を古来「**ラテン方陣**」と言う。

主対角線の１本も**ラテン型**だが，他の１本の構成は惜しくも｛1，1，1｝になる。

これは避けられない。従って，主対角線を作る場合には，定和だけを定義して，その数字内容については一切注文を付けないでおいた方が良さそうだ。
これは，他の次数の方陣分析でわかったことだが，各行列や対角線の和に関して，ラテン方陣の実現を無理に要求せず，ただ定和だけを一律に要求する定義の仕方もある。どうやらその方が普遍的な現象を良く説明し，適用範囲も広いということが次第にわかって来た。
そこで，私は『**オイラー方陣**』の概念を新たに定義し直すことにした。
Ｎ進法ユニットの各面で，全体方陣と同様に全行・全列・全主対角線の定和が成り立つこと。新概念では，それだけを求めるのである。そして，そういう方陣群全体に**オイラー方陣**の名を冠することにした。その際各部の構成内容については特に定義しない。そして，今後は，そのＮ進法ユニットのことを『**オイラー・ユニット**』または『**単位方陣**』と呼ぶ。
ラテン型の構成が単位方陣の各部分で一律可能なものに対しては，従来通り『**ギリシャ・ラテン方陣**』または『**完全オイラー方陣**』と呼ぶ。**ギリシャ・ラテン方陣＝オイラー方陣**が通例だったのだが，**オイラー方陣**と**完全オイラー方陣**との間にあえて相違を設けたい。現実には，二者が一致する場合もあるが，後者の小集合が前者の大集合に含まれることが多い。

今回は，その**オイラー方陣**を作る目的で，先ず三進法のオイラー・ユニットを構成する。
何通り出来るだろうか？
プログラムに書いてコンピュータを動かし，それを求めさせよう。三次なら手計算でもできるが，高次の場合の難問に備えて，プログラムによって作ることに極力慣れておきたい。
先ず，次のような基本図と基本式を用意しよう。最初のは，元の標準型三方陣を定義した例で，次のはそれを模倣しながら定義した「オイラー・ユニット」用のものだ。

```
//
//** Definition-1 for Standard Magic Squares of Order 3 **
//  .--.--.--. * Basic Form and Equations: C=15 *
//  | 1| 2| 3|  n1+n2+n3=C ...sm1;
//  |--+--+--|  n4+n5+n6=C ...sm2;
//  | 4| 5| 6|  n7+n8+n9=C ...sm3;
//  |--+--+--|  n1+n4+n7=C ...sm4;
//  | 7| 8| 9|  n2+n5+n8=C ...sm5;
//  '--'--'--'  n3+n6+n9=C ...sm6;
// n1+n5+n9=C ...sm7;  n3+n5+n7=C ...sm8;
//
//** Definition-2 for their Euler Units **
//  .--.--.--. * Basic Form and Equations: C=3 *
//  | 1| 2| 3|  n1+n2+n3=C ...sm1;
//  |--+--+--|  n4+n5+n6=C ...sm2;
//  | 4| 5| 6|  n7+n8+n9=C ...sm3;
//  |--+--+--|  n1+n4+n7=C ...sm4;
//  | 7| 8| 9|  n2+n5+n8=C ...sm5;
//  '--'--'--'  n3+n6+n9=C ...sm6;
// n1+n5+n9=C ...sm7;  n3+n5+n7=C ...sm8;
//
```

方陣全体とオイラー・ユニットとの間にかように緊密な「相似の構造」を仮定する。それが，**新オイラー法**の要点だ。これ無くしては，オイラー方陣は作れない。下段の定義条件下で書いたプログラムを実行して，オイラー・ユニットを合成しよう。
もう一つ重要な条件を追加する。「ユニットの各面が {0, 1, 2} の３数のみで作られること。そして，各数字を各３回ずつ使う。この回数に過不足があってはならない。」
プログラムは，例えば次のように書く。主要部分のみを示す（英語版で，ごめん！）

```
//
//** Standard Magic Squares of Order 3 Reconstructed by the **
//**  3rd Increment Number System and New Euler's Method **
//** 'NEulerMS33S.c': Dictated by Kanji Setsuda on **
//**  Apr.11, 2006, Dec. 9, 2011 and Apr.8, 2012  **
//**  Working with MacOSX 10.7.3 & Xcode 4.2.1 **
//
```

```c
#include <stdio.h>
//
short cnt, ecnt;
short LSM;
short u1,u2;
short nm[10], uflg[10], tn[10];
short anm[5][10];
short teu[5][10];
short mtc[5][5], cmtc[5][5], scmtc[5][5];
short cs[5][9];
//
void stp01(void), stp02(void), stp03(void), stp04(void), stp05(void);
void stp06(void), stp07(void), stp08(void), stp09(void);
void hansrecord(void), eurecord(void);
void preunit(void);
//
void cmbcmp(void), recprans(void), anspr(void);
//
void chksms(short x);
//
//* Main Program *
int main(void){
 short n;
  printf("\n");
  printf("** Reconstruct Standard Magic Squares of Order 3 by\n");
  printf("   the 3rd Increment Number System and New Euler's Method **\n");
  for(n=0;n<10;n++){nm[n]=0;}
  for(n=0;n<3;n++){uflg[n]=0;}
  cnt=0; LSM=3;
  printf("\n");
  printf("** List of Euler Units **\n");
  stp01();                      // Make the Euler Units
  ecnt=cnt;
  preunit();                    //* Classify and Print the Latin Units */
  printf("\n");
  printf("** List of Standard Magic Squares of Order 3 Reconstructed **\n");
  printf(" [/Euler Units: High| Low|; [Math]  [Classic];\n");
  printf("    Sum Checks of Each: |D1\\D2/Row|Column|;]\n");
  cnt=0;
  cmbcmp();                     // Combine, Compose and Print
  printf(" [Solution Counts = %d]   [OK!]\n",cnt);
  printf("** Calculated and Listed by Kanji Setsuda\n");
  printf("   on Apr. 8, 2012 with MacOSX 10.7.3 & Xcode 4.2.1 **\n");
  printf("\n");
 return 0;
}
//
//* Make the Euler Units
//** Definitions for Standard Magic Squares of Order 3 **
//.--.--.--. * Basic Form and Equations: C=3 *
//| 1| 2| 3|  n1+n2+n3=C ...sm1;
//|--+--+--|  n4+n5+n6=C ...sm2;
//| 4| 5| 6|  n7+n8+n9=C ...sm3;
//|--+--+--|  n1+n4+n7=C ...sm4;
//| 7| 8| 9|  n2+n5+n8=C ...sm5;
//'--'--'--'  n3+n6+n9=C ...sm6;
//  n1+n5+n9=C ...sm7;  n3+n5+n7=C ...sm8;
//
//Level #1:
//Set n1
void stp01(){
 short i;
  for(i=0;i<3;i++){
    if(uflg[i]<3){nm[1]=i;
      uflg[i]++; stp02(); uflg[i]--;}
```

```
  }
}
//Set n2
void stp02(){
 short i;
  for(i=0;i<3;i++){
    if(uflg[i]<3){nm[2]=i;
      uflg[i]++; stp03(); uflg[i]--;}
  }
}
//Set n3=3-n1-n2
void stp03(){
 short i;
  i=LSM-nm[1]-nm[2];
  if((0<=i)&&(i<3)){
    if(uflg[i]<3){nm[3]=i;
     uflg[i]++; stp04(); uflg[i]--;}}
}
//Set n4
void stp04(){
 short i;
  for(i=0;i<3;i++){
    if(uflg[i]<3){nm[4]=i;
      uflg[i]++; stp05(); uflg[i]--;}
  }
}
//Set n5
void stp05(){
 short i;
  for(i=0;i<3;i++){
    if(uflg[i]<3){nm[5]=i;
      uflg[i]++; stp06(); uflg[i]--;}
  }
}
//Set n6=3-n4-n5
void stp06(){
 short i;
  i=LSM-nm[4]-nm[5];
  if((0<=i)&&(i<3)){
    if(uflg[i]<3){nm[6]=i;
      uflg[i]++; stp07(); uflg[i]--;}}
}
//Set n7=3-n1-n4
void stp07(){
 short i,j;
  i=LSM-nm[1]-nm[4];
  j=LSM-nm[3]-nm[5];
  if((0<=i)&&(i<3)&&(i==j)){
    if(uflg[i]<3){nm[7]=i;
      uflg[i]++; stp08(); uflg[i]--;}}
}
/* Set n8=3-n2-n5 */
void stp08(){
 short i;
  i=LSM-nm[2]-nm[5];
  if((0<=i)&&(i<3)){
    if(uflg[i]<3){nm[8]=i;
      uflg[i]++; stp09(); uflg[i]--;}}
}
/* Set n9=3-n7-n8 & n9=3-n3-n6 & n9=3-n1-n5 */
void stp09(){
 short i,j,k;
  i=LSM-nm[7]-nm[8];
  j=LSM-nm[3]-nm[6];
  k=LSM-nm[1]-nm[5];
```

```
  if((0<=i)&&(i<3)){
    if((i==j)&&(i==k)){
      if(uflg[i]<3){nm[9]=i;
        uflg[i]++; eurecord(); uflg[i]--;}}}
}
//
//* Record the Answers *
void eurecord(){
 short n;
  teu[cnt][0]=cnt+1;
  for(n=1;n<10;n++){teu[cnt][n]=nm[n];}
  cnt++;
}
//
//* Classify and Print the Latin Units *
void preunit(){
 short t,l,l3,m,n;
  for(m=0;m<ecnt;m++){
    for(n=0;n<ecnt;n++){
      t=0;
      for(l=1;l<10;l++){if(teu[m][l]==teu[n][l]){t++;}}
      mtc[m][n]=t; scmtc[m][n]=t;
      t=0;
      for(l=1;l<10;l++){if(teu[m][l]+teu[n][l]==2){t++;}}
      cmtc[m][n]=t;
      if(scmtc[m][n]<t){scmtc[m][n]=t;}
    }
  }
  printf("%7d/%7d/%7d/%7d/\n",1,2,3,4);
  for(l=0;l<3;l++){l3=l*3;
    for(m=0;m<ecnt;m++){
      printf("  ");
      for(n=1;n<4;n++){printf("%2d",teu[m][l3+n]);}
    }
    printf("\n");
  }
  printf(" [Count of Latin Units = %d]\n",ecnt);
  printf("\n");
  printf(" [Reference Table for the Best Match]\n");
  printf("  S|");
  for(n=0;n<4;n++){printf("%3d",n+1);}
  printf("    C|");
  for(n=0;n<4;n++){printf("%3d",n+1);}
  printf("   SC|");
  for(n=0;n<4;n++){printf("%3d",n+1);}
  printf("\n");
  printf(" --+-------------  --+-------------  --+-------------\n");
  for(m=0;m<4;m++){
    printf("%3d|",m+1);
    for(n=0;n<4;n++){printf("%3d",mtc[m][n]);}
    printf("  %3d|",m+1);
    for(n=0;n<4;n++){printf("%3d",cmtc[m][n]);}
    printf("  %3d|",m+1);
    for(n=0;n<4;n++){printf("%3d",scmtc[m][n]);}
    printf("\n");
  }
}
//
//* Combine, Compose and Print *
void cmbcmp(){
 short n,d,fc;
  for(u1=0;u1<ecnt;u1++){
    for(u2=0;u2<ecnt;u2++){
      if(scmtc[u1][u2]==3){
        for(n=1;n<10;n++){uflg[n]=0;}
```

```
          for(n=0;n<10;n++){
            d=teu[u1][n]*3+teu[u2][n]+1;
            nm[n]=d; uflg[d]++;
          }
          fc=0;
          for(n=1;n<10;n++){if(uflg[n]==0){fc++;}}
          if(fc==0){recprans();}
        }
      }
    }
}
//
//* Record and Print the Solution List *
void recprans(){
 short n;
  cnt++;
  anm[0][0]=cnt;
  for(n=1;n<10;n++){tn[n]=nm[n]; anm[0][n]=tn[n];}; chksms(0);
  for(n=1;n<10;n++){tn[n]=nm[n]-1; anm[1][n]=tn[n];}; chksms(1);
  for(n=1;n<10;n++){tn[n]=teu[u1][n]; anm[2][n]=tn[n];}; chksms(2);
  for(n=1;n<10;n++){tn[n]=teu[u2][n]; anm[3][n]=tn[n];}; chksms(3);
  anm[2][0]=u1+1; anm[3][0]=u2+1;
  anspr();
}
//
void anspr(){
 short l,l3,n;
  printf(" /EU %dH|%d\\%d/    %dL|%d\\%d/  [Math]  |%2d\\%2d/  [%d]    |%2d\\%2d/",
          anm[2][0],cs[2][7],cs[2][8],anm[3][0],cs[3][7],cs[3][8],
          cs[1][7],cs[1][8],anm[0][0],cs[0][7],cs[0][8]);
  printf("\n");
  for(l=0;l<3;l++){l3=l*3;
    printf(" ");
    for(n=1;n<4;n++){printf("%2d",anm[2][l3+n]);}
    printf("|%d|%d| ",cs[2][l+1],cs[2][l+4]);
    for(n=1;n<4;n++){printf("%2d",anm[3][l3+n]);}
    printf("|%d|%d| ",cs[3][l+1],cs[3][l+4]);
    for(n=1;n<4;n++){printf("%3d",anm[1][l3+n]);}
    printf("|%2d|%2d| ",cs[1][l+1],cs[1][l+4]);
    for(n=1;n<4;n++){printf("%3d",anm[0][l3+n]);}
    printf("|%2d|%2d|",cs[0][l+1],cs[0][l+4]);
    printf("\n");
  }
}
//
//* Check Sums *
void chksms(short x){
  cs[x][1]=tn[1]+tn[2]+tn[3]; cs[x][2]=tn[4]+tn[5]+tn[6]; cs[x][3]=tn[7]+tn[8]+tn[9];
  cs[x][4]=tn[1]+tn[4]+tn[7]; cs[x][5]=tn[2]+tn[5]+tn[8]; cs[x][6]=tn[3]+tn[6]+tn[9];
  cs[x][7]=tn[1]+tn[5]+tn[9]; cs[x][8]=tn[3]+tn[5]+tn[7];
}
```

実行させたところ，オイラー・ユニットは下表のように４個出て来た。これを２個ずつ機械的に組み合わせ，計算・合成・構成した解 16個（４×４＝16）を下に示した。

```
** 三進法による『新オイラー法』で標準型の「オイラー三方陣」を再構成する 実験-１ ***
  [三進法による「オイラー・ユニット」のリスト]
     1/      2/      3/      4/
  0 2 1   1 0 2   1 2 0   2 0 1
  2 1 0   2 1 0   0 1 2   0 1 2
  1 0 2   0 2 1   2 0 1   1 2 0
  [オイラー・ユニットの総数 ＝ ４個]
```

** ４×４の組み合せで得られた全ての構成解のリスト **

[/オイラー・ユニット: 高位| 低位|; [数学的表記] [古典的表記]; チェック・サム: |対1\対2/行|列|;]

```
/EU 1H|3\3/     1L|3\3/  [Math]  |12\12/  [ 1]    |15\15/
 0 2 1|3|3|  0 2 1|3|3|   0  8  4|12|12|   1  9  5|15|15|
 2 1 0|3|3|  2 1 0|3|3|   8  4  0|12|12|   9  5  1|15|15|
 1 0 2|3|3|  1 0 2|3|3|   4  0  8|12|12|   5  1  9|15|15|

/EU 1H|3\3/     2L|3\3/  [Math]  |12\12/  [ 2]    |15\15/
 0 2 1|3|3|  1 0 2|3|3|   1  6  5|12|12|   2  7  6|15|15|
 2 1 0|3|3|  2 1 0|3|3|   8  4  0|12|12|   9  5  1|15|15|
 1 0 2|3|3|  0 2 1|3|3|   3  2  7|12|12|   4  3  8|15|15|

/EU 1H|3\3/     3L|3\3/  [Math]  |12\12/  [ 3]    |15\15/
 0 2 1|3|3|  1 2 0|3|3|   1  8  3|12|12|   2  9  4|15|15|
 2 1 0|3|3|  0 1 2|3|3|   6  4  2|12|12|   7  5  3|15|15|
 1 0 2|3|3|  2 0 1|3|3|   5  0  7|12|12|   6  1  8|15|15|

/EU 1H|3\3/     4L|3\3/  [Math]  |12\12/  [ 4]    |15\15/
 0 2 1|3|3|  2 0 1|3|3|   2  6  4|12|12|   3  7  5|15|15|
 2 1 0|3|3|  0 1 2|3|3|   6  4  2|12|12|   7  5  3|15|15|
 1 0 2|3|3|  1 2 0|3|3|   4  2  6|12|12|   5  3  7|15|15|

/EU 2H|3\3/     1L|3\3/  [Math]  |12\12/  [ 5]    |15\15/
 1 0 2|3|3|  0 2 1|3|3|   3  2  7|12|12|   4  3  8|15|15|
 2 1 0|3|3|  2 1 0|3|3|   8  4  0|12|12|   9  5  1|15|15|
 0 2 1|3|3|  1 0 2|3|3|   1  6  5|12|12|   2  7  6|15|15|

/EU 2H|3\3/     2L|3\3/  [Math]  |12\12/  [ 6]    |15\15/
 1 0 2|3|3|  1 0 2|3|3|   4  0  8|12|12|   5  1  9|15|15|
 2 1 0|3|3|  2 1 0|3|3|   8  4  0|12|12|   9  5  1|15|15|
 0 2 1|3|3|  0 2 1|3|3|   0  8  4|12|12|   1  9  5|15|15|

/EU 2H|3\3/     3L|3\3/  [Math]  |12\12/  [ 7]    |15\15/
 1 0 2|3|3|  1 2 0|3|3|   4  2  6|12|12|   5  3  7|15|15|
 2 1 0|3|3|  0 1 2|3|3|   6  4  2|12|12|   7  5  3|15|15|
 0 2 1|3|3|  2 0 1|3|3|   2  6  4|12|12|   3  7  5|15|15|

/EU 2H|3\3/     4L|3\3/  [Math]  |12\12/  [ 8]    |15\15/
 1 0 2|3|3|  2 0 1|3|3|   5  0  7|12|12|   6  1  8|15|15|
 2 1 0|3|3|  0 1 2|3|3|   6  4  2|12|12|   7  5  3|15|15|
 0 2 1|3|3|  1 2 0|3|3|   1  8  3|12|12|   2  9  4|15|15|

/EU 3H|3\3/     1L|3\3/  [Math]  |12\12/  [ 9]    |15\15/
 1 2 0|3|3|  0 2 1|3|3|   3  8  1|12|12|   4  9  2|15|15|
 0 1 2|3|3|  2 1 0|3|3|   2  4  6|12|12|   3  5  7|15|15|
 2 0 1|3|3|  1 0 2|3|3|   7  0  5|12|12|   8  1  6|15|15|

/EU 3H|3\3/     2L|3\3/  [Math]  |12\12/  [10]    |15\15/
 1 2 0|3|3|  1 0 2|3|3|   4  6  2|12|12|   5  7  3|15|15|
 0 1 2|3|3|  2 1 0|3|3|   2  4  6|12|12|   3  5  7|15|15|
 2 0 1|3|3|  0 2 1|3|3|   6  2  4|12|12|   7  3  5|15|15|

/EU 3H|3\3/     3L|3\3/  [Math]  |12\12/  [11]    |15\15/
 1 2 0|3|3|  1 2 0|3|3|   4  8  0|12|12|   5  9  1|15|15|
 0 1 2|3|3|  0 1 2|3|3|   0  4  8|12|12|   1  5  9|15|15|
 2 0 1|3|3|  2 0 1|3|3|   8  0  4|12|12|   9  1  5|15|15|

/EU 3H|3\3/     4L|3\3/  [Math]  |12\12/  [12]    |15\15/
 1 2 0|3|3|  2 0 1|3|3|   5  6  1|12|12|   6  7  2|15|15|
 0 1 2|3|3|  0 1 2|3|3|   0  4  8|12|12|   1  5  9|15|15|
 2 0 1|3|3|  1 2 0|3|3|   7  2  3|12|12|   8  3  4|15|15|

/EU 4H|3\3/     1L|3\3/  [Math]  |12\12/  [13]    |15\15/
 2 0 1|3|3|  0 2 1|3|3|   6  2  4|12|12|   7  3  5|15|15|
 0 1 2|3|3|  2 1 0|3|3|   2  4  6|12|12|   3  5  7|15|15|
 1 2 0|3|3|  1 0 2|3|3|   4  6  2|12|12|   5  7  3|15|15|
```

```
 /EU 4H|3\3/     2L|3\3/  [Math]  |12\12/  [14]    |15\15/
  2 0 1|3|3|   1 0 2|3|3|   7  0  5|12|12|   8  1  6|15|15|
  0 1 2|3|3|   2 1 0|3|3|   2  4  6|12|12|   3  5  7|15|15|
  1 2 0|3|3|   0 2 1|3|3|   3  8  1|12|12|   4  9  2|15|15|
 /EU 4H|3\3/     3L|3\3/  [Math]  |12\12/  [15]    |15\15/
  2 0 1|3|3|   1 2 0|3|3|   7  2  3|12|12|   8  3  4|15|15|
  0 1 2|3|3|   0 1 2|3|3|   0  4  8|12|12|   1  5  9|15|15|
  1 2 0|3|3|   2 0 1|3|3|   5  6  1|12|12|   6  7  2|15|15|
 /EU 4H|3\3/     4L|3\3/  [Math]  |12\12/  [16]    |15\15/
  2 0 1|3|3|   2 0 1|3|3|   8  0  4|12|12|   9  1  5|15|15|
  0 1 2|3|3|   0 1 2|3|3|   0  4  8|12|12|   1  5  9|15|15|
  1 2 0|3|3|   1 2 0|3|3|   4  8  0|12|12|   5  9  1|15|15|
 [解の総数 ＝ 16個]
** Calculated and Listed by Kanji Setsuda
    on Mar.28, 2012 with MacOSX 10.7.3 & Xcode 4.2.1 **
```

これらは，いかなる意味でも「最終解」ではない。いわば作業途中の「裏側」を暴露した姿に過ぎない。16個中８個が誤った解になっている。

第１番解は，'1'，'5'，'9' が３度ずつ出て来て，他の数字が無い。第６，11，16番解も，同様だ。これでは，「１～９の自然数を各１度ずつ使う」という魔方陣の基本約束に違反する。とても，正解とは認め難い。

第４番解は，'3'，'5'，'7' が３度ずつ出て来て，他の数字が無い。第７，10，13番解も，同様だ。

どういう時に，この種の誤答が出て来るのか？

第１，６，11，16番解は，いずれも同じ単位面を２度用いて組み合わせている。

第４，７，10，13番解は，どうか。先ず単位面 1/と単位面 4/を良く比較して見よう。この２面には，お互いにどのような関係があるか？

対応する場所の数字どうしを加えると，どれも２になるではないか。

An＋Bn＝２（n=1，2，3，...，8，9）が成り立つような２面を，我々は「２の補数面」と呼ぶ。単位面 2/と単位面 3/も，お互いに「２の補数面」になっている。

(1/，4/)，(2/，3/) の単位面を組み合わせると，順序に関係なく必ず誤答になる。

つまり，我々は，同一面の組み合せと補数面の組み合せを避けなければならないのだ。

それをプログラムで実現するには，どうしたら賢いか。

ここで，２面の関係を数値化した「参照テーブル」の作成を提案したい。４×４の表を作るのである。対称型の場合は，「同値表あるいは相似表」を作るだけでは不十分で，「補数表」も必要になる。そして，それらを合わせた１つの合成数値表を作る必要がある。

下図右端のような「参照テーブル」を作っておく。そして，２面を組み合わせて解を合成する前に，組み合せの数値を見に行く。９になる組み合せを避け，３になる組み合せのみを選ぶようにするのである。実際のプログラムは，既に示してある。`preunit();` というプロシージャの最初の方でテーブルを作成し，`cmbcmp();` でそれを参照している。

用心深くそうすれば，構成解８個が得られる(下表)。どれも，もう「誤答」ではない。

** 三進法による『新オイラー法』で標準型の「オイラー三方陣」を構成する-２ **

[最終参照テーブルの合成]

S|	1	2	3	4
1|	9	3	3	3
2|	3	9	3	3
3|	3	3	9	3
4|	3	3	3	9

C|	1	2	3	4
1|	3	3	3	9
2|	3	3	9	3
3|	3	9	3	3
4|	9	3	3	3

S&C|	1	2	3	4
1|	9	3	3	9
2|	3	9	9	3
3|	3	9	9	3
4|	9	3	3	9

[再構成した標準型三方陣の原始解８個のリスト]

```
[/オイラー・ユニット: 高H|低L|; [Math] Classic#; サム・チェック: \対1/対2|行|列|;]
 /EU 1H\3/3|     2L\3/3|  [Math]  \12/12|        1#\15/15|
  0 2 1|3|3|  1 0 2|3|3|   1  6  5|12|12|   2  7  6|15|15|
  2 1 0|3|3|  2 1 0|3|3|   8  4  0|12|12|   9  5  1|15|15|
  1 0 2|3|3|  0 2 1|3|3|   3  2  7|12|12|   4  3  8|15|15|
 /EU 1H\3/3|     3L\3/3|  [Math]  \12/12|        2#\15/15|
  0 2 1|3|3|  1 2 0|3|3|   1  8  3|12|12|   2  9  4|15|15|
  2 1 0|3|3|  0 1 2|3|3|   6  4  2|12|12|   7  5  3|15|15|
  1 0 2|3|3|  2 0 1|3|3|   5  0  7|12|12|   6  1  8|15|15|
 /EU 2H\3/3|     1L\3/3|  [Math]  \12/12|        3#\15/15|
  1 0 2|3|3|  0 2 1|3|3|   3  2  7|12|12|   4  3  8|15|15|
  2 1 0|3|3|  2 1 0|3|3|   8  4  0|12|12|   9  5  1|15|15|
  0 2 1|3|3|  1 0 2|3|3|   1  6  5|12|12|   2  7  6|15|15|
 /EU 2H\3/3|     4L\3/3|  [Math]  \12/12|        4#\15/15|
  1 0 2|3|3|  2 0 1|3|3|   5  0  7|12|12|   6  1  8|15|15|
  2 1 0|3|3|  0 1 2|3|3|   6  4  2|12|12|   7  5  3|15|15|
  0 2 1|3|3|  1 2 0|3|3|   1  8  3|12|12|   2  9  4|15|15|
 /EU 3H\3/3|     1L\3/3|  [Math]  \12/12|        5#\15/15|
  1 2 0|3|3|  0 2 1|3|3|   3  8  1|12|12|   4  9  2|15|15|
  0 1 2|3|3|  2 1 0|3|3|   2  4  6|12|12|   3  5  7|15|15|
  2 0 1|3|3|  1 0 2|3|3|   7  0  5|12|12|   8  1  6|15|15|
 /EU 3H\3/3|     4L\3/3|  [Math]  \12/12|        6#\15/15|
  1 2 0|3|3|  2 0 1|3|3|   5  6  1|12|12|   6  7  2|15|15|
  0 1 2|3|3|  0 1 2|3|3|   0  4  8|12|12|   1  5  9|15|15|
  2 0 1|3|3|  1 2 0|3|3|   7  2  3|12|12|   8  3  4|15|15|
 /EU 4H\3/3|     2L\3/3|  [Math]  \12/12|        7#\15/15|
  2 0 1|3|3|  1 0 2|3|3|   7  0  5|12|12|   8  1  6|15|15|
  0 1 2|3|3|  2 1 0|3|3|   2  4  6|12|12|   3  5  7|15|15|
  1 2 0|3|3|  0 2 1|3|3|   3  8  1|12|12|   4  9  2|15|15|
 /EU 4H\3/3|     3L\3/3|  [Math]  \12/12|        8#\15/15|
  2 0 1|3|3|  1 2 0|3|3|   7  2  3|12|12|   8  3  4|15|15|
  0 1 2|3|3|  0 1 2|3|3|   0  4  8|12|12|   1  5  9|15|15|
  1 2 0|3|3|  2 0 1|3|3|   5  6  1|12|12|   6  7  2|15|15|
 [解の総数 ＝ ８個]
```

だが，この８個も，まだ「最終解」ではない。言わば「原始解」とも「素材解」とも言うべきもので，各個を相互に比較してみると，単なる「反転形」や「回転形」ばかりである。行列の内容を調べると，並び方に違いはあるが，顔ぶれが全部同じだ。

これをそれぞれお互いに別個の解と見なすことはできない。整理の必要がある。

そこで，「リスト整形用の不等式群」が登場する。フィルターの役目をする。この不等式群を通すと，８個の解を１個に絞り込んで，代表となる標準解を出力してくれる。

```
void cmbcmp(){
  . . . . .
        for(n=1;n<10;n++){if(uflg[n]==0){fc++;}}
        if(fc==0){recprans();}
```

のところに次のように書き加えれば良い。

```
        for(n=1;n<10;n++){if(uflg[n]==0){fc++;}}
        if(fc==0){
          if((nm[1]<nm[9])&&(nm[1]<nm[3])&&(nm[3]<nm[7])){recprans();}
        }
```

［標準型三方陣の標準解１個の出力］

```
/EU 1H\3/3|      3L\3/3|  [Math]  \12/12|         2#\15/15|
 0 2 1|3|3|   1 2 0|3|3|    1  8  3|12|12|    2  9  4|15|15|
 2 1 0|3|3|   0 1 2|3|3|    6  4  2|12|12|    7  5  3|15|15|
 1 0 2|3|3|   2 0 1|3|3|    5  0  7|12|12|    6  1  8|15|15|
```

［解の総数＝１個］ OK!

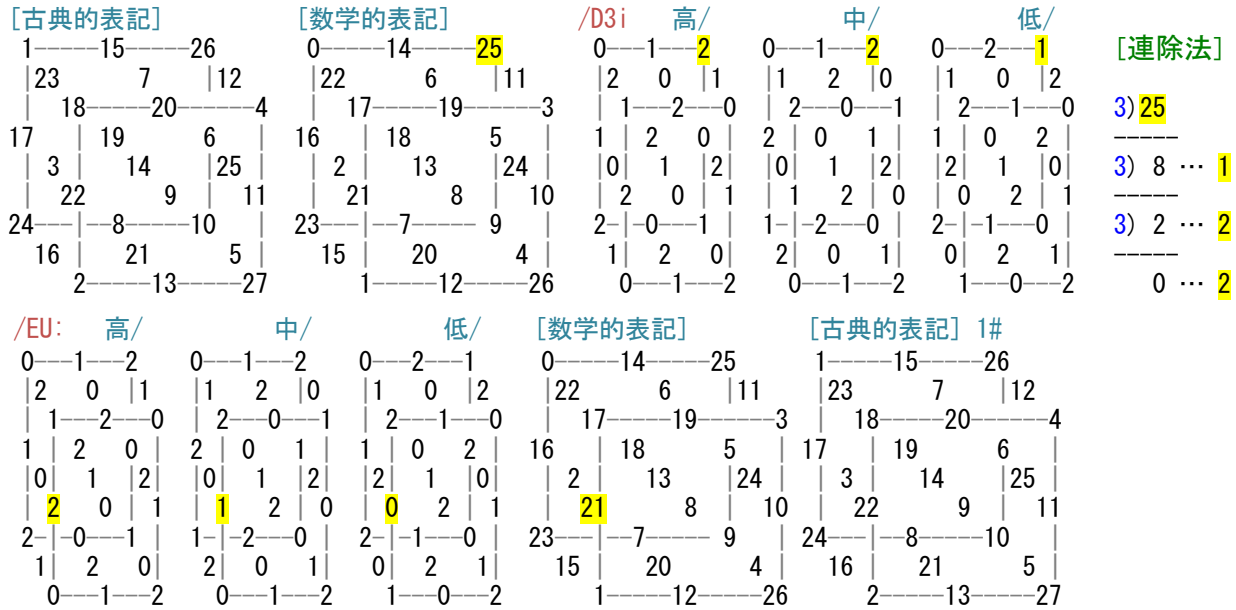
そうだ。三次の魔方陣には，もともと補数対称型の解が１個しか無いのであった。
我々は，確かに成功したと判断できる。対称三方陣は，惜しくも完全オイラー型ではなかったけれども，立派な**オイラー方陣**であった。**新オイラー法**は，どの段階でも矛盾も破綻も見せず，そのことを実証した，と思う。
なお，行列を**ラテン方陣**として設計しても，同じ結果が得られることは前に示している。

それにしても，たかが三方陣の解１個に随分と大袈裟な作り方をするものだと，呆れる読者も居られよう。それはその通りなのだが，我々の目的はあくまで『新オイラー法』の適用試験にあった。
いかに大袈裟でも，最小のｎ次について成り立つことを示さないでは，それから先の「方法の普遍性」を主張することができない。止むに止まれずにやった，と思ってほしい。
理論的研究てぇものは，本来そういうものではないのか。基礎を固めるためには，必死になって一見馬鹿げたことをしなければならないのだと，私は信じている。

２．三進法による『新オイラー法』で立方体の対称三方陣を構成する

次は，立体三方陣にチャレンジしよう。これも対称型しか作れないことがわかっている。果たして「オイラー型」と定義して，オイラー・ユニットを作れるだろうか？
下図の解の実例を見ると，有望だと言う気がする。

** 対称立体三方陣の解の三進法分解図と構成図 **



（分解図を作った「連除法」に対応するのが，（2×３＋1）×３＋0 ＝ 21 という「連乗法」だ。）

三次元立体方陣なので，３^{3}個の数字を使う。分解図が３個出来る。割り算も「連除法」が必要だ。従って，構成図も同様に３個のユニットを作り，計算・合成時には，例えば，
（2/高×３＋1/中）×３＋0/低＝合成値 21 という「連乗法」が必要になる。
上のユニットを良く観察すると，やはり１本だけ立体対角線の内容が {1, 1, 1} になっている。最初から完全オイラー型を目指さず，「オイラー型」の緩い定義を採用しよう。
基本図・基本式は，次のようにする。

```
//
//** Basic Form and Definitions: C=3; **
//   1-----10-----19         n1+n2+n3=C;    | n1+n10+n19=C;  | n1+n4+n7=C;
//   | 2     11     |20      n4+n5+n6=C;    | n2+n11+n20=C;  | n2+n5+n8=C;
//   |   3-----12-----21      n7+n8+n9=C;    | n3+n12+n21=C;  | n3+n6+n9=C;
//   4   | 13     22   |      n10+n11+n12=C; | n4+n13+n22=C;  | n10+n13+n16=C;
//   | 5 |   14    |23 |      n13+n14+n15=C; | n5+n14+n23=C;  | n11+n14+n17=C;
//   |   6     15  |  24      n16+n17+n18=C; | n6+n15+n24=C;  | n12+n15+n18=C;
//   7---|-16-----25   |      n19+n20+n21=C; | n7+n16+n25=C;  | n19+n22+n25=C;
//     8 |   17     26 |      n22+n23+n24=C; | n8+n17+n26=C;  | n20+n23+n26=C;
//       9-----18-----27      n25+n26+n27=C; | n9+n18+n27=C;  | n21+n24+n27=C;
//** 対称型補数の定義式: SC=2;
//   n1+n27=n2+n26=n3+n25=n4+n24+n5+n23
//   =n6+n22=n7+n21=n8+n20=n9+n19=n10+n18
//   =n11+n17=n12+n16=n13+n15=n14+n14=SC ...scc
//
```

定和値が変わっただけで，定義の仕方は元の方陣のものと全然変わらない。この「相似の構造」が，**新オイラー法**の根本要請の一つだ。

立体の主対角線は，次の４本あるが，これには注文を付けない。定義する必要がない。上の対称型補数の定義式のおかげで，どれも定和になることがわかっているからである。

```
    n1+n14+n27=C ...pt1;       n3+n14+n25=C ...pt2;
    n7+n14+n21=C ...pt3;       n9+n14+n19=C ...pt4;
```

n14=1 もわかっている。n14+n14=SC=2 によってそうなるからだ。こういう既知の条件は，プログラムの中では定義と同様に扱ってよい。そうしたとしても結果に影響はしない。

必ず付加する条件は，ユニットの各々で，使える数字が {0, 1, 2} に限られること。しかも，それぞれが９回ずつ使われること。この回数に過不足があってはならない。

この規則は，魔方陣一般にいつでも課せられる「使われる数字は，{1, 2, 3, 4, 5, …, 25, 26, 27} という連続する自然数 27個に限られ，しかも各１回ずつ使われること。数字の重複や欠落があってはならない」という古典的な約束に対応している。

プログラムは，例えば，次のように書く。

補数対称型であるから，(n1, n27), (n2, n26), (n3, n25), ..., (n12, n16), (n13, n15) を２数ずつまとめて一緒に決める。１つ１つのプロシージャーの長さは２倍になるが，全体のステップ数は半分になる。n14＝１は，メイン・プログラムの方で最初に定義している。

プログラムが通常に近い書き方だとがおわかりになろう。ラテン方陣の時と違い，ここではフラグの立て方が少ない。uflg[n] というフラグが監視しているのは，{0, 1, 2} が各９回ずつ使われているかどうかだ。９回未満ならば，先へ進むようにした。

```
//** Self-Complementary Magic Cubes of Order 3: Reconstructed  **
//** by the 3rd Increment N. System and the New Euler's Method **
//** 'NEulerMC3.c': in 2003 and 2005; Revised on Apr. 2, 2012; **
//**  built by Kanji Setsuda with MacOSX 10.7.3 & Xcode 4.2.1  **
//
#include <stdio.h>
//
//* Variables *
short int cnt;
short SC, LSM;
short u1, u2, u3, u4;
short teu[9][28];
short mtc[9][9];
short nm[28], uflg[28];
//
//* Sub-Routines *
void stp01(void), stp02(void), stp03(void), stp04(void);
void stp05(void), stp06(void), stp07(void), stp08(void);
void stp09(void), stp10(void), stp11(void), stp12(void);
void stp13(void), stp14(void);
```

```
void recordans(void);
void preunit(void), cmbcmp(void);
void prfmcans(void);
//
//* Main Program *
int main(){
 short n;
  printf("\n");
  printf("** Self-Complementary Magic Cubes of Order 3: Reconstructed by\n");
  printf("   the 3rd Increment Number System and the New Euler's Method **\n");
  for(n=0;n<3;n++){uflg[n]=0;}
  SC=2; LSM=3; cnt=0;
  nm[14]=1; uflg[1]=1;
  printf("\n");
  printf(" [List of Euler Units]\n");
  stp01();
  preunit();
  printf("\n");
  printf(" [Reconstruction of Self-Complementary Magic Cubes of Order 3]\n");
  printf(" (Euler Units: High/Middle/Low/; Reconstruction: [Math] Sol Number#)\n");
  cnt=0;
  cmbcmp();
  printf(" [Count of Solutions = %d]  [OK!]\n",cnt);
  printf(" ** Calculated and Listed by Kanji Setsuda\n");
  printf("    on Apr. 2, 2012 with MacOSX 10.7.3 and Xcode 4.2.1\n");
  printf("\n");
 return 0;
}
//
//* Making Euler Units *
//Level #1:
//Set n1 & n27
void stp01(){
 short i,j;
  for(i=0;i<3;i++){j=SC-i;
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[1]=i; nm[27]=j;
      uflg[i]++; uflg[j]++;
      stp02();
      uflg[j]--; uflg[i]--;}
  }
}
//Set n2 & n26
void stp02(){
 short i,j;
  for(i=2;i>=0;i--){j=SC-i;
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[2]=i; nm[26]=j;
      uflg[i]++; uflg[j]++;
      stp03();
      uflg[j]--; uflg[i]--;}
  }
}
//Set n3=LSM-n1-n2 & n25
void stp03(){
 short i,j;
  i=LSM-nm[1]-nm[2];
  j=LSM-nm[27]-nm[26];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[3]=i; nm[25]=j;
      uflg[i]++; uflg[j]++;
      stp04();
      uflg[j]--; uflg[i]--;}}
}
```

```
//Set n4 & n24
void stp04(){
 short i,j;
  for(i=2;i>=0;i--){j=SC-i;
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[4]=i; nm[24]=j;
      uflg[i]++; uflg[j]++;
      stp05();
      uflg[j]--; uflg[i]--;}
  }
}
//Set n7=LSM-n1-n4 & n21
void stp05(){
 short i,j;
  i=LSM-nm[1]-nm[4];
  j=LSM-nm[27]-nm[24];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[7]=i; nm[21]=j;
      uflg[i]++; uflg[j]++;
      stp06();
      uflg[j]--; uflg[i]--;}}
}
//Set n10 & n18
void stp06(){
 short i,j;
  for(i=2;i>=0;i--){j=SC-i;
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[10]=i; nm[18]=j;
      uflg[i]++; uflg[j]++;
      stp07();
      uflg[j]--; uflg[i]--;}
  }
}
//Set n19=LSM-n1-n10 & n9
void stp07(){
 short i,j;
  i=LSM-nm[1]-nm[10];
  j=LSM-nm[27]-nm[18];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[19]=i; nm[9]=j;
      uflg[i]++; uflg[j]++;
      stp08();
      uflg[j]--; uflg[i]--;}}
}
//Level #2:
//Set n6=LSM-n3-n9 & n22
void stp08(){
 short i,j;
  i=LSM-nm[3]-nm[9];
  j=LSM-nm[25]-nm[19];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[6]=i; nm[22]=j;
      uflg[i]++; uflg[j]++;
      stp09();
      uflg[j]--; uflg[i]--;}}
}
//Set n5=LSM-n4-n6 & n23
void stp09(){
 short i,j;
  i=LSM-nm[4]-nm[6];
  j=LSM-nm[24]-nm[22];
  if((0<=i)&&(i<3)&&(i+j==SC)){
```

```
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[5]=i; nm[23]=j;
      uflg[i]++; uflg[j]++;
      stp10();
      uflg[j]--; uflg[i]--;}}
}
//Set n8=LSM-n2-n5 & n20
void stp10(){
 short i,j,k,l;
  i=LSM-nm[2]-nm[5];
  j=LSM-nm[26]-nm[23];
  k=LSM-nm[7]-nm[9];
  l=LSM-nm[21]-nm[19];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((i==k)&&(j==l)){
      if((uflg[i]<9)&&(uflg[j]<9)){
        nm[8]=i; nm[20]=j;
        uflg[i]++; uflg[j]++;
        stp11();
        uflg[j]--; uflg[i]--;}}}
}
//Level #3:
//Set n11=LSM-n2-n20 & n17
void stp11(){
 short i,j;
  i=LSM-nm[2]-nm[20];
  j=LSM-nm[26]-nm[8];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((uflg[i]<9)&&(uflg[j]<9)){
      nm[11]=i; nm[17]=j;
      uflg[i]++; uflg[j]++;
      stp12();
      uflg[j]--; uflg[i]--;}}
}
//Set n12=LSM-n10-n11 & n16
void stp12(){
 short i,j,k,l;
  i=LSM-nm[10]-nm[11];
  j=LSM-nm[18]-nm[17];
  k=LSM-nm[3]-nm[21];
  l=LSM-nm[25]-nm[7];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((i==k)&&(j==l)){
      if((uflg[i]<9)&&(uflg[j]<9)){
        nm[12]=i; nm[16]=j;
        uflg[i]++; uflg[j]++;
        stp13();
        uflg[j]--; uflg[i]--;}}}
}
//Set n13=LSM-n10-n16 & n15
void stp13(){
 short i,j,k,l;
  i=LSM-nm[10]-nm[16];
  j=LSM-nm[18]-nm[12];
  k=LSM-nm[4]-nm[22];
  l=LSM-nm[24]-nm[6];
  if((0<=i)&&(i<3)&&(i+j==SC)){
    if((i==k)&&(j==l)){
      if((uflg[i]<9)&&(uflg[j]<9)){
        nm[13]=i; nm[15]=j;
        uflg[i]++; uflg[j]++;
        stp14();
        uflg[j]--; uflg[i]--;}}}
}
//* The Last Checks *
```

```
void stp14(){
 short sm1,sm2,sm3;
   sm1=nm[5]+nm[14]+nm[23];
   sm2=nm[11]+nm[14]+nm[17];
   sm3=nm[13]+nm[14]+nm[15];
   if((sm1==LSM)&&(sm2==LSM)&&(sm2==LSM)){recordans();}
}
//
//* Record the Euler Units */
void recordans(){
 short n;
   teu[cnt][0]=cnt+1;
   for(n=1;n<28;n++){teu[cnt][n]=nm[n];}
   cnt++;
}
//
//* Print the Euler Units *
void preunit(){
 short m,l,ls,l1,n,t;
   for(m=0;m<cnt;m=m+4){
     printf("%11d/%13d/%13d/%13d/\n",
            teu[m][0],teu[m+1][0],teu[m+2][0],teu[m+3][0]);
     for(l=0;l<9;l++){ls=l%3; l1=l+1;
       for(n=ls;n>0;n--){printf(" ");}
       printf("%4d%4d%4d  ",teu[m][l1],teu[m][l1+9],teu[m][l1+18]);
       printf("%4d%4d%4d  ",teu[m+1][l1],teu[m+1][l1+9],teu[m+1][l1+18]);
       printf("%4d%4d%4d  ",teu[m+2][l1],teu[m+2][l1+9],teu[m+2][l1+18]);
       printf("%4d%4d%4d  ",teu[m+3][l1],teu[m+3][l1+9],teu[m+3][l1+18]);
       printf("\n");}
     printf("\n");
   }
   printf(" [Count of Euler Units = %d]\n",cnt);
//* Check each pair how well they match *
   for(m=0;m<cnt;m++){
     for(l=0;l<cnt;l++){
       t=0;
       for(n=1;n<28;n++){if(teu[m][n]==teu[l][n]){t++;}}
       mtc[m][l]=t;
       t=0;
       for(n=1;n<28;n++){if(teu[m][n]+teu[l][n]==SC){t++;}}
       if(mtc[m][l]<t){mtc[m][l]=t;}
     }}
   //* Print the Reference Table *
   printf("\n");
   printf(" [Reference Table for the Best Matching]\n");
   printf(" SC|");
   for(n=1;n<=cnt;n++){printf("%3d",n);}
   printf("\n");
   printf(" --+------------------------\n");
   for(m=0;m<cnt;m++){
     printf("%3d|",m+1);
     for(n=0;n<cnt;n++){printf("%3d",mtc[m][n]);}
     printf("\n");
   }
}
//
//* Combine abd Compose *
void cmbcmp(){
 short n,d,lu,md,fc;
   lu=8; md=9;
   for(u1=0;u1<lu;u1++){
     for(u2=0;u2<lu;u2++){
       if(mtc[u2][u1]==md){
         for(u3=0;u3<lu;u3++){
           if((mtc[u3][u1]==md)&&(mtc[u3][u2]==md)){
```

```
                for(n=1;n<28;n++){uflg[n]=0;}
                for(n=1;n<28;n++){
                  d=teu[u1][n]*9+teu[u2][n]*3+teu[u3][n]+1;
                  nm[n]=d; uflg[d]++;
                }
                fc=0;
                for(n=1;n<28;n++){
                  if(uflg[n]==1){fc++;}else{break;}
                }
                if(fc==27){
                  if((nm[1]<nm[3])&&(nm[1]<nm[7])&&(nm[1]<nm[9])&&(nm[1]<nm[19])){
                    if((nm[1]<nm[21])&&(nm[1]<nm[25])&&(nm[1]<nm[27])){
                      if((nm[2]>nm[4])&&(nm[4]>nm[10])){prfmcans();
                      }}}}
              }
            }
          }
        }
      }
    }
//
//* Print 3 Forms of the Answer *
void prfmcans(){
  cnt++;
  printf("%10d/%12d/%12d/    [Math]%30d#\n",u1+1,u2+1,u3+1,cnt);
  printf("%3d---%d---%d",teu[u1][1],teu[u1][10],teu[u1][19]);
  printf("%5d---%d---%d",teu[u2][1],teu[u2][10],teu[u2][19]);
  printf("%5d---%d---%d",teu[u3][1],teu[u3][10],teu[u3][19]);
  printf("%6d-----%2d-----%2d",nm[1]-1,nm[10]-1,nm[19]-1);
  printf("%7d-----%2d-----%2d\n",nm[1],nm[10],nm[19]);
  printf("  |%d%4d  |%d",teu[u1][2],teu[u1][11],teu[u1][20]);
  printf("   |%d%4d  |%d",teu[u2][2],teu[u2][11],teu[u2][20]);
  printf("   |%d%4d  |%d",teu[u3][2],teu[u3][11],teu[u3][20]);
  printf("    |%2d%7d    |%2d",nm[2]-1,nm[11]-1,nm[20]-1);
  printf("    |%2d%7d    |%2d\n",nm[2],nm[11],nm[20]);
  printf("  |%2d---%d---%d",teu[u1][3],teu[u1][12],teu[u1][21]);
  printf("  |%2d---%d---%d",teu[u2][3],teu[u2][12],teu[u2][21]);
  printf("  |%2d---%d---%d",teu[u3][3],teu[u3][12],teu[u3][21]);
  printf("   |%4d-----%2d------%d",nm[3]-1,nm[12]-1,nm[21]-1);
  printf("  |%4d-----%2d------%d\n",nm[3],nm[12],nm[21]);
  printf("%3d |%2d%4d",teu[u1][4],teu[u1][13],teu[u1][22]);
  printf(" |%3d |%2d%4d",teu[u2][4],teu[u2][13],teu[u2][22]);
  printf(" |%3d |%2d%4d",teu[u3][4],teu[u3][13],teu[u3][22]);
  printf(" |%4d   | %2d%7d   |",nm[4]-1,nm[13]-1,nm[22]-1);
  printf("%3d   | %2d%7d   |\n",nm[4],nm[13],nm[22]);
  printf("  |%d|%3d  |%d|",teu[u1][5],teu[u1][14],teu[u1][23]);
  printf("  |%d|%3d  |%d|",teu[u2][5],teu[u2][14],teu[u2][23]);
  printf("  |%d|%3d  |%d|",teu[u3][5],teu[u3][14],teu[u3][23]);
  printf("   |%2d |%5d    |%2d |",nm[5]-1,nm[14]-1,nm[23]-1);
  printf("  |%2d |%5d    |%2d |\n",nm[5],nm[14],nm[23]);
  printf("  |%2d%4d |%2d",teu[u1][6],teu[u1][15],teu[u1][24]);
  printf("  |%2d%4d |%2d",teu[u2][6],teu[u2][15],teu[u2][24]);
  printf("  |%2d%4d |%2d",teu[u3][6],teu[u3][15],teu[u3][24]);
  printf("   |%4d%7d  |%4d",nm[6]-1,nm[15]-1,nm[24]-1);
  printf("  |%4d%7d  |%4d\n",nm[6],nm[15],nm[24]);
  printf("%3d-|-%d---%d |",teu[u1][7],teu[u1][16],teu[u1][25]);
  printf("%3d-|-%d---%d |",teu[u2][7],teu[u2][16],teu[u2][25]);
  printf("%3d-|-%d---%d |",teu[u3][7],teu[u3][16],teu[u3][25]);
  printf("%4d---|--%d-----%2d   |",nm[7]-1,nm[16]-1,nm[25]-1);
  printf("%3d---|--%d-----%2d   |\n",nm[7],nm[16],nm[25]);
  printf("%4d|%3d%4d|",teu[u1][8],teu[u1][17],teu[u1][26]);
  printf("%4d|%3d%4d|",teu[u2][8],teu[u2][17],teu[u2][26]);
  printf("%4d|%3d%4d|",teu[u3][8],teu[u3][17],teu[u3][26]);
  printf("%6d |%5d%7d |",nm[8]-1,nm[17]-1,nm[26]-1);
  printf("%5d |%5d%7d |\n",nm[8],nm[17],nm[26]);
```

```
  printf("%5d---%d---%d",teu[u1][9],teu[u1][18],teu[u1][27]);
  printf("%5d---%d---%d",teu[u2][9],teu[u2][18],teu[u2][27]);
  printf("%5d---%d---%d",teu[u3][9],teu[u3][18],teu[u3][27]);
  printf("%8d-----%2d-----%2d",nm[9]-1,nm[18]-1,nm[27]-1);
  printf("%7d-----%2d-----%2d\n",nm[9],nm[18],nm[27]);
  printf("\n");
}
//
```

単位方陣は，８個出て来た。それを３個ずつ選んで組み合わせて，解を合成する。
組み合わせる時に同じものを選ばないよう，また「２の補数ユニット」を選ばないよう，(u2, u1)，(u3, u1)，(u3, u2) と２つずつ組み合わせてチェックする必要がある。

** 対称立体三方陣用のオイラー・ユニット **

```
        1/           2/           3/           4/
 0   1   2    0   1   2    0   2   1    1   2   0
  2   0   1    1   2   0    1   0   2    2   0   1
   1   2   0    2   0   1    2   1   0    0   1   2
 1   2   0    2   0   1    1   0   2    2   0   1
  0   1   2    0   1   2    2   1   0    0   1   2
   2   0   1    1   2   0    0   2   1    1   2   0
 2   0   1    1   2   0    2   1   0    0   1   2
  1   2   0    2   0   1    0   2   1    1   2   0
   0   1   2    0   1   2    1   0   2    2   0   1

        5/           6/           7/           8/
 1   0   2    2   0   1    2   1   0    2   1   0
  0   2   1    1   2   0    1   0   2    0   2   1
   2   1   0    0   1   2    0   2   1    1   0   2
 0   2   1    1   2   0    0   2   1    1   0   2
  2   1   0    0   1   2    2   1   0    2   1   0
   1   0   2    2   0   1    1   0   2    0   2   1
 2   1   0    0   1   2    1   0   2    0   2   1
  1   0   2    2   0   1    0   2   1    1   0   2
   0   2   1    1   2   0    2   1   0    2   1   0
```

[オイラー・ユニットの総数＝８個]

[良い組み合わせを得るための参照表]

```
SC|  1  2  3  4  5  6  7  8
--+------------------------
 1| 27  9  9  9  9  9  9 27
 2|  9 27  9  9  9  9 27  9
 3|  9  9 27  9  9 27  9  9
 4|  9  9  9 27 27  9  9  9
 5|  9  9  9 27 27  9  9  9
 6|  9  9 27  9  9 27  9  9
 7|  9 27  9  9  9  9 27  9
 8| 27  9  9  9  9  9  9 27
```

** 新オイラー法によって再構成した対称立体三方陣の標準解４個 **

(/オイラー・ユニット：高位/ 中位/ 低位/ [数学的表記] 古典的表記 解番号#)

```
 /EU    1/           2/           3/    [Math]                          1#
 0---1---2    0---1---2    0---2---1     0-----14-----25       1-----15-----26
 |2   0  |1   |1   2  |0   |1   0  |2    |22       6    |11    |23       7    |12
 | 1---2---0  | 2---0---1  | 2---1---0   |  17-----19-----3   |  18-----20-----4
 1 | 2   0 |  2 | 0   1 |  1 | 0   2 |  16    | 18       5   | 17    | 19       6   |
 |0|  1  |2|  |0|  1  |2|  |2|  1  |0|   |  2 |    13    |24 |   | 3 |    14    |25 |
 | 2   0 | 1  | 1   2 | 0  | 0   2 | 1   |  21       8   |  10  |  22       9   |  11
 2-|-0---1 |  1-|-2---0 |  2-|-1---0 |  23---|--7-----  9    | 24---|--8-----10    |
  1|  2   0|   2|  0   1|   0|  2   1|     15 |   20       4 |   16 |   21       5 |
   0---1---2    0---1---2    1---0---2        1-----12-----26       2-----13-----27
```

```
 /EU    1/          2/          4/    [Math]                      2#
0---1---2   0---1---2   1---2---0    1-----14-----24      2-----15-----25
 |2   0  |1  |1   2  |0  |2   0  |1   |23      6    |10    |24      7    |11
 | 1---2---0  | 2---0---1  | 0---1---2  |  15-----19-----5  |  16-----20-----6
1 | 2   0 |  2 | 0   1 |  2 | 0   1 |  17   | 18      4   | 18   | 19      5   |
 |0|  1  |2|  |0|  1  |2|  |0|  1  |2|   | 0 |   13    |26 |   | 1 |   14    |27 |
 | 2   0 | 1  | 1   2 | 0  | 1   2 | 0  |  22      8   |  9   |  23      9   |  10
2-|-0---1 |  1-|-2---0 |  0-|-1---2 |  21---|--7-----11    |  22---|--8-----12    |
 1|  2  0|   2|  0  1|   1|  2  0|     16 |   20      3  |    17 |   21      4  |
  0---1---2    0---1---2    2---0---1       2-----12-----25       3-----13-----26

 /EU    1/          4/          2/    [Math]                      3#
0---1---2   1---2---0   0---1---2    3-----16-----20      4-----17-----21
 |2   0  |1  |2   0  |1  |1   2  |0   |25      2    |12    |26      3    |13
 | 1---2---0  | 0---1---2  | 2---0---1  |  11-----21-----7  |  12-----22-----8
1 | 2   0 |  2 | 0   1 |  2 | 0   1 |  17   | 18      4   | 18   | 19      5   |
 |0|  1  |2|  |0|  1  |2|  |0|  1  |2|   | 0 |   13    |26 |   | 1 |   14    |27 |
 | 2   0 | 1  | 1   2 | 0  | 1   2 | 0  |  22      8   |  9   |  23      9   |  10
2-|-0---1 |  0-|-1---2 |  1-|-2---0 |  19---|--5-----15    |  20---|--6-----16    |
 1|  2  0|   1|  2  0|   2|  0  1|     14 |   24      1  |    15 |   25      2  |
  0---1---2    2---0---1    0---1---2       6-----10-----23       7-----11-----24

 /EU    1/          4/          6/    [Math]                      4#
0---1---2   1---2---0   2---0---1    5-----15-----19      6-----16-----20
 |2   0  |1  |2   0  |1  |1   2  |0   |25      2    |12    |26      3    |13
 | 1---2---0  | 0---1---2  | 0---1---2  |   9-----22-----8  |  10-----23-----9
1 | 2   0 |  2 | 0   1 |  1 | 2   0 |  16   | 20      3   | 17   | 21      4   |
 |0|  1  |2|  |0|  1  |2|  |0|  1  |2|   | 0 |   13    |26 |   | 1 |   14    |27 |
 | 2   0 | 1  | 1   2 | 0  | 2   0 | 1  |  23      6   |  10  |  24      7   |  11
2-|-0---1 |  0-|-1---2 |  0-|-1---2 |  18---|--4-----17    |  19---|--5-----18    |
 1|  2  0|   1|  2  0|   2|  0  1|     14 |   24      1  |    15 |   25      2  |
  0---1---2    2---0---1    1---2---0       7-----11-----21       8-----12-----22
```

[解の総数：＝４個]　[OK!]

** Calculated and Listed by Kanji Setsuda
on Apr. 2, 2012 with MacOSX 10.7.3 and Xcode 4.2.1

この４個の解は，以前に得ていた標準解４個と同じものである。新定義のもとでも，余分なものを作り出したりはしなかったし，必要な解が欠落したりもしなかった。

全ユニットの行も列も柱も，全部が「**完全オイラー型**」だった。立方体の主対角線４本の内の１本だけがどうしても**完全オイラー型**にならないが，他の３本は {0, 1, 2} で構成されている。「‘準’完全オイラー方陣」と言って良いくらいの出来である。

何よりも，解そのものの成り立ちが三進法分解図・構成図によって示されたことが，大きい。構造解析のための新たな地平が開かれた，と言っても良いかと思う。

３．五進法による『新オイラー法』で，汎五方陣をあらためて構成する

以前に汎対角線五方陣を作ったのは，『桂馬とび構成法』によってであった。

この時は，まだ**新オイラー法**に到達していなかったので，この方法は五次に対しては依然未適用のままだった。今回それを実際に試して見よう。

汎五方陣が「**完全オイラー方陣**」であることは，既に確かめられているので，早速にも「**ラテン方陣**」の設計・構成にかかる。新オイラー法の仕組みは，次のように図解される。

** 新オイラー法の概念図：五進法による分解図と構成図（チェック・サム付き）**

[汎五方陣]	1#|Rw|Cl\P1/P2|	/D5i	高位|Rw|Cl\P1/P2|	低位|Rw|Cl\P1/P2|
1 23 20 12 9	65|65|65|65|	0 4 3 2 1	10|10|10|10|	0 2 4 1 3	10|10|10|10|
15 7 4 21 18	65|65|65|65|	2 1 0 4 3	10|10|10|10|	4 1 3 0 2	10|10|10|10|
24 16 13 10 2	65|65|65|65|	4 3 2 1 0	10|10|10|10|	3 0 2 4 1	10|10|10|10|
8 5 22 19 11	65|65|65|65|	1 0 4 3 2	10|10|10|10|	2 4 1 3 0	10|10|10|10|
17 14 6 3 25	65|65|65|65|	3 2 1 0 4	10|10|10|10|	1 3 0 2 4	10|10|10|10|

（元の数値から１を引いて，それを５で割る。整商を高位ユニットに，剰余を低位におさめる。）

［ラテン・ユニットによる構成］

```
 Lu:   4/H|Rw|Cl\P1/P2|       31/L|Rw|Cl\P1/P2|   [汎五方陣]   #|Rw|Cl\P1/P2|
 0 4 3 2 1|10|10|10|10|  0 2 4 1 3|10|10|10|10|    1 23 20 12  9|65|65|65|65|
 2 1 0 4 3|10|10|10|10|  4 1 3 0 2|10|10|10|10|   15  7  4 21 18|65|65|65|65|
 4 3 2 1 0|10|10|10|10|  3 0 2 4 1|10|10|10|10|   24 16 13 10  2|65|65|65|65|
 1 0 4 3 2|10|10|10|10|  2 4 1 3 0|10|10|10|10|    8  5 22 19 11|65|65|65|65|
 3 2 1 0 4|10|10|10|10|  1 3 0 2 4|10|10|10|10|   17 14  6  3 25|65|65|65|65|
```

（高位ユニットの数値に５を掛け，それに低位の数値を足す。最後に１を加えて元の方陣を再構成する。）

途中で１を引いたり足したりするのは，数学的表記が介在するためだ。上の図ではそれが省略されている。頭の中で補いながら見てほしい。

では，どうやって**ラテン・ユニット**を作るか。分解図の結果を再利用するような横着なことはしない。必ず新規に作らなければならない。必要な定義条件に従って実直に構成する。

定義の基本図と基本式を次のようにしよう。

```
//
//** Basic Form for the Latin Units of Pan-diagonal MS55 **
//              .--.--.--.--.--.
//    2  3  4  5| 1| 2| 3| 4| 5| 1  2  3  4
//              |--+--+--+--+--|
//    7  8  9 10| 6| 7| 8| 9|10| 6  7  8  9
//              |--+--+--+--+--|
//   12 13 14 15|11|12|13|14|15|11 12 13 14
//              |--+--+--+--+--|
//   17 18 19 20|16|17|18|19|20|16 17 18 19
//              |--+--+--+--+--|
//   22 23 24 25|21|22|23|24|25|21 22 23 24
//              '--'--'--'--'--'
//** Define Conditions for Rows and Columns: C=10 **
//    n1+n2+n3+n4+n5=C      ... rw1| n1+n6+n11+n16+n21=C   ... cl1
//    n6+n7+n8+n9+n10=C     ... rw2| n2+n7+n12+n17+n22=C   ... cl2
//   n11+n12+n13+n14+n15=C  ... rw3| n3+n8+n13+n18+n23=C   ... cl3
//   n16+n17+n18+n19+n20=C  ... rw4| n4+n9+n14+n19+n24=C   ... cl4
//   n21+n22+n23+n24+n25=C  ... rw5| n5+n10+n15+n20+n25=C  ... cl5
//** Pan-diagonal Conditions of the Latin Units: C=10 **
//    n1+n7+n13+n19+n25=C    ... pd1| n1+n10+n14+n18+n22=C  ... pd6
//    n2+n8+n14+n20+n21=C    ... pd2| n2+n6+n15+n19+n23=C   ... pd7
//    n3+n9+n15+n16+n22=C    ... pd3| n3+n7+n11+n20+n24=C   ... pd8
//    n4+n10+n11+n17+n23=C   ... pd4| n4+n8+n12+n16+n25=C   ... pd9
//    n5+n6+n12+n18+n24=C    ... pd5| n5+n9+n13+n17+n21=C   ... pd10
//
```

元の汎五方陣を定義した時の条件と「そっくり」のものを用意する。変えるのは，定和値だけだ。元の方陣とユニットの間にあるこの「相似の構造」が，新オイラー法の要請だ。

このほかに各行列・各汎対角線の数字構成に，**ラテン方陣**ならではの特別注文を付ける。どれも｛0，1，2，3，4｝の内容構成になるように，各々の場所で各数字が１度ずつしか使われないように約束をする。それを監視するために，フラグを何本も立てる。

また，例の条件も付け加える。「ユニットの各面で使う数字は，{0，1，2，3，4}の５個に限られる。その時どの数字も平等に５回ずつ使う。回数に過不足があってはならない。」

プログラムは，例えば次のように書く。

```
//**  Pan-diagonal Magic Squares of Order 5: Reconstructed by  **
//**  the 5-th Increment Number System and New Euler's Method  **
//**  File: 'CES55PD.c': Dictated by Kanji Setsuda  **
//**  in 2003, 2006, 2011, and Revised on Apr. 9, 2012 **
//**  Working with MacOSX 10.7.3 & Xcode 4.2.1  **
//
#include <stdio.h>
//
```

```
short int cnt, ecnt, cnt2;
long u1,u2;
short cnt3;
short SC, LSM;
short nm[26], uflg[26], tn[26];
short anm[8][27];
char teu[241][29];
short solt[3601][28];
short cs[4][21];
//
short rw1[5],  cl1[5],  pd1[5],  pb1[5];
short rw2[5],  cl2[5],  pd2[5],  pb2[5];
short rw3[5],  cl3[5],  pd3[5],  pb3[5];
short rw4[5],  cl4[5],  pd4[5],  pb4[5];
short rw5[5],  cl5[5],  pd5[5],  pb5[5];
//
void estp01(void), estp02(void), estp03(void), estp04(void), estp05(void);
void estp06(void), estp07(void), estp08(void), estp09(void), estp10(void);
void estp11(void), estp12(void), estp13(void), estp14(void), estp15(void);
void estp16(void), estp17(void), estp18(void), estp19(void), estp20(void);
void estp21(void), estp22(void), estp23(void), estp24(void), estp25(void);
void eurecord(void), preunit(short x);
//
short chkmtc(short x, short y);
void cmbcmp(void), solsave(void);
void prans(short x), pr2ans(void), pr1ans(void);
void chksms(short x);
//
//* Main Program *
int main(void){
 short n;
  printf("\n");
  printf("** Complete Euler Squares 5x5 for the Pan-diagonal Type: Reconstructed\n");
  printf("   by the 5-th Increment Number System and New Euler's Method **\n");
  for(n=0;n<26;n++){nm[n]=0;}
  for(n=0;n<5;n++){uflg[n]=0;
    rw1[n]=0;  cl1[n]=0;  pd1[n]=0;  pb1[n]=0;
    rw2[n]=0;  cl2[n]=0;  pd2[n]=0;  pb2[n]=0;
    rw3[n]=0;  cl3[n]=0;  pd3[n]=0;  pb3[n]=0;
    rw4[n]=0;  cl4[n]=0;  pd4[n]=0;  pb4[n]=0;
    rw5[n]=0;  cl5[n]=0;  pd5[n]=0;  pb5[n]=0;
  }
  SC=4; LSM=10; cnt=0;
  printf("\n");
  printf(" ** List of Latin Units made by /D5i **\n");
  estp01();                       // Make the Latin Units
  if(cnt3>0){preunit(cnt3);}      // Print the Rest One
  ecnt=cnt; printf("// [Count of Latin Units for High Positions = %d]\n",ecnt);
  printf("\n");
  printf("** Complete Euler Squares of Order 5 for the Pan-diagonal Type:\n");
  printf("   Reconstructed by the 5-th Increment System and New Euler's Method **\n");
  printf("  (Latin Units:/High/Low; Sol_Number#; Check_Sums:|Row|Column\\Pd1/Pd2|)\n");
  cnt=0; cnt3=0;
  cmbcmp();                       // Combine and Compose
  prans(288);                     // Print the Solutions with that Interval
  printf(" [Total Count of Solutions = %d]  [OK!]\n",cnt);
  printf("** Calculated and Listed by Kanji Setsuda\n");
  printf("   on Apr. 9, 2012 with MacOSX 10.7.3 & Xcode 4.2.1 **\n");
  printf("\n");
 return 0;
}
//
//** Program Modules for Latin Units of 'Complete Euler Squares' **
//**  of Order 5 for Pan-diagonal Magic Squares: Listed by **
//** Kanji Setsuda on Feb. 2, 2011 with MacOSX and Xcode 3.1.2 **
```

```
// Level #1:
// Set n1{1,1,1,1}
void estp01(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw1[a]==0)&&(cl1[a]==0)&&(pd1[a]==0)&&(pb1[a]==0)){
        nm[1]=a;
        uflg[a]++; rw1[a]=1; cl1[a]=1; pd1[a]=1; pb1[a]=1;
        estp02();
        rw1[a]=0; cl1[a]=0; pd1[a]=0; pb1[a]=0; uflg[a]--;}}
  }
}
// Set n25{5,5,1,4}
void estp02(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw5[a]==0)&&(cl5[a]==0)&&(pd1[a]==0)&&(pb4[a]==0)){cnt2=0;
        nm[25]=a;
        uflg[a]++; rw5[a]=1; cl5[a]=1; pd1[a]=1; pb4[a]=1;
        estp03();
        rw5[a]=0; cl5[a]=0; pd1[a]=0; pb4[a]=0; uflg[a]--;}}
  }
}
// Set n7{2,2,1,3}
void estp03(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw2[a]==0)&&(cl2[a]==0)&&(pd1[a]==0)&&(pb3[a]==0)){if(nm[1]==0){cnt2=0;}
        nm[7]=a;
        uflg[a]++; rw2[a]=1; cl2[a]=1; pd1[a]=1; pb3[a]=1;
        estp04();
        rw2[a]=0; cl2[a]=0; pd1[a]=0; pb3[a]=0; uflg[a]--;}}
  }
}
// Set n19{4,4,1,2}
void estp04(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw4[a]==0)&&(cl4[a]==0)&&(pd1[a]==0)&&(pb2[a]==0)){
        nm[19]=a;
        uflg[a]++; rw4[a]=1; cl4[a]=1; pd1[a]=1; pb2[a]=1;
        estp05();
        rw4[a]=0; cl4[a]=0; pd1[a]=0; pb2[a]=0; uflg[a]--;}}
  }
}
// Set n13{3,3,1,5}
void estp05(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw3[a]==0)&&(cl3[a]==0)&&(pd1[a]==0)&&(pb5[a]==0)){
        nm[13]=a;
        uflg[a]++; rw3[a]=1; cl3[a]=1; pd1[a]=1; pb5[a]=1;
        estp06();
        rw3[a]=0; cl3[a]=0; pd1[a]=0; pb5[a]=0; uflg[a]--;}}
  }
}
// Level #2:
// Set n2{1,2,2,2}
void estp06(){
 short a;
```

```
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw1[a]==0)&&(cl2[a]==0)&&(pd2[a]==0)&&(pb2[a]==0)){
        nm[2]=a;
        uflg[a]++; rw1[a]=1; cl2[a]=1; pd2[a]=1; pb2[a]=1;
        estp07();
        rw1[a]=0; cl2[a]=0; pd2[a]=0; pb2[a]=0; uflg[a]--;}}
  }
}
// Set n3{1,3,3,3}
void estp07(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw1[a]==0)&&(cl3[a]==0)&&(pd3[a]==0)&&(pb3[a]==0)){
        nm[3]=a;
        uflg[a]++; rw1[a]=1; cl3[a]=1; pd3[a]=1; pb3[a]=1;
        estp08();
        rw1[a]=0; cl3[a]=0; pd3[a]=0; pb3[a]=0; uflg[a]--;}}
  }
}
// Set n4{1,4,4,4}
void estp08(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw1[a]==0)&&(cl4[a]==0)&&(pd4[a]==0)&&(pb4[a]==0)){
        nm[4]=a;
        uflg[a]++; rw1[a]=1; cl4[a]=1; pd4[a]=1; pb4[a]=1;
        estp09();
        rw1[a]=0; cl4[a]=0; pd4[a]=0; pb4[a]=0; uflg[a]--;}}
  }
}
// Set n5{1,5,5,5}
void estp09(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw1[a]==0)&&(cl5[a]==0)&&(pd5[a]==0)&&(pb5[a]==0)){
        nm[5]=a;
        uflg[a]++; rw1[a]=1; cl5[a]=1; pd5[a]=1; pb5[a]=1;
        estp10();
        rw1[a]=0; cl5[a]=0; pd5[a]=0; pb5[a]=0; uflg[a]--;}}
  }
}
// Level #3:
// Set n11{3,1,4,3}
void estp10(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw3[a]==0)&&(cl1[a]==0)&&(pd4[a]==0)&&(pb3[a]==0)){
        nm[11]=a;
        uflg[a]++; rw3[a]=1; cl1[a]=1; pd4[a]=1; pb3[a]=1;
        estp11();
        rw3[a]=0; cl1[a]=0; pd4[a]=0; pb3[a]=0; uflg[a]--;}}
  }
}
// Set n16{4,1,3,4}
void estp11(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw4[a]==0)&&(cl1[a]==0)&&(pd3[a]==0)&&(pb4[a]==0)){
        nm[16]=a;
```

```
        uflg[a]++; rw4[a]=1; cl1[a]=1; pd3[a]=1; pb4[a]=1;
        estp12();
        rw4[a]=0; cl1[a]=0; pd3[a]=0; pb4[a]=0; uflg[a]--;}}
  }
}
// Set n21{5,1,2,5}
void estp12(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw5[a]==0)&&(cl1[a]==0)&&(pd2[a]==0)&&(pb5[a]==0)){
        nm[21]=a;
        uflg[a]++; rw5[a]=1; cl1[a]=1; pd2[a]=1; pb5[a]=1;
        estp13();
        rw5[a]=0; cl1[a]=0; pd2[a]=0; pb5[a]=0; uflg[a]--;}}
  }
}
// Set n6{2,1,5,2}
void estp13(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw2[a]==0)&&(cl1[a]==0)&&(pd5[a]==0)&&(pb2[a]==0)){
        nm[6]=a;
        uflg[a]++; rw2[a]=1; cl1[a]=1; pd5[a]=1; pb2[a]=1;
        estp14();
        rw2[a]=0; cl1[a]=0; pd5[a]=0; pb2[a]=0; uflg[a]--;}}
  }
}
// Level #4:
// Set n8{2,3,2,4}
void estp14(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw2[a]==0)&&(cl3[a]==0)&&(pd2[a]==0)&&(pb4[a]==0)){
        nm[8]=a;
        uflg[a]++; rw2[a]=1; cl3[a]=1; pd2[a]=1; pb4[a]=1;
        estp15();
        rw2[a]=0; cl3[a]=0; pd2[a]=0; pb4[a]=0; uflg[a]--;}}
  }
}
// Set n9{2,4,3,5}
void estp15(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw2[a]==0)&&(cl4[a]==0)&&(pd3[a]==0)&&(pb5[a]==0)){
        nm[9]=a;
        uflg[a]++; rw2[a]=1; cl4[a]=1; pd3[a]=1; pb5[a]=1;
        estp16();
        rw2[a]=0; cl4[a]=0; pd3[a]=0; pb5[a]=0; uflg[a]--;}}
  }
}
// Set n10{2,5,4,1}
void estp16(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw2[a]==0)&&(cl5[a]==0)&&(pd4[a]==0)&&(pb1[a]==0)){
        nm[10]=a;
        uflg[a]++; rw2[a]=1; cl5[a]=1; pd4[a]=1; pb1[a]=1;
        estp17();
        rw2[a]=0; cl5[a]=0; pd4[a]=0; pb1[a]=0; uflg[a]--;}}
  }
```

```
}
// Set n15{3,5,3,2}
void estp17(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw3[a]==0)&&(cl5[a]==0)&&(pd3[a]==0)&&(pb2[a]==0)){
        nm[15]=a;
        uflg[a]++; rw3[a]=1; cl5[a]=1; pd3[a]=1; pb2[a]=1;
        estp18();
        rw3[a]=0; cl5[a]=0; pd3[a]=0; pb2[a]=0; uflg[a]--;}}
  }
}
// Set n14{3,4,2,1}
void estp18(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw3[a]==0)&&(cl4[a]==0)&&(pd2[a]==0)&&(pb1[a]==0)){
        nm[14]=a;
        uflg[a]++; rw3[a]=1; cl4[a]=1; pd2[a]=1; pb1[a]=1;
        estp19();
        rw3[a]=0; cl4[a]=0; pd2[a]=0; pb1[a]=0; uflg[a]--;}}
  }
}
// Set n12{3,2,5,4}
void estp19(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw3[a]==0)&&(cl2[a]==0)&&(pd5[a]==0)&&(pb4[a]==0)){
        nm[12]=a;
        rw3[a]=1; cl2[a]=1; pd5[a]=1; pb4[a]=1;
        estp20();
        rw3[a]=0; cl2[a]=0; pd5[a]=0; pb4[a]=0; uflg[a]--;}}
  }
}
// Set n18{4,3,5,1}
void estp20(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw4[a]==0)&&(cl3[a]==0)&&(pd5[a]==0)&&(pb1[a]==0)){
        nm[18]=a;
        uflg[a]++; rw4[a]=1; cl3[a]=1; pd5[a]=1; pb1[a]=1;
        estp21();
        rw4[a]=0; cl3[a]=0; pd5[a]=0; pb1[a]=0; uflg[a]--;}}
  }
}
// Set n17{4,2,4,5}
void estp21(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw4[a]==0)&&(cl2[a]==0)&&(pd4[a]==0)&&(pb5[a]==0)){
        nm[17]=a;
        uflg[a]++; rw4[a]=1; cl2[a]=1; pd4[a]=1; pb5[a]=1;
        estp22();
        rw4[a]=0; cl2[a]=0; pd4[a]=0; pb5[a]=0; uflg[a]--;}}
  }
}
// Set n20{4,5,2,3}
void estp22(){
 short a;
  for(a=0;a<5;a++){
```

```
    if(uflg[a]<5){
      if((rw4[a]==0)&&(cl5[a]==0)&&(pd2[a]==0)&&(pb3[a]==0)){
        nm[20]=a;
        uflg[a]++; rw4[a]=1; cl5[a]=1; pd2[a]=1; pb3[a]=1;
        estp23();
        rw4[a]=0; cl5[a]=0; pd2[a]=0; pb3[a]=0; uflg[a]--;}}
  }
}
// Set n24{5,4,5,3}
void estp23(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw5[a]==0)&&(cl4[a]==0)&&(pd5[a]==0)&&(pb3[a]==0)){
        nm[24]=a;
        uflg[a]++; rw5[a]=1; cl4[a]=1; pd5[a]=1; pb3[a]=1;
        estp24();
        rw5[a]=0; cl4[a]=0; pd5[a]=0; pb3[a]=0; uflg[a]--;}}
  }
}
// Set n23{5,3,4,2}
void estp24(){
 short a;
  for(a=0;a<5;a++){
    if(uflg[a]<5){
      if((rw5[a]==0)&&(cl3[a]==0)&&(pd4[a]==0)&&(pb2[a]==0)){
        nm[23]=a;
        uflg[a]++; rw5[a]=1; cl3[a]=1; pd4[a]=1; pb2[a]=1;
        estp25();
        rw5[a]=0; cl3[a]=0; pd4[a]=0; pb2[a]=0; uflg[a]--;}}
  }
}
// Set n22{5,2,3,1}
void estp25(){
 short a;
  for(a=4;a>=0;a--){
    if(uflg[a]<5){
      if((rw5[a]==0)&&(cl2[a]==0)&&(pd3[a]==0)&&(pb1[a]==0)){
        nm[22]=a;
        uflg[a]++; rw5[a]=1; cl2[a]=1; pd3[a]=1; pb1[a]=1;
        eurecord();
        rw5[a]=0; cl2[a]=0; pd3[a]=0; pb1[a]=0; uflg[a]--;}}
  }
}
//
// Record Latin Units
void eurecord(){
 short n;
  teu[cnt][0]=cnt+1;
  for(n=1;n<26;n++){teu[cnt][n]=nm[n];}
  cnt++; cnt2++;
  if(cnt2==1){anm[cnt3][0]=cnt;
    for(n=1;n<26;n++){anm[cnt3][n]=nm[n];}
    cnt3++; if(cnt3==7){preunit(cnt3); cnt3=0;}
  }
}
//
// Print Latin Units
void preunit(short x){
 short l,m,m5,n;
  for(l=0;l<x;l++){printf("  %9d/",anm[l][0]);}
  printf("\n");
  for(m=0;m<5;m++){m5=m*5;
    for(l=0;l<x;l++){printf("  ");
      for(n=1;n<6;n++){printf("%2d",anm[l][m5+n]);}}
```

```
    printf("\n");
  }
}
//
// Check how well they match
short chkmtc(short x, short y){
 short n,d1,d2;
  d1=0; for(n=1;n<26;n++){if(teu[x][n]==teu[y][n]){d1++;}}
  d2=0; for(n=1;n<26;n++){if(teu[x][n]+teu[y][n]==SC){d2++;}}
  if(d1<d2){d1=d2;}
 return d1;
}
//
//* Combine, Compose and Print *
void cmbcmp(){
 short n,d,fc;
  for(u1=0;u1<(ecnt/2);u1++){
    for(u2=0;u2<ecnt;u2++){
      if(chkmtc(u1,u2)==5){
        for(n=1;n<26;n++){uflg[n]=0;}
        for(n=1;n<26;n++){
          d=teu[u1][n]*5+teu[u2][n]+1;
          nm[n]=d; uflg[d]++;
        }
        fc=0;
        for(n=1;n<26;n++){if(uflg[n]==0){fc++;}}
        if(fc==0){
          if((nm[1]<nm[5])&&(nm[5]<nm[21])&&(nm[1]<nm[25])){solsave();}}
      }
    }}
}
//
// Save Answers to the Solution List
void solsave(){
 short n;
  solt[cnt][0]=cnt+1;
  for(n=1;n<26;n++){solt[cnt][n]=nm[n];}
  solt[cnt][26]=u1;  solt[cnt][27]=u2;
  cnt++;
}
//
// Print the Solution List
void prans(short x){
 short s,eu1,eu2;
 short n,d;
  for(s=0;s<cnt;s=s+x){
    anm[0][0]=s+1; eu1=solt[s][26]; eu2=solt[s][27];
    for(n=1;n<26;n++){d=solt[s][n]; anm[0][n]=d; tn[n]=d;}; chksms(0);
    for(n=1;n<26;n++){d=teu[eu1][n]; anm[2][n]=d; tn[n]=d;}; chksms(2);
    for(n=1;n<26;n++){d=teu[eu2][n]; anm[3][n]=d; tn[n]=d;}; chksms(3);
    anm[2][0]=eu1+1; anm[3][0]=eu2+1;
    pr1ans();
  }
}
//
// Print the Rest One
void pr1ans(){
 short l,l5,n;
  printf("  Lu:%4d/H|Rw|Cl\\P1/P2|%9d/L|Rw|Cl\\P1/P2|",anm[2][0],anm[3][0]);
  printf("%16d#|Rw|Cl\\P1/P2|\n",anm[0][0]);
  for(l=0;l<5;l++){l5=l*5;
    printf(" ");
    for(n=1;n<6;n++){printf("%2d",anm[2][l5+n]);}
    printf("|%2d|%2d|%2d|%2d|  ",cs[2][l+1],cs[2][l+6],cs[2][l+11],cs[2][l+16]);
    for(n=1;n<6;n++){printf("%2d",anm[3][l5+n]);}
```

```
    printf("|%2d|%2d|%2d|%2d|  ",cs[3][l+1],cs[3][l+6],cs[3][l+11],cs[3][l+16]);
    for(n=1;n<6;n++){printf("%3d",anm[0][l5+n]);}
    printf("|%2d|%2d|%2d|%2d|\n",cs[0][l+1],cs[0][l+6],cs[0][l+11],cs[0][l+16]);
  }
}
//
//* Check Sums *
void chksms(short x){
  cs[x][1]=tn[1]+tn[2]+tn[3]+tn[4]+tn[5];
  cs[x][2]=tn[6]+tn[7]+tn[8]+tn[9]+tn[10];
  cs[x][3]=tn[11]+tn[12]+tn[13]+tn[14]+tn[15];
  cs[x][4]=tn[16]+tn[17]+tn[18]+tn[19]+tn[20];
  cs[x][5]=tn[21]+tn[22]+tn[23]+tn[24]+tn[25];
  cs[x][6]=tn[1]+tn[6]+tn[11]+tn[16]+tn[21];
  cs[x][7]=tn[2]+tn[7]+tn[12]+tn[17]+tn[22];
  cs[x][8]=tn[3]+tn[8]+tn[13]+tn[18]+tn[23];
  cs[x][9]=tn[4]+tn[9]+tn[14]+tn[19]+tn[24];
  cs[x][10]=tn[5]+tn[10]+tn[15]+tn[20]+tn[25];
  cs[x][11]=tn[1]+tn[7]+tn[13]+tn[19]+tn[25];
  cs[x][12]=tn[2]+tn[8]+tn[14]+tn[20]+tn[21];
  cs[x][13]=tn[3]+tn[9]+tn[15]+tn[16]+tn[22];
  cs[x][14]=tn[4]+tn[10]+tn[11]+tn[17]+tn[23];
  cs[x][15]=tn[5]+tn[6]+tn[12]+tn[18]+tn[24];
  cs[x][16]=tn[1]+tn[10]+tn[14]+tn[18]+tn[22];
  cs[x][17]=tn[2]+tn[6]+tn[15]+tn[19]+tn[23];
  cs[x][18]=tn[3]+tn[7]+tn[11]+tn[20]+tn[24];
  cs[x][19]=tn[4]+tn[8]+tn[12]+tn[16]+tn[25];
  cs[x][20]=tn[5]+tn[9]+tn[13]+tn[17]+tn[21];
}
//
```

単位面が，240面も出て来た。この中から２個ずつを選んで組み合わせ，「相性」を調べながら，解を計算で合成する。その結果を以下に抜粋で示す。

［五進法によるラテン・ユニットのリスト］

1/	5/	9/	13/	17/	21/	25/
0 4 3 2 1	0 4 3 1 2	0 4 2 1 3	0 3 4 2 1	0 3 4 1 2	0 3 2 1 4	0 3 2 1 4
2 1 0 4 3	1 2 0 4 3	1 3 0 4 2	2 1 0 3 4	1 2 0 3 4	1 4 0 3 2	2 1 4 0 3
4 3 2 1 0	4 3 1 2 0	4 2 1 3 0	3 4 2 1 0	3 4 1 2 0	3 2 1 4 0	4 0 3 2 1
1 0 4 3 2	2 0 4 3 1	3 0 4 2 1	1 0 3 4 2	2 0 3 4 1	4 0 3 2 1	3 2 1 4 0
3 2 1 0 4	3 1 2 0 4	2 1 3 0 4	4 2 1 0 3	4 1 2 0 3	2 1 4 0 3	1 4 0 3 2

29/	33/	37/	41/	45/	49/	61/
0 2 4 1 3	0 2 3 1 4	0 3 1 2 4	0 2 1 3 4	0 2 1 4 3	1 4 3 2 0	1 3 4 2 0
1 3 0 2 4	1 4 0 2 3	1 2 4 0 3	1 3 4 0 2	1 4 3 0 2	2 0 1 4 3	2 0 1 3 4
2 4 1 3 0	2 3 1 4 0	4 0 3 1 2	4 0 2 1 3	3 0 2 1 4	4 3 2 0 1	3 4 2 0 1
3 0 2 4 1	4 0 2 3 1	3 1 2 4 0	2 1 3 4 0	2 1 4 3 0	0 1 4 3 2	0 1 3 4 2
4 1 3 0 2	3 1 4 0 2	2 4 0 3 1	3 4 0 2 1	4 3 0 2 1	3 2 0 1 4	4 2 0 1 3

73/	85/	97/	109/	121/	133/	145/
1 3 2 0 4	1 3 0 2 4	2 4 3 1 0	2 3 4 1 0	2 3 1 0 4	2 3 0 1 4	3 4 2 1 0
2 0 4 1 3	0 2 4 1 3	1 0 2 4 3	1 0 2 3 4	1 0 4 2 3	0 1 4 2 3	1 0 3 4 2
4 1 3 2 0	4 1 3 0 2	4 3 1 0 2	3 4 1 0 2	4 2 3 1 0	4 2 3 0 1	4 2 1 0 3
3 2 0 4 1	3 0 2 4 1	0 2 4 3 1	0 2 3 4 1	3 1 0 4 2	3 0 1 4 2	0 3 4 2 1
0 4 1 3 2	2 4 1 3 0	3 1 0 2 4	4 1 0 2 3	0 4 2 3 1	1 4 2 3 0	2 1 0 3 4

157/	169/	181/	193/	205/	217/	229/
3 2 4 1 0	3 2 1 0 4	3 2 0 1 4	4 3 2 1 0	4 2 3 1 0	4 2 1 0 3	4 2 0 1 3
1 0 3 2 4	1 0 4 3 2	0 1 4 3 2	1 0 4 3 2	1 0 4 2 3	1 0 3 4 2	0 1 3 4 2
2 4 1 0 3	4 3 2 1 0	4 3 2 0 1	3 2 1 0 4	2 3 1 0 4	3 4 2 1 0	3 4 2 0 1
0 3 2 4 1	2 1 0 4 3	2 0 1 4 3	0 4 3 2 1	0 4 2 3 1	2 1 0 3 4	2 0 1 3 4
4 1 0 3 2	0 4 3 2 1	1 4 3 2 0	2 1 0 4 3	3 1 0 4 2	0 3 4 2 1	1 3 4 2 0

［ラテン方陣の総数 ＝ 240個］

組み合わせて調べなければならない場合の数は，240×240＝57600個もある。標準解出力を考慮して，高位のユニットを半分に減らしたとしても，120×240個は一大事だ。

** 新オイラー法で構成した汎対角線型「完全オイラー五方陣」の標準解のリスト **

（ラテン・ユニット: 高位/H 低位/L; 解番号#; チェック・サム:|行|列\汎対1/汎対2|）

Lu: 1/H|Rw|Cl\P1/P2|	2/L|Rw|Cl\P1/P2|	1#|Rw|Cl\P1/P2|
0 4 3 2 1|10|10|10|10|	0 2 4 1 3|10|10|10|10|	1 23 20 12 9|65|65|65|65|
2 1 0 4 3|10|10|10|10|	4 1 3 0 2|10|10|10|10|	15 7 4 21 18|65|65|65|65|
4 3 2 1 0|10|10|10|10|	3 0 2 4 1|10|10|10|10|	24 16 13 10 2|65|65|65|65|
1 0 4 3 2|10|10|10|10|	2 4 1 3 0|10|10|10|10|	8 5 22 19 11|65|65|65|65|
3 2 1 0 4|10|10|10|10|	1 3 0 2 4|10|10|10|10|	17 14 6 3 25|65|65|65|65|

Lu: 1/H|Rw|Cl\P1/P2|	146/L|Rw|Cl\P1/P2|	73#|Rw|Cl\P1/P2|
0 4 3 2 1|10|10|10|10|	3 1 4 0 2|10|10|10|10|	4 22 20 11 8|65|65|65|65|
2 1 0 4 3|10|10|10|10|	4 0 2 3 1|10|10|10|10|	15 6 3 24 17|65|65|65|65|
4 3 2 1 0|10|10|10|10|	2 3 1 4 0|10|10|10|10|	23 19 12 10 1|65|65|65|65|
1 0 4 3 2|10|10|10|10|	1 4 0 2 3|10|10|10|10|	7 5 21 18 14|65|65|65|65|
3 2 1 0 4|10|10|10|10|	0 2 3 1 4|10|10|10|10|	16 13 9 2 25|65|65|65|65|

Lu: 3/H|Rw|Cl\P1/P2|	50/L|Rw|Cl\P1/P2|	145#|Rw|Cl\P1/P2|
0 4 2 3 1|10|10|10|10|	1 2 4 0 3|10|10|10|10|	2 23 15 16 9|65|65|65|65|
3 1 0 4 2|10|10|10|10|	4 0 3 1 2|10|10|10|10|	20 6 4 22 13|65|65|65|65|
4 2 3 1 0|10|10|10|10|	3 1 2 4 0|10|10|10|10|	24 12 18 10 1|65|65|65|65|
1 0 4 2 3|10|10|10|10|	2 4 0 3 1|10|10|10|10|	8 5 21 14 17|65|65|65|65|
2 3 1 0 4|10|10|10|10|	0 3 1 2 4|10|10|10|10|	11 19 7 3 25|65|65|65|65|

Lu: 3/H|Rw|Cl\P1/P2|	194/L|Rw|Cl\P1/P2|	217#|Rw|Cl\P1/P2|
0 4 2 3 1|10|10|10|10|	4 1 3 0 2|10|10|10|10|	5 22 14 16 8|65|65|65|65|
3 1 0 4 2|10|10|10|10|	3 0 2 4 1|10|10|10|10|	19 6 3 25 12|65|65|65|65|
4 2 3 1 0|10|10|10|10|	2 4 1 3 0|10|10|10|10|	23 15 17 9 1|65|65|65|65|
1 0 4 2 3|10|10|10|10|	1 3 0 2 4|10|10|10|10|	7 4 21 13 20|65|65|65|65|
2 3 1 0 4|10|10|10|10|	0 2 4 1 3|10|10|10|10|	11 18 10 2 24|65|65|65|65|

Lu: 5/H|Rw|Cl\P1/P2|	98/L|Rw|Cl\P1/P2|	289#|Rw|Cl\P1/P2|
0 4 3 1 2|10|10|10|10|	2 1 4 0 3|10|10|10|10|	3 22 20 6 14|65|65|65|65|
1 2 0 4 3|10|10|10|10|	4 0 3 2 1|10|10|10|10|	10 11 4 23 17|65|65|65|65|
4 3 1 2 0|10|10|10|10|	3 2 1 4 0|10|10|10|10|	24 18 7 15 1|65|65|65|65|
2 0 4 3 1|10|10|10|10|	1 4 0 3 2|10|10|10|10|	12 5 21 19 8|65|65|65|65|
3 1 2 0 4|10|10|10|10|	0 3 2 1 4|10|10|10|10|	16 9 13 2 25|65|65|65|65|

Lu: 10/H|Rw|Cl\P1/P2|	193/L|Rw|Cl\P1/P2|	577#|Rw|Cl\P1/P2|
0 1 4 3 2|10|10|10|10|	4 3 2 1 0|10|10|10|10|	5 9 23 17 11|65|65|65|65|
4 3 2 0 1|10|10|10|10|	1 0 4 3 2|10|10|10|10|	22 16 15 4 8|65|65|65|65|
2 0 1 4 3|10|10|10|10|	3 2 1 0 4|10|10|10|10|	14 3 7 21 20|65|65|65|65|
1 4 3 2 0|10|10|10|10|	0 4 3 2 1|10|10|10|10|	6 25 19 13 2|65|65|65|65|
3 2 0 1 4|10|10|10|10|	2 1 0 4 3|10|10|10|10|	18 12 1 10 24|65|65|65|65|

Lu: 16/H|Rw|Cl\P1/P2|	50/L|Rw|Cl\P1/P2|	865#|Rw|Cl\P1/P2|
0 3 2 4 1|10|10|10|10|	1 2 4 0 3|10|10|10|10|	2 18 15 21 9|65|65|65|65|
4 1 0 3 2|10|10|10|10|	4 0 3 1 2|10|10|10|10|	25 6 4 17 13|65|65|65|65|
3 2 4 1 0|10|10|10|10|	3 1 2 4 0|10|10|10|10|	19 12 23 10 1|65|65|65|65|
1 0 3 2 4|10|10|10|10|	2 4 0 3 1|10|10|10|10|	8 5 16 14 22|65|65|65|65|
2 4 1 0 3|10|10|10|10|	0 3 1 2 4|10|10|10|10|	11 24 7 3 20|65|65|65|65|

Lu: 19/H|Rw|Cl\P1/P2|	145/L|Rw|Cl\P1/P2|	1153#|Rw|Cl\P1/P2|
0 4 3 2 1|10|10|10|10|	3 4 2 1 0|10|10|10|10|	4 25 18 12 6|65|65|65|65|
3 2 1 0 4|10|10|10|10|	1 0 3 4 2|10|10|10|10|	17 11 9 5 23|65|65|65|65|
1 0 4 3 2|10|10|10|10|	4 2 1 0 3|10|10|10|10|	10 3 22 16 14|65|65|65|65|
4 3 2 1 0|10|10|10|10|	0 3 4 2 1|10|10|10|10|	21 19 15 8 2|65|65|65|65|
2 1 0 4 3|10|10|10|10|	2 1 0 3 4|10|10|10|10|	13 7 1 24 20|65|65|65|65|

```
 Lu:  26/H|Rw|Cl\P1/P2|        2/L|Rw|Cl\P1/P2|           1441#|Rw|Cl\P1/P2|
0 2 4 3 1|10|10|10|10|  0 2 4 1 3|10|10|10|10|   1 13 25 17  9|65|65|65|65|
3 1 0 2 4|10|10|10|10|  4 1 3 0 2|10|10|10|10|  20  7  4 11 23|65|65|65|65|
2 4 3 1 0|10|10|10|10|  3 0 2 4 1|10|10|10|10|  14 21 18 10  2|65|65|65|65|
1 0 2 4 3|10|10|10|10|  2 4 1 3 0|10|10|10|10|   8  5 12 24 16|65|65|65|65|
4 3 1 0 2|10|10|10|10|  1 3 0 2 4|10|10|10|10|  22 19  6  3 15|65|65|65|65|

 Lu:  29/H|Rw|Cl\P1/P2|        98/L|Rw|Cl\P1/P2|           1729#|Rw|Cl\P1/P2|
0 2 4 1 3|10|10|10|10|  2 1 4 0 3|10|10|10|10|   3 12 25  6 19|65|65|65|65|
1 3 0 2 4|10|10|10|10|  4 0 3 2 1|10|10|10|10|  10 16  4 13 22|65|65|65|65|
2 4 1 3 0|10|10|10|10|  3 2 1 4 0|10|10|10|10|  14 23  7 20  1|65|65|65|65|
3 0 2 4 1|10|10|10|10|  1 4 0 3 2|10|10|10|10|  17  5 11 24  8|65|65|65|65|
4 1 3 0 2|10|10|10|10|  0 3 2 1 4|10|10|10|10|  21  9 18  2 15|65|65|65|65|

 Lu:  34/H|Rw|Cl\P1/P2|       193/L|Rw|Cl\P1/P2|           2017#|Rw|Cl\P1/P2|
0 1 2 4 3|10|10|10|10|  4 3 2 1 0|10|10|10|10|   5  9 13 22 16|65|65|65|65|
2 4 3 0 1|10|10|10|10|  1 0 4 3 2|10|10|10|10|  12 21 20  4  8|65|65|65|65|
3 0 1 2 4|10|10|10|10|  3 2 1 0 4|10|10|10|10|  19  3  7 11 25|65|65|65|65|
1 2 4 3 0|10|10|10|10|  0 4 3 2 1|10|10|10|10|   6 15 24 18  2|65|65|65|65|
4 3 0 1 2|10|10|10|10|  2 1 0 4 3|10|10|10|10|  23 17  1 10 14|65|65|65|65|

 Lu:  40/H|Rw|Cl\P1/P2|        50/L|Rw|Cl\P1/P2|           2305#|Rw|Cl\P1/P2|
0 1 3 4 2|10|10|10|10|  1 2 4 0 3|10|10|10|10|   2  8 20 21 14|65|65|65|65|
4 2 0 1 3|10|10|10|10|  4 0 3 1 2|10|10|10|10|  25 11  4  7 18|65|65|65|65|
1 3 4 2 0|10|10|10|10|  3 1 2 4 0|10|10|10|10|   9 17 23 15  1|65|65|65|65|
2 0 1 3 4|10|10|10|10|  2 4 0 3 1|10|10|10|10|  13  5  6 19 22|65|65|65|65|
3 4 2 0 1|10|10|10|10|  0 3 1 2 4|10|10|10|10|  16 24 12  3 10|65|65|65|65|

 Lu:  43/H|Rw|Cl\P1/P2|       145/L|Rw|Cl\P1/P2|           2593#|Rw|Cl\P1/P2|
0 4 1 3 2|10|10|10|10|  3 4 2 1 0|10|10|10|10|   4 25  8 17 11|65|65|65|65|
1 3 2 0 4|10|10|10|10|  1 0 3 4 2|10|10|10|10|   7 16 14  5 23|65|65|65|65|
2 0 4 1 3|10|10|10|10|  4 2 1 0 3|10|10|10|10|  15  3 22  6 19|65|65|65|65|
4 1 3 2 0|10|10|10|10|  0 3 4 2 1|10|10|10|10|  21  9 20 13  2|65|65|65|65|
3 2 0 4 1|10|10|10|10|  2 1 0 3 4|10|10|10|10|  18 12  1 24 10|65|65|65|65|

 Lu:  53/H|Rw|Cl\P1/P2|        2/L|Rw|Cl\P1/P2|           2881#|Rw|Cl\P1/P2|
1 4 3 0 2|10|10|10|10|  0 2 4 1 3|10|10|10|10|   6 23 20  2 14|65|65|65|65|
0 2 1 4 3|10|10|10|10|  4 1 3 0 2|10|10|10|10|   5 12  9 21 18|65|65|65|65|
4 3 0 2 1|10|10|10|10|  3 0 2 4 1|10|10|10|10|  24 16  3 15  7|65|65|65|65|
2 1 4 3 0|10|10|10|10|  2 4 1 3 0|10|10|10|10|  13 10 22 19  1|65|65|65|65|
3 0 2 1 4|10|10|10|10|  1 3 0 2 4|10|10|10|10|  17  4 11  8 25|65|65|65|65|

 Lu:  65/H|Rw|Cl\P1/P2|        98/L|Rw|Cl\P1/P2|           3169#|Rw|Cl\P1/P2|
1 3 4 0 2|10|10|10|10|  2 1 4 0 3|10|10|10|10|   8 17 25  1 14|65|65|65|65|
0 2 1 3 4|10|10|10|10|  4 0 3 2 1|10|10|10|10|   5 11  9 18 22|65|65|65|65|
3 4 0 2 1|10|10|10|10|  3 2 1 4 0|10|10|10|10|  19 23  2 15  6|65|65|65|65|
2 1 3 4 0|10|10|10|10|  1 4 0 3 2|10|10|10|10|  12 10 16 24  3|65|65|65|65|
4 0 2 1 3|10|10|10|10|  0 3 2 1 4|10|10|10|10|  21  4 13  7 20|65|65|65|65|

 Lu:  77/H|Rw|Cl\P1/P2|       194/L|Rw|Cl\P1/P2|           3457#|Rw|Cl\P1/P2|
1 2 4 0 3|10|10|10|10|  4 1 3 0 2|10|10|10|10|  10 12 24  1 18|65|65|65|65|
0 3 1 2 4|10|10|10|10|  3 0 2 4 1|10|10|10|10|   4 16  8 15 22|65|65|65|65|
2 4 0 3 1|10|10|10|10|  2 4 1 3 0|10|10|10|10|  13 25  2 19  6|65|65|65|65|
3 1 2 4 0|10|10|10|10|  1 3 0 2 4|10|10|10|10|  17  9 11 23  5|65|65|65|65|
4 0 3 1 2|10|10|10|10|  0 2 4 1 3|10|10|10|10|  21  3 20  7 14|65|65|65|65|
 [構成解の総数 ＝ 3600個]  [OK!]
** Calculated and Listed by Kanji Setsuda
    on Apr. 9, 2012 with MacOSX 10.7.3 & Xcode 4.2.1 **
```

なお，上の出力結果は初期的なもので，解の順序を見やすい形に並べ替えていない。それ以前の原始的なリストだ。今は，解の総数が要点なので，このままとする。
とにかく 3600個の解が出て来た。様々な方法で作ったものと全く同じ解集合であった。
これで、全てが**完全オイラー型**であることが，あらためて実証された。
条件の緩い**オイラー型**で定義して求めても，結果は変わらない。論理学的に「同値」を示している。従って，汎五方陣には，**ノン・オイラー型**が存在しない。奇跡の解集合だ。

方法的に見ると，「桂馬とび構成法」も「新オイラー法」も，単位面を構成する以外は，ほとんど同じことをする。結果も全然変わらない。
つまり，両者は「完全オイラー方陣構成法」として，全く同じ働きをしている。どちらも正しい，と主張することができる。
ただし，「桂馬とび」の名人芸は，必ずしもどの次数のどのタイプの魔方陣にも適用できるとは限らない。今のところ，成功したのは，五次・七次など素数次方陣だけである。
その点，新オイラー法は，単位面の設計さえできれば，ほとんど何でも作れる。次数・タイプを選ばない汎用性がある。「普遍性がある」と言っても良いだろう。
それに高速だ。例えば，汎五方陣などは，普通に作ったら，時間ばかり食って大変な作業になるのだが，それが新オイラー法で作ると，あっという間に終わってしまう。
高次方陣にチェレンジする時に，強力なアシスタントとなること，請け合いだ。
お勧めするので，ぜひこの方法と技術を身につけて活用していただきたいと思う。

次に**連立型対称汎五方陣**にも適用して「新オイラー法」をテストした。「桂馬とび構成法」の時と全く同じ結果が得られたことを，以下に抜粋した形で報告する。

```
//
//** Make Latin Units for the Simultaneous MS55 **
//** Basic Form and Equations for Definitions **
//              .--.--.--.--.--.
//    2  3  4  5| 1| 2| 3| 4| 5| 1  2  3  4
//              |--+--+--+--+--|
//    7  8  9 10| 6| 7| 8| 9|10| 6  7  8  9
//              |--+--+--+--+--|
//   12 13 14 15|11|12|13|14|15|11 12 13 14
//              |--+--+--+--+--|
//   17 18 19 20|16|17|18|19|20|16 17 18 19
//              |--+--+--+--+--|
//   22 23 24 25|21|22|23|24|25|21 22 23 24
//              '--'--'--'--'--'
//** Define Conditions for Rows and Columns: C=10 **
//  n1+n2+n3+n4+n5=C      ... rw1  | n1+n6+n11+n16+n21=C    ... cl1
//  n6+n7+n8+n9+n10=C     ... rw2  | n2+n7+n12+n17+n22=C    ... cl2
//  n11+n12+n13+n14+n15=C ... rw3  | n3+n8+n13+n18+n23=C    ... cl3
//  n16+n17+n18+n19+n20=C ... rw4  | n4+n9+n14+n19+n24=C    ... cl4
//  n21+n22+n23+n24+n25=C ... rw5  | n5+n10+n15+n20+n25=C   ... cl5
//** Pan-diagonal Conditions of the Object: C=10 **
//  n1+n7+n13+n19+n25=C   ... pd1  | n1+n10+n14+n18+n22=C   ... pd6
//  n2+n8+n14+n20+n21=C   ... pd2  | n2+n6+n15+n19+n23=C    ... pd7
//  n3+n9+n15+n16+n22=C   ... pd3  | n3+n7+n11+n20+n24=C    ... pd8
//  n4+n10+n11+n17+n23=C  ... pd4  | n4+n8+n12+n16+n25=C    ... pd9
//  n5+n6+n12+n18+n24=C   ... pd5  | n5+n9+n13+n17+n21=C    ... pd10
//** Self-complementary Conditions of the Object: SC=4 **
//  n1+n25=n2+n24=n3+n23=n4+n22=n5+n21=n6+n20=n7+n19=
//  n8+n18=n9+n17=n10+n16=n11+n15=n12+n14=n13+n13=SC   ... scc
//
// . . . . .
// Level #0:
// Set n13{3,3,1,5}
void lstp00(){
 short a;
  a=2; nm[13]=a;
  uflg[a]++; rw3[a]=1; cl3[a]=1; pd1[a]=1; pb5[a]=1;
```

```
  lstp01();
  rw3[a]=0; cl3[a]=0; pd1[a]=0; pb5[a]=0; uflg[a]--;
}
// Level #1:
// Set n1{1,1,1,1} and n25{5,5,1,4}
void lstp01(){
 short a,b;
  for(a=0;a<5;a++){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw1[a]==0)&&(cl1[a]==0)&&(pd1[a]==0)&&(pb1[a]==0)){
        if((rw5[b]==0)&&(cl5[b]==0)&&(pd1[b]==0)&&(pb4[b]==0)){
          nm[1]=a; nm[25]=b;
          uflg[a]++; rw1[a]=1; cl1[a]=1; pd1[a]=1; pb1[a]=1;
          uflg[b]++; rw5[b]=1; cl5[b]=1; pd1[b]=1; pb4[b]=1;
          lstp02();
          rw5[b]=0; cl5[b]=0; pd1[b]=0; pb4[b]=0; uflg[b]--;
          rw1[a]=0; cl1[a]=0; pd1[a]=0; pb1[a]=0; uflg[a]--;}}}
  }
}
// Set n7{2,2,1,3} and n19{4,4,1,2}
void lstp02(){
 short a,b;
  for(a=0;a<5;a++){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw2[a]==0)&&(cl2[a]==0)&&(pd1[a]==0)&&(pb3[a]==0)){
        if((rw4[b]==0)&&(cl4[b]==0)&&(pd1[b]==0)&&(pb2[b]==0)){
          nm[7]=a; nm[19]=b;
          uflg[a]++; rw2[a]=1; cl2[a]=1; pd1[a]=1; pb3[a]=1;
          uflg[b]++; rw4[b]=1; cl4[b]=1; pd1[b]=1; pb2[b]=1;
          lstp03();
          rw4[b]=0; cl4[b]=0; pd1[b]=0; pb2[b]=0; uflg[b]--;
          rw2[a]=0; cl2[a]=0; pd1[a]=0; pb3[a]=0; uflg[a]--;}}}
  }
}
// Level #2:
// Set n2{1,2,2,2} and n24{5,4,5,3}
void lstp03(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw1[a]==0)&&(cl2[a]==0)&&(pd2[a]==0)&&(pb2[a]==0)){
        if((rw5[b]==0)&&(cl4[b]==0)&&(pd5[b]==0)&&(pb3[b]==0)){
          nm[2]=a; nm[24]=b;
          uflg[a]++; rw1[a]=1; cl2[a]=1; pd2[a]=1; pb2[a]=1;
          uflg[b]++; rw5[b]=1; cl4[b]=1; pd5[b]=1; pb3[b]=1;
          lstp04();
          rw5[b]=0; cl4[b]=0; pd5[b]=0; pb3[b]=0; uflg[b]--;
          rw1[a]=0; cl2[a]=0; pd2[a]=0; pb2[a]=0; uflg[a]--;}}}
  }
}
// Set n3{1,3,3,3} and n23{5,3,4,2}
void lstp04(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw1[a]==0)&&(cl3[a]==0)&&(pd3[a]==0)&&(pb3[a]==0)){
        if((rw5[b]==0)&&(cl3[b]==0)&&(pd4[b]==0)&&(pb2[b]==0)){
          nm[3]=a; nm[23]=b;
          uflg[a]++; rw1[a]=1; cl3[a]=1; pd3[a]=1; pb3[a]=1;
          uflg[b]++; rw5[b]=1; cl3[b]=1; pd4[b]=1; pb2[b]=1;
          lstp05();
          rw5[b]=0; cl3[b]=0; pd4[b]=0; pb2[b]=0; uflg[b]--;
          rw1[a]=0; cl3[a]=0; pd3[a]=0; pb3[a]=0; uflg[a]--;}}}
  }
}
```

```
// Set n4{1,4,4,4} and n22{5,2,3,1}
void lstp05(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw1[a]==0)&&(cl4[a]==0)&&(pd4[a]==0)&&(pb4[a]==0)){
        if((rw5[b]==0)&&(cl2[b]==0)&&(pd3[b]==0)&&(pb1[b]==0)){
          nm[4]=a; nm[22]=b;
          uflg[a]++; rw1[a]=1; cl4[a]=1; pd4[a]=1; pb4[a]=1;
          uflg[b]++; rw5[b]=1; cl2[b]=1; pd3[b]=1; pb1[b]=1;
          lstp06();
          rw5[b]=0; cl2[b]=0; pd3[b]=0; pb1[b]=0; uflg[b]--;
          rw1[a]=0; cl4[a]=0; pd4[a]=0; pb4[a]=0; uflg[a]--;}}}
  }
}
// Set n5{1,5,5,5} and n21{5,1,2,5}
void lstp06(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw1[a]==0)&&(cl5[a]==0)&&(pd5[a]==0)&&(pb5[a]==0)){
        if((rw5[b]==0)&&(cl1[b]==0)&&(pd2[b]==0)&&(pb5[b]==0)){
          nm[5]=a; nm[21]=b;
          uflg[a]++; rw1[a]=1; cl5[a]=1; pd5[a]=1; pb5[a]=1;
          uflg[b]++; rw5[b]=1; cl1[b]=1; pd2[b]=1; pb5[b]=1;
          lstp07();
          rw5[b]=0; cl1[b]=0; pd2[b]=0; pb5[b]=0; uflg[b]--;
          rw1[a]=0; cl5[a]=0; pd5[a]=0; pb5[a]=0; uflg[a]--;}}}
  }
}
// Level #3:
// Set n11{3,1,4,3} and n15{3,5,3,2}
void lstp07(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw3[a]==0)&&(cl1[a]==0)&&(pd4[a]==0)&&(pb3[a]==0)){
        if((rw3[b]==0)&&(cl5[b]==0)&&(pd3[b]==0)&&(pb2[b]==0)){
          uflg[a]++; nm[11]=a; nm[15]=b;
          uflg[b]++; rw3[a]=1; cl1[a]=1; pd4[a]=1; pb3[a]=1;
          rw3[b]=1; cl5[b]=1; pd3[b]=1; pb2[b]=1;
          lstp08();
          rw3[b]=0; cl5[b]=0; pd3[b]=0; pb2[b]=0; uflg[b]--;
          rw3[a]=0; cl1[a]=0; pd4[a]=0; pb3[a]=0; uflg[a]--;}}}
  }
}
// Set n16{4,1,3,4} and n10{2,5,4,1}
void lstp08(){
 short a,b;
  for(a=0;a<5;a++){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw4[a]==0)&&(cl1[a]==0)&&(pd3[a]==0)&&(pb4[a]==0)){
        if((rw2[b]==0)&&(cl5[b]==0)&&(pd4[b]==0)&&(pb1[b]==0)){
          nm[16]=a; nm[10]=b;
          uflg[a]++; rw4[a]=1; cl1[a]=1; pd3[a]=1; pb4[a]=1;
          uflg[b]++; rw2[b]=1; cl5[b]=1; pd4[b]=1; pb1[b]=1;
          lstp09();
          rw2[b]=0; cl5[b]=0; pd4[b]=0; pb1[b]=0; uflg[b]--;
          rw4[a]=0; cl1[a]=0; pd3[a]=0; pb4[a]=0; uflg[a]--;}}}
  }
}
// Set n6{2,1,5,2} and n20{4,5,2,3}
void lstp09(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
```

```
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw2[a]==0)&&(cl1[a]==0)&&(pd5[a]==0)&&(pb2[a]==0)){
        if((rw4[b]==0)&&(cl5[b]==0)&&(pd2[b]==0)&&(pb3[b]==0)){
          nm[6]=a; nm[20]=b;
          uflg[a]++; rw2[a]=1; cl1[a]=1; pd5[a]=1; pb2[a]=1;
          uflg[b]++; rw4[b]=1; cl5[b]=1; pd2[b]=1; pb3[b]=1;
          lstp10();
          rw4[b]=0; cl5[b]=0; pd2[b]=0; pb3[b]=0; uflg[b]--;
          rw2[a]=0; cl1[a]=0; pd5[a]=0; pb2[a]=0; uflg[a]--;}}}
  }
}
// Level #4:
// Set n8{2,3,2,4} and n18{4,3,5,1}
void lstp10(){
 short a,b;
  for(a=0;a<5;a++){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw2[a]==0)&&(cl3[a]==0)&&(pd2[a]==0)&&(pb4[a]==0)){
        if((rw4[b]==0)&&(cl3[b]==0)&&(pd5[b]==0)&&(pb1[b]==0)){
          nm[8]=a; nm[18]=b;
          uflg[a]++; rw2[a]=1; cl3[a]=1; pd2[a]=1; pb4[a]=1;
          uflg[b]++; rw4[b]=1; cl3[b]=1; pd5[b]=1; pb1[b]=1;
          lstp11();
          rw4[b]=0; cl3[b]=0; pd5[b]=0; pb1[b]=0; uflg[b]--;
          rw2[a]=0; cl3[a]=0; pd2[a]=0; pb4[a]=0; uflg[a]--;}}}
  }
}
// Set n9{2,4,3,5} and n17{4,2,4,5}
void lstp11(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw2[a]==0)&&(cl4[a]==0)&&(pd3[a]==0)&&(pb5[a]==0)){
        if((rw4[b]==0)&&(cl2[b]==0)&&(pd4[b]==0)&&(pb5[b]==0)){
          nm[9]=a; nm[17]=b;
          uflg[a]++; rw2[a]=1; cl4[a]=1; pd3[a]=1; pb5[a]=1;
          uflg[b]++; rw4[b]=1; cl2[b]=1; pd4[b]=1; pb5[b]=1;
          lstp12();
          rw4[b]=0; cl2[b]=0; pd4[b]=0; pb5[b]=0; uflg[b]--;
          rw2[a]=0; cl4[a]=0; pd3[a]=0; pb5[a]=0; uflg[a]--;}}}
  }
}
// Set n12{3,2,5,4} and n14{3,4,2,1}
void lstp12(){
 short a,b;
  for(a=4;a>=0;a--){b=SC-a;
    if((uflg[a]<5)&&(uflg[b]<5)){
      if((rw3[a]==0)&&(cl2[a]==0)&&(pd5[a]==0)&&(pb4[a]==0)){
        if((rw3[b]==0)&&(cl4[b]==0)&&(pd2[b]==0)&&(pb1[b]==0)){
          nm[12]=a; nm[14]=b;
          uflg[a]++; rw3[a]=1; cl2[a]=1; pd5[a]=1; pb4[a]=1;
          uflg[b]++; rw3[b]=1; cl4[b]=1; pd2[b]=1; pb1[b]=1;
          lurecord();
          rw3[b]=0; cl4[b]=0; pd2[b]=0; pb1[b]=0; uflg[b]--;
          rw3[a]=0; cl2[a]=0; pd5[a]=0; pb4[a]=0; uflg[a]--;}}}
  }
}
//
// Record The Latin Units
void lurecord(){
 short n;
  tlu[cnt][0]=cnt+1;
  for(n=1;n<26;n++){tlu[cnt][n]=nm[n];}
  cnt++;
}
```

```
//
// Print Latin Units
void prlunit(short x){
 short t,l,l5,m,n,p;
  for(t=0;t<lcnt;t=t+x){
    p=x; if(t+x>lcnt){p=lcnt-t;}
    for(m=0;m<p;m++){printf("  %9d/",tlu[t+m][0]);}
    printf("\n");
    for(l=0;l<5;l++){l5=l*5;
      for(m=0;m<p;m++){
        printf("  ");
        for(n=1;n<6;n++){printf("%2d",tlu[t+m][l5+n]);}
      }
      printf("\n");
    }
  }
  printf(" [Count of Latin Units = %d]\n",lcnt);
 //* Measure How Similar or How Complementary each pair is *
  for(m=0;m<cnt;m++){
    for(n=0;n<cnt;n++){
      t=0;
      for(l=1;l<26;l++){if(tlu[m][l]==tlu[n][l]){t++;}}
      mtc[m][n]=t;
      t=0;
      for(l=1;l<26;l++){if(tlu[m][l]+tlu[n][l]==4){t++;}}
      if(mtc[m][n]<t){mtc[m][n]=t;}
    }
  }
  printf("\n");
  printf(" [Reference Table for the Best Combination]\n");
  printf("  0|");
  for(n=0;n<16;n++){printf("%3d",n+1);}
  printf("\n");
  printf(" --+----------------------------------------------\n");
  for(m=0;m<16;m++){
    printf("%3d|",m+1);
    for(n=0;n<16;n++){printf("%3d",mtc[m][n]);}
    printf("\n");
  }
}
//
//* Combine and Compose *
void cmbcmp(){
 short n,d,fc;
  for(u1=0;u1<(lcnt/2);u1++){
    for(u2=0;u2<lcnt;u2++){
      if(mtc[u1][u2]==5){
        for(n=1;n<26;n++){uflg[n]=0;}
        for(n=1;n<26;n++){
          d=tlu[u1][n]*5+tlu[u2][n]+1;
          nm[n]=d; uflg[d]++;
        }
        fc=0;
        for(n=1;n<26;n++){if(uflg[n]==0){fc++;}}
        if(fc==0){
          if((nm[1]<nm[5])&&(nm[5]<nm[21])&&(nm[1]<nm[25])){solsavep();}}
      }
    }
  }
}
//
// Save Answers to the Solution List and Print
void solsavep(){
 short n;
  cnt++;
```

```
  anm[0][0]=cnt;
  for(n=1;n<26;n++){anm[0][n]=nm[n]; tn[n]=nm[n];}
  chksms(0);
  for(n=1;n<26;n++){anm[2][n]=tlu[u1][n]; tn[n]=tlu[u1][n];}
  chksms(2); anm[2][26]=u1;
  for(n=1;n<26;n++){anm[3][n]=tlu[u2][n]; tn[n]=tlu[u2][n];}
  chksms(3); anm[3][26]=u2;
  pr1ans();
}
// . . . . .
//
```

** 五進法を用いた『新オイラー法』によって再構成した連立型の対称汎五方陣 ***

[ラテン方陣のリスト]

```
      1/        2/        3/        4/        5/        6/        7/        8/
0 4 3 2 1 | 0 2 4 1 3 | 0 4 1 2 3 | 0 2 4 3 1 | 1 3 4 2 0 | 1 2 3 0 4 | 1 3 0 2 4 | 1 2 3 4 0
2 1 0 4 3 | 4 1 3 0 2 | 2 3 0 4 1 | 4 3 1 0 2 | 2 0 1 3 4 | 3 0 4 1 2 | 2 4 1 3 0 | 3 4 0 1 2
4 3 2 1 0 | 3 0 2 4 1 | 4 1 2 3 0 | 1 0 2 4 3 | 3 4 2 0 1 | 4 1 2 3 0 | 3 0 2 4 1 | 0 1 2 3 4
1 0 4 3 2 | 2 4 1 3 0 | 3 0 4 1 2 | 2 4 3 1 0 | 0 1 3 4 2 | 2 3 0 4 1 | 4 1 3 0 2 | 2 3 4 0 1
3 2 1 0 4 | 1 3 0 2 4 | 1 2 3 0 4 | 3 1 0 2 4 | 4 2 0 1 3 | 0 4 1 2 3 | 0 2 4 1 3 | 4 0 1 2 3

      9/       10/       11/       12/       13/       14/       15/       16/
3 2 1 0 4 | 3 1 4 2 0 | 3 2 1 4 0 | 3 1 0 2 4 | 4 2 0 1 3 | 4 0 3 2 1 | 4 2 0 3 1 | 4 0 1 2 3
1 0 4 3 2 | 2 0 3 1 4 | 1 4 0 3 2 | 2 4 3 1 0 | 0 1 3 4 2 | 2 1 4 0 3 | 0 3 1 4 2 | 2 3 4 0 1
4 3 2 1 0 | 1 4 2 0 3 | 0 3 2 1 4 | 1 0 2 4 3 | 3 4 2 0 1 | 0 3 2 1 4 | 1 4 2 0 3 | 0 1 2 3 4
2 1 0 4 3 | 0 3 1 4 2 | 2 1 4 0 3 | 4 3 1 0 2 | 2 0 1 3 4 | 1 4 0 3 2 | 2 0 3 1 4 | 3 4 0 1 2
0 4 3 2 1 | 4 2 0 3 1 | 4 0 3 2 1 | 0 2 4 3 1 | 1 3 4 2 0 | 3 2 1 4 0 | 3 1 4 2 0 | 1 2 3 4 0
```

[ラテン方陣の総数 = 16]

[完全オイラー五方陣として再構成した連立型対称汎五方陣の標準解のリスト]

[ラテン・ユニット: 高位H 低位L; [解番号]; チェック・サム:|行|列\汎対1/汎対2|]

```
/LU     1H|Rw|Cl\P1/P2|        2L|Rw|Cl\P1/P2|  [ 1]          |Rw|Cl\P1/P2|
 0 4 3 2 1|10|10|10|10|  0 2 4 1 3|10|10|10|10|   1 23 20 12  9|65|65|65|65|
 2 1 0 4 3|10|10|10|10|  4 1 3 0 2|10|10|10|10|  15  7  4 21 18|65|65|65|65|
 4 3 2 1 0|10|10|10|10|  3 0 2 4 1|10|10|10|10|  24 16 13 10  2|65|65|65|65|
 1 0 4 3 2|10|10|10|10|  2 4 1 3 0|10|10|10|10|   8  5 22 19 11|65|65|65|65|
 3 2 1 0 4|10|10|10|10|  1 3 0 2 4|10|10|10|10|  17 14  6  3 25|65|65|65|65|

/LU     1H|Rw|Cl\P1/P2|        4L|Rw|Cl\P1/P2|  [ 2]          |Rw|Cl\P1/P2|
 0 4 3 2 1|10|10|10|10|  0 2 4 3 1|10|10|10|10|   1 23 20 14  7|65|65|65|65|
 2 1 0 4 3|10|10|10|10|  4 3 1 0 2|10|10|10|10|  15  9  2 21 18|65|65|65|65|
 4 3 2 1 0|10|10|10|10|  1 0 2 4 3|10|10|10|10|  22 16 13 10  4|65|65|65|65|
 1 0 4 3 2|10|10|10|10|  2 4 3 1 0|10|10|10|10|   8  5 24 17 11|65|65|65|65|
 3 2 1 0 4|10|10|10|10|  3 1 0 2 4|10|10|10|10|  19 12  6  3 25|65|65|65|65|

/LU     1H|Rw|Cl\P1/P2|        6L|Rw|Cl\P1/P2|  [ 3]          |Rw|Cl\P1/P2|
 0 4 3 2 1|10|10|10|10|  1 2 3 0 4|10|10|10|10|   2 23 19 11 10|65|65|65|65|
 2 1 0 4 3|10|10|10|10|  3 0 4 1 2|10|10|10|10|  14  6  5 22 18|65|65|65|65|
 4 3 2 1 0|10|10|10|10|  4 1 2 3 0|10|10|10|10|  25 17 13  9  1|65|65|65|65|
 1 0 4 3 2|10|10|10|10|  2 3 0 4 1|10|10|10|10|   8  4 21 20 12|65|65|65|65|
 3 2 1 0 4|10|10|10|10|  0 4 1 2 3|10|10|10|10|  16 15  7  3 24|65|65|65|65|

/LU     1H|Rw|Cl\P1/P2|        8L|Rw|Cl\P1/P2|  [ 4]          |Rw|Cl\P1/P2|
 0 4 3 2 1|10|10|10|10|  1 2 3 4 0|10|10|10|10|   2 23 19 15  6|65|65|65|65|
 2 1 0 4 3|10|10|10|10|  3 4 0 1 2|10|10|10|10|  14 10  1 22 18|65|65|65|65|
 4 3 2 1 0|10|10|10|10|  0 1 2 3 4|10|10|10|10|  21 17 13  9  5|65|65|65|65|
 1 0 4 3 2|10|10|10|10|  2 3 4 0 1|10|10|10|10|   8  4 25 16 12|65|65|65|65|
 3 2 1 0 4|10|10|10|10|  4 0 1 2 3|10|10|10|10|  20 11  7  3 24|65|65|65|65|

/LU     1H|Rw|Cl\P1/P2|        9L|Rw|Cl\P1/P2|  [ 5]          |Rw|Cl\P1/P2|
 0 4 3 2 1|10|10|10|10|  3 2 1 0 4|10|10|10|10|   4 23 17 11 10|65|65|65|65|
 2 1 0 4 3|10|10|10|10|  1 0 4 3 2|10|10|10|10|  12  6  5 24 18|65|65|65|65|
 4 3 2 1 0|10|10|10|10|  4 3 2 1 0|10|10|10|10|  25 19 13  7  1|65|65|65|65|
 1 0 4 3 2|10|10|10|10|  2 1 0 4 3|10|10|10|10|   8  2 21 20 14|65|65|65|65|
 3 2 1 0 4|10|10|10|10|  0 4 3 2 1|10|10|10|10|  16 15  9  3 22|65|65|65|65|
```

/LU 1H|Rw|Cl\P1/P2|										11L|Rw|Cl\P1/P2|										[6] |Rw|Cl\P1/P2|									
0	4	3	2	1	10	10	10	10		3	2	1	4	0	10	10	10	10		4	23	17	15	6	65	65	65	65	
2	1	0	4	3	10	10	10	10		1	4	0	3	2	10	10	10	10		12	10	1	24	18	65	65	65	65	
4	3	2	1	0	10	10	10	10		0	3	2	1	4	10	10	10	10		21	19	13	7	5	65	65	65	65	
1	0	4	3	2	10	10	10	10		2	1	4	0	3	10	10	10	10		8	2	25	16	14	65	65	65	65	
3	2	1	0	4	10	10	10	10		4	0	3	2	1	10	10	10	10		20	11	9	3	22	65	65	65	65	

/LU 1H|Rw|Cl\P1/P2|										13L|Rw|Cl\P1/P2|										[7] |Rw|Cl\P1/P2|									
0	4	3	2	1	10	10	10	10		4	2	0	1	3	10	10	10	10		5	23	16	12	9	65	65	65	65	
2	1	0	4	3	10	10	10	10		0	1	3	4	2	10	10	10	10		11	7	4	25	18	65	65	65	65	
4	3	2	1	0	10	10	10	10		3	4	2	0	1	10	10	10	10		24	20	13	6	2	65	65	65	65	
1	0	4	3	2	10	10	10	10		2	0	1	3	4	10	10	10	10		8	1	22	19	15	65	65	65	65	
3	2	1	0	4	10	10	10	10		1	3	4	2	0	10	10	10	10		17	14	10	3	21	65	65	65	65	

/LU 1H|Rw|Cl\P1/P2|										15L|Rw|Cl\P1/P2|										[8] |Rw|Cl\P1/P2|									
0	4	3	2	1	10	10	10	10		4	2	0	3	1	10	10	10	10		5	23	16	14	7	65	65	65	65	
2	1	0	4	3	10	10	10	10		0	3	1	4	2	10	10	10	10		11	9	2	25	18	65	65	65	65	
4	3	2	1	0	10	10	10	10		1	4	2	0	3	10	10	10	10		22	20	13	6	4	65	65	65	65	
1	0	4	3	2	10	10	10	10		2	0	3	1	4	10	10	10	10		8	1	24	17	15	65	65	65	65	
3	2	1	0	4	10	10	10	10		3	1	4	2	0	10	10	10	10		19	12	10	3	21	65	65	65	65	

/LU 4H|Rw|Cl\P1/P2|										1L|Rw|Cl\P1/P2|										[9] |Rw|Cl\P1/P2|									
0	2	4	3	1	10	10	10	10		0	4	3	2	1	10	10	10	10		1	15	24	18	7	65	65	65	65	
4	3	1	0	2	10	10	10	10		2	1	0	4	3	10	10	10	10		23	17	6	5	14	65	65	65	65	
1	0	2	4	3	10	10	10	10		4	3	2	1	0	10	10	10	10		10	4	13	22	16	65	65	65	65	
2	4	3	1	0	10	10	10	10		1	0	4	3	2	10	10	10	10		12	21	20	9	3	65	65	65	65	
3	1	0	2	4	10	10	10	10		3	2	1	0	4	10	10	10	10		19	8	2	11	25	65	65	65	65	

/LU 4H|Rw|Cl\P1/P2|										3L|Rw|Cl\P1/P2|										[10] |Rw|Cl\P1/P2|									
0	2	4	3	1	10	10	10	10		0	4	1	2	3	10	10	10	10		1	15	22	18	9	65	65	65	65	
4	3	1	0	2	10	10	10	10		2	3	0	4	1	10	10	10	10		23	19	6	5	12	65	65	65	65	
1	0	2	4	3	10	10	10	10		4	1	2	3	0	10	10	10	10		10	2	13	24	16	65	65	65	65	
2	4	3	1	0	10	10	10	10		3	0	4	1	2	10	10	10	10		14	21	20	7	3	65	65	65	65	
3	1	0	2	4	10	10	10	10		1	2	3	0	4	10	10	10	10		17	8	4	11	25	65	65	65	65	

/LU 4H|Rw|Cl\P1/P2|										5L|Rw|Cl\P1/P2|										[11] |Rw|Cl\P1/P2|									
0	2	4	3	1	10	10	10	10		1	3	4	2	0	10	10	10	10		2	14	25	18	6	65	65	65	65	
4	3	1	0	2	10	10	10	10		2	0	1	3	4	10	10	10	10		23	16	7	4	15	65	65	65	65	
1	0	2	4	3	10	10	10	10		3	4	2	0	1	10	10	10	10		9	5	13	21	17	65	65	65	65	
2	4	3	1	0	10	10	10	10		0	1	3	4	2	10	10	10	10		11	22	19	10	3	65	65	65	65	
3	1	0	2	4	10	10	10	10		4	2	0	1	3	10	10	10	10		20	8	1	12	24	65	65	65	65	

/LU 4H|Rw|Cl\P1/P2|										7L|Rw|Cl\P1/P2|										[12] |Rw|Cl\P1/P2|									
0	2	4	3	1	10	10	10	10		1	3	0	2	4	10	10	10	10		2	14	21	18	10	65	65	65	65	
4	3	1	0	2	10	10	10	10		2	4	1	3	0	10	10	10	10		23	20	7	4	11	65	65	65	65	
1	0	2	4	3	10	10	10	10		3	0	2	4	1	10	10	10	10		9	1	13	25	17	65	65	65	65	
2	4	3	1	0	10	10	10	10		4	1	3	0	2	10	10	10	10		15	22	19	6	3	65	65	65	65	
3	1	0	2	4	10	10	10	10		0	2	4	1	3	10	10	10	10		16	8	5	12	24	65	65	65	65	

/LU 4H|Rw|Cl\P1/P2|										10L|Rw|Cl\P1/P2|										[13] |Rw|Cl\P1/P2|									
0	2	4	3	1	10	10	10	10		3	1	4	2	0	10	10	10	10		4	12	25	18	6	65	65	65	65	
4	3	1	0	2	10	10	10	10		2	0	3	1	4	10	10	10	10		23	16	9	2	15	65	65	65	65	
1	0	2	4	3	10	10	10	10		1	4	2	0	3	10	10	10	10		7	5	13	21	19	65	65	65	65	
2	4	3	1	0	10	10	10	10		0	3	1	4	2	10	10	10	10		11	24	17	10	3	65	65	65	65	
3	1	0	2	4	10	10	10	10		4	2	0	3	1	10	10	10	10		20	8	1	14	22	65	65	65	65	

/LU 4H|Rw|Cl\P1/P2|										12L|Rw|Cl\P1/P2|										[14] |Rw|Cl\P1/P2|									
0	2	4	3	1	10	10	10	10		3	1	0	2	4	10	10	10	10		4	12	21	18	10	65	65	65	65	
4	3	1	0	2	10	10	10	10		2	4	3	1	0	10	10	10	10		23	20	9	2	11	65	65	65	65	
1	0	2	4	3	10	10	10	10		1	0	2	4	3	10	10	10	10		7	1	13	25	19	65	65	65	65	
2	4	3	1	0	10	10	10	10		4	3	1	0	2	10	10	10	10		15	24	17	6	3	65	65	65	65	
3	1	0	2	4	10	10	10	10		0	2	4	3	1	10	10	10	10		16	8	5	14	22	65	65	65	65	

/LU 4H|Rw|Cl\P1/P2|										14L|Rw|Cl\P1/P2|										[15] |Rw|Cl\P1/P2|									
0	2	4	3	1	10	10	10	10		4	0	3	2	1	10	10	10	10		5	11	24	18	7	65	65	65	65	
4	3	1	0	2	10	10	10	10		2	1	4	0	3	10	10	10	10		23	17	10	1	14	65	65	65	65	
1	0	2	4	3	10	10	10	10		0	3	2	1	4	10	10	10	10		6	4	13	22	20	65	65	65	65	
2	4	3	1	0	10	10	10	10		1	4	0	3	2	10	10	10	10		12	25	16	9	3	65	65	65	65	
3	1	0	2	4	10	10	10	10		3	2	1	4	0	10	10	10	10		19	8	2	15	21	65	65	65	65	

```
/LU      4H|Rw|Cl\P1/P2|        16L|Rw|Cl\P1/P2|  [16]          |Rw|Cl\P1/P2|
 0 2 4 3 1|10|10|10|10|  4 0 1 2 3|10|10|10|10|   5 11 22 18  9|65|65|65|65|
 4 3 1 0 2|10|10|10|10|  2 3 4 0 1|10|10|10|10|  23 19 10  1 12|65|65|65|65|
 1 0 2 4 3|10|10|10|10|  0 1 2 3 4|10|10|10|10|   6  2 13 24 20|65|65|65|65|
 2 4 3 1 0|10|10|10|10|  3 4 0 1 2|10|10|10|10|  14 25 16  7  3|65|65|65|65|
 3 1 0 2 4|10|10|10|10|  1 2 3 4 0|10|10|10|10|  17  8  4 15 21|65|65|65|65|
[解の総数＝16個]  [OK!]
** Calculated and Listed by Kanji Setsuda on Apr. 8, 2012 with MacOSX 10.7.3 & Xcode 4.2.1 **
```

この型の方陣は，どんな方法を用いて作っても，同じ解集合しか得られない。
つまり，全部が「完全オイラー型」であって，「ノン・オイラー型」が無い。
こういう奇跡の解集合を最初に発見・確定したのは，当時東北大教授だった鈴木睦氏だ。
従って，氏の功績を記念するために，汎五方陣と併せて２つを『鈴木方陣』と呼ぶ。
なお付記すれば，同じ五方陣でも補数対称型となると，だいぶ様子が違う。標準解の総数が 48544個。**完全オイラー型**はその内の 32個しかない。最近**オイラー型**が 7136個あることがわかった。従って，ノン・オイラー型も多数ある。第11-3節をお読みいただきたい。

４．七進法による新オイラー法を用いて対称汎七方陣を構成する

今度は，連立型の対称汎七方陣を作ろう。七進法による**新オイラー法**を用い，**ギリシャ・ラテン方陣**構成法を使用して，**完全オイラー七方陣**の解が幾つあるのかを確かめよう。

** 七進法による新オイラー法の概念図 **

[七進法による分解図]

```
[古典的表記]      *#   [数学的表記]          /D7i      */H|Rw|Cl\P1/P2|          **/L|Rw|Cl\P1/P2|
 1 28 48 19 39 10 30    0 27 47 18 38  9 29   0 3 6 2 5 1 4|21|21|21|21|  0 6 5 4 3 2 1|21|21|21|21|
38  9 29  7 27 47 18   37  8 28  6 26 46 17   5 1 4 0 3 6 2|21|21|21|21|  2 1 0 6 5 4 3|21|21|21|21|
26 46 17 37  8 35  6   25 45 16 36  7 34  5   3 6 2 5 1 4 0|21|21|21|21|  4 3 2 1 0 6 5|21|21|21|21|
14 34  5 25 45 16 36   13 33  4 24 44 15 35   1 4 0 3 6 2 5|21|21|21|21|  6 5 4 3 2 1 0|21|21|21|21|
44 15 42 13 33  4 24   43 14 41 12 32  3 23   6 2 5 1 4 0 3|21|21|21|21|  1 0 6 5 4 3 2|21|21|21|21|
32  3 23 43 21 41 12   31  2 22 42 20 40 11   4 0 3 6 2 5 1|21|21|21|21|  3 2 1 0 6 5 4|21|21|21|21|
20 40 11 31  2 22 49   19 39 10 30  1 21 48   2 5 1 4 0 3 6|21|21|21|21|  5 4 3 2 1 0 6|21|21|21|21|
```

（[古典的表記]－１＝[数学的表記]；７で割った時の整商を分解図の高位に，剰余を低位におさめる）

[七進法による構成図]

```
/Lu:      */H          **/L   [数学的表記]        |Row|Clm\Pd1/Pd2|  [古典的表記]      *#
 0 3 6 2 5 1 4   0 6 5 4 3 2 1   0 27 47 18 38  9 29|168|168|168|168|   1 28 48 19 39 10 30
 5 1 4 0 3 6 2   2 1 0 6 5 4 3  37  8 28  6 26 46 17|168|168|168|168|  38  9 29  7 27 47 18
 3 6 2 5 1 4 0   4 3 2 1 0 6 5  25 45 16 36  7 34  5|168|168|168|168|  26 46 17 37  8 35  6
 1 4 0 3 6 2 5   6 5 4 3 2 1 0  13 33  4 24 44 15 35|168|168|168|168|  14 34  5 25 45 16 36
 6 2 5 1 4 0 3   1 0 6 5 4 3 2  43 14 41 12 32  3 23|168|168|168|168|  44 15 42 13 33  4 24
 4 0 3 6 2 5 1   3 2 1 0 6 5 4  31  2 22 42 20 40 11|168|168|168|168|  32  3 23 43 21 41 12
 2 5 1 4 0 3 6   5 4 3 2 1 0 6  19 39 10 30  1 21 48|168|168|168|168|  20 40 11 31  2 22 49
```

（ラテン・ユニット高位×７＋低位＝[数学的表記]；[数学的表記]＋１＝［古典的表記］）

先ずラテン・ユニットを作る。そのための基本図と基本式を，次のようにする。

［基本図］と［基本式群］

```
//                 .--.--.--.--.--.--.--.
//   2  3  4  5  6  7| 1| 2| 3| 4| 5| 6| 7| 1  2  3  4  5  6
//                  |--+--+--+--+--+--+--|
//   9 10 11 12 13 14| 8| 9|10|11|12|13|14| 8  9 10 11 12 13
//                  |--+--+--+--+--+--+--|
//  16 17 18 19 20 21|15|16|17|18|19|20|21|15 16 17 18 19 20
//                  |--+--+--+--+--+--+--|
//  23 24 25 26 27 28|22|23|24|25|26|27|28|22 23 24 25 26 27
//                  |--+--+--+--+--+--+--|
//  30 31 32 33 34 35|29|30|31|32|33|34|35|29 30 31 32 33 34
//                  |--+--+--+--+--+--+--|
//  37 38 39 40 41 42|36|37|38|39|40|41|42|36 37 38 39 40 41
//                  |--+--+--+--+--+--+--|
//  44 45 46 47 48 49|43|44|45|46|47|48|49|43 44 45 46 47 48
//                   '--'--'--'--'--'--'--'
```

```
//** Define Conditions for Rows and Columns: C=21 **
// n1+n2+n3+n4+n5+n6+n7=C         ... rw1  | n1+n8+n15+n22+n29+n36+n43=C  ... cl1
// n8+n9+n10+n11+n12+n13+n14=C    ... rw2  | n2+n9+n16+n23+n30+n37+n44=C  ... cl2
// n15+n16+n17+n18+n19+n20+n21=C  ... rw3  | n3+n10+n17+n24+n31+n38+n45=C ... cl3
// n22+n23+n24+n25+n26+n27+n28=C  ... rw4  | n4+n11+n18+n25+n32+n39+n46=C ... cl4
// n29+n30+n31+n32+n33+n34+n35=C  ... rw5  | n5+n12+n19+n26+n33+n40+n47=C ... cl5
// n36+n37+n38+n39+n40+n41+n42=C  ... rw6  | n6+n13+n20+n27+n34+n41+n48=C ... cl6
// n43+n44+n45+n46+n47+n48+n49=C  ... rw7  | n7+n14+n21+n28+n35+n42+n49=C ... cl7
//** Pan-diagonal Conditions of the Object: C=21 **
// n1+n9+n17+n25+n33+n41+n49=C    ... pd1  | n1+n14+n20+n26+n32+n38+n44=C ... pd8
// n2+n10+n18+n26+n34+n42+n43=C   ... pd2  | n2+n8+n21+n27+n33+n39+n45=C  ... pd9
// n3+n11+n19+n27+n35+n36+n44=C   ... pd3  | n3+n9+n15+n28+n34+n40+n46=C  ... pd10
// n4+n12+n20+n28+n29+n37+n45=C   ... pd4  | n4+n10+n16+n22+n35+n41+n47=C ... pd11
// n5+n13+n21+n22+n30+n38+n46=C   ... pd5  | n5+n11+n17+n23+n29+n42+n48=C ... pd12
// n6+n14+n15+n23+n31+n39+n47=C   ... pd6  | n6+n12+n18+n24+n30+n36+n49=C ... pd13
// n7+n8+n16+n24+n32+n40+n48=C    ... pd7  | n7+n13+n19+n25+n31+n37+n43=C ... pd14
//** Self-complementary Conditions of the Object: SC=6 **
// n1+n49=n2+n48=n3+n47=n4+n46=n5+n45=n6+n44=n7+n43=n8+n42=n9+n41=
// n10+n40=n11+n39=n12+n38=n13+n37=n14+n36=n15+n35=n16+n34=n17+n33=n18+n32=
// n19+n31=n20+n30=n21+n29=n22+n28=n23+n27=n24+n26=n25+n25=SC ... scc
//
```

ラテン方陣と言うからには，全行列・全汎対角線の構成内容が必ず {0, 1, 2, 3, 4, 5, 6} となるようにする。それだけで 0+1+2+3+4+5+6=21(+) となり，定和が実現する。
また例の規則も付加する：「ラテン方陣の各面で使われる数字は {0, 1, 2, 3, 4, 5, 6} に限られる。しかも各数字が 7 回ずつ使われる。使用回数に過不足は許されない。」

プログラムは，例えば次のように書く。{1, 2, 3, 4, 5, 6, 7, 8, ..., 47, 48, 49} という数字を扱うので，プログラムも五方陣の時の倍のスケールになる。フラグも増える。
それに最新の改訂版では，7 個の数字が 7 回ずつ使われるのを監視する記述を加えた。
余りに長いリストは恐縮なので，主要部分のみを示す。

```
//** Compose 'Complete Euler Squares' of Order 7 for the Simultaneous Type: **
//**  Both Self-complementary and Pan-diagonal by 'New Euler's Method' **
//** 'CEulerS7Sml.c' Dictated by Kanji Setsuda in 2003, 2010, 2011, **
//**  and on March 18, 2012; Finally revised on Apr.11, 2012 **
//**  with MacOSX 10.7.3 and Xcode 4.3.2  **
//
#include <stdio.h>
//
short int lcnt, cnt, cnt1, cnt3;
short SC, LSM;
short u1, u2;
short nm[50], uflg[50];
short dn[5][52];
short tlu[193][50];
short mtc[193][193];
short st[3457][52];
short tn[50];
short cs[4][29];
//
short rw1[7], cl1[7], pd1[7], pb1[7];
short rw2[7], cl2[7], pd2[7], pb2[7];
short rw3[7], cl3[7], pd3[7], pb3[7];
short rw4[7], cl4[7], pd4[7], pb4[7];
short rw5[7], cl5[7], pd5[7], pb5[7];
short rw6[7], cl6[7], pd6[7], pb6[7];
short rw7[7], cl7[7], pd7[7], pb7[7];
//
//* Sub-procedures *
void stp00(void);
void stp01(void), stp02(void), stp03(void), stp04(void);
```

```
void stp05(void), stp06(void), stp07(void), stp08(void);
void stp09(void), stp10(void), stp11(void), stp12(void);
void stp13(void), stp14(void), stp15(void), stp16(void);
void stp17(void), stp18(void), stp19(void), stp20(void);
void stp21(void), stp22(void), stp23(void), stp24(void);
void lurecord(void);
void slusort(void), excslu(short x);
void refmatch(void);
void prlunit(void);
void cmbcmp(void), recsol(void);
void solsort(void), excsol(short x);
void solprint(void);
void pr2sol(void);
void prsol(short x);
void chksms(short x);
//
//* Main Program *
int main(){
 short n;
  printf("\n");
  printf("** 'Complete Euler Type' of Simultaneous Magic Squares of Order 7: **\n");
  printf("** Self-complementary and Pan-diagonal; made by New Euler's Method **\n");
  for(n=0;n<50;n++){nm[n]=0;}
  for(n=0;n<7;n++){uflg[n]=0;
    rw1[n]=0; cl1[n]=0; pd1[n]=0; pb1[n]=0;
    rw2[n]=0; cl2[n]=0; pd2[n]=0; pb2[n]=0;
    rw3[n]=0; cl3[n]=0; pd3[n]=0; pb3[n]=0;
    rw4[n]=0; cl4[n]=0; pd4[n]=0; pb4[n]=0;
    rw5[n]=0; cl5[n]=0; pd5[n]=0; pb5[n]=0;
    rw6[n]=0; cl6[n]=0; pd6[n]=0; pb6[n]=0;
    rw7[n]=0; cl7[n]=0; pd7[n]=0; pb7[n]=0;
  }
  SC=6; LSM=21; cnt=0; cnt3=0;
  stp00();                      // Make the Latin Units
  slusort();                    // Sort the Latin Units
  lcnt=cnt;
  refmatch();                   // Make the Reference Table of Matching
  cnt=0;
  printf("\n");
  printf(" [List of Latin Units]\n");
  prlunit();
  LSM=175; cnt=0; cnt1=0; cnt3=0;
  printf("\n");
  printf("** Complete Euler Type of Simultaneous MS77: Self-Complementary and Pan-diagonal; **\n");
  printf("   [Latin Units: /High /Low; Sol_Number#; Sum_Errors:|Row|Clm|Pd1\\Pd2/]\n");
  cmbcmp();                     // Combine and Compose
  solsort();                    // Sort the Solutions
  solprint();                   // Print the Solution List
  if(cnt3>0){prsol(cnt3);}      // Print The Rest One
  printf(" [Total Counts of Solutions([n1=1] All#) = [%d] %d#]   [OK!]\n",cnt1,cnt);
  printf("** Calculated and Listed by Kanji Setsuda\n");
  printf("    on Apr.11, 2012 with MacOSX 10.7.3 and Xcode 4.3.2 **\n");
  printf("\n");
 return 0;
}
//
//** The Program Modules for 'Complete Euler Squares' of Order 7: **
//**  Simultaneous Type: Both Self-Complementary and Pan-diagonal **
//* Make the Latin Units *
//Level #0:
//Set n25{4,4,1,7}
void stp00(){
 short a;
  a=3; nm[25]=a;
  uflg[a]++; rw4[a]=1; cl4[a]=1; pd1[a]=1; pb7[a]=1;
```

```
  stp01();
  rw4[a]=0; cl4[a]=0; pd1[a]=0; pb7[a]=0; uflg[a]--;
}
//Level #1:
//Set n1 & n49=SC-N1
void stp01(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw1[a]==0)&&(cl1[a]==0)&&(pd1[a]==0)&&(pb1[a]==0)){
        if((rw7[b]==0)&&(cl7[b]==0)&&(pd1[b]==0)&&(pb6[b]==0)){
          nm[1]=a; nm[49]=b;
          uflg[a]++; rw1[a]=1; cl1[a]=1; pd1[a]=1; pb1[a]=1;
          uflg[b]++; rw7[b]=1; cl7[b]=1; pd1[b]=1; pb6[b]=1;
          stp02();
          rw7[b]=0; cl7[b]=0; pd1[b]=0; pb6[b]=0; uflg[b]--;
          rw1[a]=0; cl1[a]=0; pd1[a]=0; pb1[a]=0; uflg[a]--;}}}
  }
}
//Set n9 & n41=SC-N9
void stp02(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw2[a]==0)&&(cl2[a]==0)&&(pd1[a]==0)&&(pb3[a]==0)){
        if((rw6[b]==0)&&(cl6[b]==0)&&(pd1[b]==0)&&(pb4[b]==0)){
          nm[9]=a; nm[41]=b;
          uflg[a]++; rw2[a]=1; cl2[a]=1; pd1[a]=1; pb3[a]=1;
          uflg[b]++; rw6[b]=1; cl6[b]=1; pd1[b]=1; pb4[b]=1;
          stp03();
          rw6[b]=0; cl6[b]=0; pd1[b]=0; pb4[b]=0; uflg[b]--;
          rw2[a]=0; cl2[a]=0; pd1[a]=0; pb3[a]=0; uflg[a]--;}}}
  }
}
//Set n17 & n33=SC-N17
void stp03(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw3[a]==0)&&(cl3[a]==0)&&(pd1[a]==0)&&(pb5[a]==0)){
        if((rw5[b]==0)&&(cl5[b]==0)&&(pd1[b]==0)&&(pb2[b]==0)){
          nm[17]=a; nm[33]=b;
          uflg[a]++; rw3[a]=1; cl3[a]=1; pd1[a]=1; pb5[a]=1;
          uflg[b]++; rw5[b]=1; cl5[b]=1; pd1[b]=1; pb2[b]=1;
          stp04();
          rw5[b]=0; cl5[b]=0; pd1[b]=0; pb2[b]=0; uflg[b]--;
          rw3[a]=0; cl3[a]=0; pd1[a]=0; pb5[a]=0; uflg[a]--;}}}
  }
}
//Level #2:
//Set n2 & n48=SC-N2
void stp04(){
 short a,b;
  for(a=6;a>=0;a--){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw1[a]==0)&&(cl2[a]==0)&&(pd2[a]==0)&&(pb2[a]==0)){
        if((rw7[b]==0)&&(cl6[b]==0)&&(pd7[b]==0)&&(pb5[b]==0)){
          nm[2]=a; nm[48]=b;
          uflg[a]++; rw1[a]=1; cl2[a]=1; pd2[a]=1; pb2[a]=1;
          uflg[b]++; rw7[b]=1; cl6[b]=1; pd7[b]=1; pb5[b]=1;
          stp05();
          rw7[b]=0; cl6[b]=0; pd7[b]=0; pb5[b]=0; uflg[b]--;
          rw1[a]=0; cl2[a]=0; pd2[a]=0; pb2[a]=0; uflg[a]--;}}}
  }
}
```

```
//Set n3 & n47=SC-N3
void stp05(){
 short a,b;
  for(a=6;a>=0;a--){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw1[a]==0)&&(cl3[a]==0)&&(pd3[a]==0)&&(pb3[a]==0)){
        if((rw7[b]==0)&&(cl5[b]==0)&&(pd6[b]==0)&&(pb4[b]==0)){
          nm[3]=a; nm[47]=b;
          uflg[a]++; rw1[a]=1; cl3[a]=1; pd3[a]=1; pb3[a]=1;
          uflg[b]++; rw7[b]=1; cl5[b]=1; pd6[b]=1; pb4[b]=1;
          stp06();
          rw7[b]=0; cl5[b]=0; pd6[b]=0; pb4[b]=0; uflg[b]--;
          rw1[a]=0; cl3[a]=0; pd3[a]=0; pb3[a]=0; uflg[a]--;}}}
  }
}
//Set n4 & n46=SC-N4
void stp06(){
 short a,b;
  for(a=6;a>=0;a--){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw1[a]==0)&&(cl4[a]==0)&&(pd4[a]==0)&&(pb4[a]==0)){
        if((rw7[b]==0)&&(cl4[b]==0)&&(pd5[b]==0)&&(pb3[b]==0)){
          nm[4]=a; nm[46]=b;
          uflg[a]++; rw1[a]=1; cl4[a]=1; pd4[a]=1; pb4[a]=1;
          uflg[b]++; rw7[b]=1; cl4[b]=1; pd5[b]=1; pb3[b]=1;
          stp07();
          rw7[b]=0; cl4[b]=0; pd5[b]=0; pb3[b]=0; uflg[b]--;
          rw1[a]=0; cl4[a]=0; pd4[a]=0; pb4[a]=0; uflg[a]--;}}}
  }
}
//Set n5 & n45=SC-N5
void stp07(){
 short a,b;
  for(a=6;a>=0;a--){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw1[a]==0)&&(cl5[a]==0)&&(pd5[a]==0)&&(pb5[a]==0)){
        if((rw7[b]==0)&&(cl3[b]==0)&&(pd4[b]==0)&&(pb2[b]==0)){
          nm[5]=a; nm[45]=b;
          uflg[a]++; rw1[a]=1; cl5[a]=1; pd5[a]=1; pb5[a]=1;
          uflg[b]++; rw7[b]=1; cl3[b]=1; pd4[b]=1; pb2[b]=1;
          stp08();
          rw7[b]=0; cl3[b]=0; pd4[b]=0; pb2[b]=0; uflg[b]--;
          rw1[a]=0; cl5[a]=0; pd5[a]=0; pb5[a]=0; uflg[a]--;}}}
  }
}
//Set n6 & n44=SC-N6
void stp08(){
 short a,b;
  for(a=6;a>=0;a--){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw1[a]==0)&&(cl6[a]==0)&&(pd6[a]==0)&&(pb6[a]==0)){
        if((rw7[b]==0)&&(cl2[b]==0)&&(pd3[b]==0)&&(pb1[b]==0)){
          nm[6]=a; nm[44]=b;
          uflg[a]++; rw1[a]=1; cl6[a]=1; pd6[a]=1; pb6[a]=1;
          uflg[b]++; rw7[b]=1; cl2[b]=1; pd3[b]=1; pb1[b]=1;
          stp09();
          rw7[b]=0; cl2[b]=0; pd3[b]=0; pb1[b]=0; uflg[b]--;
          rw1[a]=0; cl6[a]=0; pd6[a]=0; pb6[a]=0; uflg[a]--;}}}
  }
}
//Set n7 & n43=SC-N7
void stp09(){
 short a,b;
  for(a=6;a>=0;a--){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
```

```
      if((rw1[a]==0)&&(cl7[a]==0)&&(pd7[a]==0)&&(pb7[a]==0)){
        if((rw7[b]==0)&&(cl1[b]==0)&&(pd2[b]==0)&&(pb7[b]==0)){
          nm[7]=a; nm[43]=b;
          uflg[a]++; rw1[a]=1; cl7[a]=1; pd7[a]=1; pb7[a]=1;
          uflg[b]++; rw7[b]=1; cl1[b]=1; pd2[b]=1; pb7[b]=1;
          stp10();
          rw7[b]=0; cl1[b]=0; pd2[b]=0; pb7[b]=0; uflg[b]--;
          rw1[a]=0; cl7[a]=0; pd7[a]=0; pb7[a]=0; uflg[a]--;}}}
  }
}
//Level #3:
//Set n8 & n42=SC-N8
void stp10(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw2[a]==0)&&(cl1[a]==0)&&(pd7[a]==0)&&(pb2[a]==0)){
        if((rw6[b]==0)&&(cl7[b]==0)&&(pd2[b]==0)&&(pb5[b]==0)){
          nm[8]=a; nm[42]=b;
          uflg[a]++; rw2[a]=1; cl1[a]=1; pd7[a]=1; pb2[a]=1;
          uflg[b]++; rw6[b]=1; cl7[b]=1; pd2[b]=1; pb5[b]=1;
          stp11();
          rw6[b]=0; cl7[b]=0; pd2[b]=0; pb5[b]=0; uflg[b]--;
          rw2[a]=0; cl1[a]=0; pd7[a]=0; pb2[a]=0; uflg[a]--;}}}
  }
}
//Set n15 & n35=SC-N15
void stp11(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw3[a]==0)&&(cl1[a]==0)&&(pd6[a]==0)&&(pb3[a]==0)){
        if((rw5[b]==0)&&(cl7[b]==0)&&(pd3[b]==0)&&(pb4[b]==0)){
          nm[15]=a; nm[35]=b;
          uflg[a]++; rw3[a]=1; cl1[a]=1; pd6[a]=1; pb3[a]=1;
          uflg[b]++; rw5[b]=1; cl7[b]=1; pd3[b]=1; pb4[b]=1;
          stp12();
          rw5[b]=0; cl7[b]=0; pd3[b]=0; pb4[b]=0; uflg[b]--;
          rw3[a]=0; cl1[a]=0; pd6[a]=0; pb3[a]=0; uflg[a]--;}}}
  }
}
//Set n22 & n28=SC-N22
void stp12(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw4[a]==0)&&(cl1[a]==0)&&(pd5[a]==0)&&(pb4[a]==0)){
        if((rw4[b]==0)&&(cl7[b]==0)&&(pd4[b]==0)&&(pb3[b]==0)){
          nm[22]=a; nm[28]=b;
          uflg[a]++; rw4[a]=1; cl1[a]=1; pd5[a]=1; pb4[a]=1;
          uflg[b]++; rw4[b]=1; cl7[b]=1; pd4[b]=1; pb3[b]=1;
          stp13();
          rw4[b]=0; cl7[b]=0; pd4[b]=0; pb3[b]=0; uflg[b]--;
          rw4[a]=0; cl1[a]=0; pd5[a]=0; pb4[a]=0; uflg[a]--;}}}
  }
}
//Set n29 & n21=SC-N29
void stp13(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw5[a]==0)&&(cl1[a]==0)&&(pd4[a]==0)&&(pb5[a]==0)){
        if((rw3[b]==0)&&(cl7[b]==0)&&(pd5[b]==0)&&(pb2[b]==0)){
          nm[29]=a; nm[21]=b;
          uflg[a]++; rw5[a]=1; cl1[a]=1; pd4[a]=1; pb5[a]=1;
```

```
            uflg[b]++; rw3[b]=1; cl7[b]=1; pd5[b]=1; pb2[b]=1;
            stp14();
            rw3[b]=0; cl7[b]=0; pd5[b]=0; pb2[b]=0; uflg[b]--;
            rw5[a]=0; cl1[a]=0; pd4[a]=0; pb5[a]=0; uflg[a]--;}}}
  }
}
//Set n36 & n14=SC-N36
void stp14(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw6[a]==0)&&(cl1[a]==0)&&(pd3[a]==0)&&(pb6[a]==0)){
        if((rw2[b]==0)&&(cl7[b]==0)&&(pd6[b]==0)&&(pb1[b]==0)){
          nm[36]=a; nm[14]=b;
          uflg[a]++; rw6[a]=1; cl1[a]=1; pd3[a]=1; pb6[a]=1;
          uflg[b]++; rw2[b]=1; cl7[b]=1; pd6[b]=1; pb1[b]=1;
          stp15();
          rw2[b]=0; cl7[b]=0; pd6[b]=0; pb1[b]=0; uflg[b]--;
          rw6[a]=0; cl1[a]=0; pd3[a]=0; pb6[a]=0; uflg[a]--;}}}
  }
}
//Level #4:
//Set n13 & n37=SC-N13
//Set n19 & n31=SC-N19
//Set n10 & n40=SC-N10
//Set n12 & n38=SC-N12
//  ...（途中省略あり）...
//Set n16 & n34=SC-N16
void stp20(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw3[a]==0)&&(cl2[a]==0)&&(pd7[a]==0)&&(pb4[a]==0)){
        if((rw5[b]==0)&&(cl6[b]==0)&&(pd2[b]==0)&&(pb3[b]==0)){
          nm[16]=a; nm[34]=b;
          uflg[a]++; rw3[a]=1; cl2[a]=1; pd7[a]=1; pb4[a]=1;
          uflg[b]++; rw5[b]=1; cl6[b]=1; pd2[b]=1; pb3[b]=1;
          stp21();
          rw5[b]=0; cl6[b]=0; pd2[b]=0; pb3[b]=0; uflg[b]--;
          rw3[a]=0; cl2[a]=0; pd7[a]=0; pb4[a]=0; uflg[a]--;}}}
  }
}
//Set n23 & n27=SC-N23
void stp21(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw4[a]==0)&&(cl2[a]==0)&&(pd6[a]==0)&&(pb5[a]==0)){
        if((rw4[b]==0)&&(cl6[b]==0)&&(pd3[b]==0)&&(pb2[b]==0)){
          nm[23]=a; nm[27]=b;
          uflg[a]++; rw4[a]=1; cl2[a]=1; pd6[a]=1; pb5[a]=1;
          uflg[b]++; rw4[b]=1; cl6[b]=1; pd3[b]=1; pb2[b]=1;
          stp22();
          rw4[b]=0; cl6[b]=0; pd3[b]=0; pb2[b]=0; uflg[b]--;
          rw4[a]=0; cl2[a]=0; pd6[a]=0; pb5[a]=0; uflg[a]--;}}}
  }
}
//Set n30 & n20=SC-N30
void stp22(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw5[a]==0)&&(cl2[a]==0)&&(pd5[a]==0)&&(pb6[a]==0)){
        if((rw3[b]==0)&&(cl6[b]==0)&&(pd4[b]==0)&&(pb1[b]==0)){
          nm[30]=a; nm[20]=b;
```

```
            uflg[a]++; rw5[a]=1; cl2[a]=1; pd5[a]=1; pb6[a]=1;
            uflg[b]++; rw3[b]=1; cl6[b]=1; pd4[b]=1; pb1[b]=1;
            stp23();
            rw3[b]=0; cl6[b]=0; pd4[b]=0; pb1[b]=0; uflg[b]--;
            rw5[a]=0; cl2[a]=0; pd5[a]=0; pb6[a]=0; uflg[a]--;}}}
  }
}
//Set n18 & n32=SC-N18
void stp23(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw3[a]==0)&&(cl4[a]==0)&&(pd2[a]==0)&&(pb6[a]==0)){
        if((rw5[b]==0)&&(cl4[b]==0)&&(pd7[b]==0)&&(pb1[b]==0)){
          nm[18]=a; nm[32]=b;
          uflg[a]++; rw3[a]=1; cl4[a]=1; pd2[a]=1; pb6[a]=1;
          uflg[b]++; rw5[b]=1; cl4[b]=1; pd7[b]=1; pb1[b]=1;
          stp24();
          rw5[b]=0; cl4[b]=0; pd7[b]=0; pb1[b]=0; uflg[b]--;
          rw3[a]=0; cl4[a]=0; pd2[a]=0; pb6[a]=0; uflg[a]--;}}}
  }
}
//Set n24 & n26=SC-N24
void stp24(){
 short a,b;
  for(a=0;a<7;a++){b=SC-a;
    if((uflg[a]<7)&&(uflg[b]<7)){
      if((rw4[a]==0)&&(cl3[a]==0)&&(pd7[a]==0)&&(pb6[a]==0)){
        if((rw4[b]==0)&&(cl5[b]==0)&&(pd2[b]==0)&&(pb1[b]==0)){
          nm[24]=a; nm[26]=b;
          uflg[a]++; rw4[a]=1; cl3[a]=1; pd7[a]=1; pb6[a]=1;
          uflg[b]++; rw4[b]=1; cl5[b]=1; pd2[b]=1; pb1[b]=1;
          lurecord();
          rw4[b]=0; cl5[b]=0; pd2[b]=0; pb1[b]=0; uflg[b]--;
          rw4[a]=0; cl3[a]=0; pd7[a]=0; pb6[a]=0; uflg[a]--;}}}
  }
}
//
//* Record Each of the Latin Units *
void lurecord(){
 short n;
  tlu[cnt][0]=cnt+1;
  for(n=1;n<50;n++){tlu[cnt][n]=nm[n];}
  cnt++;
}
//  ...
//  ...（途中省略あり）...
//* Combine and Compose *
void cmbcmp(){
 short n,d,fc;
  for(u1=0;u1<(lcnt/2);u1++){
    for(u2=0;u2<lcnt;u2++){
      if(mtc[u1][u2]==7){
        for(n=1;n<50;n++){uflg[n]=0;}
        for(n=1;n<50;n++){
          d=tlu[u1][n]*7+tlu[u2][n]+1;
          nm[n]=d; uflg[d]++;
        }
        fc=0;
        for(n=1;n<50;n++){if(uflg[n]==1){fc++;}else{break;}}
        if(fc==49){
          if((nm[1]<nm[49])&&(nm[1]<nm[43])&&(nm[43]<nm[7])){recsol();}
        }
      }
    }
```

```
  }
}
//
//* Record Each of the Solutions *
void recsol(){
 short n;
  cnt++;
  st[cnt][0]=cnt;
  for(n=1;n<50;n++){st[cnt][n]=nm[n];}
  st[cnt][50]=u1+1; st[cnt][51]=u2+1;
  if(nm[1]==1){cnt1++;}
}
//
//  ...（以下省略）...
//
```

単位面は，全部で 192面が出て来た。これも「桂馬とび構成法」の時と同じである。
この中から２面ずつを選んで組み合わせて計算・合成する。様々なチェックも必要だ。
目的の完全オイラー型対称汎七方陣は，3456個の標準解が得られた。これも桂馬とび構成法の時と全く同じだ（そうでなくてはならない！）。

** 新オイラー法で対称汎七方陣を構成する **

[ラテン・ユニットのリスト]

1/	2/	3/	4/	5/	6/	7/	8/
0654321	0531642	0362514	0246135	0652341	0531624	0426135	0364512
2106543	3164205	5140362	6135024	4106523	3162405	6135042	5120364
4321065	6420531	3625140	5024613	2341065	6240531	5042613	3645120
6543210	2053164	1403625	4613502	6523410	4053162	2613504	1203645
1065432	5316420	6251403	3502461	1065234	5316240	3504261	6451203
3210654	1642053	4036251	2461350	3410652	1624053	4261350	2036451
5432106	4205316	2514036	1350246	5234106	2405316	1350426	4512036

49/	50/	51/	52/	53/	54/	55/	56/
1526304	1350246	1234560	1065432	1605432	1520364	1356240	1234506
0415263	2461350	3456012	5432106	5432160	6415203	2401356	3450612
6304152	3502461	5601234	2106543	2160543	0364152	3562401	5061234
5263041	4613502	0123456	6543210	0543216	5203641	4013562	6123450
4152630	5024613	2345601	3210654	3216054	4152036	5624013	2345061
3041526	6135024	4560123	0654321	6054321	3641520	0135624	4506123
2630415	0246135	6012345	4321065	4321605	2036415	6240135	0612345

97/	98/	99/	100/	101/	102/	103/	104/
4630251	4321605	4265310	4152036	4630215	4512036	4325601	4261350
3025146	6054321	1042653	2036415	3021546	2036451	6014325	5042613
2514630	3216054	5310426	6415203	2154630	6451203	3256014	1350426
1463025	0543216	2653104	5203641	5463021	1203645	0143256	2613504
6302514	2160543	0426531	3641520	6302154	3645120	2560143	0426135
0251463	5432160	3104265	1520364	0215463	5120364	1432560	3504261
5146302	1605432	6531042	0364152	1546302	0364512	5601432	6135042

145/	146/	147/	148/	149/	150/	151/	152/
5310246	5234160	5126304	5061432	5601432	5316240	5234106	5120364
2465310	3416052	0451263	1432506	1432560	2405316	3410652	6451203
3102465	1605234	6304512	2506143	2560143	3162405	1065234	0364512
4653102	0523416	1263045	6143250	0143256	4053162	6523410	1203645
1024653	2341605	4512630	3250614	3256014	1624053	2341065	4512036
6531024	4160523	3045126	0614325	6014325	0531624	4106523	3645120
0246531	6052341	2630451	4325061	4325601	6240531	0652341	2036451

[ラテン方陣の総数 = 192]

** 「完全オイラー型」対称汎七方陣のリスト: **

[標準解: 使用ユニット: 高位/H 低位/L; 解番号#; サム・チェックのエラー表示: |行|列\汎対1/汎対2|]

3/H	1/L							1#	R	C\P/P	
0362514	0654321	1	28	48	19	39	10	30	0	0	0	0
5140362	2106543	38	9	29	7	27	47	18	0	0	0	0
3625140	4321065	26	46	17	37	8	35	6	0	0	0	0
1403625	6543210	14	34	5	25	45	16	36	0	0	0	0
6251403	1065432	44	15	42	13	33	4	24	0	0	0	0
4036251	3210654	32	3	23	43	21	41	12	0	0	0	0
2514036	5432106	20	40	11	31	2	22	49	0	0	0	0

3/H	2/L							2#	R	C\P/P	
0362514	0531642	1	27	46	16	42	12	31	0	0	0	0
5140362	3164205	39	9	35	5	24	43	20	0	0	0	0
3625140	6420531	28	47	17	36	13	32	2	0	0	0	0
1403625	2053164	10	29	6	25	44	21	40	0	0	0	0
6251403	5316420	48	18	37	14	33	3	22	0	0	0	0
4036251	1642053	30	7	26	45	15	41	11	0	0	0	0
2514036	4205316	19	38	8	34	4	23	49	0	0	0	0

3/H	4/L							3#	R	C\P/P	
0362514	0246135	1	24	47	21	37	11	34	0	0	0	0
5140362	6135024	42	9	32	6	22	45	19	0	0	0	0
3625140	5024613	27	43	17	40	14	30	4	0	0	0	0
1403625	4613502	12	35	2	25	48	15	38	0	0	0	0
6251403	3502461	46	20	36	10	33	7	23	0	0	0	0
4036251	2461350	31	5	28	44	18	41	8	0	0	0	0
2514036	1350246	16	39	13	29	3	26	49	0	0	0	0

4/H	1/L							4#	R	C\P/P	
0246135	0654321	1	21	34	47	11	24	37	0	0	0	0
6135024	2106543	45	9	22	42	6	19	32	0	0	0	0
5024613	4321065	40	4	17	30	43	14	27	0	0	0	0
4613502	6543210	35	48	12	25	38	2	15	0	0	0	0
3502461	1065432	23	36	7	20	33	46	10	0	0	0	0
2461350	3210654	18	31	44	8	28	41	5	0	0	0	0
1350246	5432106	13	26	39	3	16	29	49	0	0	0	0

4/H	2/L							5#	R	C\P/P	
0246135	0531642	1	20	32	44	14	26	38	0	0	0	0
6135024	3164205	46	9	28	40	3	15	34	0	0	0	0
5024613	6420531	42	5	17	29	48	11	23	0	0	0	0
4613502	2053164	31	43	13	25	37	7	19	0	0	0	0
3502461	5316420	27	39	2	21	33	45	8	0	0	0	0
2461350	1642053	16	35	47	10	22	41	4	0	0	0	0
1350246	4205316	12	24	36	6	18	30	49	0	0	0	0

4/H	3/L							6#	R	C\P/P	
0246135	0362514	1	18	35	45	13	23	40	0	0	0	0
6135024	5140362	48	9	26	36	4	21	31	0	0	0	0
5024613	3625140	39	7	17	34	44	12	22	0	0	0	0
4613502	1403625	30	47	8	25	42	3	20	0	0	0	0
3502461	6251403	28	38	6	16	33	43	11	0	0	0	0
2461350	4036251	19	29	46	14	24	41	2	0	0	0	0
1350246	2514036	10	27	37	5	15	32	49	0	0	0	0

3/H	5/L							7#	R	C\P/P	
0362514	0652341	1	28	48	17	39	12	30	0	0	0	0
5140362	4106523	40	9	29	7	27	45	18	0	0	0	0
3625140	2341065	24	46	19	37	8	35	6	0	0	0	0
1403625	6523410	14	34	3	25	47	16	36	0	0	0	0
6251403	1065234	44	15	42	13	31	4	26	0	0	0	0
4036251	3410652	32	5	23	43	21	41	10	0	0	0	0
2514036	5234106	20	38	11	33	2	22	49	0	0	0	0

3/H	6/L							8#	R	C\P/P	
0362514	0531624	1	27	46	16	42	10	33	0	0	0	0
5140362	3162405	39	9	35	3	26	43	20	0	0	0	0
3625140	6240531	28	45	19	36	13	32	2	0	0	0	0
1403625	4053162	12	29	6	25	44	21	38	0	0	0	0
6251403	5316240	48	18	37	14	31	5	22	0	0	0	0
4036251	1624053	30	7	24	47	15	41	11	0	0	0	0
2514036	2405316	17	40	8	34	4	23	49	0	0	0	0

3/H	7/L							9#	R	C\P/P	
0362514	0426135	1	26	45	21	37	11	34	0	0	0	0
5140362	6135042	42	9	32	6	22	47	17	0	0	0	0
3625140	5042613	27	43	19	38	14	30	4	0	0	0	0
1403625	2613504	10	35	2	25	48	15	40	0	0	0	0
6251403	3504261	46	20	36	12	31	7	23	0	0	0	0
4036251	4261350	33	3	28	44	18	41	8	0	0	0	0
2514036	1350426	16	39	13	29	5	24	49	0	0	0	0

4/H	5/L							10#	R	C\P/P	
0246135	0652341	1	21	34	45	11	26	37	0	0	0	0
6135024	4106523	47	9	22	42	6	17	32	0	0	0	0
5024613	2341065	38	4	19	30	43	14	27	0	0	0	0
4613502	6523410	35	48	10	25	40	2	15	0	0	0	0
3502461	1065234	23	36	7	20	31	46	12	0	0	0	0
2461350	3410652	18	33	44	8	28	41	3	0	0	0	0
1350246	5234106	13	24	39	5	16	29	49	0	0	0	0

4/H	6/L							11#	R	C\P/P	
0246135	0531624	1	20	32	44	14	24	40	0	0	0	0
6135024	3162405	46	9	28	38	5	15	34	0	0	0	0
5024613	6240531	42	3	19	29	48	11	23	0	0	0	0
4613502	4053162	33	43	13	25	37	7	17	0	0	0	0
3502461	5316240	27	39	2	21	31	47	8	0	0	0	0
2461350	1624053	16	35	45	12	22	41	4	0	0	0	0
1350246	2405316	10	26	36	6	18	30	49	0	0	0	0

4/H	8/L							12#	R	C\P/P	
0246135	0364512	1	18	35	47	13	23	38	0	0	0	0
6135024	5120364	48	9	24	36	4	21	33	0	0	0	0
5024613	3645120	39	7	19	34	44	10	22	0	0	0	0
4613502	1203645	30	45	8	25	42	5	20	0	0	0	0
3502461	6451203	28	40	6	16	31	43	11	0	0	0	0
2461350	2036451	17	29	46	14	26	41	2	0	0	0	0
1350246	4512036	12	27	37	3	15	32	49	0	0	0	0

6/H	1/L							13#	R	C\P/P	
0531624	0654321	1	42	27	12	46	17	30	0	0	0	0
3162405	2106543	24	9	43	21	34	5	39	0	0	0	0
6240531	4321065	47	18	31	2	36	28	13	0	0	0	0
4053162	6543210	35	6	40	25	10	44	15	0	0	0	0
5316240	1065432	37	22	14	48	19	32	3	0	0	0	0
1624053	3210654	11	45	16	29	7	41	26	0	0	0	0
2405316	5432106	20	33	4	38	23	8	49	0	0	0	0

6/H	3/L							14#	R	C\P/P	
0531624	0362514	1	39	28	10	48	16	33	0	0	0	0
3162405	5140362	27	9	47	15	32	7	38	0	0	0	0
6240531	3625140	46	21	31	6	37	26	8	0	0	0	0
4053162	1403625	30	5	36	25	14	45	20	0	0	0	0
5316240	6251403	42	24	13	44	19	29	4	0	0	0	0
1624053	4036251	12	43	18	35	3	41	23	0	0	0	0
2405316	2514036	17	34	2	40	22	11	49	0	0	0	0

6/H	4/L							15#	R	C\P	P	
0531624	0246135	1	38	26	14	44	18	34	0	0	0	0
3162405	6135024	28	9	46	20	29	3	40	0	0	0	0
6240531	5024613	48	15	31	5	42	23	11	0	0	0	0
4053162	4613502	33	7	37	25	13	43	17	0	0	0	0
5316240	3502461	39	27	8	45	19	35	2	0	0	0	0
1624053	2461350	10	47	21	30	4	41	22	0	0	0	0
2405316	1350246	16	32	6	36	24	12	49	0	0	0	0

7/H	1/L							16#	R	C\P	P	
0426135	0654321	1	35	20	47	11	24	37	0	0	0	0
6135042	2106543	45	9	22	42	6	33	18	0	0	0	0
5042613	4321065	40	4	31	16	43	14	27	0	0	0	0
2613504	6543210	21	48	12	25	38	2	29	0	0	0	0
3504261	1065432	23	36	7	34	19	46	10	0	0	0	0
4261350	3210654	32	17	44	8	28	41	5	0	0	0	0
1350426	5432106	13	26	39	3	30	15	49	0	0	0	0

7/H	2/L							17#	R	C\P	P	
0426135	0531642	1	34	18	44	14	26	38	0	0	0	0
6135042	3164205	46	9	28	40	3	29	20	0	0	0	0
5042613	6420531	42	5	31	15	48	11	23	0	0	0	0
2613504	2053164	17	43	13	25	37	7	33	0	0	0	0
3504261	5316420	27	39	2	35	19	45	8	0	0	0	0
4261350	1642053	30	21	47	10	22	41	4	0	0	0	0
1350426	4205316	12	24	36	6	32	16	49	0	0	0	0

7/H	3/L							18#	R	C\P	P	
0426135	0362514	1	32	21	45	13	23	40	0	0	0	0
6135042	5140362	48	9	26	36	4	35	17	0	0	0	0
5042613	3625140	39	7	31	20	44	12	22	0	0	0	0
2613504	1403625	16	47	8	25	42	3	34	0	0	0	0
3504261	6251403	28	38	6	30	19	43	11	0	0	0	0
4261350	4036251	33	15	46	14	24	41	2	0	0	0	0
1350426	2514036	10	27	37	5	29	18	49	0	0	0	0

6/H	5/L							19#	R	C\P	P	
0531624	0652341	1	42	27	10	46	19	30	0	0	0	0
3162405	4106523	26	9	43	21	34	3	39	0	0	0	0
6240531	2341065	45	18	33	2	36	28	13	0	0	0	0
4053162	6523410	35	6	38	25	12	44	15	0	0	0	0
5316240	1065234	37	22	14	48	17	32	5	0	0	0	0
1624053	3410652	11	47	16	29	7	41	24	0	0	0	0
2405316	5234106	20	31	4	40	23	8	49	0	0	0	0

6/H	7/L							20#	R	C\P	P	
0531624	0426135	1	40	24	14	44	18	34	0	0	0	0
3162405	6135042	28	9	46	20	29	5	38	0	0	0	0
6240531	5042613	48	15	33	3	42	23	11	0	0	0	0
4053162	2613504	31	7	37	25	13	43	19	0	0	0	0
5316240	3504261	39	27	8	47	17	35	2	0	0	0	0
1624053	4261350	12	45	21	30	4	41	22	0	0	0	0
2405316	1350426	16	32	6	36	26	10	49	0	0	0	0

6/H	8/L							21#	R	C\P	P	
0531624	0364512	1	39	28	12	48	16	31	0	0	0	0
3162405	5120364	27	9	45	15	32	7	40	0	0	0	0
6240531	3645120	46	21	33	6	37	24	8	0	0	0	0
4053162	1203645	30	3	36	25	14	47	20	0	0	0	0
5316240	6451203	42	26	13	44	17	29	4	0	0	0	0
1624053	2036451	10	43	18	35	5	41	23	0	0	0	0
2405316	4512036	19	34	2	38	22	11	49	0	0	0	0

7/H	5/L							22#	R	C\P	P	
0426135	0652341	1	35	20	45	11	26	37	0	0	0	0
6135042	4106523	47	9	22	42	6	31	18	0	0	0	0
5042613	2341065	38	4	33	16	43	14	27	0	0	0	0
2613504	6523410	21	48	10	25	40	2	29	0	0	0	0
3504261	1065234	23	36	7	34	17	46	12	0	0	0	0
4261350	3410652	32	19	44	8	28	41	3	0	0	0	0
1350426	5234106	13	24	39	5	30	15	49	0	0	0	0

7/H	6/L							23#	R	C\P	P	
0426135	0531624	1	34	18	44	14	24	40	0	0	0	0
6135042	3162405	46	9	28	38	5	29	20	0	0	0	0
5042613	6240531	42	3	33	15	48	11	23	0	0	0	0
2613504	4053162	19	43	13	25	37	7	31	0	0	0	0
3504261	5316240	27	39	2	35	17	47	8	0	0	0	0
4261350	1624053	30	21	45	12	22	41	4	0	0	0	0
1350426	2405316	10	26	36	6	32	16	49	0	0	0	0

7/H	8/L							24#	R	C\P	P	
0426135	0364512	1	32	21	47	13	23	38	0	0	0	0
6135042	5120364	48	9	24	36	4	35	19	0	0	0	0
5042613	3645120	39	7	33	20	44	10	22	0	0	0	0
2613504	1203645	16	45	8	25	42	5	34	0	0	0	0
3504261	6451203	28	40	6	30	17	43	11	0	0	0	0
4261350	2036451	31	15	46	14	26	41	2	0	0	0	0
1350426	4512036	12	27	37	3	29	18	49	0	0	0	0

3/H	9/L	25#							3/H	17/L	49#								3/H	25/L	73#								11/H	1/L	97#							
1	28	47	20	39	9	31			1	28	45	20	39	9	33				1	28	44	19	39	10	34				1	28	48	12	32	17	37			
37	10	29	7	26	48	18			37	12	29	7	24	48	18				38	13	29	7	23	47	18				31	16	36	7	27	47	11			
27	46	16	38	8	35	5			27	46	16	40	8	35	3				26	46	17	41	8	35	2				26	46	10	30	15	42	6			
14	33	6	25	44	17	36			14	31	6	25	44	19	36				14	30	5	25	45	20	36				21	41	5	25	45	9	29			
45	15	42	12	34	4	23			47	15	42	10	34	4	23				48	15	42	9	33	4	24				44	8	35	20	40	4	24			
32	2	24	43	21	40	13			32	2	26	43	21	38	13				32	3	27	43	21	37	12				39	3	23	43	14	34	19			
19	41	11	30	3	22	49			17	41	11	30	5	22	49				16	40	11	31	6	22	49				13	33	18	38	2	22	49			

11/H	9/L	121#							11/H	17/L	145#								11/H	25/L	169#								17/H	2/L	193#							
1	28	47	13	32	16	38			1	28	45	13	32	16	40				1	28	44	12	32	17	41				1	48	18	37	28	12	31			
30	17	36	7	26	48	11			30	19	36	7	24	48	11				31	20	36	7	23	47	11				11	30	7	47	17	36	27			
27	46	9	31	15	42	5			27	46	9	33	15	42	3				26	46	10	34	15	42	2				42	26	10	29	6	46	16			
21	40	6	25	44	10	29			21	38	6	25	44	12	29				21	37	5	25	45	13	29				45	15	41	25	9	35	5			
45	8	35	19	41	4	23			47	8	35	17	41	4	23				48	8	35	16	40	4	24				34	4	44	21	40	24	8			
39	2	24	43	14	33	20			39	2	26	43	14	31	20				39	3	27	43	14	30	19				23	14	33	3	43	20	39			
12	34	18	37	3	22	49			10	34	18	37	5	22	49				9	33	18	38	6	22	49				19	38	22	13	32	2	49			

```
  17/H   10/L   217#     17/H   18/L   241#     17/H   26/L   265#     25/H    2/L   289#
 1 47 18 38 28 13 30    1 46 21 37 24 12 34    1 46 21 38 23 13 33    1 48 11 30 28 19 38
11 31  7 48 16 36 26   10 33  6 43 18 42 23    9 34  5 43 18 42 24   18 37  7 47 10 29 27
42 27  9 29  5 46 17   39 28  9 31  5 48 15   39 28 10 30  6 47 15   35 26 17 36  6 46  9
44 15 40 25 10 35  6   47 20 36 25 14 30  3   48 19 36 25 14 31  2   45  8 34 25 16 42  5
33  4 45 21 41 23  8   35  2 45 19 41 22 11   35  3 44 20 40 22 11   41  4 44 14 33 24 15
24 14 34  2 43 19 39   27  8 32  7 44 17 40   26  8 32  7 45 16 41   23 21 40  3 43 13 32
20 37 22 12 32  3 49   16 38 26 13 29  4 49   17 37 27 12 29  4 49   12 31 22 20 39  2 49

  25/H   10/L   313#     25/H   18/L   337#     25/H   26/L   361#      3/H   33/L   385#
 1 47 11 31 28 20 37    1 46 14 30 24 19 41    1 46 14 31 23 20 40    2 28 46 15 41 12 31
18 38  7 48  9 29 26   17 40  6 43 11 35 23   16 41  5 43 11 35 24   39  8 34  5 24 44 21
35 27 16 36  5 46 10   32 28 16 38  5 48  8   32 28 17 37  6 47  8   27 47 17 37 14 32  1
44  8 33 25 17 42  6   47 13 29 25 21 37  3   48 12 29 25 21 38  2   10 30  7 25 43 20 40
40  4 45 14 34 23 15   42  2 45 12 34 22 18   42  3 44 13 33 22 18   49 18 36 13 33  3 23
24 21 41  2 43 12 32   27 15 39  7 44 10 33   26 15 39  7 45  9 34   29  6 26 45 16 42 11
13 30 22 19 39  3 49    9 31 26 20 36  4 49   10 30 27 19 36  4 49   19 38  9 35  4 22 48

   3/H   65/L   769#      3/H   97/L  1153#      3/H  129/L  1537#      3/H  161/L  1921#
 3 28 46 15 40 13 30    5 28 46 15 38 13 30    6 28 46 15 37 12 31    7 27 46 16 36 12 31
39  8 33  6 23 45 21   39  8 31  6 23 47 21   39  8 30  5 24 48 21   39  9 29  5 24 49 20
26 48 16 38 14 32  1   24 48 16 40 14 32  1   23 47 17 41 14 32  1   22 47 17 42 13 32  2
 9 31  7 25 43 19 41    9 33  7 25 43 17 41   10 34  7 25 43 16 40   10 35  6 25 44 15 40
49 18 36 12 34  2 24   49 18 36 10 34  2 26   49 18 36  9 33  3 27   48 18 37  8 33  3 28
29  5 27 44 17 42 11   29  3 27 44 19 42 11   29  2 26 45 20 42 11   30  1 26 45 21 41 11
20 37 10 35  4 22 47   20 37 12 35  4 22 45   19 38 13 35  4 22 44   19 38 14 34  4 23 43

   35/H    1/L  2305#     35/H   33/L  2497#     35/H   65/L  2689#     35/H   97/L  2881#
 8 28 41 19 46  3 30    9 28 39 15 48  5 31   10 28 39 15 47  6 30   12 28 39 15 45  6 30
45  2 29 14 27 40 18   46  1 34 12 24 37 21   46  1 33 13 23 38 21   46  1 31 13 23 40 21
26 39 17 44  1 35 13   27 40 17 44  7 32  8   26 41 16 45  7 32  8   24 41 16 47  7 32  8
 7 34 12 25 38 16 43    3 30 14 25 36 20 47    2 31 14 25 36 19 48    2 33 14 25 36 17 48
37 15 49  6 33 11 24   42 18 43  6 33 10 23   42 18 43  5 34  9 24   42 18 43  3 34  9 26
32 10 23 36 21 48  5   29 13 26 38 16 49  4   29 12 27 37 17 49  4   29 10 27 37 19 49  4
20 47  4 31  9 22 42   19 45  2 35 11 22 41   20 44  3 35 11 22 40   20 44  5 35 11 22 38

   35/H  129/L  3073#     35/H  161/L  3265#
13 28 39 15 44  5 31   14 27 39 16 43  5 31
46  1 30 12 24 41 21   46  2 29 12 24 42 20
23 40 17 48  7 32  8   22 40 17 49  6 32  9
 3 34 14 25 36 16 47    3 35 13 25 37 15 47
42 18 43  2 33 10 27   41 18 44  1 33 10 28
29  9 26 38 20 49  4   30  8 26 38 21 48  4
19 45  6 35 11 22 37   19 45  7 34 11 23 36
 [解の総数([n1＝1] All#) = [384] 3456#]    [OK!]
** Calculated and Listed by Kanji Setsuda
    on Apr.11, 2012 with MacOSX 10.7.3 and Xcode 4.3.2 **
```

『桂馬とび構成法』の時と全く同じ結果が得られたことで，七次でも七進法による「新オイラー法」が通用することが確かめられた。

ついでに,**完全オイラー型**の**汎対角線七方陣**も構成してみた。

単位面だけで，20160面も出て来た。２面ずつを組み合わせて，選別しながら構成すると，38102400個の解がめでたく得られたことを報告する。この総数もまた，以前に桂馬とび構成法で得たものと全く変わらない。

これならば，奇数次にも自信を持って「新オイラー法」を適用できるわけだ。

最新の研究では，七次ではラテン条件に従わない緩い定義での「**オイラー型**」の構成も可能で，その結果そのタイプの対称汎七方陣の解の総数は 150552個あることがわかった。

完全オイラー型の 3456個は，その中に下位集合として含まれる。

だから，オイラー型と完全オイラー型は，今後は明快に区別して扱う必要がある。
第11-4節で詳しく報告しているので，是非お読みいただきたい。

最後に，私が何故これほどまでに**オイラー方陣**にこだわるのかについて，一つ付記しておきたい。その際，「美しいからだ」という審美的理由についてはあえて触れない。
一つには，オイラー型構成法の実行スピードが速いことがある。特に完全オイラー型のそれは，すこぶる速い。この高速性が，高次方陣を研究する時に大いに助かるのだ。
五次以上の魔方陣では，通常の方法でその型の「全数」を求めるのは，いつでも大変な作業になる。最新のコンピューターを使っても，時間を食うばかりだ。
六次以上の「全貌」は，現在でもおよそわからないままだ。
どの研究者も，だから高次では希少価値の高い「美しい」魔方陣の構成に向かわざるを得ない。さもないと，いつまで経ってもその型の総数が求まらないからだ。
そんな際にオイラー方陣構成法の存在が救いになってくれる。あっという間に「全貌」を明らかにしてくれるからだ。現在のところ五，七，八，九，十一，十三次ぐらいまで通用することがわかっている。
もちろんオイラー型は，「珍品」しか作れない。総数が少ない。だから求められるのだが，「全貌」にほど遠いかというと、そういう訳でもない。私は，オイラー解が一般解の良き「アブストラクト（抽象）」になってくれているように思う。その型の魔方陣本来の属性（＝個性）を損なわないまま，見事に圧縮して全貌を見せてくれるからである。
その「圧縮」のためにN進法を使う。ユニットに分解する。そして全体方陣とユニットの間に「相似の構造」を仮定する。その結果の素敵さは，見て御覧の通りだ。
相手がたとえ何であれ，総数が求まると，ただそれだけで嬉しい。ファンの「さが」てぇものには歯止めが無い。おかげさまで，私は今や世界一幸福な好事家で居られる！？

（初稿：2003年10月；改訂稿：2005年９月；2011年３月27日；
最新改訂版：2012年４月13日 by 摂田 寛二(Kanji Setsuda)）

E-Mail Address: jag12001@nifty.com
http://homepage2.nifty.com/KanjiSetsuda/

付記：今節は，ついに最長になってしまった。50 ページ以内におさめたい一心だったが，そのために最後の三つのプログラムが「抜粋」あるいは掲載省略となってしまい，全部を御覧になりたい方々に済まない気持ちが残った。
そこで，以前にもそうしたように別稿に「プログラム・シート」を載せ，付録とすることにした。内容は，Xcode の最新版を使って，新しいものに改訂してある。
目次で見て，すぐ次の節をクリックしていただきたい。