

第4部：『魔方陣の最新研究』： 撰田 寛二

第4章： 解説『魔方陣研究の諸技法』II

第4-1節：「完全オイラー五方陣」を正攻法で作る

0. 先ず定義に忠実な構成法で「完全オイラー五方陣」を作ろう

前節では、ひな形を完全オイラー方陣から選別し、原初方陣とひな形変換法によって五次と七次の完全オイラー方陣を構成した。なかなかスマートな結果が出たと思っているが、五次は良しとしても七次の場合は、それで全てを尽くしているかどうか保証の限りではなかった。何しろ誰もまだ目的方陣の全貌を知らず、いったいまとめて数えて幾つくらいあるのか想像さえつかなかったからである。

七次以上の魔方陣の全数計数は、全く骨が折れる。簡単に百万個千万個を超えるし、汎魔方陣などは、高速の最新型パソコンでも遅々たる計数しかなし得ない。途中調べなければならぬ「場合の数」が多過ぎるからである。

調べる場合の数を効果的に減らし、高速に動くプログラムを是が非でも開発しなければならない。そう知ってはいても、良い策をなかなか思い付かない。「これは、次世代に委ねるしかないか」と諦めかけていた折に、ものは試しにとテストした実験が大当たりした。

古人の知恵の結晶をそのままプログラムにしたら、どうなるか。「オイラー方陣」の定義そのものを忠実にプログラミングしてみたならどうなるかを、実験して調べたのである。

その成果をここで詳しく報告したい。

1. [完全オイラー五方陣] の定義

五方陣の場合には、先ず五進法分解図を作る。それが攻略の第一歩だ。

分解図は、古典的表記図の各数値からそれぞれ1を引いた近代的表記法を基にする。そして、各数を基数5で割った整商を上位面に、剰余を下位面に集めるのが、約束である。

下図の例解のいずれも右側2面が、その五進法分解図である。

左端の古典的表記とこの分解図との間には、同じ番地で見ても各数値の間に

$$1 = (0 \times 5 + 0) + 1; \quad 23 = (4 \times 5 + 2) + 1; \quad 15 = (2 \times 5 + 4) + 1; \\ 22 = (4 \times 5 + 1) + 1; \quad 17 = (3 \times 5 + 1) + 1; \quad 19 = (3 \times 5 + 3) + 1; \quad \dots$$

という関係式で表わされる対応関係が成り立つ。

** 完全オイラー方陣の実例1： 対称汎五方陣の場合 **

1/ /D5i					2/ /D5i																								
1	23	20	12	9	0	4	3	2	1	1	23	20	14	7	0	4	3	2	1	0	2	4	3	1					
15	7	4	21	18	2	1	0	4	3	4	1	3	0	2	15	9	2	21	18	2	1	0	4	3	4	3	1	0	2
24	16	13	10	2	4	3	2	1	0	3	0	2	4	1	22	16	13	10	4	4	3	2	1	0	1	0	2	4	3
8	5	22	19	11	1	0	4	3	2	2	4	1	3	0	8	5	24	17	11	1	0	4	3	2	2	4	3	1	0
17	14	6	3	25	3	2	1	0	4	1	3	0	2	4	19	12	6	3	25	3	2	1	0	4	3	1	0	2	4

上の例解を良く調べてみると、次のような特徴を備えていることがわかる。

- (1) 各行・各列の内容が、どれも {0, 1, 2, 3, 4} の数値を各1回ずつ使っていて、重複やもれ落ちが無い。汎対角線でも、どうやらそのようになっている。
- (2) したがって、上位/下位の各面でそれぞれの数値 {0, 1, 2, 3, 4} が、各5回使われている。これは(1)の性質が成り立てば、必ず成り立つ性質だ。
- (3) 上位と下位の数値の組み合わせ {00, 01, 02, 03, 04, ..., 41, 42, 43, 44} にも、同じ一回性が守られていて、重複やもれ落ちが無い。

この(3)の性質は、元の古典的表記で言えば「1~5²の自然数を各1回ずつ使い重複やもれ落ちが無い」という魔方陣の基本性質(約束)と、対応している。

こういう珍しい性質を持つ方陣の存在は、ヨーロッパでは随分と早くから知られていて、「ラテン方陣」と呼んで珍重されていた。近代にいたって、伝説の数学者オイラー Leonhard Euler (1707-1783) が、位取りに応じて上位と下位に分解して2面を作り、その両面で「ラテ

ン方陣」となるタイプを「グレコ・ラテン方陣」と呼んで特に珍重していた。

オイラーは、これをもって、ある種の魔方陣群の本質であると考えていたふしがある。

退官前の鈴木睦教授に教わった話では、彼自身は汎対角線までは定義に入れていなかったと言う。ところが、上の2例では、汎対角線までも(1)の性質が成り立っている。

各行/列に加えて汎対角線まで(1)の性質が成り立つこの方陣のことを、小生は珍重して、あえて「完全オイラー方陣」と呼びたいと思う。

もちろん、そういう方陣は、汎魔方陣の中にしか存在しない。

何故(1)の性質が重要かと言うと、この特徴があると、各行/列および汎対角線の5数の和が同一形式の計算式で計算できるからである。

$$\{0+1+2+3+4\} \times 5 + \{0+1+2+3+4\} = 10 \times 5 + 10 = 60(+)$$

{ } 内の順序は問わない。必ず各和が一定になることから、これで行列・汎対角線の定和が実現する。汎魔方陣の基本性質(約束)を実現するのである。

したがって、もし上の性質をもつ「グレコ・ラテン方陣」をみごとに構成できれば、その結果は必ず汎魔方陣になる。それは、上の定義によって論理的に保証される。

では、どうやって具体的に完全オイラー方陣を構成するか。そして、どうやって完全オイラー方陣の解の全数を計数すべきか？

2. 完全オイラー方陣の定義通りにプログラムするための二、三の工夫

ここからは、プログラミング・テクニックの解説になる。

言語は、それしか持ち合わせが無いので「C言語」にするが、殆ど「ミニ・パスカル」みたいな組み方をしているので、小生はさほど難解ではないつもりである。

(1) 次の基本図を用意しておく(汎魔方陣だから、空間を最小限拡張してある)。五進法分解図上位/下位の2面を先に作り、その各々について「ラテン方陣」を構成する。

24	25	21	22	23	24	25	21	22	23	4	4	4	4	4	4	4	4	4	4	4	4	4	4	3	4	0	1	2	3	4	0	1	2	
4	5	1	2	3	4	5	1	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	4	0	1	2	3	4	0	1	2
9	10	6	7	8	9	10	6	7	8	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3	4	0	1	2	3	4	0	1	2
14	15	11	12	13	14	15	11	12	13	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	4	0	1	2	3	4	0	1	2
19	20	16	17	18	19	20	16	17	18	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	4	0	1	2	3	4	0	1	2
24	25	21	22	23	24	25	21	22	23	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	3	4	0	1	2	3	4	0	1	2
4	5	1	2	3	4	5	1	2	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	4	0	1	2	3	4	0	1	2
[番地名]										[五進法分解図：上位面／下位面]																								

通常では、例えば、上の基本図の1番地に入る数値nの値を次々に変えたい時には、下図左側のようにプログラムを書くのだが、今回は、その作業を五進法の上位と下位に一々分けて行う。それを、例えば、右側のように書く。

```
// | //
for(n=0;n<25;n++){ | for(a=0;a<5;a++){ // a==d/5
  nm[1]=n; | for(b=0;b<5;b++){ // b==d%5
  // ...; | d=a*5+b; // 'd' stands for n itself
} | n[1]=d;
// | // ...;
| }
| }
| //
```

(2) 次にそのnの値が既に使われているかどうかを調べなければならないとする。

今までなら、使用登録のために用意した配列変数 `uflag[25]` 1本の状態を調べて確認すれば良かった。そしてあらためて自分が数値 `n` を使う時に、その数値の場所に `{ uflag[n]=1; }` と書き、「使用中」を登録する。使い終わったら、その時には `{ uflag[n]=0; }` と書いて、そのつど「未使用」状態に戻すのであった。

```
//
for(n=0;n<25;n++){
  if(uflag[n]==0){ // Check if the value N is used or not
    nm[3]=n; // If it is not used, then ...
    uflag[n]=1; // Set Used_flag of the value N
    nextproc(); // Go to the next procedure
    uflag[n]=0; // Reset Used_flag of the value N
  }
}
//
```

しかるに今回は、この作業を五進法上位/下位について別々に行う必要がある。

```
//
for(a=0;a<5;a++){
  if(uflag_high[a]==0){ // Check if it is used or not
    for(b=0;b<5;b++){
      if(uflag_low[b]==0){ // Check if it is used or not
        n=a*5+b; nm[3]=n; // Produce N from a and b ...
        if(uflag[n]==0){ // Check if the value N is used or not
          uflag[n]=1; // Set Used_flag of the value N
          uflag_high[a]=1; // Set Used_flag_high of Value a
          uflag_low[b]=1; // Set Used_flag_low of Value b
          nextproc(); // Go to the next procedure
          uflag_high[a]=0; // Reset Used_flag_high of Value a
          uflag_low[b]=0; // Reset Used_flag_low of Value b
          uflag[n]=0; // Reset Used_flag of Value N
        }
      }
    }
  }
}
//
```

3群の配列変数 `uflag[25]`, `uflag_high[5]`, `uflag_low[5]` を一々用意してそれを逐一調べる作業は、大変な仕事だ。我々には苦労だが、コンピュータは幸いにこういう単純作業の連続を得意にしている。命令した通りに高速に仕事をしてくれる。

その数値が「使用中」かそうでないかを調べる作業は重要で、省略することができない。「使用中」もしくは「未使用」の旗（フラグ）をそのつど立てることで、それを解決する。これをうまく使いこなさないと、`{ 0, 1, 2, 3, 4 }` の数値をそれぞれ1回しか使ってはいけないという「完全オイラー方陣」の厳しい制約に応えることができない。

では、そうしたフラグ配列を、全部で何本用意すべきか。

まず各行列について上位/下位の2本ずつ必要だから、次の20本が先ず必要になる。

```
//
short column1_high[5], column1_low[5], row1_high[5], row1_low[5];
short column2_high[5], column2_low[5], row2_high[5], row2_low[5];
short column3_high[5], column3_low[5], row3_high[5], row3_low[5];
short column4_high[5], column4_low[5], row4_high[5], row4_low[5];
short column5_high[5], column5_low[5], row5_high[5], row5_low[5];
//
```

同様に汎対角線10本についても、上位/下位の2本ずつ必要だ。これが20本。合計40本。

```
//
```

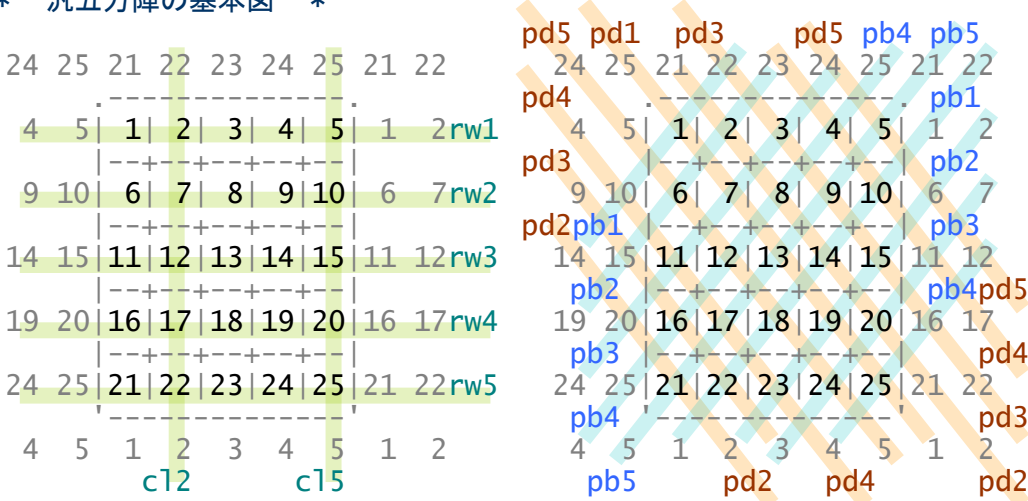
```

short pand1_high[5], pand1_low[5], panb1_high[5], panb1_low[5];
short pand2_high[5], pand2_low[5], panb2_high[5], panb2_low[5];
short pand3_high[5], pand3_low[5], panb3_high[5], panb3_low[5];
short pand4_high[5], pand4_low[5], panb4_high[5], panb4_low[5];
short pand5_high[5], pand5_low[5], panb5_high[5], panb5_low[5];
//

```

下にその場所を図示する。同じ場所に、上位/下位の区別を付けて2本ずつ立てる。

*** 汎五方陣の基本図 ***



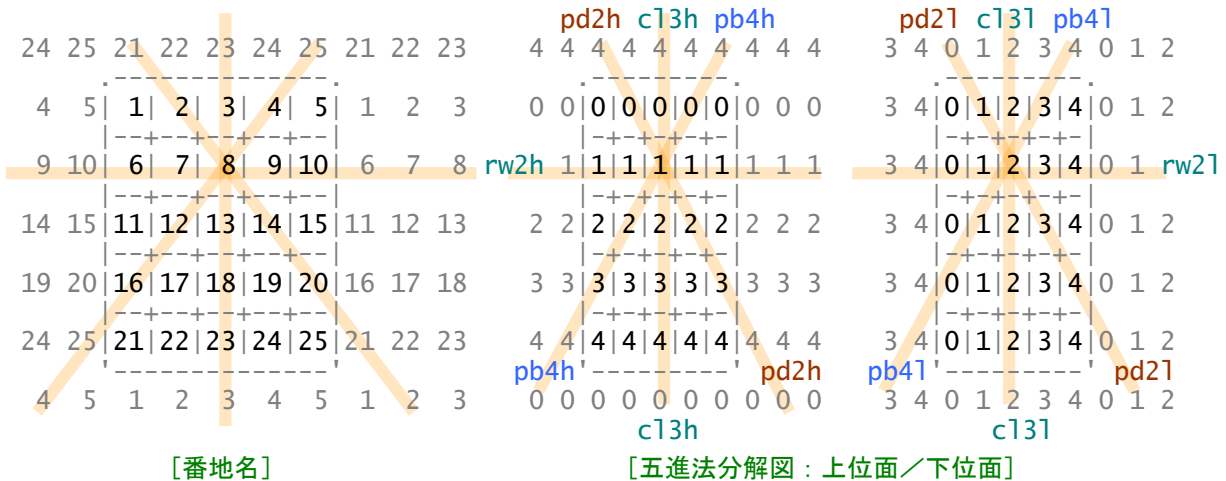
*** 基本条件式: C=60; ***

$$\begin{array}{ll}
 n1+n2+n3+n4+n5=C & \dots \text{rw1} \mid n1+n6+n11+n16+n21=C \dots c11 \\
 n6+n7+n8+n9+n10=C & \dots \text{rw2} \mid n2+n7+n12+n17+n22=C \dots c12 \\
 n11+n12+n13+n14+n15=C & \dots \text{rw3} \mid n3+n8+n13+n18+n23=C \dots c13 \\
 n16+n17+n18+n19+n20=C & \dots \text{rw4} \mid n4+n9+n14+n19+n24=C \dots c14 \\
 n21+n22+n23+n24+n25=C & \dots \text{rw5} \mid n5+n10+n15+n20+n25=C \dots c15
 \end{array}$$

*** 汎対角線条件式: C=60; ***

$$\begin{array}{ll}
 n1+n7+n13+n19+n25=C & \dots \text{pd1} \mid n1+n10+n14+n18+n22=C \dots \text{pb1} \\
 n2+n8+n14+n20+n21=C & \dots \text{pd2} \mid n2+n6+n15+n19+n23=C \dots \text{pb2} \\
 n3+n9+n15+n16+n22=C & \dots \text{pd3} \mid n3+n7+n11+n20+n24=C \dots \text{pb3} \\
 n4+n10+n11+n17+n23=C & \dots \text{pd4} \mid n4+n8+n12+n16+n25=C \dots \text{pb4} \\
 n5+n6+n12+n18+n24=C & \dots \text{pd5} \mid n5+n9+n13+n17+n21=C \dots \text{pb5}
 \end{array}$$

(3) では、1つの手続き中に何本のフラグ配列を調べるか。例えば、下図の8番地に数値7(+)を代入したい時に、どの配列のどこを調べるべきか？



[番地名]

[五進法分解図: 上位面/下位面]

7(+) = 1 × 5 + 2 だから、上位の8番地に1を、下位の8番地に2を入れたい。その1や

2が既に使われているかどうか、次の登録配列を調べに行く。

```
{c13h[1], c13l[2], rw2h[1], rw2l[2], pd2h[1], pd2l[2], pb4h[1], pb4l[2]}
```

そして uflg[7] も調べる。1つの手続き中で9個ものフラグを調べる。その全てについて「未使用」の数値を次々に使って行けば、何とか「完全オイラー方陣」を作れそうだ。

3. プログラムの組み方：その実際例

途中の説明は省略して、小生が組んだ実際のプログラムを観ていただこう。その方が、わかる人には、わかりが早かろう。取り急ぎ組んだテスト・プログラムなので、未整理・未清書の恨みが残るが、思考の流れを意識的に追跡していただきたい。特にフラグの組織的な利用法を念入りに観察していただきたい。

```
/** 'Complete Euler Squares' 5x5 of Pan-Diagonal Type **
/** File: 'CEuler5PD.c' Dictated by Kanji Setsuda **
/** on Sep. 9, 2003; Revised on Feb.16, 2007; **
/** Recompiled on Feb.14, 2015 **
/** with MacOSX 10.10.2 and Xcode 6.1.1 **
//
#include <stdio.h>
//
short int cnt, cnt1;
short LSM, cnt2, cnt3;
short nm[26], uflg[26];
short anm[7][26];
short c11h[6], rw1h[6], pd1h[6], pb1h[6];
short c11l[6], rw1l[6], pd1l[6], pb1l[6];
short c12h[6], rw2h[6], pd2h[6], pb2h[6];
short c12l[6], rw2l[6], pd2l[6], pb2l[6];
short c13h[6], rw3h[6], pd3h[6], pb3h[6];
short c13l[6], rw3l[6], pd3l[6], pb3l[6];
short c14h[6], rw4h[6], pd4h[6], pb4h[6];
short c14l[6], rw4l[6], pd4l[6], pb4l[6];
short c15h[6], rw5h[6], pd5h[6], pb5h[6];
short c15l[6], rw5l[6], pd5l[6], pb5l[6];
//
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 stp25(void), ansprint(void), pr3ans(void);
void pr2ans(void), pr1ans(void);
//
/** Main Program *
int main(){
short n;
printf("\n");
printf("*** 'Complete Euler Squares' of Order 5: Pan-Diagonal Type ***\n");
printf("** Compact List of the 3600 Standard Solutions **\n");
printf(" [Normalizing Inequality Conditions: n1<n25, n1<n5, n1<21 and n2>n6;]\n");
printf("\n");
for(n=0;n<26;n++){nm[n]=0; uflg[n]=0;}
for(n=0;n<6;n++){
c11h[n]=0; rw1h[n]=0; pd1h[n]=0; pb1h[n]=0;
c11l[n]=0; rw1l[n]=0; pd1l[n]=0; pb1l[n]=0;
c12h[n]=0; rw2h[n]=0; pd2h[n]=0; pb2h[n]=0;
c12l[n]=0; rw2l[n]=0; pd2l[n]=0; pb2l[n]=0;
c13h[n]=0; rw3h[n]=0; pd3h[n]=0; pb3h[n]=0;
c13l[n]=0; rw3l[n]=0; pd3l[n]=0; pb3l[n]=0;
c14h[n]=0; rw4h[n]=0; pd4h[n]=0; pb4h[n]=0;
c14l[n]=0; rw4l[n]=0; pd4l[n]=0; pb4l[n]=0;
c15h[n]=0; rw5h[n]=0; pd5h[n]=0; pb5h[n]=0;
```

```

    c15l[n]=0; rw5l[n]=0; pd5l[n]=0; pb5l[n]=0;}
LSM=60; cnt=0; cnt3=0;
stp01();    /* Begin the Calculations *
if((0<cnt3)&&(cnt3<2)){pr1ans();}
else if(cnt3>1){pr2ans();}
printf("\n");
printf(" [Total Solution Counts(n1=1/All) = %d/%d] OK!\n",cnt1,cnt);
printf("\n");
printf("*** Calculated and Listed by Kanji Setsuda on\n");
printf(" Feb.14, 2015 with MacOSX 10.10.2 and Xcode 6.1.1 **\n");
printf("\n");
return 0;
}
//
/* Begin the Calculations *
/* Set n1 *
void stp01(){
short a,b,d;
for(a=0;a<5;a++){
if((c11h[a]==0)&&(rw1h[a]==0)&&(pd1h[a]==0)&&(pb1h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if(uflg[d]==0){
if((c11l[b]==0)&&(rw1l[b]==0)&&(pd1l[b]==0)&&(pb1l[b]==0)){
nm[1]=d; uflg[d]=1; cnt2=0;
c11h[a]=1; rw1h[a]=1; pd1h[a]=1; pb1h[a]=1;
c11l[b]=1; rw1l[b]=1; pd1l[b]=1; pb1l[b]=1;
stp02();
c11h[a]=0; rw1h[a]=0; pd1h[a]=0; pb1h[a]=0;
c11l[b]=0; rw1l[b]=0; pd1l[b]=0; pb1l[b]=0;
uflg[d]=0;}}
}}
}
}
/* Set n25 & n1<n25 *
void stp02(){
short a,b,d;
for(a=4;a>=0;a--){
if((c15h[a]==0)&&(rw5h[a]==0)&&(pd1h[a]==0)&&(pb4h[a]==0)){
for(b=4;b>=0;b--){d=a*5+b;
if((nm[1]<d)&&(d<25)&&(uflg[d]==0)){
if((c15l[b]==0)&&(rw5l[b]==0)&&(pd1l[b]==0)&&(pb4l[b]==0)){
nm[25]=d; uflg[d]=1; if(nm[1]==0){cnt2=0;}
c15h[a]=1; rw5h[a]=1; pd1h[a]=1; pb4h[a]=1;
c15l[b]=1; rw5l[b]=1; pd1l[b]=1; pb4l[b]=1;
stp03();
c15h[a]=0; rw5h[a]=0; pd1h[a]=0; pb4h[a]=0;
c15l[b]=0; rw5l[b]=0; pd1l[b]=0; pb4l[b]=0;
uflg[d]=0;}}
}}
}
}
/* Set n7 *
void stp03(){
short a,b,d;
for(a=0;a<5;a++){
if((c12h[a]==0)&&(rw2h[a]==0)&&(pd1h[a]==0)&&(pb3h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if(uflg[d]==0){
if((c12l[b]==0)&&(rw2l[b]==0)&&(pd1l[b]==0)&&(pb3l[b]==0)){
nm[7]=d; uflg[d]=1;
c12h[a]=1; rw2h[a]=1; pd1h[a]=1; pb3h[a]=1;
c12l[b]=1; rw2l[b]=1; pd1l[b]=1; pb3l[b]=1;
stp04();
c12h[a]=0; rw2h[a]=0; pd1h[a]=0; pb3h[a]=0;
c12l[b]=0; rw2l[b]=0; pd1l[b]=0; pb3l[b]=0;

```

```

        uflg[d]=0;}}
    }}
}
}
}
/* Set n13 *
void stp04(){
short a,b,d;
for(a=0;a<5;a++){
if((c13h[a]==0)&&(rw3h[a]==0)&&(pd1h[a]==0)&&(pb5h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if(uflg[d]==0){
if((c13l[b]==0)&&(rw3l[b]==0)&&(pd1l[b]==0)&&(pb5l[b]==0)){
nm[13]=d; uflg[d]=1;
c13h[a]=1; rw3h[a]=1; pd1h[a]=1; pb5h[a]=1;
c13l[b]=1; rw3l[b]=1; pd1l[b]=1; pb5l[b]=1;
stp05();
c13h[a]=0; rw3h[a]=0; pd1h[a]=0; pb5h[a]=0;
c13l[b]=0; rw3l[b]=0; pd1l[b]=0; pb5l[b]=0;
uflg[d]=0;}}
}}
}
}
}
/* Set n19=LSM-n1-n7-n13-n25 *
void stp05(){
short a,b,d;
for(a=0;a<5;a++){
if((c14h[a]==0)&&(rw4h[a]==0)&&(pd1h[a]==0)&&(pb2h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if((nm[1]+nm[7]+nm[13]+d+nm[25]==LSM)&&(uflg[d]==0)){
if((c14l[b]==0)&&(rw4l[b]==0)&&(pd1l[b]==0)&&(pb2l[b]==0)){
nm[19]=d; uflg[d]=1;
c14h[a]=1; rw4h[a]=1; pd1h[a]=1; pb2h[a]=1;
c14l[b]=1; rw4l[b]=1; pd1l[b]=1; pb2l[b]=1;
stp06();
c14h[a]=0; rw4h[a]=0; pd1h[a]=0; pb2h[a]=0;
c14l[b]=0; rw4l[b]=0; pd1l[b]=0; pb2l[b]=0;
uflg[d]=0;}}
}}
}
}
}
/* Set n5 & n1<n5 *
void stp06(){
short a,b,d;
for(a=0;a<5;a++){
if((c15h[a]==0)&&(rw1h[a]==0)&&(pd5h[a]==0)&&(pb5h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if((d>nm[1])&&(uflg[d]==0)){
if((c15l[b]==0)&&(rw1l[b]==0)&&(pd5l[b]==0)&&(pb5l[b]==0)){
nm[5]=d; uflg[d]=1;
c15h[a]=1; rw1h[a]=1; pd5h[a]=1; pb5h[a]=1;
c15l[b]=1; rw1l[b]=1; pd5l[b]=1; pb5l[b]=1;
stp07();
c15h[a]=0; rw1h[a]=0; pd5h[a]=0; pb5h[a]=0;
c15l[b]=0; rw1l[b]=0; pd5l[b]=0; pb5l[b]=0;
uflg[d]=0;}}
}}
}
}
}
/* Set n21 & n1<n21 *
void stp07(){
short a,b,d;
for(a=4;a>=0;a--){
if((c11h[a]==0)&&(rw5h[a]==0)&&(pd2h[a]==0)&&(pb5h[a]==0)){
for(b=4;b>=0;b--){d=a*5+b;
if((nm[1]<d)&&(uflg[d]==0)){

```

```

        if((cl1l[b]==0)&&(rw5l[b]==0)&&(pd2l[b]==0)&&(pb5l[b]==0)){
            nm[21]=d; uflg[d]=1;
            cl1h[a]=1; rw5h[a]=1; pd2h[a]=1; pb5h[a]=1;
            cl1l[b]=1; rw5l[b]=1; pd2l[b]=1; pb5l[b]=1;
            stp08C);
            cl1h[a]=0; rw5h[a]=0; pd2h[a]=0; pb5h[a]=0;
            cl1l[b]=0; rw5l[b]=0; pd2l[b]=0; pb5l[b]=0;
            uflg[d]=0;}}
    }}
}
}
}
/* Set n9 *
void stp08C){
    short a,b,d;
    for(a=0;a<5;a++){
        if((cl4h[a]==0)&&(rw2h[a]==0)&&(pd3h[a]==0)&&(pb5h[a]==0)){
            for(b=0;b<5;b++){d=a*5+b;
                if(uflg[d]==0){
                    if((cl4l[b]==0)&&(rw2l[b]==0)&&(pd3l[b]==0)&&(pb5l[b]==0)){
                        nm[9]=d; uflg[d]=1;
                        cl4h[a]=1; rw2h[a]=1; pd3h[a]=1; pb5h[a]=1;
                        cl4l[b]=1; rw2l[b]=1; pd3l[b]=1; pb5l[b]=1;
                        stp09C);
                        cl4h[a]=0; rw2h[a]=0; pd3h[a]=0; pb5h[a]=0;
                        cl4l[b]=0; rw2l[b]=0; pd3l[b]=0; pb5l[b]=0;
                        uflg[d]=0;}}
                }}
            }
        }
}
/* Set n17=LSM-n5-n9-n13-n21 *
void stp09C){
    short a,b,d;
    for(a=0;a<5;a++){
        if((cl2h[a]==0)&&(rw4h[a]==0)&&(pd4h[a]==0)&&(pb5h[a]==0)){
            for(b=0;b<5;b++){d=a*5+b;
                if((nm[5]+nm[9]+nm[13]+d+nm[21]==LSM)&&(uflg[d]==0)){
                    if((cl2l[b]==0)&&(rw4l[b]==0)&&(pd4l[b]==0)&&(pb5l[b]==0)){
                        nm[17]=d; uflg[d]=1;
                        cl2h[a]=1; rw4h[a]=1; pd4h[a]=1; pb5h[a]=1;
                        cl2l[b]=1; rw4l[b]=1; pd4l[b]=1; pb5l[b]=1;
                        stp10C);
                        cl2h[a]=0; rw4h[a]=0; pd4h[a]=0; pb5h[a]=0;
                        cl2l[b]=0; rw4l[b]=0; pd4l[b]=0; pb5l[b]=0;
                        uflg[d]=0;}}
                    }}
                }
            }
        }
}
}
}
/* Level #2: *
/* Set n2 *
void stp10C){
    short a,b,d;
    for(a=0;a<5;a++){
        if((cl2h[a]==0)&&(rw1h[a]==0)&&(pd2h[a]==0)&&(pb2h[a]==0)){
            for(b=0;b<5;b++){d=a*5+b;
                if(uflg[d]==0){
                    if((cl2l[b]==0)&&(rw1l[b]==0)&&(pd2l[b]==0)&&(pb2l[b]==0)){
                        nm[2]=d; uflg[d]=1;
                        cl2h[a]=1; rw1h[a]=1; pd2h[a]=1; pb2h[a]=1;
                        cl2l[b]=1; rw1l[b]=1; pd2l[b]=1; pb2l[b]=1;
                        stp11C);
                        cl2h[a]=0; rw1h[a]=0; pd2h[a]=0; pb2h[a]=0;
                        cl2l[b]=0; rw1l[b]=0; pd2l[b]=0; pb2l[b]=0;
                        uflg[d]=0;}}
                    }}
                }
            }
        }
}
}
}

```



```

}
/* Set n3 *
void stp11(){
short a,b,d;
for(a=0;a<5;a++){
if((cl3h[a]==0)&&(rw1h[a]==0)&&(pd3h[a]==0)&&(pb3h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if(uflg[d]==0){
if((cl3l[b]==0)&&(rw1l[b]==0)&&(pd3l[b]==0)&&(pb3l[b]==0)){
nm[3]=d; uflg[d]=1;
cl3h[a]=1; rw1h[a]=1; pd3h[a]=1; pb3h[a]=1;
cl3l[b]=1; rw1l[b]=1; pd3l[b]=1; pb3l[b]=1;
stp12();
cl3h[a]=0; rw1h[a]=0; pd3h[a]=0; pb3h[a]=0;
cl3l[b]=0; rw1l[b]=0; pd3l[b]=0; pb3l[b]=0;
uflg[d]=0;}}
}}
}
}
/* Set n4=LSM-n1-n2-n3-n5 *
void stp12(){
short a,b,d;
for(a=0;a<5;a++){
if((cl4h[a]==0)&&(rw1h[a]==0)&&(pd4h[a]==0)&&(pb4h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if((nm[1]+nm[2]+nm[3]+d+nm[5]==LSM)&&(uflg[d]==0)){
if((cl4l[b]==0)&&(rw1l[b]==0)&&(pd4l[b]==0)&&(pb4l[b]==0)){
nm[4]=d; uflg[d]=1;
cl4h[a]=1; rw1h[a]=1; pd4h[a]=1; pb4h[a]=1;
cl4l[b]=1; rw1l[b]=1; pd4l[b]=1; pb4l[b]=1;
stp13();
cl4h[a]=0; rw1h[a]=0; pd4h[a]=0; pb4h[a]=0;
cl4l[b]=0; rw1l[b]=0; pd4l[b]=0; pb4l[b]=0;
uflg[d]=0;}}
}}
}
}
/* Set n6 & n2>n6 *
void stp13(){
short a,b,d;
for(a=0;a<5;a++){
if((cl1h[a]==0)&&(rw2h[a]==0)&&(pd5h[a]==0)&&(pb2h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if((d<nm[2])&&(uflg[d]==0)){
if((cl1l[b]==0)&&(rw2l[b]==0)&&(pd5l[b]==0)&&(pb2l[b]==0)){
nm[6]=d; uflg[d]=1;
cl1h[a]=1; rw2h[a]=1; pd5h[a]=1; pb2h[a]=1;
cl1l[b]=1; rw2l[b]=1; pd5l[b]=1; pb2l[b]=1;
stp14();
cl1h[a]=0; rw2h[a]=0; pd5h[a]=0; pb2h[a]=0;
cl1l[b]=0; rw2l[b]=0; pd5l[b]=0; pb2l[b]=0;
uflg[d]=0;}}
}}
}
}
/* Set n11 *
void stp14(){
short a,b,d;
for(a=0;a<5;a++){
if((cl1h[a]==0)&&(rw3h[a]==0)&&(pd4h[a]==0)&&(pb3h[a]==0)){
for(b=0;b<5;b++){d=a*5+b;
if(uflg[d]==0){
if((cl1l[b]==0)&&(rw3l[b]==0)&&(pd4l[b]==0)&&(pb3l[b]==0)){
nm[11]=d; uflg[d]=1;
cl1h[a]=1; rw3h[a]=1; pd4h[a]=1; pb3h[a]=1;

```

```

        c11l[b]=1; rw3l[b]=1; pd4l[b]=1; pb3l[b]=1;
        stp15();
        c11h[a]=0; rw3h[a]=0; pd4h[a]=0; pb3h[a]=0;
        c11l[b]=0; rw3l[b]=0; pd4l[b]=0; pb3l[b]=0;
        uflg[d]=0;}}
    }}
}
}
/* Set n16=LSM-n1-n6-n11-n21 *
void stp15(){
    short a,b,d;
    for(a=0;a<5;a++){
        if((c11h[a]==0)&&(rw4h[a]==0)&&(pd3h[a]==0)&&(pb4h[a]==0)){
            for(b=0;b<5;b++){d=a*5+b;
                if((nm[1]+nm[6]+nm[11]+d+nm[21]==LSM)&&(uflg[d]==0)){
                    if((c11l[b]==0)&&(rw4l[b]==0)&&(pd3l[b]==0)&&(pb4l[b]==0)){
                        nm[16]=d; uflg[d]=1;
                        c11h[a]=1; rw4h[a]=1; pd3h[a]=1; pb4h[a]=1;
                        c11l[b]=1; rw4l[b]=1; pd3l[b]=1; pb4l[b]=1;
                        stp16();
                        c11h[a]=0; rw4h[a]=0; pd3h[a]=0; pb4h[a]=0;
                        c11l[b]=0; rw4l[b]=0; pd3l[b]=0; pb4l[b]=0;
                        uflg[d]=0;}}
                    }}
            }
        }
}
/* Set n8 *
void stp16(){
    short a,b,d;
    for(a=0;a<5;a++){
        if((c13h[a]==0)&&(rw2h[a]==0)&&(pd2h[a]==0)&&(pb4h[a]==0)){
            for(b=0;b<5;b++){d=a*5+b;
                if(uflg[d]==0){
                    if((c13l[b]==0)&&(rw2l[b]==0)&&(pd2l[b]==0)&&(pb4l[b]==0)){
                        nm[8]=d; uflg[d]=1;
                        c13h[a]=1; rw2h[a]=1; pd2h[a]=1; pb4h[a]=1;
                        c13l[b]=1; rw2l[b]=1; pd2l[b]=1; pb4l[b]=1;
                        stp17();
                        c13h[a]=0; rw2h[a]=0; pd2h[a]=0; pb4h[a]=0;
                        c13l[b]=0; rw2l[b]=0; pd2l[b]=0; pb4l[b]=0;
                        uflg[d]=0;}}
                    }}
            }
        }
}
/* Set n12=LSM-n4-n8-n16-n25 *
void stp17(){
    short a,b,d;
    for(a=0;a<5;a++){
        if((c12h[a]==0)&&(rw3h[a]==0)&&(pd5h[a]==0)&&(pb4h[a]==0)){
            for(b=0;b<5;b++){d=a*5+b;
                if((nm[4]+nm[8]+d+nm[16]+nm[25]==LSM)&&(uflg[d]==0)){
                    if((c12l[b]==0)&&(rw3l[b]==0)&&(pd5l[b]==0)&&(pb4l[b]==0)){
                        nm[12]=d; uflg[d]=1;
                        c12h[a]=1; rw3h[a]=1; pd5h[a]=1; pb4h[a]=1;
                        c12l[b]=1; rw3l[b]=1; pd5l[b]=1; pb4l[b]=1;
                        stp18();
                        c12h[a]=0; rw3h[a]=0; pd5h[a]=0; pb4h[a]=0;
                        c12l[b]=0; rw3l[b]=0; pd5l[b]=0; pb4l[b]=0;
                        uflg[d]=0;}}
                    }}
            }
        }
}
/* Set n10=LSM-n6-n7-n8-n9 *
void stp18(){
    short a,b,d;

```

```

for(a=0;a<5;a++){
  if((c15h[a]==0)&&(rw2h[a]==0)&&(pd4h[a]==0)&&(pb1h[a]==0)){
    for(b=0;b<5;b++){d=a*5+b;
      if((nm[6]+nm[7]+nm[8]+nm[9]+d==LSM)&&(uflg[d]==0)){
        if((c15l[b]==0)&&(rw2l[b]==0)&&(pd4l[b]==0)&&(pb1l[b]==0)){
          nm[10]=d; uflg[d]=1;
          c15h[a]=1; rw2h[a]=1; pd4h[a]=1; pb1h[a]=1;
          c15l[b]=1; rw2l[b]=1; pd4l[b]=1; pb1l[b]=1;
          stp19C);
          c15h[a]=0; rw2h[a]=0; pd4h[a]=0; pb1h[a]=0;
          c15l[b]=0; rw2l[b]=0; pd4l[b]=0; pb1l[b]=0;
          uflg[d]=0;}}
        }}
    }
}
}
/* Set n22=LSM-n2-n7-n12-n17 */
void stp19C){
short a,b,d;
for(a=0;a<5;a++){
  if((c12h[a]==0)&&(rw5h[a]==0)&&(pd3h[a]==0)&&(pb1h[a]==0)){
    for(b=0;b<5;b++){d=a*5+b;
      if((nm[2]+nm[7]+nm[12]+nm[17]+d==LSM)&&(uflg[d]==0)){
        if((c12l[b]==0)&&(rw5l[b]==0)&&(pd3l[b]==0)&&(pb1l[b]==0)){
          nm[22]=d; uflg[d]=1;
          c12h[a]=1; rw5h[a]=1; pd3h[a]=1; pb1h[a]=1;
          c12l[b]=1; rw5l[b]=1; pd3l[b]=1; pb1l[b]=1;
          stp20C);
          c12h[a]=0; rw5h[a]=0; pd3h[a]=0; pb1h[a]=0;
          c12l[b]=0; rw5l[b]=0; pd3l[b]=0; pb1l[b]=0;
          uflg[d]=0;}}
        }}
    }
}
}
/* Level #3: */
/* Set n23=LSM-n4-n10-n11-n17 */
void stp20C){
short a,b,d;
for(a=0;a<5;a++){
  if((c13h[a]==0)&&(rw5h[a]==0)&&(pd4h[a]==0)&&(pb2h[a]==0)){
    for(b=0;b<5;b++){d=a*5+b;
      if((nm[4]+nm[10]+nm[11]+nm[17]+d==LSM)&&(uflg[d]==0)){
        if((c13l[b]==0)&&(rw5l[b]==0)&&(pd4l[b]==0)&&(pb2l[b]==0)){
          nm[23]=d; uflg[d]=1;
          c13h[a]=1; rw5h[a]=1; pd4h[a]=1; pb2h[a]=1;
          c13l[b]=1; rw5l[b]=1; pd4l[b]=1; pb2l[b]=1;
          stp21C);
          c13h[a]=0; rw5h[a]=0; pd4h[a]=0; pb2h[a]=0;
          c13l[b]=0; rw5l[b]=0; pd4l[b]=0; pb2l[b]=0;
          uflg[d]=0;}}
        }}
    }
}
}
/* Set n18=LSM-n3-n8-n13-n23 */
void stp21C){
short a,b,d;
for(a=0;a<5;a++){
  if((c13h[a]==0)&&(rw4h[a]==0)&&(pd5h[a]==0)&&(pb1h[a]==0)){
    for(b=0;b<5;b++){d=a*5+b;
      if((nm[3]+nm[8]+nm[13]+d+nm[23]==LSM)&&(uflg[d]==0)){
        if((c13l[b]==0)&&(rw4l[b]==0)&&(pd5l[b]==0)&&(pb1l[b]==0)){
          nm[18]=d; uflg[d]=1;
          c13h[a]=1; rw4h[a]=1; pd5h[a]=1; pb1h[a]=1;
          c13l[b]=1; rw4l[b]=1; pd5l[b]=1; pb1l[b]=1;
          stp22C);
          c13h[a]=0; rw4h[a]=0; pd5h[a]=0; pb1h[a]=0;

```

```

        c13l[b]=0; rw4l[b]=0; pd5l[b]=0; pb1l[b]=0;
        uflg[d]=0;}}
    }}
}
}
/** Set n14=LSM-n1-n10-n18-n22 *
void stp22(){
short a,b,d;
for(a=0;a<5;a++){
    if((c14h[a]==0)&&(rw3h[a]==0)&&(pd2h[a]==0)&&(pb1h[a]==0)){
        for(b=0;b<5;b++){d=a*5+b;
            if((nm[1]+nm[10]+d+nm[18]+nm[22]==LSM)&&(uflg[d]==0)){
                if((c14l[b]==0)&&(rw3l[b]==0)&&(pd2l[b]==0)&&(pb1l[b]==0)){
                    nm[14]=d; uflg[d]=1;
                    c14h[a]=1; rw3h[a]=1; pd2h[a]=1; pb1h[a]=1;
                    c14l[b]=1; rw3l[b]=1; pd2l[b]=1; pb1l[b]=1;
                    stp23();
                    c14h[a]=0; rw3h[a]=0; pd2h[a]=0; pb1h[a]=0;
                    c14l[b]=0; rw3l[b]=0; pd2l[b]=0; pb1l[b]=0;
                    uflg[d]=0;}}
            }}
        }
    }
}
/** Set n15=LSM-n3-n9-n16-n22 *
void stp23(){
short a,b,d;
for(a=0;a<5;a++){
    if((c15h[a]==0)&&(rw3h[a]==0)&&(pd3h[a]==0)&&(pb2h[a]==0)){
        for(b=0;b<5;b++){d=a*5+b;
            if((nm[3]+nm[9]+d+nm[16]+nm[22]==LSM)&&(uflg[d]==0)){
                if((c15l[b]==0)&&(rw3l[b]==0)&&(pd3l[b]==0)&&(pb2l[b]==0)){
                    nm[15]=d; uflg[d]=1;
                    c15h[a]=1; rw3h[a]=1; pd3h[a]=1; pb2h[a]=1;
                    c15l[b]=1; rw3l[b]=1; pd3l[b]=1; pb2l[b]=1;
                    stp24();
                    c15h[a]=0; rw3h[a]=0; pd3h[a]=0; pb2h[a]=0;
                    c15l[b]=0; rw3l[b]=0; pd3l[b]=0; pb2l[b]=0;
                    uflg[d]=0;}}
            }}
        }
    }
}
/** Set n20=LSM-n2-n8-n14-n21 *
void stp24(){
short a,b,d;
for(a=0;a<5;a++){
    if((c15h[a]==0)&&(rw4h[a]==0)&&(pd2h[a]==0)&&(pb3h[a]==0)){
        for(b=0;b<5;b++){d=a*5+b;
            if((nm[2]+nm[8]+nm[14]+d+nm[21]==LSM)&&(uflg[d]==0)){
                if((c15l[b]==0)&&(rw4l[b]==0)&&(pd2l[b]==0)&&(pb3l[b]==0)){
                    nm[20]=d; uflg[d]=1;
                    c15h[a]=1; rw4h[a]=1; pd2h[a]=1; pb3h[a]=1;
                    c15l[b]=1; rw4l[b]=1; pd2l[b]=1; pb3l[b]=1;
                    stp25();
                    c15h[a]=0; rw4h[a]=0; pd2h[a]=0; pb3h[a]=0;
                    c15l[b]=0; rw4l[b]=0; pd2l[b]=0; pb3l[b]=0;
                    uflg[d]=0;}}
            }}
        }
    }
}
}
/** Set n24=LSM-n3-n7-n11-n20 *
void stp25(){
short a,b,d;
for(a=0;a<5;a++){
    if((c14h[a]==0)&&(rw5h[a]==0)&&(pd5h[a]==0)&&(pb3h[a]==0)){
        for(b=0;b<5;b++){d=a*5+b;

```

```

        if((nm[3]+nm[7]+nm[11]+nm[20]+d==LSM)&&(uflg[d]==0)){
            if((cl4l[b]==0)&&(rw5l[b]==0)&&(pd5l[b]==0)&&(pb3l[b]==0)){
                nm[24]=d; uflg[d]=1;
                cl4h[a]=1; rw5h[a]=1; pd5h[a]=1; pb3h[a]=1;
                cl4l[b]=1; rw5l[b]=1; pd5l[b]=1; pb3l[b]=1;
                ansprint();
                cl4h[a]=0; rw5h[a]=0; pd5h[a]=0; pb3h[a]=0;
                cl4l[b]=0; rw5l[b]=0; pd5l[b]=0; pb3l[b]=0;
                uflg[d]=0;}}
        }}
    }
}
//
/** Print the Answers *
void ansprint(){
    short n;
    cnt++; if(nm[1]==1){cnt1++;}
    cnt2++;
    if(cnt2==1){
        anm[cnt3*2][0]=cnt;
        for(n=1;n<26;n++){anm[cnt3*2][n]=nm[n]+1; anm[cnt3*2+1][n]=nm[n];}
        cnt3++;
        if(cnt3==3){pr3ans(); cnt3=0;
            anm[2][0]=0; anm[4][0]=0;
            for(n=1;n<26;n++){anm[2][n]=0; anm[3][n]=0;}
        }
    }
}
//
/** Print 3 Answers *
void pr3ans(){
    short l,15,n;
    printf(" /D5i%22d# /D5i%22d# /D5i%22d#\n",anm[0][0],anm[2][0],anm[4][0]);
    for(l=0;l<5;l++){l5=l*5;
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[1][l5+n])/5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[1][l5+n])%5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%3d", anm[0][l5+n]);}
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[3][l5+n])/5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[3][l5+n])%5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%3d", anm[2][l5+n]);}
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[5][l5+n])/5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[5][l5+n])%5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%3d", anm[4][l5+n]);}
        printf("\n");
    }
}
//
/** Print 2 Answers *
void pr2ans(){
    short l,15,n;
    printf(" /D5i%22d# /D5i%22d#\n",anm[0][0],anm[2][0]);
    for(l=0;l<5;l++){l5=l*5;
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[1][l5+n])/5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%d", (anm[1][l5+n])%5);}

```

```

printf(" ");
for(n=1;n<6;n++){printf("%3d",anm[0][15+n]);}
printf(" ");
for(n=1;n<6;n++){printf("%d", (anm[3][15+n])/5);}
printf(" ");
for(n=1;n<6;n++){printf("%d", (anm[3][15+n])%5);}
printf(" ");
for(n=1;n<6;n++){printf("%3d",anm[2][15+n]);}
printf("\n");
}
}
//
/* Print 1 Answer *
void pr1ans(){
short l,15,n;
printf(" /D5i%22d#\n",anm[0][0]);
for(l=0;l<5;l++){l5=l*5;
printf(" ");
for(n=1;n<6;n++){printf("%d", (anm[1][15+n])/5);}
printf(" ");
for(n=1;n<6;n++){printf("%d", (anm[1][15+n])%5);}
printf(" ");
for(n=1;n<6;n++){printf("%3d",anm[0][15+n]);}
printf("\n");
}
}
//

```

4. 計算結果のリスト

早速にも、このプログラムの実行結果を示そう。下のリストがそれである。

```

* 完全オイラー型の汎対角線五方陣の標準解のリスト (抜粋) *
/D5i 1# /D5i 37# /D5i 73#
04321 02413 1 23 20 12 9 04321 02314 1 23 19 12 10 04321 03214 1 24 18 12 10
21043 41302 15 7 4 21 18 21043 31402 14 7 5 21 18 21043 21403 13 7 5 21 19
43210 30241 24 16 13 10 2 43210 40231 25 16 13 9 2 43210 40321 25 16 14 8 2
10432 24130 8 5 22 19 11 10432 23140 8 4 22 20 11 10432 32140 9 3 22 20 11
32104 13024 17 14 6 3 25 32104 14023 17 15 6 3 24 32104 14032 17 15 6 4 23

/D5i 109# /D5i 145# /D5i 181#
04321 03124 1 24 17 13 10 03421 02413 1 18 25 12 9 03421 02314 1 18 24 12 10
21043 12403 12 8 5 21 19 21034 41302 15 7 4 16 23 21034 31402 14 7 5 16 23
43210 40312 25 16 14 7 3 34210 30241 19 21 13 10 2 34210 40231 20 21 13 9 2
10432 31240 9 2 23 20 11 10342 24130 8 5 17 24 11 10342 23140 8 4 17 25 11
32104 24031 18 15 6 4 22 42103 13024 22 14 6 3 20 42103 14023 22 15 6 3 19

/D5i 217# /D5i 253# /D5i 289#
03421 03214 1 19 23 12 10 03421 03124 1 19 22 13 10 03214 04321 1 20 14 8 22
21034 21403 13 7 5 16 24 21034 12403 12 8 5 16 24 21403 21043 13 7 21 5 19
34210 40321 20 21 14 8 2 34210 40312 20 21 14 7 3 40321 43210 25 4 18 12 6
10342 32140 9 3 17 25 11 10342 31240 9 2 18 25 11 32140 10432 17 11 10 24 3
42103 14032 22 15 6 4 18 42103 24031 23 15 6 4 17 14032 32104 9 23 2 16 15

/D5i 325# /D5i 361# /D5i 397#
03214 03421 1 19 15 8 22 03214 02431 1 18 15 9 22 03214 01432 1 17 15 9 23
21403 21034 13 7 21 4 20 21403 31024 14 7 21 3 20 21403 32014 14 8 21 2 20
40321 34210 24 5 18 12 6 40321 24310 23 5 19 12 6 40321 14320 22 5 19 13 6
32140 10342 17 11 9 25 3 32140 10243 17 11 8 25 4 32140 20143 18 11 7 25 4
14032 42103 10 23 2 16 14 14032 43102 10 24 2 16 13 14032 43201 10 24 3 16 12

/D5i 433# /D5i 469# /D5i 505#
03124 04321 1 20 9 13 22 03124 03421 1 19 10 13 22 03124 02431 1 18 10 14 22
12403 21043 8 12 21 5 19 12403 21034 8 12 21 4 20 12403 31024 9 12 21 3 20
40312 43210 25 4 18 7 11 40312 34210 24 5 18 7 11 40312 24310 23 5 19 7 11
31240 10432 17 6 15 24 3 31240 10342 17 6 14 25 3 31240 10243 17 6 13 25 4
24031 32104 14 23 2 16 10 24031 42103 15 23 2 16 9 24031 43102 15 24 2 16 8

```

/D5i	541#	/D5i	577#	/D5i	1153#
03124 01432	1 17 10 14 23	04321 12403	2 23 20 11 9	04321 21403	3 22 20 11 9
12403 32014	9 13 21 2 20	21043 40312	15 6 4 22 18	21043 40321	15 6 4 23 17
40312 14320	22 5 19 8 11	43210 31240	24 17 13 10 1	43210 32140	24 18 12 10 1
31240 20143	18 6 12 25 4	10432 24031	8 5 21 19 12	10432 14032	7 5 21 19 13
24031 43201	15 24 3 16 7	32104 03124	16 14 7 3 25	32104 03214	16 14 8 2 25
/D5i	1729#	/D5i	2305#	/D5i	2881#
04321 31402	4 22 20 11 8	04321 41302	5 22 19 11 8	14302 02413	6 23 20 2 14
21043 40231	15 6 3 24 17	21043 30241	14 6 3 25 17	02143 41302	5 12 9 21 18
43210 23140	23 19 12 10 1	43210 24130	23 20 12 9 1	43021 30241	24 16 3 15 7
10432 14023	7 5 21 18 14	10432 13024	7 4 21 18 15	21430 24130	13 10 22 19 1
32104 02314	16 13 9 2 25	32104 02413	16 13 10 2 24	30214 13024	17 4 11 8 25
/D5i	3025#	/D5i	3169#	/D5i	3313#
14302 12403	7 23 20 1 14	14302 21403	8 22 20 1 14	14302 31402	9 22 20 1 13
02143 40312	5 11 9 22 18	02143 40321	5 11 9 23 17	02143 40231	5 11 8 24 17
43021 31240	24 17 3 15 6	43021 32140	24 18 2 15 6	43021 23140	23 19 2 15 6
21430 24031	13 10 21 19 2	21430 14032	12 10 21 19 3	21430 14023	12 10 21 18 4
30214 03124	16 4 12 8 25	30214 03214	16 4 13 7 25	30214 02314	16 3 14 7 25
/D5i	3457#				
14302 41302	10 22 19 1 13				
02143 30241	4 11 8 25 17				
43021 24130	23 20 2 14 6				
21430 13024	12 9 21 18 5				
30214 02413	16 3 15 7 24				

[Total Solution Counts(n1=1/All) = 576/3600] OK!

** Calculated and Listed by Kanji Setsuda on
Feb.14, 2015 with MacOSX 10.10.2 and Xcode 6.1.1 **

解は、全部で3600個得られた。以前に計数した汎五方陣(「鈴木方陣」)の解集合と完全に一致する。

これは、何を意味するだろうか。

我々のプログラムが正しかったことが確かめられただけではない。我々が作ろうとしたのは最初から「完全オイラー方陣」だったから、これで汎五方陣の解は全て完全オイラー方陣であると決定したことになる。このことは前からわかっていたが、別の方法でも確かめられたわけである。

それとは別に、実験中に体感したのは、このプログラムの実行スピードである。非常に速い。調べる場合の数が、途中の段階でも、良く絞り込まれているようだ。無駄なく核心に迫るように感じる。

5. 「完全オイラー方陣」を作る：その2：対称汎五方陣の構成

もう1つの「鈴木方陣」である連立型の対称汎五方陣を、この際「完全オイラー方陣」として構成することにチャレンジしてみよう。

連立型は、対称型でもあるために「24の補数ペア」12組：{(0, 24), (1, 23), (2, 22), ..., (10, 14), (11, 13)} が、点対称に配置される。数値12は、必ず13番地に置かれる。

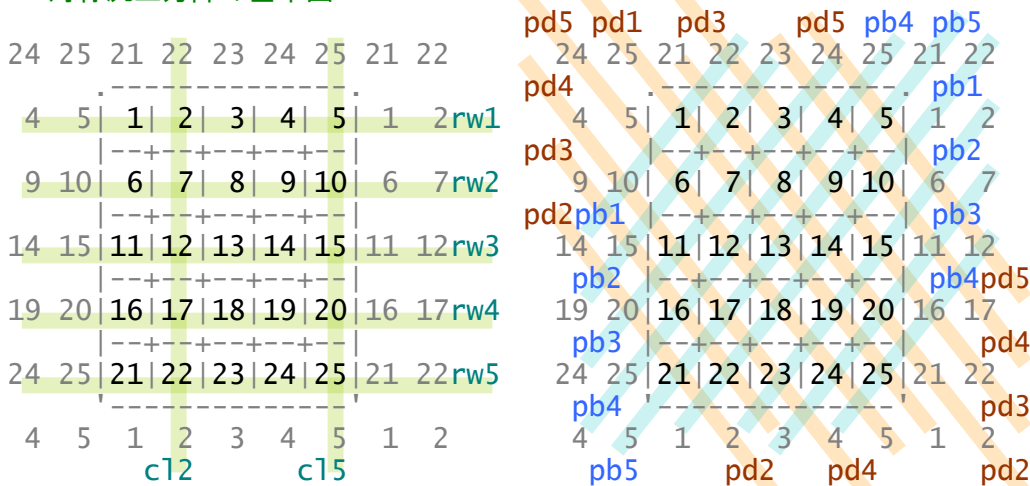
n1が決まれば、n25は $n_{25}=24-n_1$ と決まる。n2が決まれば、n24= $24-n_2$ も決まる。n3が決まれば、n23も自動的に $n_{23}=24-n_3$ と決まる。...

1つの手続きの中で(n1, n25), (n2, n24), (n3, n23), ..., (n10, n14)などのペアを同時に決めて行く。こうすると、手続きの総数は半分になるが、1つの手続きの中身は倍の長さになる。複雑になるので、慎重に書き続けなければならない。

プログラムの実際は、例えば小生ならば次のように書く。

(後々もう少し洗練されたプログラムの書き方を開発したが、今はあえて当初のスタイルをここに書き留めておきたい。出発点の記念碑としての意味をこめて...)

* 対称汎五方陣の基本図 *



* 基本条件式: C=60; *

$$\begin{array}{ll}
 n1+n2+n3+n4+n5=C & \dots \text{rw1} \mid n1+n6+n11+n16+n21=C \dots \text{c1} \\
 n6+n7+n8+n9+n10=C & \dots \text{rw2} \mid n2+n7+n12+n17+n22=C \dots \text{c2} \\
 n11+n12+n13+n14+n15=C & \dots \text{rw3} \mid n3+n8+n13+n18+n23=C \dots \text{c3} \\
 n16+n17+n18+n19+n20=C & \dots \text{rw4} \mid n4+n9+n14+n19+n24=C \dots \text{c4} \\
 n21+n22+n23+n24+n25=C & \dots \text{rw5} \mid n5+n10+n15+n20+n25=C \dots \text{c5}
 \end{array}$$

* 汎対角線条件式: C=60; *

$$\begin{array}{ll}
 n1+n7+n13+n19+n25=C & \dots \text{pd1} \mid n1+n10+n14+n18+n22=C \dots \text{pb1} \\
 n2+n8+n14+n20+n21=C & \dots \text{pd2} \mid n2+n6+n15+n19+n23=C \dots \text{pb2} \\
 n3+n9+n15+n16+n22=C & \dots \text{pd3} \mid n3+n7+n11+n20+n24=C \dots \text{pb3} \\
 n4+n10+n11+n17+n23=C & \dots \text{pd4} \mid n4+n8+n12+n16+n25=C \dots \text{pb4} \\
 n5+n6+n12+n18+n24=C & \dots \text{pd5} \mid n5+n9+n13+n17+n21=C \dots \text{pb5}
 \end{array}$$

* 補数ペアの対称配置: CC=24 *

$$\begin{aligned}
 &n1+n25=n2+n24=n3+n23=n4+n22=n5+n21=n6+n20=n7+n19 \\
 &=n8+n18=n9+n17=n10+n16=n11+n15=n12+n14=n13+n13=CC \dots (\text{sc})
 \end{aligned}$$

* リスト整形出力用の不等式 *

$$n1 < n25; n1 < n5; n1 < n21; \text{ and } n2 > n6;$$

* 完全オイラー型の対称汎五方陣を構成するプログラム *

```

/** Compose 'Complete Euler Squares' of Order 5: **
/** Simultaneous Type: Both Self-Complementary and Pan-Diagonal **
/** File: 'CEuler5Sml.c' Dictated by Kanji Setsuda in 2003; **
/** Revised on Feb.16, 2007; Recompiled on Feb.14, 2015 **
/** working with MacOSX 10.10.2 and Xcode 6.1.1; **
//
#include <stdio.h>
//
short int cnt, cnt1, cnt3;
short LSM, CC;
short nm[26], uflg[26];
short anm[5][26];
short c1h[6], rw1h[6], pd1h[6], pb1h[6];
short c1l[6], rw1l[6], pd1l[6], pb1l[6];
short c2h[6], rw2h[6], pd2h[6], pb2h[6];
short c2l[6], rw2l[6], pd2l[6], pb2l[6];
short c3h[6], rw3h[6], pd3h[6], pb3h[6];
short c3l[6], rw3l[6], pd3l[6], pb3l[6];
short c4h[6], rw4h[6], pd4h[6], pb4h[6];
short c4l[6], rw4l[6], pd4l[6], pb4l[6];
short c5h[6], rw5h[6], pd5h[6], pb5h[6];
short c5l[6], rw5l[6], pd5l[6], pb5l[6];

```



```

//
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 ansprint(void), prans(void);
//
/** Main Program *
int main(){
short n;
printf("\n");
printf("*** 'Complete Euler Squares' of Order 5:\n");
printf(" Simultaneous Type: Both Self-Complementary and Pan-Diagonal **\n");
printf("** List of the 16 Standard Solutions **\n");
printf(" [Normalizing Inequality Conditions: n1<n25, n1<n5, n1<21 and n2>n6;]\n");
printf("\n");
for(n=0;n<26;n++){nm[n]=0; uflg[n]=0;}
for(n=0;n<6;n++){
cl1h[n]=0; rw1h[n]=0; pd1h[n]=0; pb1h[n]=0;
cl1l[n]=0; rw1l[n]=0; pd1l[n]=0; pb1l[n]=0;
cl2h[n]=0; rw2h[n]=0; pd2h[n]=0; pb2h[n]=0;
cl2l[n]=0; rw2l[n]=0; pd2l[n]=0; pb2l[n]=0;
cl3h[n]=0; rw3h[n]=0; pd3h[n]=0; pb3h[n]=0;
cl3l[n]=0; rw3l[n]=0; pd3l[n]=0; pb3l[n]=0;
cl4h[n]=0; rw4h[n]=0; pd4h[n]=0; pb4h[n]=0;
cl4l[n]=0; rw4l[n]=0; pd4l[n]=0; pb4l[n]=0;
cl5h[n]=0; rw5h[n]=0; pd5h[n]=0; pb5h[n]=0;
cl5l[n]=0; rw5l[n]=0; pd5l[n]=0; pb5l[n]=0;}
LSM=60; CC=24; cnt=0; cnt3=0;
nm[13]=12; uflg[12]=1;
cl3h[2]=1; rw3h[2]=1; pd1h[2]=1; pb5h[2]=1;
cl3l[2]=1; rw3l[2]=1; pd1l[2]=1; pb5l[2]=1;
stp01();      /* Begin The Calculations *
if(cnt3>0){prans();}
printf("\n");
printf(" [Total Solution Counts(n1=1/All) = %d/%d] OK!\n",cnt1,cnt);
printf("\n");
printf("** Calculated and Listed by Kanji Setsuda on\n");
printf(" Feb.14, 2015 with MacOSX 10.10.2 and Xcode 6.1.1 **\n");
printf("\n");
return 0;
}
//
/** Begin The Calculations *
/** Set n1 & n25 & n1<n25 *
void stp01(){
short a,na,b,nb,d,nd;
for(a=0;a<5;a++){na=4-a;
if((cl1h[a]==0)&&(rw1h[a]==0)&&(pd1h[a]==0)&&(pb1h[a]==0)){
if((cl5h[na]==0)&&(rw5h[na]==0)&&(pd1h[na]==0)&&(pb4h[na]==0)){
for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
if((uflg[d]==0)&&(uflg[nd]==0)&&(d<nd)){
if((cl1l[b]==0)&&(rw1l[b]==0)&&(pd1l[b]==0)&&(pb1l[b]==0)){
if((cl5l[nb]==0)&&(rw5l[nb]==0)&&(pd1l[nb]==0)&&(pb4l[nb]==0)){
nm[1]=d; nm[25]=nd; uflg[d]=1; uflg[nd]=1;
cl1h[a]=1; rw1h[a]=1; pd1h[a]=1; pb1h[a]=1;
cl1l[b]=1; rw1l[b]=1; pd1l[b]=1; pb1l[b]=1;
cl5h[na]=1; rw5h[na]=1; pd1h[na]=1; pb4h[na]=1;
cl5l[nb]=1; rw5l[nb]=1; pd1l[nb]=1; pb4l[nb]=1;
stp02();
cl5h[na]=0; rw5h[na]=0; pd1h[na]=0; pb4h[na]=0;
cl5l[nb]=0; rw5l[nb]=0; pd1l[nb]=0; pb4l[nb]=0;
cl1h[a]=0; rw1h[a]=0; pd1h[a]=0; pb1h[a]=0;
cl1l[b]=0; rw1l[b]=0; pd1l[b]=0; pb1l[b]=0;
uflg[nd]=0; uflg[d]=0;}}}}
}}}}

```

```

}
}
}
/* Set n7 & n19 & n1+n7+n13+n19+n25=LSM? *
void stp02(){
short a,na,b,nb,d,nd;
for(a=0;a<5;a++){na=4-a;
if((c12h[a]==0)&&(rw2h[a]==0)&&(pd1h[a]==0)&&(pb3h[a]==0)){
if((c14h[na]==0)&&(rw4h[na]==0)&&(pd1h[na]==0)&&(pb2h[na]==0)){
for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
if((nm[1]+d+nm[13]+nd+nm[25]==LSM)&&(uflg[d]==0)&&(uflg[nd]==0)){
if((c12l[b]==0)&&(rw2l[b]==0)&&(pd1l[b]==0)&&(pb3l[b]==0)){
if((c14l[nb]==0)&&(rw4l[nb]==0)&&(pd1l[nb]==0)&&(pb2l[nb]==0)){
nm[7]=d; nm[19]=nd; uflg[d]=1; uflg[nd]=1;
c12h[a]=1; rw2h[a]=1; pd1h[a]=1; pb3h[a]=1;
c12l[b]=1; rw2l[b]=1; pd1l[b]=1; pb3l[b]=1;
c14h[na]=1; rw4h[na]=1; pd1h[na]=1; pb2h[na]=1;
c14l[nb]=1; rw4l[nb]=1; pd1l[nb]=1; pb2l[nb]=1;
stp03();
c14h[na]=0; rw4h[na]=0; pd1h[na]=0; pb2h[na]=0;
c14l[nb]=0; rw4l[nb]=0; pd1l[nb]=0; pb2l[nb]=0;
c12h[a]=0; rw2h[a]=0; pd1h[a]=0; pb3h[a]=0;
c12l[b]=0; rw2l[b]=0; pd1l[b]=0; pb3l[b]=0;
uflg[nd]=0; uflg[d]=0;}}}}
}}}
}
}
}
/* Set n5 & n21 & n1<n5 & n1<n21 *
void stp03(){
short a,na,b,nb,d,nd;
for(a=0;a<5;a++){na=4-a;
if((c15h[a]==0)&&(rw1h[a]==0)&&(pd5h[a]==0)&&(pb5h[a]==0)){
if((c11h[na]==0)&&(rw5h[na]==0)&&(pd2h[na]==0)&&(pb5h[na]==0)){
for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
if((nm[1]<d)&&(nm[1]<nd)&&(uflg[d]==0)&&(uflg[nd]==0)){
if((c15l[b]==0)&&(rw1l[b]==0)&&(pd5l[b]==0)&&(pb5l[b]==0)){
if((c11l[nb]==0)&&(rw5l[nb]==0)&&(pd2l[nb]==0)&&(pb5l[nb]==0)){
nm[5]=d; nm[21]=nd; uflg[d]=1; uflg[nd]=1;
c15h[a]=1; rw1h[a]=1; pd5h[a]=1; pb5h[a]=1;
c15l[b]=1; rw1l[b]=1; pd5l[b]=1; pb5l[b]=1;
c11h[na]=1; rw5h[na]=1; pd2h[na]=1; pb5h[na]=1;
c11l[nb]=1; rw5l[nb]=1; pd2l[nb]=1; pb5l[nb]=1;
stp04();
c11h[na]=0; rw5h[na]=0; pd2h[na]=0; pb5h[na]=0;
c11l[nb]=0; rw5l[nb]=0; pd2l[nb]=0; pb5l[nb]=0;
c15h[a]=0; rw1h[a]=0; pd5h[a]=0; pb5h[a]=0;
c15l[b]=0; rw1l[b]=0; pd5l[b]=0; pb5l[b]=0;
uflg[nd]=0; uflg[d]=0;}}}}
}}}
}
}
}
/* Set n9 & n17 & n5+n9+n13+n17+n21=LSM? *
void stp04(){
short a,na,b,nb,d,nd;
for(a=0;a<5;a++){na=4-a;
if((c14h[a]==0)&&(rw2h[a]==0)&&(pd3h[a]==0)&&(pb5h[a]==0)){
if((c12h[na]==0)&&(rw4h[na]==0)&&(pd4h[na]==0)&&(pb5h[na]==0)){
for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
if((nm[5]+d+nm[13]+nd+nm[21]==LSM)&&(uflg[d]==0)&&(uflg[nd]==0)){
if((c14l[b]==0)&&(rw2l[b]==0)&&(pd3l[b]==0)&&(pb5l[b]==0)){
if((c12l[nb]==0)&&(rw4l[nb]==0)&&(pd4l[nb]==0)&&(pb5l[nb]==0)){
nm[9]=d; nm[17]=nd; uflg[d]=1; uflg[nd]=1;
c14h[a]=1; rw2h[a]=1; pd3h[a]=1; pb5h[a]=1;
c14l[b]=1; rw2l[b]=1; pd3l[b]=1; pb5l[b]=1;
c12h[na]=1; rw4h[na]=1; pd4h[na]=1; pb5h[na]=1;
c12l[nb]=1; rw4l[nb]=1; pd4l[nb]=1; pb5l[nb]=1;
}}}
}}}
}
}
}

```

```

        stp05();
        cl2h[na]=0; rw4h[na]=0; pd4h[na]=0; pb5h[na]=0;
        cl2l[nb]=0; rw4l[nb]=0; pd4l[nb]=0; pb5l[nb]=0;
        cl4h[a]=0; rw2h[a]=0; pd3h[a]=0; pb5h[a]=0;
        cl4l[b]=0; rw2l[b]=0; pd3l[b]=0; pb5l[b]=0;
        uflg[nd]=0; uflg[d]=0;}}}
    }
}
/* Level #2: *
/* Set n2 & n24 *
void stp05(){
    short a,na,b,nb,d,nd;
    for(a=0;a<5;a++){na=4-a;
        if((cl2h[a]==0)&&(rw1h[a]==0)&&(pd2h[a]==0)&&(pb2h[a]==0)){
            if((cl4h[na]==0)&&(rw5h[na]==0)&&(pd5h[na]==0)&&(pb3h[na]==0)){
                for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
                    if((uflg[d]==0)&&(uflg[nd]==0)){
                        if((cl2l[b]==0)&&(rw1l[b]==0)&&(pd2l[b]==0)&&(pb2l[b]==0)){
                            if((cl4l[nb]==0)&&(rw5l[nb]==0)&&(pd5l[nb]==0)&&(pb3l[nb]==0)){
                                nm[2]=d; nm[24]=nd; uflg[d]=1; uflg[nd]=1;
                                cl2h[a]=1; rw1h[a]=1; pd2h[a]=1; pb2h[a]=1;
                                cl2l[b]=1; rw1l[b]=1; pd2l[b]=1; pb2l[b]=1;
                                cl4h[na]=1; rw5h[na]=1; pd5h[na]=1; pb3h[na]=1;
                                cl4l[nb]=1; rw5l[nb]=1; pd5l[nb]=1; pb3l[nb]=1;
                                stp06();
                                cl4h[na]=0; rw5h[na]=0; pd5h[na]=0; pb3h[na]=0;
                                cl4l[nb]=0; rw5l[nb]=0; pd5l[nb]=0; pb3l[nb]=0;
                                cl2h[a]=0; rw1h[a]=0; pd2h[a]=0; pb2h[a]=0;
                                cl2l[b]=0; rw1l[b]=0; pd2l[b]=0; pb2l[b]=0;
                                uflg[nd]=0; uflg[d]=0;}}}
                    }
                }
            }
        }
    }
}
/* Set n3 & n23 *
void stp06(){
    short a,na,b,nb,d,nd;
    for(a=0;a<5;a++){na=4-a;
        if((cl3h[a]==0)&&(rw1h[a]==0)&&(pd3h[a]==0)&&(pb3h[a]==0)){
            if((cl3h[na]==0)&&(rw5h[na]==0)&&(pd4h[na]==0)&&(pb2h[na]==0)){
                for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
                    if((uflg[d]==0)&&(uflg[nd]==0)){
                        if((cl3l[b]==0)&&(rw1l[b]==0)&&(pd3l[b]==0)&&(pb3l[b]==0)){
                            if((cl3l[nb]==0)&&(rw5l[nb]==0)&&(pd4l[nb]==0)&&(pb2l[nb]==0)){
                                nm[3]=d; nm[23]=nd; uflg[d]=1; uflg[nd]=1;
                                cl3h[a]=1; rw1h[a]=1; pd3h[a]=1; pb3h[a]=1;
                                cl3l[b]=1; rw1l[b]=1; pd3l[b]=1; pb3l[b]=1;
                                cl3h[na]=1; rw5h[na]=1; pd4h[na]=1; pb2h[na]=1;
                                cl3l[nb]=1; rw5l[nb]=1; pd4l[nb]=1; pb2l[nb]=1;
                                stp07();
                                cl3h[na]=0; rw5h[na]=0; pd4h[na]=0; pb2h[na]=0;
                                cl3l[nb]=0; rw5l[nb]=0; pd4l[nb]=0; pb2l[nb]=0;
                                cl3h[a]=0; rw1h[a]=0; pd3h[a]=0; pb3h[a]=0;
                                cl3l[b]=0; rw1l[b]=0; pd3l[b]=0; pb3l[b]=0;
                                uflg[nd]=0; uflg[d]=0;}}}
                    }
                }
            }
        }
    }
}
/* Set n4=LSM-n1-n2-n3-n5 & n22 *
void stp07(){
    short a,na,b,nb,d,nd;
    for(a=0;a<5;a++){na=4-a;
        if((cl4h[a]==0)&&(rw1h[a]==0)&&(pd4h[a]==0)&&(pb4h[a]==0)){
            if((cl2h[na]==0)&&(rw5h[na]==0)&&(pd3h[na]==0)&&(pb1h[na]==0)){
                for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;

```

```

if((nm[1]+nm[2]+nm[3]+d+nm[5]==LSM)&&(uflg[d]==0)&&(uflg[nd]==0)){
  if((c14l[b]==0)&&(rw1l[b]==0)&&(pd4l[b]==0)&&(pb4l[b]==0)){
    if((c12l[nb]==0)&&(rw5l[nb]==0)&&(pd3l[nb]==0)&&(pb1l[nb]==0)){
      nm[4]=d; nm[22]=nd; uflg[d]=1; uflg[nd]=1;
      c14h[a]=1; rw1h[a]=1; pd4h[a]=1; pb4h[a]=1;
      c14l[b]=1; rw1l[b]=1; pd4l[b]=1; pb4l[b]=1;
      c12h[na]=1; rw5h[na]=1; pd3h[na]=1; pb1h[na]=1;
      c12l[nb]=1; rw5l[nb]=1; pd3l[nb]=1; pb1l[nb]=1;
      stp08C);
      c12h[na]=0; rw5h[na]=0; pd3h[na]=0; pb1h[na]=0;
      c12l[nb]=0; rw5l[nb]=0; pd3l[nb]=0; pb1l[nb]=0;
      c14h[a]=0; rw1h[a]=0; pd4h[a]=0; pb4h[a]=0;
      c14l[b]=0; rw1l[b]=0; pd4l[b]=0; pb4l[b]=0;
      uflg[nd]=0; uflg[d]=0;}}}}
}
}
}
/** Set n6 & n20 & n6<n2 *
void stp08C){
  short a,na,b,nb,d,nd;
  for(a=0;a<5;a++){na=4-a;
    if((c11h[a]==0)&&(rw2h[a]==0)&&(pd5h[a]==0)&&(pb2h[a]==0)){
      if((c15h[na]==0)&&(rw4h[na]==0)&&(pd2h[na]==0)&&(pb3h[na]==0)){
        for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
          if((d<nm[2])&&(uflg[d]==0)&&(uflg[nd]==0)){
            if((c11l[b]==0)&&(rw2l[b]==0)&&(pd5l[b]==0)&&(pb2l[b]==0)){
              if((c15l[nb]==0)&&(rw4l[nb]==0)&&(pd2l[nb]==0)&&(pb3l[nb]==0)){
                nm[6]=d; nm[20]=nd; uflg[d]=1; uflg[nd]=1;
                c11h[a]=1; rw2h[a]=1; pd5h[a]=1; pb2h[a]=1;
                c11l[b]=1; rw2l[b]=1; pd5l[b]=1; pb2l[b]=1;
                c15h[na]=1; rw4h[na]=1; pd2h[na]=1; pb3h[na]=1;
                c15l[nb]=1; rw4l[nb]=1; pd2l[nb]=1; pb3l[nb]=1;
                stp09C);
                c15h[na]=0; rw4h[na]=0; pd2h[na]=0; pb3h[na]=0;
                c15l[nb]=0; rw4l[nb]=0; pd2l[nb]=0; pb3l[nb]=0;
                c11h[a]=0; rw2h[a]=0; pd5h[a]=0; pb2h[a]=0;
                c11l[b]=0; rw2l[b]=0; pd5l[b]=0; pb2l[b]=0;
                uflg[nd]=0; uflg[d]=0;}}}}
          }}}
        }
      }
    }
}
}
/** Set n11=LSM-n3-n7-n20-n24 & n15 *
void stp09C){
  short a,na,b,nb,d,nd;
  for(a=0;a<5;a++){na=4-a;
    if((c11h[a]==0)&&(rw3h[a]==0)&&(pd4h[a]==0)&&(pb3h[a]==0)){
      if((c15h[na]==0)&&(rw3h[na]==0)&&(pd3h[na]==0)&&(pb2h[na]==0)){
        for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
          if((nm[3]+nm[7]+d+nm[20]+nm[24]==LSM)&&(uflg[d]==0)&&(uflg[nd]==0)){
            if((c11l[b]==0)&&(rw3l[b]==0)&&(pd4l[b]==0)&&(pb3l[b]==0)){
              if((c15l[nb]==0)&&(rw3l[nb]==0)&&(pd3l[nb]==0)&&(pb2l[nb]==0)){
                nm[11]=d; nm[15]=nd; uflg[d]=1; uflg[nd]=1;
                c11h[a]=1; rw3h[a]=1; pd4h[a]=1; pb3h[a]=1;
                c11l[b]=1; rw3l[b]=1; pd4l[b]=1; pb3l[b]=1;
                c15h[na]=1; rw3h[na]=1; pd3h[na]=1; pb2h[na]=1;
                c15l[nb]=1; rw3l[nb]=1; pd3l[nb]=1; pb2l[nb]=1;
                stp10C);
                c15h[na]=0; rw3h[na]=0; pd3h[na]=0; pb2h[na]=0;
                c15l[nb]=0; rw3l[nb]=0; pd3l[nb]=0; pb2l[nb]=0;
                c11h[a]=0; rw3h[a]=0; pd4h[a]=0; pb3h[a]=0;
                c11l[b]=0; rw3l[b]=0; pd4l[b]=0; pb3l[b]=0;
                uflg[nd]=0; uflg[d]=0;}}}}
          }}}
        }
      }
    }
}
}
}

```

```

/* Set n16=LSM-n1-n6-n11-n21 & n10 *
void stp10(){
short a,na,b,nb,d,nd;
for(a=0;a<5;a++){na=4-a;
if((c11h[a]==0)&&(rw4h[a]==0)&&(pd3h[a]==0)&&(pb4h[a]==0)){
if((c15h[na]==0)&&(rw2h[na]==0)&&(pd4h[na]==0)&&(pb1h[na]==0)){
for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
if((nm[1]+nm[6]+nm[11]+d+nm[21]==LSM)&&(uflg[d]==0)&&(uflg[nd]==0)){
if((c11l[b]==0)&&(rw4l[b]==0)&&(pd3l[b]==0)&&(pb4l[b]==0)){
if((c15l[nb]==0)&&(rw2l[nb]==0)&&(pd4l[nb]==0)&&(pb1l[nb]==0)){
nm[16]=d; nm[10]=nd; uflg[d]=1; uflg[nd]=1;
c11h[a]=1; rw4h[a]=1; pd3h[a]=1; pb4h[a]=1;
c11l[b]=1; rw4l[b]=1; pd3l[b]=1; pb4l[b]=1;
c15h[na]=1; rw2h[na]=1; pd4h[na]=1; pb1h[na]=1;
c15l[nb]=1; rw2l[nb]=1; pd4l[nb]=1; pb1l[nb]=1;
stp11();
c15h[na]=0; rw2h[na]=0; pd4h[na]=0; pb1h[na]=0;
c15l[nb]=0; rw2l[nb]=0; pd4l[nb]=0; pb1l[nb]=0;
c11h[a]=0; rw4h[a]=0; pd3h[a]=0; pb4h[a]=0;
c11l[b]=0; rw4l[b]=0; pd3l[b]=0; pb4l[b]=0;
uflg[nd]=0; uflg[d]=0;}}}}
}}}
}
}
/* Set n8=LSM-n6-n7-n9-n10 & n18 *
void stp11(){
short a,na,b,nb,d,nd;
for(a=0;a<5;a++){na=4-a;
if((c13h[a]==0)&&(rw2h[a]==0)&&(pd2h[a]==0)&&(pb4h[a]==0)){
if((c13h[na]==0)&&(rw4h[na]==0)&&(pd5h[na]==0)&&(pb1h[na]==0)){
for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
if((nm[6]+nm[7]+d+nm[9]+nm[10]==LSM)&&(uflg[d]==0)&&(uflg[nd]==0)){
if((c13l[b]==0)&&(rw2l[b]==0)&&(pd2l[b]==0)&&(pb4l[b]==0)){
if((c13l[nb]==0)&&(rw4l[nb]==0)&&(pd5l[nb]==0)&&(pb1l[nb]==0)){
nm[8]=d; nm[18]=nd; uflg[d]=1; uflg[nd]=1;
c13h[a]=1; rw2h[a]=1; pd2h[a]=1; pb4h[a]=1;
c13l[b]=1; rw2l[b]=1; pd2l[b]=1; pb4l[b]=1;
c13h[na]=1; rw4h[na]=1; pd5h[na]=1; pb1h[na]=1;
c13l[nb]=1; rw4l[nb]=1; pd5l[nb]=1; pb1l[nb]=1;
stp12();
c13h[na]=0; rw4h[na]=0; pd5h[na]=0; pb1h[na]=0;
c13l[nb]=0; rw4l[nb]=0; pd5l[nb]=0; pb1l[nb]=0;
c13h[a]=0; rw2h[a]=0; pd2h[a]=0; pb4h[a]=0;
c13l[b]=0; rw2l[b]=0; pd2l[b]=0; pb4l[b]=0;
uflg[nd]=0; uflg[d]=0;}}}}
}}}
}
}
/* Set n12=LSM-n2-n7-n17-n22 & n14 *
void stp12(){
short a,na,b,nb,d,nd;
for(a=0;a<5;a++){na=4-a;
if((c12h[a]==0)&&(rw3h[a]==0)&&(pd5h[a]==0)&&(pb4h[a]==0)){
if((c14h[na]==0)&&(rw3h[na]==0)&&(pd2h[na]==0)&&(pb1h[na]==0)){
for(b=0;b<5;b++){nb=4-b; d=a*5+b; nd=CC-d;
if((nm[2]+nm[7]+d+nm[17]+nm[22]==LSM)&&(uflg[d]==0)&&(uflg[nd]==0)){
if((c12l[b]==0)&&(rw3l[b]==0)&&(pd5l[b]==0)&&(pb4l[b]==0)){
if((c14l[nb]==0)&&(rw3l[nb]==0)&&(pd2l[nb]==0)&&(pb1l[nb]==0)){
nm[12]=d; nm[14]=nd; uflg[d]=1; uflg[nd]=1;
c12h[a]=1; rw3h[a]=1; pd5h[a]=1; pb4h[a]=1;
c12l[b]=1; rw3l[b]=1; pd5l[b]=1; pb4l[b]=1;
c14h[na]=1; rw3h[na]=1; pd2h[na]=1; pb1h[na]=1;
c14l[nb]=1; rw3l[nb]=1; pd2l[nb]=1; pb1l[nb]=1;
ansprint();
c14h[na]=0; rw3h[na]=0; pd2h[na]=0; pb1h[na]=0;
}}}
}}}
}
}

```

```

        c14l[nb]=0; rw3l[nb]=0; pd2l[nb]=0; pb1l[nb]=0;
        c12h[a]=0; rw3h[a]=0; pd5h[a]=0; pb4h[a]=0;
        c12l[b]=0; rw3l[b]=0; pd5l[b]=0; pb4l[b]=0;
        uflg[nd]=0; uflg[d]=0;}}}}
    }
}
//
/** Print the Answers *
void ansprint(){
    short n;
    cnt++; if(nm[1]==1){cnt1++;}
    anm[cnt3*2][0]=cnt;
    for(n=1;n<26;n++){anm[cnt3*2][n]=nm[n]+1; anm[cnt3*2+1][n]=nm[n];}
    cnt3++; if(cnt3==2){prans(); cnt3=0;}
}
//
/** Print Two Answers *
void prans(){
    short l,l5,n;
    printf(" /D5i  H/          L/%15d#  /D5i  H/          L/%15d#\n",
           anm[0][0],anm[2][0]);
    for(l=0;l<5;l++){l5=l*5;
        for(n=1;n<6;n++){printf("%2d", (anm[1][l5+n])/5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%2d", (anm[1][l5+n])%5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%3d", anm[0][l5+n]);}
        printf(" ");
        for(n=1;n<6;n++){printf("%2d", (anm[3][l5+n])/5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%2d", (anm[3][l5+n])%5);}
        printf(" ");
        for(n=1;n<6;n++){printf("%3d", anm[2][l5+n]);}
        printf("\n");
    }
}
//

```

6. 計算結果のリスト

早速にも、このプログラムの実行結果を示そう。下のリストがそれである。

* 「完全オイラー型」対称汎五方陣の標準解 16 個のリスト (全) *

[正規化のための不等式: $n1 < n25$, $n1 < n5$, $n1 < 21$ and $n2 > n6$;]

/D5i	H/	L/	1#	/D5i	H/	L/	2#
0 4 3 2 1	0 2 4 1 3	1 23 20 12 9		0 4 3 2 1	0 2 4 3 1	1 23 20 14 7	
2 1 0 4 3	4 1 3 0 2	15 7 4 21 18		2 1 0 4 3	4 3 1 0 2	15 9 2 21 18	
4 3 2 1 0	3 0 2 4 1	24 16 13 10 2		4 3 2 1 0	1 0 2 4 3	22 16 13 10 4	
1 0 4 3 2	2 4 1 3 0	8 5 22 19 11		1 0 4 3 2	2 4 3 1 0	8 5 24 17 11	
3 2 1 0 4	1 3 0 2 4	17 14 6 3 25		3 2 1 0 4	3 1 0 2 4	19 12 6 3 25	
/D5i	H/	L/	3#	/D5i	H/	L/	4#
0 4 1 2 3	0 2 4 1 3	1 23 10 12 19		0 4 1 2 3	0 2 4 3 1	1 23 10 14 17	
2 3 0 4 1	4 1 3 0 2	15 17 4 21 8		2 3 0 4 1	4 3 1 0 2	15 19 2 21 8	
4 1 2 3 0	3 0 2 4 1	24 6 13 20 2		4 1 2 3 0	1 0 2 4 3	22 6 13 20 4	
3 0 4 1 2	2 4 1 3 0	18 5 22 9 11		3 0 4 1 2	2 4 3 1 0	18 5 24 7 11	
1 2 3 0 4	1 3 0 2 4	7 14 16 3 25		1 2 3 0 4	3 1 0 2 4	9 12 16 3 25	
/D5i	H/	L/	5#	/D5i	H/	L/	6#
0 4 3 2 1	1 2 3 0 4	2 23 19 11 10		0 4 3 2 1	1 2 3 4 0	2 23 19 15 6	
2 1 0 4 3	3 0 4 1 2	14 6 5 22 18		2 1 0 4 3	3 4 0 1 2	14 10 1 22 18	
4 3 2 1 0	4 1 2 3 0	25 17 13 9 1		4 3 2 1 0	0 1 2 3 4	21 17 13 9 5	
1 0 4 3 2	2 3 0 4 1	8 4 21 20 12		1 0 4 3 2	2 3 4 0 1	8 4 25 16 12	
3 2 1 0 4	0 4 1 2 3	16 15 7 3 24		3 2 1 0 4	4 0 1 2 3	20 11 7 3 24	

/D5i	H/	L/	7#	/D5i	H/	L/	8#
0 4 1 2 3	1 2 3 0 4	2 23 9 11 20		0 4 1 2 3	1 2 3 4 0	2 23 9 15 16	
2 3 0 4 1	3 0 4 1 2	14 16 5 22 8		2 3 0 4 1	3 4 0 1 2	14 20 1 22 8	
4 1 2 3 0	4 1 2 3 0	25 7 13 19 1		4 1 2 3 0	0 1 2 3 4	21 7 13 19 5	
3 0 4 1 2	2 3 0 4 1	18 4 21 10 12		3 0 4 1 2	2 3 4 0 1	18 4 25 6 12	
1 2 3 0 4	0 4 1 2 3	6 15 17 3 24		1 2 3 0 4	4 0 1 2 3	10 11 17 3 24	
/D5i	H/	L/	9#	/D5i	H/	L/	10#
0 4 3 2 1	3 2 1 0 4	4 23 17 11 10		0 4 3 2 1	3 2 1 4 0	4 23 17 15 6	
2 1 0 4 3	1 0 4 3 2	12 6 5 24 18		2 1 0 4 3	1 4 0 3 2	12 10 1 24 18	
4 3 2 1 0	4 3 2 1 0	25 19 13 7 1		4 3 2 1 0	0 3 2 1 4	21 19 13 7 5	
1 0 4 3 2	2 1 0 4 3	8 2 21 20 14		1 0 4 3 2	2 1 4 0 3	8 2 25 16 14	
3 2 1 0 4	0 4 3 2 1	16 15 9 3 22		3 2 1 0 4	4 0 3 2 1	20 11 9 3 22	
/D5i	H/	L/	11#	/D5i	H/	L/	12#
0 4 1 2 3	3 2 1 0 4	4 23 7 11 20		0 4 1 2 3	3 2 1 4 0	4 23 7 15 16	
2 3 0 4 1	1 0 4 3 2	12 16 5 24 8		2 3 0 4 1	1 4 0 3 2	12 20 1 24 8	
4 1 2 3 0	4 3 2 1 0	25 9 13 17 1		4 1 2 3 0	0 3 2 1 4	21 9 13 17 5	
3 0 4 1 2	2 1 0 4 3	18 2 21 10 14		3 0 4 1 2	2 1 4 0 3	18 2 25 6 14	
1 2 3 0 4	0 4 3 2 1	6 15 19 3 22		1 2 3 0 4	4 0 3 2 1	10 11 19 3 22	
/D5i	H/	L/	13#	/D5i	H/	L/	14#
0 4 3 2 1	4 2 0 1 3	5 23 16 12 9		0 4 3 2 1	4 2 0 3 1	5 23 16 14 7	
2 1 0 4 3	0 1 3 4 2	11 7 4 25 18		2 1 0 4 3	0 3 1 4 2	11 9 2 25 18	
4 3 2 1 0	3 4 2 0 1	24 20 13 6 2		4 3 2 1 0	1 4 2 0 3	22 20 13 6 4	
1 0 4 3 2	2 0 1 3 4	8 1 22 19 15		1 0 4 3 2	2 0 3 1 4	8 1 24 17 15	
3 2 1 0 4	1 3 4 2 0	17 14 10 3 21		3 2 1 0 4	3 1 4 2 0	19 12 10 3 21	
/D5i	H/	L/	15#	/D5i	H/	L/	16#
0 4 1 2 3	4 2 0 1 3	5 23 6 12 19		0 4 1 2 3	4 2 0 3 1	5 23 6 14 17	
2 3 0 4 1	0 1 3 4 2	11 17 4 25 8		2 3 0 4 1	0 3 1 4 2	11 19 2 25 8	
4 1 2 3 0	3 4 2 0 1	24 10 13 16 2		4 1 2 3 0	1 4 2 0 3	22 10 13 16 4	
3 0 4 1 2	2 0 1 3 4	18 1 22 9 15		3 0 4 1 2	2 0 3 1 4	18 1 24 7 15	
1 2 3 0 4	1 3 4 2 0	7 14 20 3 21		1 2 3 0 4	3 1 4 2 0	9 12 20 3 21	

[Total Solution Counts(n1=1/A11)] = 4/16] OK!

** Calculated and Listed by Kanji Setsuda on
Feb.14, 2015 with MacOSX 10.10.2 and Xcode 6.1.1 **

かくの如く16個を作れた。どれも「鈴木方陣」と同じ結果である。構成法が大掛かりなわりには、結果そのものに新味は無い。しかし、これで連立型の対称汎五方陣の解もまた全て「完全オイラー方陣」であることが、実際に確かめられた。この検証の意味は断然大きい。

7. 対称五方陣では「オイラー方陣」が幾つあるか？

次に、対称五方陣にも果たして「オイラー方陣」があるのかどうかを調べておこう。

対称型であるから、汎対角線でのラテン構造を追求しない。

最初に、主対角線2本もまた「オイラー型」になっているものを調べた。

* 「オイラー型」対称五方陣の標準解 32 個のリスト (全) *

(Used_Unit:H/L/; Sol_Numb#; Sum_Checks:\Diag1/Diag2|Row|Column|)

3/	5/	1#\65/65	4/	6/	2#\65/65
04321 02413	1 23 20 12	9 65 65	03421 02143	1 18 22 15	9 65 65
21043 41302	15 7 4 21	18 65 65	21340 31402	14 7 20 21	3 65 65
43210 30241	24 16 13 10	2 65 65	14203 43210	10 24 13 2	16 65 65
10432 24130	8 5 22 19	11 65 65	40132 24031	23 5 6 19	12 65 65
32104 13024	17 14 6 3	25 65 65	32014 10324	17 11 4 8	25 65 65
3/	9/	3#\65/65	4/	10/	4#\65/65
04321 02431	1 23 20 14	7 65 65	03421 02341	1 18 24 15	7 65 65
21043 43102	15 9 2 21	18 65 65	21340 13402	12 9 20 21	3 65 65
43210 10243	22 16 13 10	4 65 65	14203 41230	10 22 13 4	16 65 65
10432 24310	8 5 24 17	11 65 65	40132 24013	23 5 6 17	14 65 65
32104 31024	19 12 6 3	25 65 65	32014 30124	19 11 2 8	25 65 65

15/	5/					5#\65/65		10/	4/					6#\65/65
04123	02413	1	23	10	12	19 65 65		02341	03421	1	14	20	23	7 65 65
23041	41302	15	17	4	21	8 65 65		13402	21340	8	17	24	5	11 65 65
41230	30241	24	6	13	20	2 65 65		41230	14203	22	10	13	16	4 65 65
30412	24130	18	5	22	9	11 65 65		24013	40132	15	21	2	9	18 65 65
12304	13024	7	14	16	3	25 65 65		30124	32014	19	3	6	12	25 65 65

15/	9/					7#\65/65		10/	16/					8#\65/65
04123	02431	1	23	10	14	17 65 65		02341	01423	1	12	20	23	9 65 65
23041	43102	15	19	2	21	8 65 65		13402	23140	8	19	22	5	11 65 65
41230	10243	22	6	13	20	4 65 65		41230	34201	24	10	13	16	2 65 65
30412	24310	18	5	24	7	11 65 65		24013	40312	15	21	4	7	18 65 65
12304	31024	9	12	16	3	25 65 65		30124	12034	17	3	6	14	25 65 65

3/ 21/	9#	4/ 22/	10#	3/ 26/	11#	4/ 25/	12#
2 23 19 11 10		2 18 21 14 10		2 23 19 15 6		2 18 25 14 6	
14 6 5 22 18		15 6 19 22 3		14 10 1 22 18		11 10 19 22 3	
25 17 13 9 1		9 25 13 1 17		21 17 13 9 5		9 21 13 5 17	
8 4 21 20 12		23 4 7 20 11		8 4 25 16 12		23 4 7 16 15	
16 15 7 3 24		16 12 5 8 24		20 11 7 3 24		20 12 1 8 24	

15/ 21/	13#	10/ 19/	14#	15/ 26/	15#	10/ 32/	16#
2 23 9 11 20		2 15 19 23 6		2 23 9 15 16		2 11 19 23 10	
14 16 5 22 8		8 16 25 4 12		14 20 1 22 8		8 20 21 4 12	
25 7 13 19 1		21 9 13 17 5		21 7 13 19 5		25 9 13 17 1	
18 4 21 10 12		14 22 1 10 18		18 4 25 6 12		14 22 5 6 18	
6 15 17 3 24		20 3 7 11 24		10 11 17 3 24		16 3 7 15 24	

3/ 39/	17#	4/ 40/	18#	3/ 44/	19#	4/ 43/	20#
4 23 17 11 10		4 18 21 12 10		4 23 17 15 6		4 18 25 12 6	
12 6 5 24 18		15 6 17 24 3		12 10 1 24 18		11 10 17 24 3	
25 19 13 7 1		7 25 13 1 19		21 19 13 7 5		7 21 13 5 19	
8 2 21 20 14		23 2 9 20 11		8 2 25 16 14		23 2 9 16 15	
16 15 9 3 22		16 14 5 8 22		20 11 9 3 22		20 14 1 8 22	

15/ 39/	21#	10/ 33/	22#	15/ 44/	23#	10/ 46/	24#
4 23 7 11 20		4 15 17 23 6		4 23 7 15 16		4 11 17 23 10	
12 16 5 24 8		8 16 25 2 14		12 20 1 24 8		8 20 21 2 14	
25 9 13 17 1		21 7 13 19 5		21 9 13 17 5		25 7 13 19 1	
18 2 21 10 14		12 24 1 10 18		18 2 25 6 14		12 24 5 6 18	
6 15 19 3 22		20 3 9 11 22		10 11 19 3 22		16 3 9 15 22	

3/ 56/	25#	4/ 55/	26#	3/ 60/	27#	4/ 59/	28#
5 23 16 12 9		5 18 22 11 9		5 23 16 14 7		5 18 24 11 7	
11 7 4 25 18		14 7 16 25 3		11 9 2 25 18		12 9 16 25 3	
24 20 13 6 2		6 24 13 2 20		22 20 13 6 4		6 22 13 4 20	
8 1 22 19 15		23 1 10 19 12		8 1 24 17 15		23 1 10 17 14	
17 14 10 3 21		17 15 4 8 21		19 12 10 3 21		19 15 2 8 21	

15/ 56/	29#	10/ 49/	30#	15/ 60/	31#	10/ 61/	32#
5 23 6 12 19		5 14 16 23 7		5 23 6 14 17		5 12 16 23 9	
11 17 4 25 8		8 17 24 1 15		11 19 2 25 8		8 19 22 1 15	
24 10 13 16 2		22 6 13 20 4		22 10 13 16 4		24 6 13 20 2	
18 1 22 9 15		11 25 2 9 18		18 1 24 7 15		11 25 4 7 18	
7 14 20 3 21		19 3 10 12 21		9 12 20 3 21		17 3 10 14 21	

[Total Counts of Solutions(n1=1/A11) = 8/32] OK!

** Calculated and Listed by Kanji Setsuda on Jun.27, 2010;
 Recompiled on Feb.14, 2015 with MacOSX 10.10.2 and Xcode 6.1.1; **

主対角線 2 本についてもラテン構造を持つものが、32 個あった。この結果は、前に調べたものと同じである。

次に、対角線にはいかなる注文も付けずに(とは言っても、主対角線 2 本の定和条件は外せ

ないが), ただ行列のみにラテン構造を追求して計数して見たら, 次の成果を得た。

* 行列のみ「オイラー型」の対称五方陣の標準解のリスト (抜粋) *

2/ 18/ 1#	2/ 36/ 2#	2/ 40/ 3#	2/ 41/ 4#
1 24 7 15 18	1 23 10 12 19	1 23 10 14 17	1 22 9 15 18
14 17 5 23 6	15 17 4 21 8	15 19 2 21 8	12 19 5 23 6
22 10 13 16 4	24 6 13 20 2	22 6 13 20 4	24 10 13 16 2
20 3 21 9 12	18 5 22 9 11	18 5 24 7 11	20 3 21 7 14
8 11 19 2 25	7 14 16 3 25	9 12 16 3 25	8 11 17 4 25
3/ 20/ 25#	3/ 34/ 26#	3/ 38/ 27#	3/ 43/ 28#
1 24 7 15 18	1 23 10 12 19	1 23 10 14 17	1 22 9 15 18
20 3 21 9 12	18 5 22 9 11	18 5 24 7 11	20 3 21 7 14
22 10 13 16 4	24 6 13 20 2	22 6 13 20 4	24 10 13 16 2
14 17 5 23 6	15 17 4 21 8	15 19 2 21 8	12 19 5 23 6
8 11 19 2 25	7 14 16 3 25	9 12 16 3 25	8 11 17 4 25
6/ 23/ 49#	6/ 26/ 50#	6/ 32/ 51#	6/ 47/ 52#
1 24 10 18 12	1 23 7 20 14	1 23 9 20 12	1 22 10 18 14
20 11 22 9 3	18 15 21 9 2	18 15 21 7 4	20 11 24 7 3
7 5 13 21 19	10 4 13 22 16	10 2 13 24 16	9 5 13 21 17
23 17 4 15 6	24 17 5 11 8	22 19 5 11 8	23 19 2 15 6
14 8 16 2 25	12 6 19 3 25	14 6 17 3 25	12 8 16 4 25
8/ 21/ 73#	8/ 29/ 74#	9/ 23/ 85#	9/ 26/ 86#
1 24 10 18 12	1 23 9 20 12	1 24 20 8 12	1 23 17 10 14
23 17 4 15 6	22 19 5 11 8	10 11 22 19 3	8 15 21 19 2
7 5 13 21 19	10 2 13 24 16	17 5 13 21 9	20 4 13 22 6
20 11 22 9 3	18 15 21 7 4	23 7 4 15 16	24 7 5 11 18
14 8 16 2 25	14 6 17 3 25	14 18 6 2 25	12 16 9 3 25
9/ 32/ 87#	9/ 47/ 88#	11/ 21/ 109#	11/ 29/ 110#
1 23 19 10 12	1 22 20 8 14	1 24 20 8 12	1 23 19 10 12
8 15 21 17 4	10 11 24 17 3	23 7 4 15 16	22 9 5 11 18
20 2 13 24 6	19 5 13 21 7	17 5 13 21 9	20 2 13 24 6
22 9 5 11 18	23 9 2 15 16	10 11 22 19 3	8 15 21 17 4
14 16 7 3 25	12 18 6 4 25	14 18 6 2 25	14 16 7 3 25
13/ 20/ 121#	13/ 34/ 122#	13/ 38/ 123#	13/ 43/ 124#
1 24 17 15 8	1 23 20 12 9	1 23 20 14 7	1 22 19 15 8
10 3 21 19 12	8 5 22 19 11	8 5 24 17 11	10 3 21 17 14
22 20 13 6 4	24 16 13 10 2	22 16 13 10 4	24 20 13 6 2
14 7 5 23 16	15 7 4 21 18	15 9 2 21 18	12 9 5 23 16
18 11 9 2 25	17 14 6 3 25	19 12 6 3 25	18 11 7 4 25
16/ 18/ 145#	16/ 36/ 146#	16/ 40/ 147#	16/ 41/ 148#
1 24 17 15 8	1 23 20 12 9	1 23 20 14 7	1 22 19 15 8
14 7 5 23 16	15 7 4 21 18	15 9 2 21 18	12 9 5 23 16
22 20 13 6 4	24 16 13 10 2	22 16 13 10 4	24 20 13 6 2
10 3 21 19 12	8 5 22 19 11	8 5 24 17 11	10 3 21 17 14
18 11 9 2 25	17 14 6 3 25	19 12 6 3 25	18 11 7 4 25
18/ 2/ 169#	18/ 16/ 170#	21/ 8/ 181#	21/ 11/ 182#
1 20 7 23 14	1 20 9 23 12	1 20 22 14 8	1 20 24 12 8
18 9 21 15 2	18 7 21 15 4	15 9 16 23 2	15 7 16 23 4
10 22 13 4 16	10 24 13 2 16	7 21 13 5 19	9 21 13 5 17
24 11 5 17 8	22 11 5 19 8	24 3 10 17 11	22 3 10 19 11
12 3 19 6 25	14 3 17 6 25	18 12 4 6 25	18 14 2 6 25
21/ 27/ 183#	21/ 29/ 184#	26/ 6/ 205#	26/ 9/ 206#
1 18 22 15 9	1 18 24 15 7	1 15 7 24 18	1 15 9 22 18
14 7 20 21 3	12 9 20 21 3	14 23 5 17 6	12 23 5 19 6
10 24 13 2 16	10 22 13 4 16	22 16 13 10 4	24 16 13 10 2
23 5 6 19 12	23 5 6 17 14	20 9 21 3 12	20 7 21 3 14
17 11 4 8 25	19 11 2 8 25	8 2 19 11 25	8 4 17 11 25

29/ 8/ 217#	29/ 11/ 218#	29/ 21/ 219#	29/ 45/ 220#
1 15 17 24 8	1 15 19 22 8	1 14 20 23 7	1 12 20 23 9
10 19 21 3 12	10 17 21 3 14	8 17 24 5 11	8 19 22 5 11
22 6 13 20 4	24 6 13 20 2	22 10 13 16 4	24 10 13 16 2
14 23 5 7 16	12 23 5 9 16	15 21 2 9 18	15 21 4 7 18
18 2 9 11 25	18 4 7 11 25	19 3 6 12 25	17 3 6 14 25
32/ 6/ 241#	32/ 9/ 242#	34/ 3/ 253#	34/ 13/ 254#
1 15 17 24 8	1 15 19 22 8	1 15 22 8 19	1 15 24 8 17
14 23 5 7 16	12 23 5 9 16	14 21 10 17 3	12 21 10 19 3
22 6 13 20 4	24 6 13 20 2	20 2 13 24 6	20 4 13 22 6
10 19 21 3 12	10 17 21 3 14	23 9 16 5 12	23 7 16 5 14
18 2 9 11 25	18 4 7 11 25	7 18 4 11 25	9 18 2 11 25
38/ 3/ 265#	38/ 13/ 266#	41/ 2/ 277#	41/ 16/ 278#
1 15 22 18 9	1 15 24 18 7	1 10 17 23 14	1 10 19 23 12
14 21 20 7 3	12 21 20 9 3	8 19 21 15 2	8 17 21 15 4
10 2 13 24 16	10 4 13 22 16	20 22 13 4 6	20 24 13 2 6
23 19 6 5 12	23 17 6 5 14	24 11 5 7 18	22 11 5 9 18
17 8 4 11 25	19 8 2 11 25	12 3 9 16 25	14 3 7 16 25

[Count(n1=1/A11)] = 48/384]

上のリストでは、 $n1 = 1$ の時の解 48 個しか載せていないが、総数 384 個しか無かった。対称五方陣は全部で 48544 個もあるというのに、「オイラー方陣」は何故にこうも少ないのか。これが何を意味するのかについては、ずっと思案中である。

8. その後の研究成果について

この稿で報告した「完全オイラー方陣」の構成法は、小生にとって最初の成功例である。その後に別のやり方を考えて、もう 2 つ発明して「完全オイラー-五方陣」の構成に成功している：「桂馬跳び構成法」と「新オイラー法」である。

どちらも、今後の数節で報告しているので、ぜひ読んでほしい。やり方は少し複雑だが、実行スピードは、すこぶる速い。この稿のやり方より、数段速い。

だから、この稿は省略・削除しても構わないのだが、小生としては葬るにしのびない。

ぜひ残したい思いで、初稿の趣旨を極力尊重しつつ清書してみた。構成法は幾つあっても良いのではないかと、という気がしている。

次の「完全オイラー-七方陣」の構成こそが真の挑戦である。五次での苦労は、七次で実る。

(初稿 : March 31, 2003; 再計算清書稿 : Feb.19, 2007 and Feb.14, 2015
with MacOSX 10.10.2 and Xcode 6.1.1: by 撰田 寛二(Kanji Setsuda))

E-Mail Address <jag12001@nifty.com>