

Part 4: "New Advanced Study of Magic Squares and Cubes"
 Chapter 4: Commentary Articles No.2 by **Kanji Setsuda**:
 "Various Arts and Tools for Studying Magic Squares"

Section 3-1: How to make 'Complete Euler Squares' of Order 5

#1. What is the 'Complete Euler Square' like?

It is sometimes called "Greco-Latin Square". They say basic idea of that type was mentioned first by Legendary Leonhard Euler(1707-1783). What does it look like?

First of all we have to change our number system from classical one into modern.
 You know about the so-called 'Positional Number System of Base N', don't you?

** Basic Form and our Object Square in Classical Notation **

[Basic Form]					[Pan-diagonal]														
24	25	21	22	23	24	25	21	22	23	3	25	17	14	6	3	25	17	14	6
4	5	1	2	3	4	5	1	2	3	12	9	1	23	20	12	9	1	23	20
9	10	6	7	8	9	10	6	7	8	21	18	15	7	4	21	18	15	7	4
14	15	11	12	13	14	15	11	12	13	10	2	24	16	13	10	2	24	16	13
19	20	16	17	18	19	20	16	17	18	19	11	8	5	22	19	11	8	5	22
24	25	21	22	23	24	25	21	22	23	3	25	17	14	6	3	25	17	14	6
4	5	1	2	3	4	5	1	2	3	12	9	1	23	20	12	9	1	23	20

** Pan-diagonal Magic Square 5x5 in the Modern Notation **

[Pan-diagonal]					[Decomposed by PNS of Base 5]																								
02	44	31	23	10	02	44	31	23	10	0	4	3	2	1	0	4	3	2	1	2	4	1	3	0	2	4	1	3	0
21	13	00	42	34	21	13	00	42	34	2	1	0	4	3	2	1	0	4	3	1	3	0	2	4	1	3	0	2	4
40	32	24	11	03	40	32	24	11	03	4	3	2	1	0	4	3	2	1	0	0	2	4	1	3	0	2	4	1	3
14	01	43	30	22	14	01	43	30	22	1	0	4	3	2	1	0	4	3	2	4	1	3	0	2	4	1	3	0	2
33	20	12	04	41	33	20	12	04	41	3	2	1	0	4	3	2	1	0	4	3	0	2	4	1	3	0	2	4	1
02	44	31	23	10	02	44	31	23	10	0	4	3	2	1	0	4	3	2	1	2	4	1	3	0	2	4	1	3	0
21	13	00	42	34	21	13	00	42	34	2	1	0	4	3	2	1	0	4	3	1	3	0	2	4	1	3	0	2	4

Let me take a system of base 5 for instance. Watch and study the figures above, especially the last two decomposed patterns of 'Complete Euler Square' 5x5.

Each positional layer for high values or low ones is drawn separately as shown above. Any value of 25 numbers is divided by 5. The integer part of quotient is listed in the high layer and the remainder modulo(5) is put in the low one.

According to this system each sum of every column, of every row and even of every pan-diagonal in this example is always calculated by such the same form of equation:

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

The equal sum means magic constant.

Yes. It is certainly an example of 'Complete Euler Square'.

The constant sum 60 of this example is equivalent to 65 in our classical notation,

since modern system uses integers: 0~24, instead of natural numbers: 1~25.

#2. Unique Properties of 'Complete Euler Squares'

The 'Complete Euler Square' has such beautiful properties as follows:

- (1) In each layer every column, every row and every pan-diagonal consists of {0, 1, 2, 3 and 4} and it has neither repetition, nor drop-off of any number.
- (2) Any value of {0, 1, 2, 3 or 4} appears always five times in each layer.
- (3) The value combination of high layer and low one in any location is logically the same with the value of the same location in the figure on the left hand side.

$34(N5i)=3 \times 5 + 4 (=15+4=19(\text{Decimal}))$: equivalent to 20 in Classical Notation)

Neither repetition nor drop-off of a certain combination should be found.

That means any of all integers 0~24 must be used strictly once to make our object.

Check and find that it has such beautiful properties in the example put above.

They say Euler did not mention about pan-diagonals, but I want all pan-diagonals also have this property(1), and I want to call those which has all the miraculous properties above "Complete Euler Squares" from now on.

#3. How to construct such 'Complete Euler Squares'

Can we make such a beautiful square as 'Complete Euler Square' directly?

Why don't you try to dictate our program simply expressing the idea literally after these beautiful definitions themselves?

[1] We usually want the variable **N** to take any value of 0~24 as dictated below.

But why don't you use two variables **N_high** and **N_low**, each of which takes any value of {0, 1, 2, 3 or 4}? Each variable is expected to stand for any value of its own positional layer.

How about dictating such a number generator program with double loops of `for(...){...}` sentences as follows?

```
/* Old */
for(n=0; n<25; n++){
    nm[1]=n;
    // ...;
}
/**/

/* New */
for(a=0; a<5; a++){           // a==d/5: for high
    for(b=0; b<5; b++){       // b==d%5: for low
        d=a*5+b;              // 'd' stands for N itself
        n[1]=d;
        // ...;
    }
}
/**/
```

[2] How do we realize that we must take any value of {0, 1, 2, 3 or 4} strictly once and must not use it twice or more often in any row, any column and any pan-diagonal of our object?

We have usually used the so-called "used_flag" for this purpose.

When you use the value **N**, you should set the used_flag for it as 'true', meaning 'used'. And when you finish using **N**, you should reset the flag as 'false', meaning 'not-used', and may return to the former step.

Whenever you make a new job, you should always check if the used_flag is 'true' or 'false' before all.

```
/* Old */
for(n=0; n<25; n++){
    if(used[n]==0){           // Check if the value N is used or not
        nm[3]=n;              // If it is not used, then ...
    }
}
```

```

    ufl g[n]=1;      // Set Used_flag for the value N
    nextproc();     // Go to the next procedure
    ufl g[n]=0;     // Reset Used_flag for the value N
}
}
/**/

```

Why don't you remake it into the new style below to meet our double structure?

```

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

```

[3] How many sets of used_flags in all do we have to prepare, then?

We have to prepare 10 sets of used_flags for 5 columns high and low, 10 for 5 rows high and low, and 20 for 10 pan-diagonals high and low: We need 40 sets in all!

**** Basic Forms for Pan-diagonal Magic Squares of Order 5 ****



* Basic Conditions: C=60; *

$$\begin{array}{l}
 n_1+n_2+n_3+n_4+n_5=C \quad \dots \text{rw1} \quad | \quad n_1+n_6+n_{11}+n_{16}+n_{21}=C \quad \dots \text{cl 1} \\
 n_6+n_7+n_8+n_9+n_{10}=C \quad \dots \text{rw2} \quad | \quad n_2+n_7+n_{12}+n_{17}+n_{22}=C \quad \dots \text{cl 2} \\
 n_{11}+n_{12}+n_{13}+n_{14}+n_{15}=C \quad \dots \text{rw3} \quad | \quad n_3+n_8+n_{13}+n_{18}+n_{23}=C \quad \dots \text{cl 3} \\
 n_{16}+n_{17}+n_{18}+n_{19}+n_{20}=C \quad \dots \text{rw4} \quad | \quad n_4+n_9+n_{14}+n_{19}+n_{24}=C \quad \dots \text{cl 4} \\
 n_{21}+n_{22}+n_{23}+n_{24}+n_{25}=C \quad \dots \text{rw5} \quad | \quad n_5+n_{10}+n_{15}+n_{20}+n_{25}=C \quad \dots \text{cl 5}
 \end{array}$$

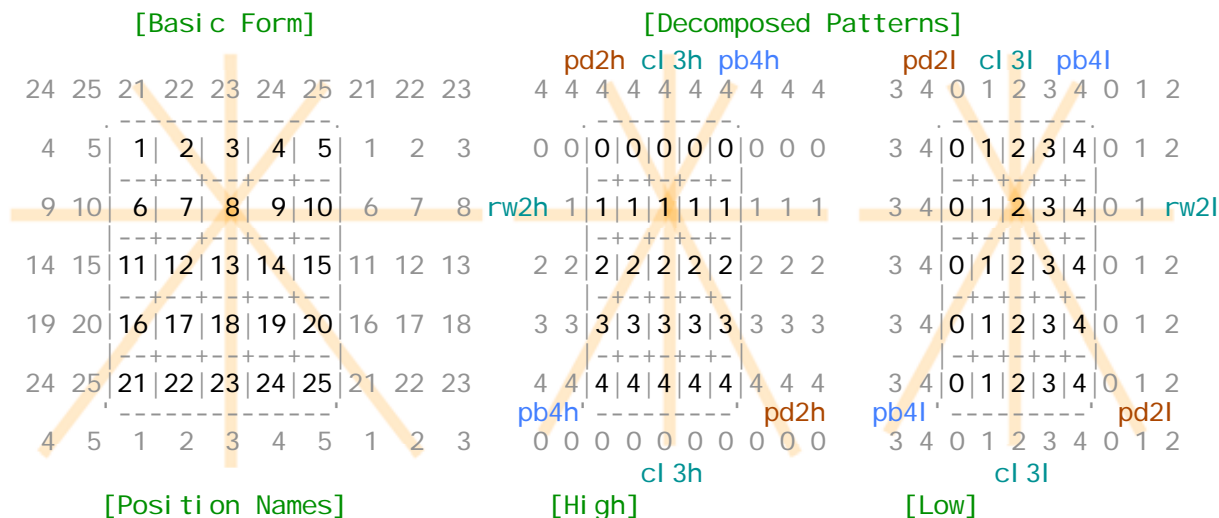
* Pan-diagonal Conditions: C=60; *

$$\begin{array}{l}
 n_1+n_7+n_{13}+n_{19}+n_{25}=C \quad \dots \text{pd1} \quad | \quad n_1+n_{10}+n_{14}+n_{18}+n_{22}=C \quad \dots \text{pb1} \\
 n_2+n_8+n_{14}+n_{20}+n_{21}=C \quad \dots \text{pd2} \quad | \quad n_2+n_6+n_{15}+n_{19}+n_{23}=C \quad \dots \text{pb2}
 \end{array}$$

n3+n9+n15+n16+n22=C ... pd3 | n3+n7+n11+n20+n24=C ... pb3
n4+n10+n11+n17+n23=C ... pd4 | n4+n8+n12+n16+n25=C ... pb4
n5+n6+n12+n18+n24=C ... pd5 | n5+n9+n13+n17+n21=C ... pb5

We have to put their names each by each as shown above.

[4] Which set of used_flags should we check in one procedure when you want to take a certain value for any single variable?



Suppose if you want to take the value of N8 with 7(Decimal) for instance, you should check if the states of {cl3h[1], cl3l[2], rw2h[1], rw2l[2], pd2h[1], pd2l[2], pb4h[1], pb4l[2] and uflag[7]} are all 'false', meaning 'not-used'. You should watch and check 8 flags of 4 lines high and low and check one more flag for d itself.

#4. How to build our Computer Program for 'Complete Euler Square' 5x5

Let me skip my further explanation and come to my conclusion.

I will show you the list of my actual program here. It will tell you how I could make the 3600 pan-diagonal magic squares by my new way of programming for 'Complete Euler Squares'. It may be easier for you experts to understand.

```

/** 'Complete Euler Squares' 5x5 of Pan-Diagonal Magic Type */
/** 'CEuler5PD.c' Built by Kanji Setsuda */
/** on Sep. 9, 2003; Revised on Feb.16, 2007 */
/** Working with MacOSX and Xcode2.2 */
/**/
#include <stdio.h>
/**/
short int cnt;
short LSM, cnt2, cnt3;
short nm[26], uflag[26];
short anm[7][26];
short cl1h[6], rw1h[6], pd1h[6], pb1h[6];
short cl1l[6], rw1l[6], pd1l[6], pb1l[6];
short cl2h[6], rw2h[6], pd2h[6], pb2h[6];
short cl2l[6], rw2l[6], pd2l[6], pb2l[6];
short cl3h[6], rw3h[6], pd3h[6], pb3h[6];
short cl3l[6], rw3l[6], pd3l[6], pb3l[6];
short cl4h[6], rw4h[6], pd4h[6], pb4h[6];
short cl4l[6], rw4l[6], pd4l[6], pb4l[6];
short cl5h[6], rw5h[6], pd5h[6], pb5h[6];
short cl5l[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);
/**/
/* Main Program */
int main(){
short n;
printf("\n** 'Complete Euler Squares' for Pan-Diagonal Magic Type 5x5 **\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; cnt=0; cnt3=0;
stp01();
if(cnt3>0){pr3ans(); }
printf("** [Count = %d] OK! *\n", cnt);
return 0;
}
/* Begin The Search */
/* Set n1 */
void stp01(){
short a, b, d;
for(a=0; a<5; a++){
if((cl1h[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((cl1l[b]==0)&&(rw1l[b]==0)&&(pd1l[b]==0)&&(pb1l[b]==0)){
nm[1]=d; uflg[d]=1; cnt2=0;
cl1h[a]=1; rw1h[a]=1; pd1h[a]=1; pb1h[a]=1;
cl1l[b]=1; rw1l[b]=1; pd1l[b]=1; pb1l[b]=1;
stp02();
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[d]=0; }}
}}
}
}
/* Set n25 & n1<n25 */
void stp02(){
short a, b, d;
for(a=4; a>=0; a--){
if((cl5h[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((cl5l[b]==0)&&(rw5l[b]==0)&&(pd1l[b]==0)&&(pb4l[b]==0)){
nm[25]=d; uflg[d]=1;
cl5h[a]=1; rw5h[a]=1; pd1h[a]=1; pb4h[a]=1;
cl5l[b]=1; rw5l[b]=1; pd1l[b]=1; pb4l[b]=1;
stp03();
cl5h[a]=0; rw5h[a]=0; pd1h[a]=0; pb4h[a]=0;
cl5l[b]=0; rw5l[b]=0; pd1l[b]=0; pb4l[b]=0;
}
}
}
}
}
}

```

```

        ufl g[d]=0; }}
    }}
}
}
/* Set n7 */
void stp03(){
    short a, b, d;
    for(a=0; a<5; a++){
        if((cl 2h[a]==0)&&(rw2h[a]==0)&&(pd1h[a]==0)&&(pb3h[a]==0)){
            for(b=0; b<5; b++){d=a*5+b;
                if(ufl g[d]==0){
                    if((cl 2l [b]==0)&&(rw2l [b]==0)&&(pd1l [b]==0)&&(pb3l [b]==0)){
                        nm[7]=d; ufl g[d]=1;
                        cl 2h[a]=1; rw2h[a]=1; pd1h[a]=1; pb3h[a]=1;
                        cl 2l [b]=1; rw2l [b]=1; pd1l [b]=1; pb3l [b]=1;
                        stp04();
                        cl 2h[a]=0; rw2h[a]=0; pd1h[a]=0; pb3h[a]=0;
                        cl 2l [b]=0; rw2l [b]=0; pd1l [b]=0; pb3l [b]=0;
                        ufl g[d]=0; }}
                    }}
            }
        }
}
/* Set n13 */
void stp04(){
    short a, b, d;
    for(a=0; a<5; a++){
        if((cl 3h[a]==0)&&(rw3h[a]==0)&&(pd1h[a]==0)&&(pb5h[a]==0)){
            for(b=0; b<5; b++){d=a*5+b;
                if(ufl g[d]==0){
                    if((cl 3l [b]==0)&&(rw3l [b]==0)&&(pd1l [b]==0)&&(pb5l [b]==0)){
                        nm[13]=d; ufl g[d]=1;
                        cl 3h[a]=1; rw3h[a]=1; pd1h[a]=1; pb5h[a]=1;
                        cl 3l [b]=1; rw3l [b]=1; pd1l [b]=1; pb5l [b]=1;
                        stp05();
                        cl 3h[a]=0; rw3h[a]=0; pd1h[a]=0; pb5h[a]=0;
                        cl 3l [b]=0; rw3l [b]=0; pd1l [b]=0; pb5l [b]=0;
                        ufl g[d]=0; }}
                    }}
            }
        }
}
/* Set n19=LSM-n1-n7-n13-n25 */
void stp05(){
    short a, b, d;
    for(a=0; a<5; a++){
        if((cl 4h[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)&&(ufl g[d]==0)){
                    if((cl 4l [b]==0)&&(rw4l [b]==0)&&(pd1l [b]==0)&&(pb2l [b]==0)){
                        nm[19]=d; ufl g[d]=1;
                        cl 4h[a]=1; rw4h[a]=1; pd1h[a]=1; pb2h[a]=1;
                        cl 4l [b]=1; rw4l [b]=1; pd1l [b]=1; pb2l [b]=1;
                        stp06();
                        cl 4h[a]=0; rw4h[a]=0; pd1h[a]=0; pb2h[a]=0;
                        cl 4l [b]=0; rw4l [b]=0; pd1l [b]=0; pb2l [b]=0;
                        ufl g[d]=0; }}
                    }}
            }
        }
}
}
/* Set n5 & n1<n5 */
void stp06(){
    short a, b, d;
    for(a=0; a<5; a++){
        if((cl 5h[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((cl5l[b]==0)&&(rw1l[b]==0)&&(pd5l[b]==0)&&(pb5l[b]==0)){
            nm[5]=d; uflg[d]=1;
            cl5h[a]=1; rw1h[a]=1; pd5h[a]=1; pb5h[a]=1;
            cl5l[b]=1; rw1l[b]=1; pd5l[b]=1; pb5l[b]=1;
            stp07();
            cl5h[a]=0; rw1h[a]=0; pd5h[a]=0; pb5h[a]=0;
            cl5l[b]=0; rw1l[b]=0; pd5l[b]=0; pb5l[b]=0;
            uflg[d]=0;}}
    }}
}
}
/* Set n21 & n5<n21 */
void stp07(){
    short a, b, d;
    for(a=4; a>=0; a--){
        if((cl1h[a]==0)&&(rw5h[a]==0)&&(pd2h[a]==0)&&(pb5h[a]==0)){
            for(b=4; b>=0; b--){d=a*5+b;
                if((nm[5]<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;
                        stp08();
                        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 stp08(){
    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;
                        stp09();
                        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 stp09(){
    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;
                        stp10();
                        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;}}
                    }}
            }
        }
}
}
}

```

```

}
}
/* Search Level 2 */
/* Set n2 */
void stp10(){
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;
stp11();
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 */
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(uflg[d]==0){

```

```

    i f((cl 1l [b]==0)&&(rw2l [b]==0)&&(pd5l [b]==0)&&(pb2l [b]==0)){
        nm[6]=d; ufl g[d]=1;
        cl 1h[a]=1; rw2h[a]=1; pd5h[a]=1; pb2h[a]=1;
        cl 1l [b]=1; rw2l [b]=1; pd5l [b]=1; pb2l [b]=1;
        stp14();
        cl 1h[a]=0; rw2h[a]=0; pd5h[a]=0; pb2h[a]=0;
        cl 1l [b]=0; rw2l [b]=0; pd5l [b]=0; pb2l [b]=0;
        ufl g[d]=0;}}
    }}
}
}
/* Set n11 */
void stp14(){
    short a, b, d;
    for(a=0; a<5; a++){
        i f((cl 1h[a]==0)&&(rw3h[a]==0)&&(pd4h[a]==0)&&(pb3h[a]==0)){
            for(b=0; b<5; b++){d=a*5+b;
                i f(ufl g[d]==0){
                    i f((cl 1l [b]==0)&&(rw3l [b]==0)&&(pd4l [b]==0)&&(pb3l [b]==0)){
                        nm[11]=d; ufl g[d]=1;
                        cl 1h[a]=1; rw3h[a]=1; pd4h[a]=1; pb3h[a]=1;
                        cl 1l [b]=1; rw3l [b]=1; pd4l [b]=1; pb3l [b]=1;
                        stp15();
                        cl 1h[a]=0; rw3h[a]=0; pd4h[a]=0; pb3h[a]=0;
                        cl 1l [b]=0; rw3l [b]=0; pd4l [b]=0; pb3l [b]=0;
                        ufl g[d]=0;}}
                    }}
            }
        }
}
/* Set n16=LSM-n1-n6-n11-n21 */
void stp15(){
    short a, b, d;
    for(a=0; a<5; a++){
        i f((cl 1h[a]==0)&&(rw4h[a]==0)&&(pd3h[a]==0)&&(pb4h[a]==0)){
            for(b=0; b<5; b++){d=a*5+b;
                i f((nm[1]+nm[6]+nm[11]+d+nm[21]==LSM)&&(ufl g[d]==0)){
                    i f((cl 1l [b]==0)&&(rw4l [b]==0)&&(pd3l [b]==0)&&(pb4l [b]==0)){
                        nm[16]=d; ufl g[d]=1;
                        cl 1h[a]=1; rw4h[a]=1; pd3h[a]=1; pb4h[a]=1;
                        cl 1l [b]=1; rw4l [b]=1; pd3l [b]=1; pb4l [b]=1;
                        stp16();
                        cl 1h[a]=0; rw4h[a]=0; pd3h[a]=0; pb4h[a]=0;
                        cl 1l [b]=0; rw4l [b]=0; pd3l [b]=0; pb4l [b]=0;
                        ufl g[d]=0;}}
                    }}
            }
        }
}
}
/* Set n8 */
void stp16(){
    short a, b, d;
    for(a=0; a<5; a++){
        i f((cl 3h[a]==0)&&(rw2h[a]==0)&&(pd2h[a]==0)&&(pb4h[a]==0)){
            for(b=0; b<5; b++){d=a*5+b;
                i f(ufl g[d]==0){
                    i f((cl 3l [b]==0)&&(rw2l [b]==0)&&(pd2l [b]==0)&&(pb4l [b]==0)){
                        nm[8]=d; ufl g[d]=1;
                        cl 3h[a]=1; rw2h[a]=1; pd2h[a]=1; pb4h[a]=1;
                        cl 3l [b]=1; rw2l [b]=1; pd2l [b]=1; pb4l [b]=1;
                        stp17();
                        cl 3h[a]=0; rw2h[a]=0; pd2h[a]=0; pb4h[a]=0;
                        cl 3l [b]=0; rw2l [b]=0; pd2l [b]=0; pb4l [b]=0;
                        ufl g[d]=0;}}
                    }}
            }
        }
}
}

```

```

}
/* Set n12=LSM-n4-n8-n16-n25 */
void stp17(){
short a, b, d;
for(a=0; a<5; a++){
if((cl2h[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((cl2l[b]==0)&&(rw3l[b]==0)&&(pd5l[b]==0)&&(pb4l[b]==0)){
nm[12]=d; uflg[d]=1;
cl2h[a]=1; rw3h[a]=1; pd5h[a]=1; pb4h[a]=1;
cl2l[b]=1; rw3l[b]=1; pd5l[b]=1; pb4l[b]=1;
stp18();
cl2h[a]=0; rw3h[a]=0; pd5h[a]=0; pb4h[a]=0;
cl2l[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((cl5h[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((cl5l[b]==0)&&(rw2l[b]==0)&&(pd4l[b]==0)&&(pb1l[b]==0)){
nm[10]=d; uflg[d]=1;
cl5h[a]=1; rw2h[a]=1; pd4h[a]=1; pb1h[a]=1;
cl5l[b]=1; rw2l[b]=1; pd4l[b]=1; pb1l[b]=1;
stp19();
cl5h[a]=0; rw2h[a]=0; pd4h[a]=0; pb1h[a]=0;
cl5l[b]=0; rw2l[b]=0; pd4l[b]=0; pb1l[b]=0;
uflg[d]=0; }}
}}
}
}
/* Set n22=LSM-n2-n7-n12-n17 */
void stp19(){
short a, b, d;
for(a=0; a<5; a++){
if((cl2h[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((cl2l[b]==0)&&(rw5l[b]==0)&&(pd3l[b]==0)&&(pb1l[b]==0)){
nm[22]=d; uflg[d]=1;
cl2h[a]=1; rw5h[a]=1; pd3h[a]=1; pb1h[a]=1;
cl2l[b]=1; rw5l[b]=1; pd3l[b]=1; pb1l[b]=1;
stp20();
cl2h[a]=0; rw5h[a]=0; pd3h[a]=0; pb1h[a]=0;
cl2l[b]=0; rw5l[b]=0; pd3l[b]=0; pb1l[b]=0;
uflg[d]=0; }}
}}
}
}
/* Search Level 3 */
/* Set n23=LSM-n4-n10-n11-n17 */
void stp20(){
short a, b, d;
for(a=0; a<5; a++){
if((cl3h[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((cl3l[b]==0)&&(rw5l[b]==0)&&(pd4l[b]==0)&&(pb2l[b]==0)){

```

```

        nm[23]=d; ufl g[d]=1;
        cl 3h[a]=1; rw5h[a]=1; pd4h[a]=1; pb2h[a]=1;
        cl 3l [b]=1; rw5l [b]=1; pd4l [b]=1; pb2l [b]=1;
        stp21();
        cl 3h[a]=0; rw5h[a]=0; pd4h[a]=0; pb2h[a]=0;
        cl 3l [b]=0; rw5l [b]=0; pd4l [b]=0; pb2l [b]=0;
        ufl g[d]=0; }}
    }}
}
}
/* Set n18=LSM-n3-n8-n13-n23 */
void stp21(){
short a, b, d;
for(a=0; a<5; a++){
    if((cl 3h[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)&&(ufl g[d]==0)){
                if((cl 3l [b]==0)&&(rw4l [b]==0)&&(pd5l [b]==0)&&(pb1l [b]==0)){
                    nm[18]=d; ufl g[d]=1;
                    cl 3h[a]=1; rw4h[a]=1; pd5h[a]=1; pb1h[a]=1;
                    cl 3l [b]=1; rw4l [b]=1; pd5l [b]=1; pb1l [b]=1;
                    stp22();
                    cl 3h[a]=0; rw4h[a]=0; pd5h[a]=0; pb1h[a]=0;
                    cl 3l [b]=0; rw4l [b]=0; pd5l [b]=0; pb1l [b]=0;
                    ufl g[d]=0; }}
                }}
        }
}
/* Set n14=LSM-n1-n10-n18-n22 */
void stp22(){
short a, b, d;
for(a=0; a<5; a++){
    if((cl 4h[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)&&(ufl g[d]==0)){
                if((cl 4l [b]==0)&&(rw3l [b]==0)&&(pd2l [b]==0)&&(pb1l [b]==0)){
                    nm[14]=d; ufl g[d]=1;
                    cl 4h[a]=1; rw3h[a]=1; pd2h[a]=1; pb1h[a]=1;
                    cl 4l [b]=1; rw3l [b]=1; pd2l [b]=1; pb1l [b]=1;
                    stp23();
                    cl 4h[a]=0; rw3h[a]=0; pd2h[a]=0; pb1h[a]=0;
                    cl 4l [b]=0; rw3l [b]=0; pd2l [b]=0; pb1l [b]=0;
                    ufl g[d]=0; }}
                }}
        }
}
}
/* Set n15=LSM-n3-n9-n16-n22 */
void stp23(){
short a, b, d;
for(a=0; a<5; a++){
    if((cl 5h[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)&&(ufl g[d]==0)){
                if((cl 5l [b]==0)&&(rw3l [b]==0)&&(pd3l [b]==0)&&(pb2l [b]==0)){
                    nm[15]=d; ufl g[d]=1;
                    cl 5h[a]=1; rw3h[a]=1; pd3h[a]=1; pb2h[a]=1;
                    cl 5l [b]=1; rw3l [b]=1; pd3l [b]=1; pb2l [b]=1;
                    stp24();
                    cl 5h[a]=0; rw3h[a]=0; pd3h[a]=0; pb2h[a]=0;
                    cl 5l [b]=0; rw3l [b]=0; pd3l [b]=0; pb2l [b]=0;
                    ufl g[d]=0; }}
                }}
        }
}
}
}

```

```

/* Set n20=LSM-n2-n8-n14-n21 */
void stp24(){
short a, b, d;
for(a=0; a<5; a++){
if((cl5h[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((cl5l[b]==0)&&(rw4l[b]==0)&&(pd2l[b]==0)&&(pb3l[b]==0)){
nm[20]=d; uflg[d]=1;
cl5h[a]=1; rw4h[a]=1; pd2h[a]=1; pb3h[a]=1;
cl5l[b]=1; rw4l[b]=1; pd2l[b]=1; pb3l[b]=1;
stp25();
cl5h[a]=0; rw4h[a]=0; pd2h[a]=0; pb3h[a]=0;
cl5l[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((cl4h[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;
ansprnt();
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 ansprnt(){
short n;
cnt++; 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, l5, 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");
}
}
/**/

```

#5. Result List of My Recent Computation

** 'Complete Euler Squares' for Pan-Diagonal Magic Type 5x5 (Part) **

/D5i	1#	/D5i	5#	/D5i	9#
04321 02413	1 23 20 12 9	04321 01423	1 22 20 13 9	04321 01432	1 22 20 14 8
21043 41302	15 7 4 21 18	21043 42301	15 8 4 21 17	21043 43201	15 9 3 21 17
43210 30241	24 16 13 10 2	43210 30142	24 16 12 10 3	43210 20143	23 16 12 10 4
10432 24130	8 5 22 19 11	10432 14230	7 5 23 19 11	10432 14320	7 5 24 18 11
32104 13024	17 14 6 3 25	32104 23014	18 14 6 2 25	32104 32014	19 13 6 2 25
/D5i	13#	/D5i	17#	/D5i	21#
04312 02413	1 23 20 7 14	04312 01423	1 22 20 8 14	04312 01432	1 22 20 9 13
12043 41302	10 12 4 21 18	12043 42301	10 13 4 21 17	12043 43201	10 14 3 21 17
43120 30241	24 16 8 15 2	43120 30142	24 16 7 15 3	43120 20143	23 16 7 15 4
20431 24130	13 5 22 19 6	20431 14230	12 5 23 19 6	20431 14320	12 5 24 18 6
31204 13024	17 9 11 3 25	31204 23014	18 9 11 2 25	31204 32014	19 8 11 2 25
/D5i	25#	/D5i	29#	/D5i	33#
01432 04321	1 10 24 18 12	01432 04312	1 10 24 17 13	01432 04213	1 10 23 17 14
43201 21043	23 17 11 5 9	43201 12043	22 18 11 5 9	43201 13042	22 19 11 5 8
20143 43210	15 4 8 22 16	20143 43120	15 4 7 23 16	20143 42130	15 3 7 24 16
14320 10432	7 21 20 14 3	14320 20431	8 21 20 14 2	14320 30421	9 21 20 13 2
32014 32104	19 13 2 6 25	32014 31204	19 12 3 6 25	32014 21304	18 12 4 6 25
/D5i	37#	/D5i	41#	/D5i	45#
04321 02314	1 23 19 12 10	04321 01324	1 22 19 13 10	04321 01342	1 22 19 15 8
21043 31402	14 7 5 21 18	21043 32401	14 8 5 21 17	21043 34201	14 10 3 21 17
43210 40231	25 16 13 9 2	43210 40132	25 16 12 9 3	43210 20134	23 16 12 9 5
10432 23140	8 4 22 20 11	10432 13240	7 4 23 20 11	10432 13420	7 4 25 18 11
32104 14023	17 15 6 3 24	32104 24013	18 15 6 2 24	32104 42013	20 13 6 2 24
/D5i	49#	/D5i	53#	/D5i	57#
04312 02314	1 23 19 7 15	04312 01324	1 22 19 8 15	04312 01342	1 22 19 10 13
12043 31402	9 12 5 21 18	12043 32401	9 13 5 21 17	12043 34201	9 15 3 21 17
43120 40231	25 16 8 14 2	43120 40132	25 16 7 14 3	43120 20134	23 16 7 14 5
20431 23140	13 4 22 20 6	20431 13240	12 4 23 20 6	20431 13420	12 4 25 18 6
31204 14023	17 10 11 3 24	31204 24013	18 10 11 2 24	31204 42013	20 8 11 2 24
/D5i	61#	/D5i	65#	/D5i	69#
01432 03421	1 9 25 18 12	01432 03412	1 9 25 17 13	01432 03214	1 9 23 17 15
43201 21034	23 17 11 4 10	43201 12034	22 18 11 4 10	43201 14032	22 20 11 4 8
20143 34210	14 5 8 22 16	20143 34120	14 5 7 23 16	20143 32140	14 3 7 25 16
14320 10342	7 21 19 15 3	14320 20341	8 21 19 15 2	14320 40321	10 21 19 13 2
32014 42103	20 13 2 6 24	32014 41203	20 12 3 6 24	32014 21403	18 12 5 6 24
/D5i	73#	/D5i	77#	/D5i	81#
04321 03214	1 24 18 12 10	04321 01234	1 22 18 14 10	04321 01243	1 22 18 15 9
21043 21403	13 7 5 21 19	21043 23401	13 9 5 21 17	21043 24301	13 10 4 21 17
43210 40321	25 16 14 8 2	43210 40123	25 16 12 8 4	43210 30124	24 16 12 8 5
10432 32140	9 3 22 20 11	10432 12340	7 3 24 20 11	10432 12430	7 3 25 19 11
32104 14032	17 15 6 4 23	32104 34012	19 15 6 2 23	32104 43012	20 14 6 2 23

/D5i	85#	/D5i	89#	/D5i	93#
04312 03214 1 24 18 7 15		04312 01234 1 22 18 9 15		04312 01243 1 22 18 10 14	
12043 21403 8 12 5 21 19		12043 23401 8 14 5 21 17		12043 24301 8 15 4 21 17	
43120 40321 25 16 9 13 2		43120 40123 25 16 7 13 4		43120 30124 24 16 7 13 5	
20431 32140 14 3 22 20 6		20431 12340 12 3 24 20 6		20431 12430 12 3 25 19 6	
31204 14032 17 10 11 4 23		31204 34012 19 10 11 2 23		31204 43012 20 9 11 2 23	
/D5i	97#	/D5i	101#	/D5i	105#
01432 02431 1 8 25 19 12		01432 02413 1 8 25 17 14		01432 02314 1 8 24 17 15	
43201 31024 24 17 11 3 10		43201 13024 22 19 11 3 10		43201 14023 22 20 11 3 9	
20143 24310 13 5 9 22 16		20143 24130 13 5 7 24 16		20143 23140 13 4 7 25 16	
14320 10243 7 21 18 15 4		14320 30241 9 21 18 15 2		14320 40231 10 21 18 14 2	
32014 43102 20 14 2 6 23		32014 41302 20 12 4 6 23		32014 31402 19 12 5 6 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		02431 02413 1 13 25 17 9	
21034 21403 13 7 5 16 24		21034 12403 12 8 5 16 24		31024 41302 20 7 4 11 23	
34210 40321 20 21 14 8 2		34210 40312 20 21 14 7 3		24310 30241 14 21 18 10 2	
10342 32140 9 3 17 25 11		10342 31240 9 2 18 25 11		10243 24130 8 5 12 24 16	
42103 14032 22 15 6 4 18		42103 24031 23 15 6 4 17		43102 13024 22 19 6 3 15	
/D5i	325#	/D5i	361#	/D5i	397#
02431 02314 1 13 24 17 10		02431 03214 1 14 23 17 10		02431 03124 1 14 22 18 10	
31024 31402 19 7 5 11 23		31024 21403 18 7 5 11 24		31024 12403 17 8 5 11 24	
24310 40231 15 21 18 9 2		24310 40321 15 21 19 8 2		24310 40312 15 21 19 7 3	
10243 23140 8 4 12 25 16		10243 32140 9 3 12 25 16		10243 31240 9 2 13 25 16	
43102 14023 22 20 6 3 14		43102 14032 22 20 6 4 13		43102 24031 23 20 6 4 12	
/D5i	433#	/D5i	469#	/D5i	505#
01432 02413 1 8 25 17 14		01432 02314 1 8 24 17 15		01432 03214 1 9 23 17 15	
32014 41302 20 12 4 6 23		32014 31402 19 12 5 6 23		32014 21403 18 12 5 6 24	
14320 30241 9 21 18 15 2		14320 40231 10 21 18 14 2		14320 40321 10 21 19 13 2	
20143 24130 13 5 7 24 16		20143 23140 13 4 7 25 16		20143 32140 14 3 7 25 16	
43201 13024 22 19 11 3 10		43201 14023 22 20 11 3 9		43201 14032 22 20 11 4 8	
/D5i	541#	/D5i	577#	/D5i	613#
01432 03124 1 9 22 18 15		04321 12403 2 23 20 11 9		04321 12304 2 23 19 11 10	
32014 12403 17 13 5 6 24		21043 40312 15 6 4 22 18		21043 30412 14 6 5 22 18	
14320 40312 10 21 19 12 3		43210 31240 24 17 13 10 1		43210 41230 25 17 13 9 1	
20143 31240 14 2 8 25 16		10432 24031 8 5 21 19 12		10432 23041 8 4 21 20 12	
43201 24031 23 20 11 4 7		32104 03124 16 14 7 3 25		32104 04123 16 15 7 3 24	
/D5i	649#	/D5i	685#	/D5i	721#
04321 13204 2 24 18 11 10		04321 13024 2 24 16 13 10		03421 12403 2 18 25 11 9	
21043 20413 13 6 5 22 19		21043 02413 11 8 5 22 19		21034 40312 15 6 4 17 23	
43210 41320 25 17 14 8 1		43210 41302 25 17 14 6 3		34210 31240 19 22 13 10 1	
10432 32041 9 3 21 20 12		10432 30241 9 1 23 20 12		10342 24031 8 5 16 24 12	
32104 04132 16 15 7 4 23		32104 24130 18 15 7 4 21		42103 03124 21 14 7 3 20	
/D5i	757#	/D5i	793#	/D5i	829#
03421 12304 2 18 24 11 10		03421 13204 2 19 23 11 10		03421 13024 2 19 21 13 10	
21034 30412 14 6 5 17 23		21034 20413 13 6 5 17 24		21034 02413 11 8 5 17 24	
34210 41230 20 22 13 9 1		34210 41320 20 22 14 8 1		34210 41302 20 22 14 6 3	
10342 23041 8 4 16 25 12		10342 32041 9 3 16 25 12		10342 30241 9 1 18 25 12	
42103 04123 21 15 7 3 19		42103 04132 21 15 7 4 18		42103 24130 23 15 7 4 16	
/D5i	865#	/D5i	901#	/D5i	937#
02431 12403 2 13 25 16 9		02431 12304 2 13 24 16 10		02431 13204 2 14 23 16 10	
31024 40312 20 6 4 12 23		31024 30412 19 6 5 12 23		31024 20413 18 6 5 12 24	
24310 31240 14 22 18 10 1		24310 41230 15 22 18 9 1		24310 41320 15 22 19 8 1	
10243 24031 8 5 11 24 17		10243 23041 8 4 11 25 17		10243 32041 9 3 11 25 17	
43102 03124 21 19 7 3 15		43102 04123 21 20 7 3 14		43102 04132 21 20 7 4 13	

/D5i	973#	/D5i	1009#	/D5i	1045#
02431 13024 2 14 21 18 10		01432 12403 2 8 25 16 14		01432 12304 2 8 24 16 15	
31024 02413 16 8 5 12 24		32014 40312 20 11 4 7 23		32014 30412 19 11 5 7 23	
24310 41302 15 22 19 6 3		14320 31240 9 22 18 15 1		14320 41230 10 22 18 14 1	
10243 30241 9 1 13 25 17		20143 24031 13 5 6 24 17		20143 23041 13 4 6 25 17	
43102 24130 23 20 7 4 11		43201 03124 21 19 12 3 10		43201 04123 21 20 12 3 9	
/D5i	1081#	/D5i	1117#	/D5i	1153#
01432 13204 2 9 23 16 15		01432 13024 2 9 21 18 15		04321 21403 3 22 20 11 9	
32014 20413 18 11 5 7 24		32014 02413 16 13 5 7 24		21043 40321 15 6 4 23 17	
14320 41320 10 22 19 13 1		14320 41302 10 22 19 11 3		43210 32140 24 18 12 10 1	
20143 32041 14 3 6 25 17		20143 30241 14 1 8 25 17		10432 14032 7 5 21 19 13	
43201 04132 21 20 12 4 8		43201 24130 23 20 12 4 6		32104 03214 16 14 8 2 25	
/D5i	1189#	/D5i	1225#	/D5i	1261#
04321 21304 3 22 19 11 10		04321 23104 3 24 17 11 10		04321 23014 3 24 16 12 10	
21043 30421 14 6 5 23 17		21043 10423 12 6 5 23 19		21043 01423 11 7 5 23 19	
43210 42130 25 18 12 9 1		43210 42310 25 18 14 7 1		43210 42301 25 18 14 6 2	
10432 13042 7 4 21 20 13		10432 31042 9 2 21 20 13		10432 30142 9 1 22 20 13	
32104 04213 16 15 8 2 24		32104 04231 16 15 8 4 22		32104 14230 17 15 8 4 21	
/D5i	1297#	/D5i	1333#	/D5i	1369#
03421 21403 3 17 25 11 9		03421 21304 3 17 24 11 10		03421 23104 3 19 22 11 10	
21034 40321 15 6 4 18 22		21034 30421 14 6 5 18 22		21034 10423 12 6 5 18 24	
34210 32140 19 23 12 10 1		34210 42130 20 23 12 9 1		34210 42310 20 23 14 7 1	
10342 14032 7 5 16 24 13		10342 13042 7 4 16 25 13		10342 31042 9 2 16 25 13	
42103 03214 21 14 8 2 20		42103 04213 21 15 8 2 19		42103 04231 21 15 8 4 17	
/D5i	1405#	/D5i	1441#	/D5i	1477#
03421 23014 3 19 21 12 10		02431 21403 3 12 25 16 9		02431 21304 3 12 24 16 10	
21034 01423 11 7 5 18 24		31024 40321 20 6 4 13 22		31024 30421 19 6 5 13 22	
34210 42301 20 23 14 6 2		24310 32140 14 23 17 10 1		24310 42130 15 23 17 9 1	
10342 30142 9 1 17 25 13		10243 14032 7 5 11 24 18		10243 13042 7 4 11 25 18	
42103 14230 22 15 8 4 16		43102 03214 21 19 8 2 15		43102 04213 21 20 8 2 14	
/D5i	1513#	/D5i	1549#	/D5i	1585#
02431 23104 3 14 22 16 10		02431 23014 3 14 21 17 10		01432 21403 3 7 25 16 14	
31024 10423 17 6 5 13 24		31024 01423 16 7 5 13 24		32014 40321 20 11 4 8 22	
24310 42310 15 23 19 7 1		24310 42301 15 23 19 6 2		14320 32140 9 23 17 15 1	
10243 31042 9 2 11 25 18		10243 30142 9 1 12 25 18		20143 14032 12 5 6 24 18	
43102 04231 21 20 8 4 12		43102 14230 22 20 8 4 11		43201 03214 21 19 13 2 10	
/D5i	1621#	/D5i	1657#	/D5i	1693#
01432 21304 3 7 24 16 15		01432 23104 3 9 22 16 15		01432 23014 3 9 21 17 15	
32014 30421 19 11 5 8 22		32014 10423 17 11 5 8 24		32014 01423 16 12 5 8 24	
14320 42130 10 23 17 14 1		14320 42310 10 23 19 12 1		14320 42301 10 23 19 11 2	
20143 13042 12 4 6 25 18		20143 31042 14 2 6 25 18		20143 30142 14 1 7 25 18	
43201 04213 21 20 13 2 9		43201 04231 21 20 13 4 7		43201 14230 22 20 13 4 6	
/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					

* [Count = 3600] OK! *


```

/** Magic Type: Both Self-Complementary and Pan-Diagonal **/
/** File: 'CEuler5SDP2.c' built by Kanji Setsuda **/
/** on Apr. 22, 2003; Revised on Feb. 16, 2007 **/
/** Working with MacOSX and Xcode2.2 **/
/**/
#include <stdio.h>
/**/
short int cnt, cnt3;
short LSM, CC;
short nm[26], uflg[26];
short anm[5][26];
short cl1h[6], rw1h[6], pd1h[6], pb1h[6];
short cl1l[6], rw1l[6], pd1l[6], pb1l[6];
short cl2h[6], rw2h[6], pd2h[6], pb2h[6];
short cl2l[6], rw2l[6], pd2l[6], pb2l[6];
short cl3h[6], rw3h[6], pd3h[6], pb3h[6];
short cl3l[6], rw3l[6], pd3l[6], pb3l[6];
short cl4h[6], rw4h[6], pd4h[6], pb4h[6];
short cl4l[6], rw4l[6], pd4l[6], pb4l[6];
short cl5h[6], rw5h[6], pd5h[6], pb5h[6];
short cl5l[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), pr2ans(void);
/**/
/* Main Program */
int main(){
short n;
printf("\n** 'Complete Euler Squares' of Simultaneous Type: **\n");
printf("*** Both Self-Complementary and Pan-Diagonal 5x5 ***\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();
if(cnt3>0){pr2ans();}
printf("** [Count = %d] OK! *\n", cnt);
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((cl2h[a]==0)&&(rw2h[a]==0)&&(pd1h[a]==0)&&(pb3h[a]==0)){
            if((cl4h[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((cl2l[b]==0)&&(rw2l[b]==0)&&(pd1l[b]==0)&&(pb3l[b]==0)){
                            if((cl4l[nb]==0)&&(rw4l[nb]==0)&&(pd1l[nb]==0)&&(pb2l[nb]==0)){
                                nm[7]=d; nm[19]=nd; uflg[d]=1; uflg[nd]=1;
                                cl2h[a]=1; rw2h[a]=1; pd1h[a]=1; pb3h[a]=1;
                                cl2l[b]=1; rw2l[b]=1; pd1l[b]=1; pb3l[b]=1;
                                cl4h[na]=1; rw4h[na]=1; pd1h[na]=1; pb2h[na]=1;
                                cl4l[nb]=1; rw4l[nb]=1; pd1l[nb]=1; pb2l[nb]=1;
                                stp03();
                                cl4h[na]=0; rw4h[na]=0; pd1h[na]=0; pb2h[na]=0;
                                cl4l[nb]=0; rw4l[nb]=0; pd1l[nb]=0; pb2l[nb]=0;
                                cl2h[a]=0; rw2h[a]=0; pd1h[a]=0; pb3h[a]=0;
                                cl2l[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((cl5h[a]==0)&&(rw1h[a]==0)&&(pd5h[a]==0)&&(pb5h[a]==0)){
            if((cl1h[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((cl5l[b]==0)&&(rw1l[b]==0)&&(pd5l[b]==0)&&(pb5l[b]==0)){
                            if((cl1l[nb]==0)&&(rw5l[nb]==0)&&(pd2l[nb]==0)&&(pb5l[nb]==0)){
                                nm[5]=d; nm[21]=nd; uflg[d]=1; uflg[nd]=1;
                                cl5h[a]=1; rw1h[a]=1; pd5h[a]=1; pb5h[a]=1;
                                cl5l[b]=1; rw1l[b]=1; pd5l[b]=1; pb5l[b]=1;
                                cl1h[na]=1; rw5h[na]=1; pd2h[na]=1; pb5h[na]=1;
                                cl1l[nb]=1; rw5l[nb]=1; pd2l[nb]=1; pb5l[nb]=1;
                                stp04();
                                cl1h[na]=0; rw5h[na]=0; pd2h[na]=0; pb5h[na]=0;
                                cl1l[nb]=0; rw5l[nb]=0; pd2l[nb]=0; pb5l[nb]=0;
                                cl5h[a]=0; rw1h[a]=0; pd5h[a]=0; pb5h[a]=0;
                                cl5l[b]=0; rw1l[b]=0; pd5l[b]=0; pb5l[b]=0;
                                uflg[nd]=0; uflg[d]=0;}}}}
                    }}}
    }
}
}

```



```

        stp07();
        cl 3h[na]=0; rw5h[na]=0; pd4h[na]=0; pb2h[na]=0;
        cl 3l[nb]=0; rw5l[nb]=0; pd4l[nb]=0; pb2l[nb]=0;
        cl 3h[a]=0; rw1h[a]=0; pd3h[a]=0; pb3h[a]=0;
        cl 3l[b]=0; rw1l[b]=0; pd3l[b]=0; pb3l[b]=0;
        ufl g[nd]=0; ufl g[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((cl 4h[a]==0)&&(rw1h[a]==0)&&(pd4h[a]==0)&&(pb4h[a]==0)){
            if((cl 2h[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)&&(ufl g[d]==0)&&(ufl g[nd]==0)){
                        if((cl 4l[b]==0)&&(rw1l[b]==0)&&(pd4l[b]==0)&&(pb4l[b]==0)){
                            if((cl 2l[nb]==0)&&(rw5l[nb]==0)&&(pd3l[nb]==0)&&(pb1l[nb]==0)){
                                nm[4]=d; nm[22]=nd; ufl g[d]=1; ufl g[nd]=1;
                                cl 4h[a]=1; rw1h[a]=1; pd4h[a]=1; pb4h[a]=1;
                                cl 4l[b]=1; rw1l[b]=1; pd4l[b]=1; pb4l[b]=1;
                                cl 2h[na]=1; rw5h[na]=1; pd3h[na]=1; pb1h[na]=1;
                                cl 2l[nb]=1; rw5l[nb]=1; pd3l[nb]=1; pb1l[nb]=1;
                                stp08();
                                cl 2h[na]=0; rw5h[na]=0; pd3h[na]=0; pb1h[na]=0;
                                cl 2l[nb]=0; rw5l[nb]=0; pd3l[nb]=0; pb1l[nb]=0;
                                cl 4h[a]=0; rw1h[a]=0; pd4h[a]=0; pb4h[a]=0;
                                cl 4l[b]=0; rw1l[b]=0; pd4l[b]=0; pb4l[b]=0;
                                ufl g[nd]=0; ufl g[d]=0; }}}
                            }}}
                    }
                }
            }
        }
    }
}
/* Set n6 & n20 & n6<n2 */
void stp08(){
    short a, na, b, nb, d, nd;
    for(a=0; a<5; a++){na=4-a;
        if((cl 1h[a]==0)&&(rw2h[a]==0)&&(pd5h[a]==0)&&(pb2h[a]==0)){
            if((cl 5h[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])&&(ufl g[d]==0)&&(ufl g[nd]==0)){
                        if((cl 1l[b]==0)&&(rw2l[b]==0)&&(pd5l[b]==0)&&(pb2l[b]==0)){
                            if((cl 5l[nb]==0)&&(rw4l[nb]==0)&&(pd2l[nb]==0)&&(pb3l[nb]==0)){
                                nm[6]=d; nm[20]=nd; ufl g[d]=1; ufl g[nd]=1;
                                cl 1h[a]=1; rw2h[a]=1; pd5h[a]=1; pb2h[a]=1;
                                cl 1l[b]=1; rw2l[b]=1; pd5l[b]=1; pb2l[b]=1;
                                cl 5h[na]=1; rw4h[na]=1; pd2h[na]=1; pb3h[na]=1;
                                cl 5l[nb]=1; rw4l[nb]=1; pd2l[nb]=1; pb3l[nb]=1;
                                stp09();
                                cl 5h[na]=0; rw4h[na]=0; pd2h[na]=0; pb3h[na]=0;
                                cl 5l[nb]=0; rw4l[nb]=0; pd2l[nb]=0; pb3l[nb]=0;
                                cl 1h[a]=0; rw2h[a]=0; pd5h[a]=0; pb2h[a]=0;
                                cl 1l[b]=0; rw2l[b]=0; pd5l[b]=0; pb2l[b]=0;
                                ufl g[nd]=0; ufl g[d]=0; }}}
                            }}}
                    }
                }
            }
        }
    }
}
/* Set n11=LSM-n3-n7-n20-n24 & n15 */
void stp09(){
    short a, na, b, nb, d, nd;
    for(a=0; a<5; a++){na=4-a;
        if((cl 1h[a]==0)&&(rw3h[a]==0)&&(pd4h[a]==0)&&(pb3h[a]==0)){
            if((cl 5h[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((cl1l[b]==0)&&(rw3l[b]==0)&&(pd4l[b]==0)&&(pb3l[b]==0)){
    if((cl5l[nb]==0)&&(rw3l[nb]==0)&&(pd3l[nb]==0)&&(pb2l[nb]==0)){
    nm[11]=d; nm[15]=nd; uflg[d]=1; uflg[nd]=1;
    cl1h[a]=1; rw3h[a]=1; pd4h[a]=1; pb3h[a]=1;
    cl1l[b]=1; rw3l[b]=1; pd4l[b]=1; pb3l[b]=1;
    cl5h[na]=1; rw3h[na]=1; pd3h[na]=1; pb2h[na]=1;
    cl5l[nb]=1; rw3l[nb]=1; pd3l[nb]=1; pb2l[nb]=1;
    stp10();
    cl5h[na]=0; rw3h[na]=0; pd3h[na]=0; pb2h[na]=0;
    cl5l[nb]=0; rw3l[nb]=0; pd3l[nb]=0; pb2l[nb]=0;
    cl1h[a]=0; rw3h[a]=0; pd4h[a]=0; pb3h[a]=0;
    cl1l[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((cl1h[a]==0)&&(rw4h[a]==0)&&(pd3h[a]==0)&&(pb4h[a]==0)){
if((cl5h[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((cl1l[b]==0)&&(rw4l[b]==0)&&(pd3l[b]==0)&&(pb4l[b]==0)){
if((cl5l[nb]==0)&&(rw2l[nb]==0)&&(pd4l[nb]==0)&&(pb1l[nb]==0)){
nm[16]=d; nm[10]=nd; uflg[d]=1; uflg[nd]=1;
cl1h[a]=1; rw4h[a]=1; pd3h[a]=1; pb4h[a]=1;
cl1l[b]=1; rw4l[b]=1; pd3l[b]=1; pb4l[b]=1;
cl5h[na]=1; rw2h[na]=1; pd4h[na]=1; pb1h[na]=1;
cl5l[nb]=1; rw2l[nb]=1; pd4l[nb]=1; pb1l[nb]=1;
stp11();
cl5h[na]=0; rw2h[na]=0; pd4h[na]=0; pb1h[na]=0;
cl5l[nb]=0; rw2l[nb]=0; pd4l[nb]=0; pb1l[nb]=0;
cl1h[a]=0; rw4h[a]=0; pd3h[a]=0; pb4h[a]=0;
cl1l[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((cl3h[a]==0)&&(rw2h[a]==0)&&(pd2h[a]==0)&&(pb4h[a]==0)){
if((cl3h[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((cl3l[b]==0)&&(rw2l[b]==0)&&(pd2l[b]==0)&&(pb4l[b]==0)){
if((cl3l[nb]==0)&&(rw4l[nb]==0)&&(pd5l[nb]==0)&&(pb1l[nb]==0)){
nm[8]=d; nm[18]=nd; uflg[d]=1; uflg[nd]=1;
cl3h[a]=1; rw2h[a]=1; pd2h[a]=1; pb4h[a]=1;
cl3l[b]=1; rw2l[b]=1; pd2l[b]=1; pb4l[b]=1;
cl3h[na]=1; rw4h[na]=1; pd5h[na]=1; pb1h[na]=1;
cl3l[nb]=1; rw4l[nb]=1; pd5l[nb]=1; pb1l[nb]=1;
stp12();
cl3h[na]=0; rw4h[na]=0; pd5h[na]=0; pb1h[na]=0;
cl3l[nb]=0; rw4l[nb]=0; pd5l[nb]=0; pb1l[nb]=0;
cl3h[a]=0; rw2h[a]=0; pd2h[a]=0; pb4h[a]=0;
cl3l[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((cl 2h[a]==0)&&(rw3h[a]==0)&&(pd5h[a]==0)&&(pb4h[a]==0)){
if((cl 4h[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)&&(ufl g[d]==0)&&(ufl g[nd]==0)){
if((cl 2l [b]==0)&&(rw3l [b]==0)&&(pd5l [b]==0)&&(pb4l [b]==0)){
if((cl 4l [nb]==0)&&(rw3l [nb]==0)&&(pd2l [nb]==0)&&(pb1l [nb]==0)){
nm[12]=d; nm[14]=nd; ufl g[d]=1; ufl g[nd]=1;
cl 2h[a]=1; rw3h[a]=1; pd5h[a]=1; pb4h[a]=1;
cl 2l [b]=1; rw3l [b]=1; pd5l [b]=1; pb4l [b]=1;
cl 4h[na]=1; rw3h[na]=1; pd2h[na]=1; pb1h[na]=1;
cl 4l [nb]=1; rw3l [nb]=1; pd2l [nb]=1; pb1l [nb]=1;
anspri nt();
cl 4h[na]=0; rw3h[na]=0; pd2h[na]=0; pb1h[na]=0;
cl 4l [nb]=0; rw3l [nb]=0; pd2l [nb]=0; pb1l [nb]=0;
cl 2h[a]=0; rw3h[a]=0; pd5h[a]=0; pb4h[a]=0;
cl 2l [b]=0; rw3l [b]=0; pd5l [b]=0; pb4l [b]=0;
ufl g[nd]=0; ufl g[d]=0;}}}}
}}}
}
}
/**/
/* Print the Answers */
void anspri nt(){
short n;
cnt++;
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){pr2ans(); anm[2][0]=0; cnt3=0; }
}
/**/
/* Print the 2 Answers */
void pr2ans(){
short l, l5, n;
pri ntf(" /D5i H/ L/%3d# /D5i H/ L/%3d#\n",
anm[0][0], anm[2][0]);
for(l=0; l<5; l++){l5=l*5;
for(n=1; n<6; n++){pri ntf("%2d", (anm[1][l5+n])/5); }
pri ntf(" ");
for(n=1; n<6; n++){pri ntf("%2d", (anm[1][l5+n])%5); }
pri ntf(" ");
for(n=1; n<6; n++){pri ntf("%3d", anm[0][l5+n]); }
pri ntf(" ");
for(n=1; n<6; n++){pri ntf("%2d", (anm[3][l5+n])/5); }
pri ntf(" ");
for(n=1; n<6; n++){pri ntf("%2d", (anm[3][l5+n])%5); }
pri ntf(" ");
for(n=1; n<6; n++){pri ntf("%3d", anm[2][l5+n]); }
pri ntf("\n");
}
}
}
/**/

```

I am afraid it may look more complicated, but each procedure is written in such a similar way to the others that you can copy one and modify it a little to make another.

#8. Result List of My Recent Computation

```

** 'Complete Euler Squares' of Simul taneous Type: **
*** Both Self-Complementary and Pan-Di agonal 5x5 ***

```

/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	

* [Count = 16] OK! *

All of these results are just the same with those 16 Simultaneous magic squares 5x5: Self-complementary and Pan-diagonal ("Suzuki Squares").

It does not only mean this program could work all right, but also means it could verify completely that all of Suzuki Squares 5x5 are really "Complete Euler Squares".

How happy I am to know that!

#9. 'Euler Squares' 5x5 of Self-Complementary Magic Type

How about Self-complementary type? Can we make any 'Complete Euler Square' for that type? I mean 'Non-Simultaneous' type and pure Self-complementary MS55.

No, we can't.

According to our definition we call 'C.E.S.' for what has all pan-diagonals made up of {0, 1, 2, 3 and 4} using each strictly once. But non-Simultaneous type does not always have to take the constant sum of all pan-diagonals.

It is the problem of definition.

If you want to have any 'Euler Square' 5x5 of Self-complementary type, you should assume the 'Latin Structure' only for every column and every row.

The next list shows some extract data of such 'Euler Squares' of order 5.

** ' Euler Squares 5x5' of Self-Complementary Magic Type: **

1/ /D5i					5/ /D5i																								
1	20	22	14	8	0	3	4	2	1	1	18	24	15	7	0	3	4	2	1	0	2	3	4	1					
24	3	10	17	11	4	0	1	3	2	3	2	4	1	0	23	5	6	17	14	4	0	1	3	2	2	4	0	1	3
7	21	13	5	19	1	4	2	0	3	1	0	2	4	3	10	22	13	4	16	1	4	2	0	3	4	1	2	3	0
15	9	16	23	2	2	1	3	4	0	4	3	0	2	1	12	9	20	21	3	2	1	3	4	0	1	3	4	0	2
18	12	4	6	25	3	2	0	1	4	2	1	3	0	4	19	11	2	8	25	3	2	0	1	4	3	0	1	2	4
9/ /D5i					15/ /D5i																								
1	20	24	12	8	0	3	4	2	1	0	4	3	1	2	1	18	24	15	7	0	3	4	2	1	0	2	3	4	1
15	7	16	23	4	2	1	3	4	0	4	1	0	2	3	12	9	20	21	3	2	1	3	4	0	1	3	4	0	2
9	21	13	5	17	1	4	2	0	3	3	0	2	4	1	10	22	13	4	16	1	4	2	0	3	4	1	2	3	0
22	3	10	19	11	4	0	1	3	2	1	2	4	3	0	23	5	6	17	14	4	0	1	3	2	2	4	0	1	3
18	14	2	6	25	3	2	0	1	4	2	3	1	0	4	19	11	2	8	25	3	2	0	1	4	3	0	1	2	4
21/ /D5i					25/ /D5i																								
1	10	19	23	12	0	1	3	4	2	0	4	3	2	1	1	8	20	24	12	0	1	3	4	2	0	2	4	3	1
22	11	5	9	18	4	2	0	1	3	1	0	4	3	2	23	15	4	7	16	4	2	0	1	3	2	4	3	1	0
20	24	13	2	6	3	4	2	0	1	4	3	2	1	0	17	21	13	5	9	3	4	2	0	1	1	0	2	4	3
8	17	21	15	4	1	3	4	2	0	2	1	0	4	3	10	19	22	11	3	1	3	4	2	0	4	3	1	0	2
14	3	7	16	25	2	0	1	3	4	3	2	1	0	4	14	2	6	18	25	2	0	1	3	4	3	1	0	2	4
29/ /D5i					35/ /D5i																								
1	14	20	23	7	0	2	3	4	1	0	3	4	2	1	1	12	24	20	8	0	2	4	3	1	0	1	3	4	2
8	17	24	5	11	1	3	4	0	2	2	1	3	4	0	22	19	10	3	11	4	3	1	0	2	1	3	4	2	0
22	10	13	16	4	4	1	2	3	0	1	4	2	0	3	9	5	13	21	17	1	0	2	4	3	3	4	2	0	1
15	21	2	9	18	2	4	0	1	3	4	0	1	3	2	15	23	16	7	4	2	4	3	1	0	4	2	0	1	3
19	3	6	12	25	3	0	1	2	4	3	2	0	1	4	18	6	2	14	25	3	1	0	2	4	2	0	1	3	4
41/ /D5i					45/ /D5i																								
1	14	20	23	7	0	2	3	4	1	0	3	4	2	1	1	12	24	20	8	0	2	4	3	1	0	1	3	4	2
15	21	2	9	18	2	4	0	1	3	4	0	1	3	2	15	23	16	7	4	2	4	3	1	0	4	2	0	1	3
22	10	13	16	4	4	1	2	3	0	1	4	2	0	3	9	5	13	21	17	1	0	2	4	3	3	4	2	0	1
8	17	24	5	11	1	3	4	0	2	2	1	3	4	0	22	19	10	3	11	4	3	1	0	2	1	3	4	2	0
19	3	6	12	25	3	0	1	2	4	3	2	0	1	4	18	6	2	14	25	3	1	0	2	4	2	0	1	3	4
49/ /D5i					53/ /D5i																								
2	19	21	15	8	0	3	4	2	1	1	3	0	4	2	2	18	25	14	6	0	3	4	2	1	1	2	4	3	0
25	3	9	16	12	4	0	1	3	2	4	2	3	0	1	23	4	7	16	15	4	0	1	3	2	2	3	1	0	4
6	22	13	4	20	1	4	2	0	3	0	1	2	3	4	9	21	13	5	17	1	4	2	0	3	3	0	2	4	1
14	10	17	23	1	2	1	3	4	0	3	4	1	2	0	11	10	19	22	3	2	1	3	4	0	0	4	3	1	2
18	11	5	7	24	3	2	0	1	4	2	0	4	1	3	20	12	1	8	24	3	2	0	1	4	4	1	0	2	3
57/ /D5i					63/ /D5i																								
2	19	25	11	8	0	3	4	2	1	1	3	4	0	2	2	18	25	14	6	0	3	4	2	1	1	2	4	3	0
14	6	17	23	5	2	1	3	4	0	3	0	1	2	4	11	10	19	22	3	2	1	3	4	0	0	4	3	1	2
10	22	13	4	16	1	4	2	0	3	4	1	2	3	0	9	21	13	5	17	1	4	2	0	3	3	0	2	4	1
21	3	9	20	12	4	0	1	3	2	0	2	3	4	1	23	4	7	16	15	4	0	1	3	2	2	3	1	0	4
18	15	1	7	24	3	2	0	1	4	2	4	0	1	3	20	12	1	8	24	3	2	0	1	4	4	1	0	2	3

<p>69/ /D5i</p> <p>2 9 20 23 11 0 1 3 4 2 1 3 4 2 0</p> <p>21 12 4 10 18 4 2 0 1 3 0 1 3 4 2</p> <p>19 25 13 1 7 3 4 2 0 1 3 4 2 0 1</p> <p>8 16 22 14 5 1 3 4 2 0 2 0 1 3 4</p> <p>15 3 6 17 24 2 0 1 3 4 4 2 0 1 3</p> <p>77/ /D5i</p> <p>2 14 25 18 6 0 2 4 3 1 1 3 4 2 0</p> <p>23 16 7 4 15 4 3 1 0 2 2 0 1 3 4</p> <p>9 5 13 21 17 1 0 2 4 3 3 4 2 0 1</p> <p>11 22 19 10 3 2 4 3 1 0 0 1 3 4 2</p> <p>20 8 1 12 24 3 1 0 2 4 4 2 0 1 3</p> <p>89/ /D5i</p> <p>2 14 25 18 6 0 2 4 3 1 1 3 4 2 0</p> <p>11 22 19 10 3 2 4 3 1 0 0 1 3 4 2</p> <p>9 5 13 21 17 1 0 2 4 3 3 4 2 0 1</p> <p>23 16 7 4 15 4 3 1 0 2 2 0 1 3 4</p> <p>20 8 1 12 24 3 1 0 2 4 4 2 0 1 3</p> <p>97/ /D5i</p> <p>3 20 21 14 7 0 3 4 2 1 2 4 0 3 1</p> <p>22 1 9 18 15 4 0 1 3 2 1 0 3 2 4</p> <p>10 24 13 2 16 1 4 2 0 3 4 3 2 1 0</p> <p>11 8 17 25 4 2 1 3 4 0 0 2 1 4 3</p> <p>19 12 5 6 23 3 2 0 1 4 3 1 4 0 2</p>	<p>73/ /D5i</p> <p>2 8 19 25 11 0 1 3 4 2 1 2 3 4 0</p> <p>23 14 5 6 17 4 2 0 1 3 2 3 4 0 1</p> <p>16 22 13 4 10 3 4 2 0 1 0 1 2 3 4</p> <p>9 20 21 12 3 1 3 4 2 0 3 4 0 1 2</p> <p>15 1 7 18 24 2 0 1 3 4 4 0 1 2 3</p> <p>83/ /D5i</p> <p>2 11 25 19 8 0 2 4 3 1 1 0 4 3 2</p> <p>21 20 9 3 12 4 3 1 0 2 0 4 3 2 1</p> <p>10 4 13 22 16 1 0 2 4 3 4 3 2 1 0</p> <p>14 23 17 6 5 2 4 3 1 0 3 2 1 0 4</p> <p>18 7 1 15 24 3 1 0 2 4 2 1 0 4 3</p> <p>93/ /D5i</p> <p>2 11 25 19 8 0 2 4 3 1 1 0 4 3 2</p> <p>14 23 17 6 5 2 4 3 1 0 3 2 1 0 4</p> <p>10 4 13 22 16 1 0 2 4 3 4 3 2 1 0</p> <p>21 20 9 3 12 4 3 1 0 2 0 4 3 2 1</p> <p>18 7 1 15 24 3 1 0 2 4 2 1 0 4 3</p> <p>101/ /D5i</p> <p>3 19 22 15 6 0 3 4 2 1 2 3 1 4 0</p> <p>21 2 10 18 14 4 0 1 3 2 0 1 4 2 3</p> <p>9 25 13 1 17 1 4 2 0 3 3 4 2 0 1</p> <p>12 8 16 24 5 2 1 3 4 0 1 2 0 3 4</p> <p>20 11 4 7 23 3 2 0 1 4 4 0 3 1 2</p>
---	--

. . . . (Ski p). . . .

<p>365/ /D5i</p> <p>9 2 25 18 11 1 0 4 3 2 3 1 4 2 0</p> <p>3 21 19 12 10 0 4 3 2 1 2 0 3 1 4</p> <p>22 20 13 6 4 4 3 2 1 0 1 4 2 0 3</p> <p>16 14 7 5 23 3 2 1 0 4 0 3 1 4 2</p> <p>15 8 1 24 17 2 1 0 4 3 4 2 0 3 1</p> <p>369/ /D5i</p> <p>10 19 21 3 12 1 3 4 0 2 4 3 0 2 1</p> <p>18 2 9 11 25 3 0 1 2 4 2 1 3 0 4</p> <p>22 6 13 20 4 4 1 2 3 0 1 0 2 4 3</p> <p>1 15 17 24 8 0 2 3 4 1 0 4 1 3 2</p> <p>14 23 5 7 16 2 4 0 1 3 3 2 4 1 0</p> <p>373/ /D5i</p> <p>10 3 21 19 12 1 0 4 3 2 4 2 0 3 1</p> <p>18 11 9 2 25 3 2 1 0 4 2 0 3 1 4</p> <p>22 20 13 6 4 4 3 2 1 0 1 4 2 0 3</p> <p>1 24 17 15 8 0 4 3 2 1 0 3 1 4 2</p> <p>14 7 5 23 16 2 1 0 4 3 3 1 4 2 0</p> <p>381/ /D5i</p> <p>10 1 24 18 12 1 0 4 3 2 4 0 3 2 1</p> <p>3 22 20 11 9 0 4 3 2 1 2 1 4 0 3</p> <p>21 19 13 7 5 4 3 2 1 0 0 3 2 1 4</p> <p>17 15 6 4 23 3 2 1 0 4 1 4 0 3 2</p> <p>14 8 2 25 16 2 1 0 4 3 3 2 1 4 0</p>	<p>367/ /D5i</p> <p>9 3 22 20 11 1 0 4 3 2 3 2 1 4 0</p> <p>2 25 16 14 8 0 4 3 2 1 1 4 0 3 2</p> <p>21 19 13 7 5 4 3 2 1 0 0 3 2 1 4</p> <p>18 12 10 1 24 3 2 1 0 4 2 1 4 0 3</p> <p>15 6 4 23 17 2 1 0 4 3 4 0 3 2 1</p> <p>371/ /D5i</p> <p>10 18 24 1 12 1 3 4 0 2 4 2 3 0 1</p> <p>17 4 6 15 23 3 0 1 2 4 1 3 0 4 2</p> <p>21 7 13 19 5 4 1 2 3 0 0 1 2 3 4</p> <p>3 11 20 22 9 0 2 3 4 1 2 0 4 1 3</p> <p>14 25 2 8 16 2 4 0 1 3 3 4 1 2 0</p> <p>377/ /D5i</p> <p>10 1 24 18 12 1 0 4 3 2 4 0 3 2 1</p> <p>17 15 6 4 23 3 2 1 0 4 1 4 0 3 2</p> <p>21 19 13 7 5 4 3 2 1 0 0 3 2 1 4</p> <p>3 22 20 11 9 0 4 3 2 1 2 1 4 0 3</p> <p>14 8 2 25 16 2 1 0 4 3 3 2 1 4 0</p> <p>383/ /D5i</p> <p>10 3 21 19 12 1 0 4 3 2 4 2 0 3 1</p> <p>1 24 17 15 8 0 4 3 2 1 0 3 1 4 2</p> <p>22 20 13 6 4 4 3 2 1 0 1 4 2 0 3</p> <p>18 11 9 2 25 3 2 1 0 4 2 0 3 1 4</p> <p>14 7 5 23 16 2 1 0 4 3 3 1 4 2 0</p>
---	---

* [Count = 384] OK! *

We could find only a few solutions for that as many as 384.
 I haven't yet known anything about this, while I have been long thinking.
 But don't you think it is nice we could build 'Euler Squares' even for S-C type?

#10. Comment

This is the first successful result of my elaborate computation to try to make any solution set of "Complete Euler Squares". Though it might look too complicated, I am proud of my experience I could surely have got some skills to solve the problem and have any special key in my hand to open the door to the new world.

I am lucky I could have invented another two ways of composing Complete Euler Squares: "Compositions by Knight's Tour", and "New Euler's Method". I want you to read my articles I explained about them in the following sections.

Those new methods can do our job very fast, faster than anything else.

Let's go on to the next challenge to build Complete Euler Squares of Order 7.

(Original Written in English on March 27, 2003 by Kanji Setsuda
Retyped Edition on February 21, 2007 with MacOSX and Xcode2.2)

E-Mail Address <jag12001@ni fty. com>