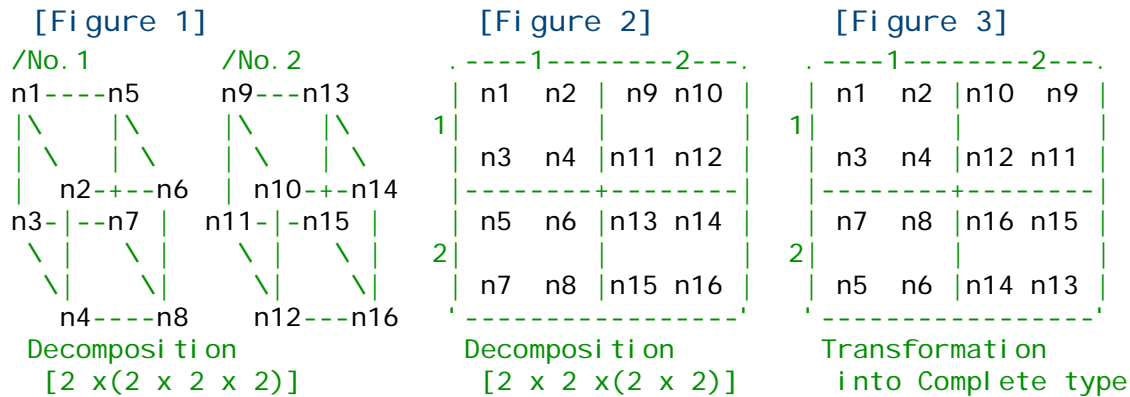


Chapter 6: Fundamental Studies of Multi-Dimensional Extra-Cubic Magic Forms and their Decomposed Ones: **Kanji Setsuda**

Section 2: Study of 4-Dimensional Extra-Cubic Magic Objects of Order 2 and their Decomposed Forms

#1. Let's make a 4-dimensional magic form of order 2 in the 2 x 2 x 2 x 2 array. Why do I intend to do so? It is because I want to know of the highest symmetry of magic forms by increasing the dimensions up to 4. It is also because I want to know what the 'master key' of various magic squares of order 4 is.

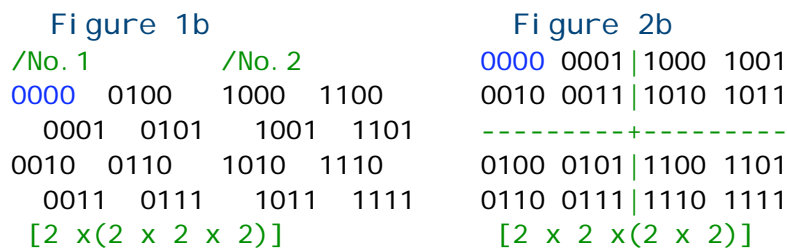
#2. The structural data of the consecutive whole numbers from 1 to 16 are stored in the 4-dimensional array of 2 x 2 x 2 x 2. No one can draw a visible picture of that, but it is too difficult for us to think of it without using any kind of figures. So I tried to prepare a few figures of 'Decomposed Forms', so that we could understand the concept clear.



Decomposed Figure 1 has a pair of cubes. Both of them stand for a single extra-cubic thing. Decomposed Figure 2 contains 4 sets of squares of order 2, and each of them comes from each surface of the decomposed cubes (1), though at a whole sight the total figure (2) looks like a magic square of order 4.

We suppose these two figures as the basic forms to make everything with.

#3. We can put the same figures in a different way with binary notations.



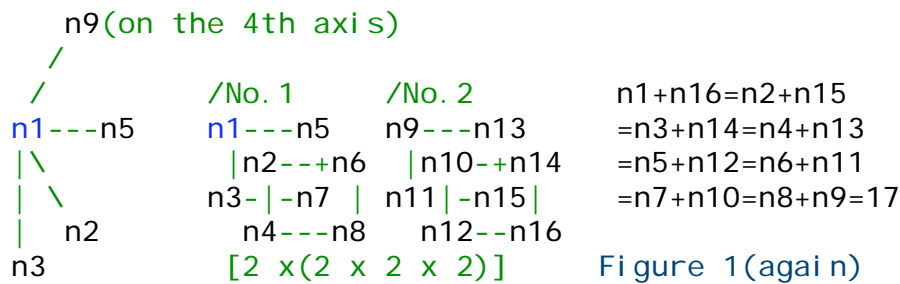
The position n(0,0,0,0) stands for 'n1' in the classic style; and n(0,0,0,1) for n2, n(0,0,1,0) for n3, n(0,0,1,1) for n4, . . . , n(1,1,1,0) for n15, and n(1,1,1,1) for n16.

Say, n(0,0,0,0) is the dimensional center or the origin four axes meet on. Then what are the 'axes'? Each of them is supposed to have both n(0,0,0,0) and the next position adjacent to it.

Then what are the 'next' positions to n1? No doubt, they are n2, n3 and n5. And

then what is the rest one?

It is n9. Because it is placed in n(1,0,0,0). In the 4-dimensional system n2(0,0,0,1), n3(0,0,1,0), n5(0,1,0,0), and n9(1,0,0,0) are supposed to be the next positions to the origin n1(0,0,0,0).



#4. What reasonable conditions do we put to this 4-dimensional object? How do we make it for the 'magic' one?

Of course, we cannot put such equations as n1+n2=C, n1+n3=C, n1+n5=C, and n1+n9=C. Because the logical result n2=n3=n5=n9(=C-n1) is never acceptable. Four different numbers can never take the same value.

So we must imagine and realize anything else for the magic constant. ...

What about the sum of any sets of 4 entries?

At first suppose the constant sum of every 4 numbers on the same surfaces of cubes. For example,

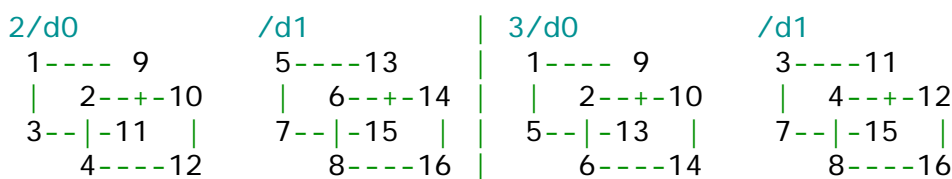
$$\begin{aligned}
 &n1+n2+n3+n4=C, \quad n1+n2+n5+n6=C, \quad n1+n3+n5+n7=C, \\
 &n2+n4+n6+n9=C, \quad n3+n4+n7+n8=C, \quad n5+n6+n7+n8=C \quad \dots \text{(Figure 1/No. 1)} \\
 &n9+n10+n11+n12=C, \quad n9+n10+n13+n14=C, \quad n9+n11+n13+n15=C, \\
 &n10+n12+n14+n16=C, \quad n11+n12+n15+n16=C, \quad n13+n14+n15+n16=C \quad \dots \text{(/No. 2)}
 \end{aligned}$$

Twelve simultaneous equations are thus put to be solved.

But they are not enough. It does not look like 4-dimensional yet.

$$\begin{aligned}
 &n1+n2+n9+n10=C, \quad n1+n3+n9+n11=C, \quad n1+n5+n9+n13=C, \\
 &n2+n4+n10+n12=C, \quad n2+n6+n10+n14=C, \quad n3+n4+n11+n12=C, \\
 &n3+n7+n11+n15=C, \quad n4+n8+n12+n16=C, \quad n5+n6+n13+n14=C, \\
 &n5+n7+n13+n15=C, \quad n6+n8+n14+n16=C, \quad n7+n8+n15+n16=C
 \end{aligned}$$

Twelve more equations above should be added. You can easily understand this, if you have another 2 view-forms of decomposition for the same object.



#5. How should we think of self-complementary pairs and their symmetrical locations?

Take a look at the figures above again. Combinations and locations are quite natural and reasonable. You can make 8 complementary pairs of 17 and place them in their own symmetrical locations.

$$n1+n16=n2+n15=n3+n14=n4+n13=n5+n12=n6+n11=n7+n10=n8+n9=17$$

Let's assume these conditions become true for the first definition stage of our object. Then you don't have to assume as many 'composite conditions' as 24, but you have to take only the following 12 equations at first.

\*\* The 12 Essential 'Composite Conditions' \*\*

$$\begin{array}{lll}
 n_1+n_2+n_3+n_4=C \dots (1); & n_1+n_2+n_5+n_6=C \dots (2); & n_1+n_2+n_9+n_{10}=C \dots (3); \\
 n_1+n_3+n_5+n_7=C \dots (4); & n_1+n_3+n_9+n_{11}=C \dots (5); & n_1+n_5+n_9+n_{13}=C \dots (6); \\
 n_2+n_4+n_6+n_8=C \dots (7); & n_2+n_4+n_{10}+n_{12}=C \dots (8); & n_2+n_6+n_{10}+n_{14}=C \dots (9); \\
 n_3+n_4+n_7+n_8=C \dots (10); & n_3+n_4+n_{11}+n_{12}=C \dots (11); & n_5+n_6+n_7+n_8=C \dots (12);
 \end{array}$$

#6. Under these strict conditions, examine how you can exchange the entries of this 4-dimensional object, and know how often you can see the different solutions. You can use your computer to count them for the purpose.

If you assume  $n_1 < n_{16}$ , and  $n_1 < n_2 < n_3 < n_5 < n_9$ , then only one solution could come out. If you assume  $n_1 < n_{16}$ ,  $n_1 < n_9$ , and  $n_1 < n_2 < n_3 < n_5$ , then you could get three solutions. If you assume  $n_1 < n_{16}$ ,  $n_1 < n_9$ ,  $n_1 < n_5$ ,  $n_1 < n_3$  and  $n_1 < n_2$ , then you could get more solutions.

Check the solution list below. This is the first 24 primitive solutions of the largest 'Mother Set'. How do the four axes and the numbers next to  $n_1$  appear?

1/N1	/N2	1/Seif-compl em.	/Pan-magic
1---12	8---13	1 15 8 10	1 15 10 8
15---+6	10---+3	14 4 11 5	14 4 5 11
14- ---7	11- ---2	12 6 13 3	7 9 16 2
4----9	5---16	7 9 2 16	12 6 3 13
2/N1	/N2	2/SS4*4	/PS4*4
1 8 12 13		1 15 12 6	1 15 6 12
15 10 6 3		14 4 7 9	14 4 9 7
14 11 7 2		8 10 13 3	11 5 16 2
4 5 9 16		11 5 2 16	8 10 3 13
3/N1	/N2	3/SS4*4	/PS4*4
1 14 8 11		1 15 8 10	1 15 10 8
15 4 10 5		12 6 13 3	12 6 3 13
12 7 13 2		14 4 11 5	7 9 16 2
6 9 3 16		7 9 2 16	14 4 5 11
4/N1	/N2	4/SS4*4	/PS4*4
1 8 14 11		1 15 14 4	1 15 4 14
15 10 4 5		12 6 7 9	12 6 9 7
12 13 7 2		8 10 11 5	13 3 16 2
6 3 9 16		13 3 2 16	8 10 5 11
5/N1	/N2	5/SS4*4	/PS4*4
1 14 12 7		1 15 12 6	1 15 6 12
15 4 6 9		8 10 13 3	8 10 3 13
8 11 13 2		14 4 7 9	11 5 16 2
10 5 3 16		11 5 2 16	14 4 9 7
6/N1	/N2	6/SS4*4	/PS4*4
1 12 14 7		1 15 14 4	1 15 4 14
15 6 4 9		8 10 11 5	8 10 5 11
8 13 11 2		12 6 7 9	13 3 16 2
10 3 5 16		13 3 2 16	12 6 9 7
7/N1	/N2	7/SS4*4	/PS4*4
1---12	8---13	1 14 8 11	1 14 11 8
14---+7	11---+2	15 4 10 5	15 4 5 10
15- -6	10- -3	12 7 13 2	6 9 16 3
4----9	5---16	6 9 3 16	12 7 2 13

8/N1	/N2	8/SS4*4	/PS4*4
1----8	12---13	1 14 12 7	1 14 7 12
14---+11	7---+2	15 4 6 9	15 4 9 6
15- -10	6- --3	8 11 13 2	10 5 16 3
4-----5	9---16	10 5 3 16	8 11 2 13
9/N1	/N2	9/SS4*4	/PS4*4
1 15 8 10		1 14 8 11	1 14 11 8
14 4 11 5		12 7 13 2	12 7 2 13
12 6 13 3		15 4 10 5	6 9 16 3
7 9 2 16		6 9 3 16	15 4 5 10
10/N1	/N2	10/SS4*4	/PS4*4
1 8 15 10		1 14 15 4	1 14 4 15
14 11 4 5		12 7 6 9	12 7 9 6
12 13 6 3		8 11 10 5	13 2 16 3
7 2 9 16		13 2 3 16	8 11 5 10
11/N1	/N2	11/SS4*4	/PS4*4
1 15 12 6		1 14 12 7	1 14 7 12
14 4 7 9		8 11 13 2	8 11 2 13
8 10 13 3		15 4 6 9	10 5 16 3
11 5 2 16		10 5 3 16	15 4 9 6
12/N1	/N2	12/SS4*4	/PS4*4
1 12 15 6		1 14 15 4	1 14 4 15
14 7 4 9		8 11 10 5	8 11 5 10
8 13 10 3		12 7 6 9	13 2 16 3
11 2 5 16		13 2 3 16	12 7 9 6
13/N1	/N2	13/SS4*4	/PS4*4
1 14 8 11		1 12 8 13	1 12 13 8
12 7 13 2		15 6 10 3	15 6 3 10
15 4 10 5		14 7 11 2	4 9 16 5
6 9 3 16		4 9 5 16	14 7 2 11
14/N1	/N2	14/SS4*4	/PS4*4
1 8 14 11		1 12 14 7	1 12 7 14
12 13 7 2		15 6 4 9	15 6 9 4
15 10 4 5		8 13 11 2	10 3 16 5
6 3 9 16		10 3 5 16	8 13 2 11
15/N1	/N2	15/SS4*4	/PS4*4
1 15 8 10		1 12 8 13	1 12 13 8
12 6 13 3		14 7 11 2	14 7 2 11
14 4 11 5		15 6 10 3	4 9 16 5
7 9 2 16		4 9 5 16	15 6 3 10
16/N1	/N2	16/SS4*4	/PS4*4
1 8 15 10		1 12 15 6	1 12 6 15
12 13 6 3		14 7 4 9	14 7 9 4
14 11 4 5		8 13 10 3	11 2 16 5
7 2 9 16		11 2 5 16	8 13 3 10
17/N1	/N2	17/SS4*4	/PS4*4
1---15	14-----4	1 12 14 7	1 12 7 14
12---+6	7---+9	8 13 11 2	8 13 2 11
8- -10	11- --5	15 6 4 9	10 3 16 5
13-----3	2---16	10 3 5 16	15 6 9 4

18/N1	/N2	18/SS4*4	/PS4*4
1---14	15--- 4	1 12 15 6	1 12 6 15
12---+ 7	6---+ 9	8 13 10 3	8 13 3 10
8- -11	10- - 5	14 7 4 9	11 2 16 5
13--- 2	3---16	11 2 5 16	14 7 9 4
19/N1	/N2	19/SS4*4	/PS4*4
1 14 12 7		1 8 12 13	1 8 13 12
8 11 13 2		15 10 6 3	15 10 3 6
15 4 6 9		14 11 7 2	4 5 16 9
10 5 3 16		4 5 9 16	14 11 2 7
20/N1	/N2	20/SS4*4	/PS4*4
1 12 14 7		1 8 14 11	1 8 11 14
8 13 11 2		15 10 4 5	15 10 5 4
15 6 4 9		12 13 7 2	6 3 16 9
10 3 5 16		6 3 9 16	12 13 2 7
21/N1	/N2	21/SS4*4	/PS4*4
1 15 12 6		1 8 12 13	1 8 13 12
8 10 13 3		14 11 7 2	14 11 2 7
14 4 7 9		15 10 6 3	4 5 16 9
11 5 2 16		4 5 9 16	15 10 3 6
22/N1	/N2	22/SS4*4	/PS4*4
1 12 15 6		1 8 15 10	1 8 10 15
8 13 10 3		14 11 4 5	14 11 5 4
14 7 4 9		12 13 6 3	7 2 16 9
11 2 5 16		7 2 9 16	12 13 3 6
23/N1	/N2	23/SS4*4	/PS4*4
1 15 14 4		1 8 14 11	1 8 11 14
8 10 11 5		12 13 7 2	12 13 2 7
12 6 7 9		15 10 4 5	6 3 16 9
13 3 2 16		6 3 9 16	15 10 5 4
24/N1	/N2	24/Sel f-compl em.	/Pan-magi c
1---14	15--- 4	1 8 15 10	1 8 10 15
8---+11	10---+ 5	12 13 6 3	12 13 3 6
12- - 7	6- - 9	14 11 4 5	7 2 16 9
13--- 2	3---16	7 2 9 16	14 11 5 4

Do you notice that the four axes are exchangeable with one another? The numbers next to 1 are only 15, 14, 12 and 8. And they are also exchangeable with one another. They are equally free to go anywhere next to 1.

Say, this is the evidence of our success, isn't it? It is highly symmetrical in itself. We are now able to make the 4-dimensional magic form of order 2!!

**#7.** Check the solution list so that the same pattern as each other might not appear many times. If you remove all simple rotations and reflections, you could get the smaller list than the one above.

In fact they are all the same. There proved to be only one solution.

**#8.** Watch the last two decomposed patterns of each solution. They look like magic squares of order 4.

The central square named /SS4\*4 is made by putting all entries into a 4 x 4 form in order. It looks like a self-complementary magic square of order 4. All

complementary pairs of 17 are regularly located in symmetrical positions. Yes. This is really a self-complementary magic square of order 4.

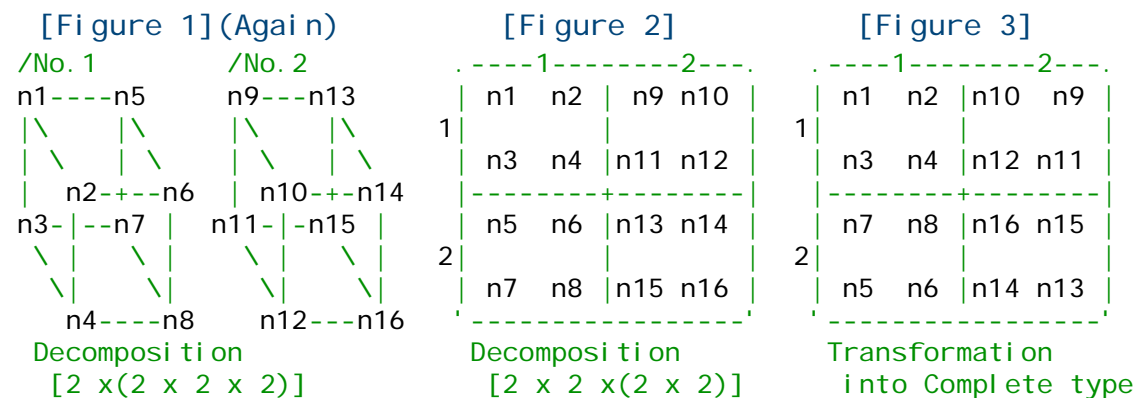
The pattern named /PS4\*4 on the right hand is transformed from the central one. The 3rd and 4th rows were exchanged with each other. The 3rd and 4th columns were also exchanged with each other. It is made into a pan-magic square of order 4. This also happens to be a 'composite and complete' square at the same time. It has the property that any 2 x 2 block of adjacent entries sum up to the same total. But it is neither an accident, nor an arbitrary arrangement made by someone too ambitious.

You can always take the two types of magic squares of order 4 out of the 4-dimensional object of order 2. Because each of them is one-to-one corresponding to the other two. This shows the possibility of 'dimension converter', doesn't it?

If you stand on the side of magic squares of order 4, you should have the 'smallest set' that contains 3 'initial' solutions as shown below. The magic squares of lower dimension and higher order must have more 'initial' solutions than the ones of higher dimension and lower order.

1/n1	/n2	1/SS4*4	/CS4*4
1---12	8---13	1 15 8 10	1 15 10 8
15---+6	10---+3	14 4 11 5	14 4 5 11
14- -7	11- -2	12 6 13 3	7 9 16 2
4----9	5---16	7 9 2 16	12 6 3 13
2/n1	/n2	2/SS4*4	/CS4*4
1----8	12---13	1 15 12 6	1 15 6 12
15---+10	6---+3	14 4 7 9	14 4 9 7
14- -11	7- -2	8 10 13 3	11 5 16 2
4----5	9---16	11 5 2 16	8 10 3 13
3/n1	/n2	3/SS4*4	/CS4*4
1----8	14---11	1 15 14 4	1 15 4 14
15---+10	4---+5	12 6 7 9	12 6 9 7
12- -13	7- -2	8 10 11 5	13 3 16 2
6----3	9---16	13 3 2 16	8 10 5 11

[Count = 3]



Where are the four numbers next to the origin n1? Where did they go in the

magic squares of order 4? There exist  $n_2$ ,  $n_3$ ,  $n_5$  and  $n_9$  on the two axes: 2 for each. Although you may not accept  $n_5$  and  $n_9$  as the next numbers to  $n_1$ , they are reasonably located. Because you can easily exchange all 4 entries of the row or column that include  $n_2$  and  $n_9$  at any time you want, and also exchange  $n_3$  and  $n_5$  with each other.

The smallest count 3 of solutions is, therefore, determined by the permutation of 4 numbers next to 1 on the 2 axes of squares of order 4. Divide  $4C_2$  by  $2P_2$ . You can get the answer 3.

#9. You might be surprised, if I say that there proved to be the 'origin' of various magic squares of order 4. They are surely derived from the only one original concept of higher dimensions, so that the offspring from it might be quite similar to one another.

(The original was written in English on June 24, 2001;  
Revised on May 6, 2005; Worked on MacOS Xcode 1.5 by Kanji Setsuda)

\*\*\* E-Mail Address: <jag12001@nifty.ne.jp>

```

/* Appendix: Program List */
/** Make the 4-Dimensional Magic Objects of Order 2 and **/
/** Transform them into 'Composite and Complete' MS44: **/
/** 'EC024A384.c' built by Kanji Setsuda **/
/** on MacOS X and Xcode 1.5; Mar. 20, 2005 **/
/**/
#include <stdio.h>
/**/
/* Variables */
short int cnt;
short CC, LSM, cnt2;
short nm[17], uflg[17];
short t[384][17];
/**/
/* Sub-Routines */
void stp01(void), stp02(void), stp03(void);
void stp04(void), stp05(void), stp06(void);
void stp07(void), stp08(void), stp09(void);
void stp10(void), stp11(void), stp12(void);
void ansrecrd(void);
void pr4f24(void), pr4cms4(void);
/**/
/* Main Program */
main(){
    short n;
    printf("\n*** Make the 4-Dimensional Magic Objects of Order 2 and ***\n");
    printf("** Transform them into Two Types of MS44: 'S-C' and 'C&C' **\n");
    for(n=0; n<17; n++){nm[n]=0; uflg[n]=0;}
    CC=17; LSM=34; cnt=0;
    stp01(); /* Begin the Calculation */
    pr4f24(); /* Print the Answer in 4 Forms */
    pr4cms4(); /* Print the Answer only in CCMS44 Form */
    printf("\n [Count = %d]\n", cnt);
    printf(" OK!\n");
    return 0;
}
/**/

```

```

/* Calculations */
/* Set n1 & n16 */
void stp01(){
  short a, b;
  for(a=1; a<17; a++){b=CC-a;
    if((ufl g[a]==0)&&(ufl g[b]==0)){
      nm[1]=a; nm[16]=b;
      ufl g[a]=1; ufl g[b]=1;
      stp02();
      ufl g[b]=0; ufl g[a]=0; }
  }
}
/* Set n2 & n15 */
void stp02(){
  short a, b;
  for(a=16; a>0; a--){b=CC-a;
    if((ufl g[a]==0)&&(ufl g[b]==0)){
      nm[2]=a; nm[15]=b;
      ufl g[a]=1; ufl g[b]=1;
      stp03();
      ufl g[b]=0; ufl g[a]=0; }
  }
}
/* Set n5 & n12 */
void stp03(){
  short a, b;
  for(a=16; a>0; a--){b=CC-a;
    if((ufl g[a]==0)&&(ufl g[b]==0)){
      nm[5]=a; nm[12]=b;
      ufl g[a]=1; ufl g[b]=1;
      stp04();
      ufl g[b]=0; ufl g[a]=0; }
  }
}
/* Set n6=LSM-n1-n2-n5 & n11 */
void stp04(){
  short a, b;
  a=LSM-nm[1]-nm[2]-nm[5];
  if((0<a)&&(a<17)){b=CC-a;
    if((ufl g[a]==0)&&(ufl g[b]==0)){
      nm[6]=a; nm[11]=b;
      ufl g[a]=1; ufl g[b]=1;
      stp05();
      ufl g[b]=0; ufl g[a]=0; }}
}
/* Set n3 & n14 */
void stp05(){
  short a, b;
  for(a=16; a>0; a--){b=CC-a;
    if((ufl g[a]==0)&&(ufl g[b]==0)){
      nm[3]=a; nm[14]=b;
      ufl g[a]=1; ufl g[b]=1;
      stp06();
      ufl g[b]=0; ufl g[a]=0; }
  }
}
/* Set n4=LSM-n1-n2-n3 & n13 */
void stp06(){
  short a, b;

```

```

a=LSM-nm[1]-nm[2]-nm[3];
if((0<a)&&(a<17)){b=CC-a;
  if((ufl g[a]==0)&&(ufl g[b]==0)){
    nm[4]=a; nm[13]=b;
    ufl g[a]=1; ufl g[b]=1;
    stp07();
    ufl g[b]=0; ufl g[a]=0; }}
}
/* Set n7=LSM-n1-n3-n5 & n10 */
void stp07(){
  short a, b;
  a=LSM-nm[1]-nm[3]-nm[5];
  if((0<a)&&(a<17)){b=CC-a;
    if((0<b)&&(ufl g[a]==0)&&(ufl g[b]==0)){
      nm[7]=a; nm[10]=b;
      ufl g[a]=1; ufl g[b]=1;
      stp08();
      ufl g[b]=0; ufl g[a]=0; }}
}
/* Set n8=LSM-n2-n4-n6 & n9 */
void stp08(){
  short a, b, c;
  a=LSM-nm[2]-nm[4]-nm[6];
  b=LSM-nm[3]-nm[4]-nm[7];
  c=LSM-nm[5]-nm[6]-nm[7];
  if((0<a)&&(a<17)&&(a==b)&&(a==c)){b=CC-a;
    if((ufl g[a]==0)&&(ufl g[b]==0)){
      nm[8]=a; nm[9]=b;
      ufl g[a]=1; ufl g[b]=1; cnt2=0;
      stp09();
      ufl g[b]=0; ufl g[a]=0; }}
}
/* Check some Line-sums */
void stp09(){
  short sm1, sm2, sm3;
  sm1=nm[1]+nm[2]+nm[9]+nm[10];
  sm2=nm[1]+nm[3]+nm[9]+nm[11];
  sm3=nm[1]+nm[5]+nm[9]+nm[13];
  if((sm1==LSM)&&(sm2==LSM)&&(sm3==LSM)){stp10();}
}
/* Check some Line-sums */
void stp10(){
  short sm1, sm2, sm3, sm4;
  sm1=nm[2]+nm[4]+nm[10]+nm[12];
  sm2=nm[2]+nm[6]+nm[10]+nm[14];
  if((sm1==LSM)&&(sm2==LSM)){
    ansreocrd();}
}
/**/
void ansreocrd(void){
  short n;
  t[cnt][0]=cnt+1;
  for(n=1; n<17; n++){t[cnt][n]=nm[n];}
  cnt++;
}
/*
*** Diagrams expressing Basic Positions of Elements ***
BD/d0      /d1      SC/      CC/
1----- 2      3----- 4      1 2 3 4      1 2 4 3

```

```

| 9---10 | 11---12 | 5 6 7 8 | 5 6 8 7
5--|- 6 | 7--|- 8 | 9 10 11 12 | 13 14 16 15
13----14 | 15----16 | 13 14 15 16 | 9 10 12 11
*/
/* Print the Answer of Order 2^4 in 4 Forms */
void pr4f24(void){
short m, n;
short tn[17];
for(m=0; m<24; m++){
for(n=0; n<17; n++){tn[n]=t[m][n];}
printf("%3d/d0 /d1 SC/ CC/\n", tn[0]);
printf("%4d----%2d%7d----%2d ", tn[1], tn[2], tn[3], tn[4]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[1], tn[2], tn[3], tn[4], tn[1], tn[2], tn[4], tn[3]);
printf(" | %2d---%2d | %2d---%2d", tn[9], tn[10], tn[11], tn[12]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[5], tn[6], tn[7], tn[8], tn[5], tn[6], tn[8], tn[7]);
printf("%4d--|-%2d |%4d--|-%2d |", tn[5], tn[6], tn[7], tn[8]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[9], tn[10], tn[11], tn[12], tn[13], tn[14], tn[16], tn[15]);
printf("%7d----%2d%7d----%2d", tn[13], tn[14], tn[15], tn[16]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[13], tn[14], tn[15], tn[16], tn[9], tn[10], tn[12], tn[11]);
}
}
/**/
/* Print the Solutions only in CCMS44 Style */
void pr4cms4(void){
short m, n, p;
short tn[4][17];
for(m=24; m<cnt; m=m+4){
for(p=0; p<4; p++){for(n=0; n<17; n++){tn[p][n]=t[m+p][n];}}
printf("%3d/CC%11d/CC%11d/CC%11d/CC\n",
tn[0][0], tn[1][0], tn[2][0], tn[3][0]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[0][1], tn[0][2], tn[0][4], tn[0][3], tn[1][1], tn[1][2], tn[1][4], tn[1][3],
tn[2][1], tn[2][2], tn[2][4], tn[2][3], tn[3][1], tn[3][2], tn[3][4], tn[3][3]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[0][5], tn[0][6], tn[0][8], tn[0][7], tn[1][5], tn[1][6], tn[1][8], tn[1][7],
tn[2][5], tn[2][6], tn[2][8], tn[2][7], tn[3][5], tn[3][6], tn[3][8], tn[3][7]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[0][13], tn[0][14], tn[0][16], tn[0][15], tn[1][13], tn[1][14], tn[1][16], tn[1][15],
tn[2][13], tn[2][14], tn[2][16], tn[2][15], tn[3][13], tn[3][14], tn[3][16], tn[3][15]);
printf(" %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d %3d%3d%3d%3d\n",
tn[0][9], tn[0][10], tn[0][12], tn[0][11], tn[1][9], tn[1][10], tn[1][12], tn[1][11],
tn[2][9], tn[2][10], tn[2][12], tn[2][11], tn[3][9], tn[3][10], tn[3][12], tn[3][11]);
}
}
}
/* EOF */

```