Part 4: "New Advanced Study of Magic Squares and Cubes"

Chapter 4: Commentary Articles No.2: **Kanji Setsuda**
"Various Arts and Tools for Studying Magic Squares"

Section 10: How to Make 'Complete Euler Squares' 9x9 by New Euler's Method using PNS of Base 3

0. Making 'Complete Euler Squares' 9x9 by the "New Euler's Method"

Let's build various types of Magic Squares 9x9 here by our "New Euler's Method" using **P**ositional **N**umber **S**ystem of Base 3. We should apply it even to every pan-diagonal of the object, if necessary.

Now that we already know we cannot compose any 'Complete Euler Squares' of order 9 by PNS of Base 9, we use PNS of Base 3 instead this time.

We have to do with it as carefully as we can all the time, because it is the first experience for us to make everything with only '0', '1' and '2'.

At first let me show you a solution example of 'Complete Euler Square' 9x9 in the classical style, with the forms of notation /N3i and decompositions /D3i by PNS of Base 3 below.

** Example of Complete Euler Square 9x9 **

Classic/									/N3i								
1	68	54	7	39	77	4	45	74	0000	2111	1222	0020	1102	2211	0010	1122	2201
42	25	56	65	22	36	71	19	33	1112	0220	2001	2101	0210	1022	2121	0200	1012
80	30	13	51	62	10	48	59	16	2221	1002	0110	1212	2021	0100	1202	2011	0120
47	58	18	53	3	67	50	61	12	1201	2010	0122	1221	0002	2110	1211	2020	0102
6	44	73	55	41	27	9	38	76	0012	1121	2200	2000	1111	0222	0022	1101	2210
70	21	32	15	79	29	64	24	35	2120	0202	1011	0112	2220	1001	2100	0212	1021
66	23	34	72	20	31	69	52	2	2102	0211	1020	2122	0201	1010	2112	1220	0001
49	63	11	46	60	17	26	57	40	1210	2022	0101	1200	2012	0121	0221	2002	1110
8	37	78	5	43	75	28	14	81	0021	1100	2212	0011	1120	2202	1000	0111	2222

/D3i								A*27/								B*9/								C*3/								D*1/							
0	2	1	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	2	0	1	1	2	0	0	1	2	0	2	1	0	2	1
1	0	2	2	0	1	2	0	1	1	2	0	1	2	0	1	2	0	1	2	0	0	1	2	2	0	1	2	0	1	1	0	2	1	0	2
2	1	0	1	2	0	1	2	0	2	0	1	2	0	1	2	0	1	2	0	1	1	2	0	0	1	2	1	2	0	2	1	0	2	1	0
1	2	0	1	0	2	1	2	0	2	0	1	2	0	1	2	0	1	2	0	1	2	2	0	1	1	2	0	1	0	2	1	2	0	1	0	2
0	1	2	2	1	0	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	0	1	2	2	0	1	2	1	0	0	1	2	2	1	0
2	0	1	0	2	1	2	0	1	1	2	0	1	2	0	1	2	0	1	2	0	1	1	2	0	0	1	2	0	2	1	2	0	1	0	2	1
2	0	1	2	0	1	2	1	0	1	2	0	1	2	0	1	2	0	1	2	0	0	1	2	2	0	1	1	2	0	2	1	0	2	0	1
1	2	0	1	2	0	0	2	1	2	0	1	2	0	1	2	0	1	2	0	1	1	2	0	0	1	2	2	0	1	0	2	1	0	2	1	2	0
0	1	2	0	1	2	1	0	2	0	1	2	0	1	2	0	1	2	0	1	2	2	0	1	1	2	0	0	1	2	1	0	2	1	0	2	0	1	2

Each number of the three forms above is related to one another by the next equation:

Vn = An*27 + Bn*9 + Cn*3 + Dn*1 + 1 (n=1, 2, 3, 4, 5, ... , 63, 64)

1. Definition of 'Complete Euler Square' 9x9 by PNS of Base 3

Take your careful look at the 4 decomposed layers.

[I] In each layer every row and every column is made up of three '0', three '1' and three '2'. No other patterns may be found. Therefore every sum of any row or any column is calculated in the same manner as:

(3*0 +3*1 +3*2)*27 + (3*0 +3*1 +3*2)*9 + (3*0 +3*1 +3*2)*3 + (3*0 +3*1 +3*2)*1
= (3*0 +3*1 +3*2) * (27+9+3+1) = 9 * 40 = 360(Decimal)

It means every sum is equal to the Magic Constant that should be necessarily

realized. 360(Dec) here is logically equivalent to 369(Dec) in our classical notation.

Whenever every pan-diagonal takes the same pattern and the same sum, we would call "Complete Euler Squares" 9x9 for those which have thist fine structure.

[II] Every layer consists of twenty seven '0', twenty seven '1' and twenty seven '2',

[III] When you pick up each number of the same position of each layer and combine these four numbers, each result must be any one of {0000, 0001, 0002, 0010, 0011, 0012, 0020, 0021, 0022, 0100, ... , 2220, 2221, 2222(N3i)} and neither repetition nor drop-off of any value must be found.

This definition is logically equivalent to one of the most basic promises in our classical notation: We must use the series of natural numbers 1~81 to make the object magic square taking each number strictly once, and neither repeating usage nor dropping-off of a certain number must be found.

2. Our "New Euler's Method"

"New Euler's Method" is the way how we make the object by realizing those definitions above one by one literally.

(1) Design and compose the Latin Units for the decomposed layers at first.

Prepare the necessary simultaneous equations defining the object and basic circumstance, and you can dictate your program codes to make the Latin Units using only '0', '1' and '2'.

(2) Choose any appropriate four units among the composed ones and pick them up and combine to make them act as the four Decomposed Layers of a solution. Calculate each value of every position by the next equation:

Vn = An*27 + Bn*9 + Cn*3 + Dn*1 + 1 (n=1, 2, 3, 4, 5, ... , 63, 64)

(3) Check if your composition is against the definition [III] or not, and put your passing answer through the normalizing inequality condition. And you will get your correct set of standard solutions at last.

These steps are just the same as the ones in the previous sections.

But you might come across a new certain problem related to the PNS of Base 3.

It is caused by the ambiguous answers to the next equation. It cannot be determined definitely as this: If (A+B+C=3) then {A, B, C}={0, 1, 2}

Since [If (A+B+C=3) then {A, B, C}={1, 1, 1}] is also possible.

[If (A+B+C+D+E+F+G+H+I=9) then {A,B,C,D,E,F,G,H,I} = {0,0,0,1,1,1,2,2,2}] is not always correct. Since [If (A+B+C+D+E+F+G+H+I=9) then {A,B,C,D,E,F,G,H,I} = {1,1,1,1,1,1,1,1,1}] is also possible.

You must choose either one of these answer patterns at the definition stage in advance. If you want to have every pan-diagonal take the definition [I] as true, you should prepare the check flag for each and know how often each number is used.

3. Let's make Simultaneous type of Magic Squares 9x9 both S-C and P-D.

Let's try to build Simultaneous MS99 both Self-complementary and Pan-diagonal type by our New Euler's Method using PNS of Base 3. The total solution count of this type is still unknown.

We know there are many 'Non-Euler Type' of solutions, far more than the 'Complete Euler Squares', but we want to study the latter type now and count it up through.

```
*** Basic Things for Simultaneous Magic Squares ***
** 9*9: Both Self-Complementary and Pan-Diagonal **

** Basic Conditions defining all Rows and Columns: **
  n1+n2+n3+n4+n5+n6+n7+n8+n9=C             ... rw1;
  n10+n11+n12+n13+n14+n15+n16+n17+n18=C    ... rw2;
```

```
  n19+n20+n21+n22+n23+n24+n25+n26+n27=C    ... rw3;
  n28+n29+n30+n31+n32+n33+n34+n35+n36=C    ... rw4;
  n37+n38+n39+n40+n41+n42+n43+n44+n45=C    ... rw5;
  n46+n47+n48+n49+n50+n51+n52+n53+n54=C    ... rw6;
  n55+n56+n57+n58+n59+n60+n61+n62+n63=C    ... rw7;
  n64+n65+n66+n67+n68+n69+n70+n71+n72=C    ... rw8;
  n73+n74+n75+n76+n77+n78+n79+n80+n81=C    ... rw9;

  n1+n10+n19+n28+n37+n46+n55+n64+n73=C     ... cl1;
  n2+n11+n20+n29+n38+n47+n56+n65+n74=C     ... cl2;
  n3+n12+n21+n30+n39+n48+n57+n66+n75=C     ... cl3;
  n4+n13+n22+n31+n40+n49+n58+n67+n76=C     ... cl4;
  n5+n14+n23+n32+n41+n50+n59+n68+n77=C     ... cl5;
  n6+n15+n24+n33+n42+n51+n60+n69+n78=C     ... cl6;
  n7+n16+n25+n34+n43+n52+n61+n70+n79=C     ... cl7;
  n8+n17+n26+n35+n44+n53+n62+n71+n80=C     ... cl8;
  n9+n18+n27+n36+n45+n54+n63+n72+n81=C     ... cl9;
```

** Self-Complementary Conditions: **

```
  n1+n81=CC;   n2+n80=CC;   n3+n79=CC;   n4+n78=CC;
  n5+n77=CC;   n6+n76=CC;   n7+n75=CC;   n8+n74=CC;
  n9+n73=CC;   n10+n72=CC;  n11+n71=CC;  n12+n70=CC;
  n13+n69=CC;  n14+n68=CC;  n15+n67=CC;  n16+n66=CC;
  n17+n65=CC;  n18+n64=CC;  n19+n63=CC;  n20+n62=CC;
  n21+n61=CC;  n22+n60=CC;  n23+n59=CC;  n24+n58=CC;
  n25+n57=CC;  n26+n56=CC;  n27+n55=CC;  n28+n54=CC;
  n29+n53=CC;  n30+n52=CC;  n31+n51=CC;  n32+n50=CC;
  n33+n49=CC;  n34+n48=CC;  n35+n47=CC;  n36+n46=CC;
  n37+n45=CC;  n38+n44=CC;  n39+n43=CC;  n40+n42=CC;
  n41+n41=CC;  n42+n40=CC;  ......;
```

** Basic Form **

```
          .-------------------------.
 6  7  8  9| 1| 2| 3| 4| 5| 6| 7| 8| 9| 1  2  3  4  5
          |--+--+--+--+--+--+--+--+--|
15 16 17 18|10|11|12|13|14|15|16|17|18|10 11 12 13 14
          |--+--+--+--+--+--+--+--+--|
24 25 26 27|19|20|21|22|23|24|25|26|27|19 20 21 22 23
          |--+--+--+--+--+--+--+--+--|
33 34 35 36|28|29|30|31|32|33|34|35|36|28 29 30 31 32
          |--+--+--+--+--+--+--+--+--|
42 43 44 45|37|38|39|40|41|42|43|44|45|37 38 39 40 41
          |--+--+--+--+--+--+--+--+--|
51 52 53 54|46|47|48|49|50|51|52|53|54|46 47 48 49 50
          |--+--+--+--+--+--+--+--+--|
60 61 62 63|55|56|57|58|59|60|61|62|63|55 56 57 58 59
          |--+--+--+--+--+--+--+--+--|
69 70 71 72|64|65|66|67|68|69|70|71|72|64 65 66 67 68
          |--+--+--+--+--+--+--+--+--|
78 79 80 81|73|74|75|76|77|78|79|80|81|73 74 75 76 77
          '-------------------------'
```

** Pan-diagonal Conditions: **

```
  n1+n11+n21+n31+n41+n51+n61+n71+n81=C     ... pd1;
  n2+n12+n22+n32+n42+n52+n62+n72+n73=C     ... pd2;
  n3+n13+n23+n33+n43+n53+n63+n64+n74=C     ... pd3;
  n4+n14+n24+n34+n44+n54+n55+n65+n75=C     ... pd4;
  n5+n15+n25+n35+n45+n46+n56+n66+n76=C     ... pd5;
  n6+n16+n26+n36+n37+n47+n57+n67+n77=C     ... pd6;
```

```
    n7+n17+n27+n28+n38+n48+n58+n68+n78=C     ... pd7;
    n8+n18+n19+n29+n39+n49+n59+n69+n79=C     ... pd8;
    n9+n10+n20+n30+n40+n50+n60+n70+n80=C     ... pd9;

    n1+n18+n26+n34+n42+n50+n58+n66+n74=C     ... pb1;
    n2+n10+n27+n35+n43+n51+n59+n67+n75=C     ... pb2;
    n3+n11+n19+n36+n44+n52+n60+n68+n76=C     ... pb3;
    n4+n12+n20+n28+n45+n53+n61+n69+n77=C     ... pb4;
    n5+n13+n21+n29+n37+n54+n62+n70+n78=C     ... pb5;
    n6+n14+n22+n30+n38+n46+n63+n71+n79=C     ... pb6;
    n7+n15+n23+n31+n39+n47+n55+n72+n80=C     ... pb7;
    n8+n16+n24+n32+n40+n48+n56+n64+n81=C     ... pb8;
    n9+n17+n25+n33+n41+n49+n57+n65+n73=C     ... pb9;
```

Let me show you a core part of the program list I wrote recently as follows.

```
//** Simultaneous MS99: Self-Complementary and Pan-Diagonal **
//**  Designed and Composed by the "New Euler's Method"  **
//** File: 'CES9Sml.c': Dictated by Kanji Setsuda on Sep.17,'03; **
//**  Revised on May 16,'06 on MacOSX and Xcode 2.2; **
//**  Recompiled on Mar.13,'15 on MacOSX 10.10.2 and Xcode 6.2; **
//
//* Main Program *
//
int main(){
 short m,n;
  printf("\n* Simultaneous MS99: Self-complementary and Pan-diagonal *\n");
  printf("  **  Designed and Composed by 'New Euler's Method'  **\n");
  for(n=0;n<82;n++){nm[n]=-1; uflg[n]=0;}
  for(m=0;m<10;m++){
    for(n=0;n<3;n++){
      rw[m][n]=0; cl[m][n]=0; pd[m][n]=0; pb[m][n]=0;
    }}
  LSM=9; CC=2; cnt=0; cnt3=0;
  nm[41]=1; rw[5][1]=1; cl[5][1]=1; pd[1][1]=1; pb[9][1]=1;
  printf(" [List of Latin Units]\n");
  stp01();
  if(cnt3>3){pr4f();}
  else if(cnt3>0){pr2f();}
  printf(" [Count of Latin Units = %ld]  OK!\n",cnt);
 return 0;
}
//
//* Making Latin Units *
//* Set n1 & n81 *
void stp01(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[1][a]<3)&&(pd[1][a]<3)&&(pb[1][a]<3)){
      if((rw[9][b]<3)&&(cl[9][b]<3)&&(pd[1][b]<3)&&(pb[8][b]<3)){
        nm[1]=a; nm[81]=b;
        rw[1][a]++; cl[1][a]++; pd[1][a]++; pb[1][a]++;
        rw[9][b]++; cl[9][b]++; pd[1][b]++; pb[8][b]++;
        stp02();
        rw[1][a]--; cl[1][a]--; pd[1][a]--; pb[1][a]--;
        rw[9][b]--; cl[9][b]--; pd[1][b]--; pb[8][b]--;
      }}
  }
}
//* Set n9 & n73 *
void stp02(){
 short a,b;
```

```
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[9][a]<3)&&(pd[9][a]<3)&&(pb[9][a]<3)){
      if((rw[9][b]<3)&&(cl[1][b]<3)&&(pd[2][b]<3)&&(pb[9][b]<3)){
        nm[9]=a; nm[73]=b;
        rw[1][a]++; cl[9][a]++; pd[9][a]++; pb[9][a]++;
        rw[9][b]++; cl[1][b]++; pd[2][b]++; pb[9][b]++;
        stp03();
        rw[1][a]--; cl[9][a]--; pd[9][a]--; pb[9][a]--;
        rw[9][b]--; cl[1][b]--; pd[2][b]--; pb[9][b]--;
      }}
  }
}
//* Set n11 & n71 *
void stp03(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[2][a]<3)&&(cl[2][a]<3)&&(pd[1][a]<3)&&(pb[3][a]<3)){
      if((rw[8][b]<3)&&(cl[8][b]<3)&&(pd[1][b]<3)&&(pb[6][b]<3)){
        nm[11]=a; nm[71]=b;
        rw[2][a]++; cl[2][a]++; pd[1][a]++; pb[3][a]++;
        rw[8][b]++; cl[8][b]++; pd[1][b]++; pb[6][b]++;
        stp04();
        rw[2][a]--; cl[2][a]--; pd[1][a]--; pb[3][a]--;
        rw[8][b]--; cl[8][b]--; pd[1][b]--; pb[6][b]--;
      }}
  }
}
//* Set n17 & n65 *
void stp04(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[2][a]<3)&&(cl[8][a]<3)&&(pd[7][a]<3)&&(pb[9][a]<3)){
      if((rw[8][b]<3)&&(cl[2][b]<3)&&(pd[4][b]<3)&&(pb[9][b]<3)){
        nm[17]=a; nm[65]=b; cnt2=0;
        rw[2][a]++; cl[8][a]++; pd[7][a]++; pb[9][a]++;
        rw[8][b]++; cl[2][b]++; pd[4][b]++; pb[9][b]++;
        stp05();
        rw[2][a]--; cl[8][a]--; pd[7][a]--; pb[9][a]--;
        rw[8][b]--; cl[2][b]--; pd[4][b]--; pb[9][b]--;
      }}
  }
}
//* Set n21 & n61 *
void stp05(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[3][a]<3)&&(cl[3][a]<3)&&(pd[1][a]<3)&&(pb[5][a]<3)){
      if((rw[7][b]<3)&&(cl[7][b]<3)&&(pd[1][b]<3)&&(pb[4][b]<3)){
        nm[21]=a; nm[61]=b;
        rw[3][a]++; cl[3][a]++; pd[1][a]++; pb[5][a]++;
        rw[7][b]++; cl[7][b]++; pd[1][b]++; pb[4][b]++;
        stp06();
        rw[3][a]--; cl[3][a]--; pd[1][a]--; pb[5][a]--;
        rw[7][b]--; cl[7][b]--; pd[1][b]--; pb[4][b]--;
      }}
  }
}
//* Set n25 & n57 *
void stp06(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[3][a]<3)&&(cl[7][a]<3)&&(pd[5][a]<3)&&(pb[9][a]<3)){
      if((rw[7][b]<3)&&(cl[3][b]<3)&&(pd[6][b]<3)&&(pb[9][b]<3)){
        nm[25]=a; nm[57]=b;
        rw[3][a]++; cl[7][a]++; pd[5][a]++; pb[9][a]++;
```

```
          rw[7][b]++; cl[3][b]++; pd[6][b]++; pb[9][b]++;
          stp07();
          rw[3][a]--; cl[7][a]--; pd[5][a]--; pb[9][a]--;
          rw[7][b]--; cl[3][b]--; pd[6][b]--; pb[9][b]--;
        }}
    }
}
//* Set n31 & n51 *
void stp07(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[4][a]<3)&&(cl[4][a]<3)&&(pd[1][a]<3)&&(pb[7][a]<3)){
      if((rw[6][b]<3)&&(cl[6][b]<3)&&(pd[1][b]<3)&&(pb[2][b]<3)){
        nm[31]=a; nm[51]=b;
        rw[4][a]++; cl[4][a]++; pd[1][a]++; pb[7][a]++;
        rw[6][b]++; cl[6][b]++; pd[1][b]++; pb[2][b]++;
        stp08();
        rw[4][a]--; cl[4][a]--; pd[1][a]--; pb[7][a]--;
        rw[6][b]--; cl[6][b]--; pd[1][b]--; pb[2][b]--;
      }}
  }
}
//* Set n33 & n49 *
void stp08(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[4][a]<3)&&(cl[6][a]<3)&&(pd[3][a]<3)&&(pb[9][a]<3)){
      if((rw[6][b]<3)&&(cl[4][b]<3)&&(pd[8][b]<3)&&(pb[9][b]<3)){
        nm[33]=a; nm[49]=b;
        rw[4][a]++; cl[6][a]++; pd[3][a]++; pb[9][a]++;
        rw[6][b]++; cl[4][b]++; pd[8][b]++; pb[9][b]++;
        stp09();
        rw[4][a]--; cl[6][a]--; pd[3][a]--; pb[9][a]--;
        rw[6][b]--; cl[4][b]--; pd[8][b]--; pb[9][b]--;
      }}
  }
}
//
//* Search Level 2: *
//* Set n2 & n80 *
void stp09(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[2][a]<3)&&(pd[2][a]<3)&&(pb[2][a]<3)){
      if((rw[9][b]<3)&&(cl[8][b]<3)&&(pd[9][b]<3)&&(pb[7][b]<3)){
        nm[2]=a; nm[80]=b;
        rw[1][a]++; cl[2][a]++; pd[2][a]++; pb[2][a]++;
        rw[9][b]++; cl[8][b]++; pd[9][b]++; pb[7][b]++;
        stp10();
        rw[1][a]--; cl[2][a]--; pd[2][a]--; pb[2][a]--;
        rw[9][b]--; cl[8][b]--; pd[9][b]--; pb[7][b]--;
      }}
  }
}
//* Set n3 & n79 *
void stp10(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[3][a]<3)&&(pd[3][a]<3)&&(pb[3][a]<3)){
      if((rw[9][b]<3)&&(cl[7][b]<3)&&(pd[8][b]<3)&&(pb[6][b]<3)){
        nm[3]=a; nm[79]=b;
        rw[1][a]++; cl[3][a]++; pd[3][a]++; pb[3][a]++;
        rw[9][b]++; cl[7][b]++; pd[8][b]++; pb[6][b]++;
        stp11();
        rw[1][a]--; cl[3][a]--; pd[3][a]--; pb[3][a]--;
```

```
        rw[9][b]--; cl[7][b]--; pd[8][b]--; pb[6][b]--;
      }}
  }
}
//* Set n4 & n78 *
void stp11(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[4][a]<3)&&(pd[4][a]<3)&&(pb[4][a]<3)){
      if((rw[9][b]<3)&&(cl[6][b]<3)&&(pd[7][b]<3)&&(pb[5][b]<3)){
        nm[4]=a; nm[78]=b;
        rw[1][a]++; cl[4][a]++; pd[4][a]++; pb[4][a]++;
        rw[9][b]++; cl[6][b]++; pd[7][b]++; pb[5][b]++;
        stp12();
        rw[1][a]--; cl[4][a]--; pd[4][a]--; pb[4][a]--;
        rw[9][b]--; cl[6][b]--; pd[7][b]--; pb[5][b]--;
      }}
  }
}
//* Set n5 & n77 *
void stp12(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[5][a]<3)&&(pd[5][a]<3)&&(pb[5][a]<3)){
      if((rw[9][b]<3)&&(cl[5][b]<3)&&(pd[6][b]<3)&&(pb[4][b]<3)){
        nm[5]=a; nm[77]=b;
        rw[1][a]++; cl[5][a]++; pd[5][a]++; pb[5][a]++;
        rw[9][b]++; cl[5][b]++; pd[6][b]++; pb[4][b]++;
        stp13();
        rw[1][a]--; cl[5][a]--; pd[5][a]--; pb[5][a]--;
        rw[9][b]--; cl[5][b]--; pd[6][b]--; pb[4][b]--;
      }}
  }
}
//* Set n6 & n76 *
void stp13(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[6][a]<3)&&(pd[6][a]<3)&&(pb[6][a]<3)){
      if((rw[9][b]<3)&&(cl[4][b]<3)&&(pd[5][b]<3)&&(pb[3][b]<3)){
        nm[6]=a; nm[76]=b;
        rw[1][a]++; cl[6][a]++; pd[6][a]++; pb[6][a]++;
        rw[9][b]++; cl[4][b]++; pd[5][b]++; pb[3][b]++;
        stp14();
        rw[1][a]--; cl[6][a]--; pd[6][a]--; pb[6][a]--;
        rw[9][b]--; cl[4][b]--; pd[5][b]--; pb[3][b]--;
      }}
  }
}
//* Set n7 & n75 *
void stp14(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[7][a]<3)&&(pd[7][a]<3)&&(pb[7][a]<3)){
      if((rw[9][b]<3)&&(cl[3][b]<3)&&(pd[4][b]<3)&&(pb[2][b]<3)){
        nm[7]=a; nm[75]=b;
        rw[1][a]++; cl[7][a]++; pd[7][a]++; pb[7][a]++;
        rw[9][b]++; cl[3][b]++; pd[4][b]++; pb[2][b]++;
        stp15();
        rw[1][a]--; cl[7][a]--; pd[7][a]--; pb[7][a]--;
        rw[9][b]--; cl[3][b]--; pd[4][b]--; pb[2][b]--;
      }}
  }
}
//* Set n8 & n74 *
```

```
void stp15(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[1][a]<3)&&(cl[8][a]<3)&&(pd[8][a]<3)&&(pb[8][a]<3)){
      if((rw[9][b]<3)&&(cl[2][b]<3)&&(pd[3][b]<3)&&(pb[1][b]<3)){
        nm[8]=a; nm[74]=b;
        rw[1][a]++; cl[8][a]++; pd[8][a]++; pb[8][a]++;
        rw[9][b]++; cl[2][b]++; pd[3][b]++; pb[1][b]++;
        stp16();
        rw[1][a]--; cl[8][a]--; pd[8][a]--; pb[8][a]--;
        rw[9][b]--; cl[2][b]--; pd[3][b]--; pb[1][b]--;
      }}
  }
}
//* Set n10 & n72 *
void stp16(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw[2][a]<3)&&(cl[1][a]<3)&&(pd[9][a]<3)&&(pb[2][a]<3)){
      if((rw[8][b]<3)&&(cl[9][b]<3)&&(pd[2][b]<3)&&(pb[7][b]<3)){
        nm[10]=a; nm[72]=b;
        rw[2][a]++; cl[1][a]++; pd[9][a]++; pb[2][a]++;
        rw[8][b]++; cl[9][b]++; pd[2][b]++; pb[7][b]++;
        stp17();
        rw[2][a]--; cl[1][a]--; pd[9][a]--; pb[2][a]--;
        rw[8][b]--; cl[9][b]--; pd[2][b]--; pb[7][b]--;
      }}
  }
}
//
//* ---(Skip)--- *
//
```

I have got so many Latin units as about 45-thousands. What a great number!

```
* Simultaneous MS99: Self-complementary and Pan-diagonal *
  **  Designed and Composed by "New Euler's Method"  **
 [List of Latin Units(Part)]
         1/       521/      1443/      1969/      2891/      3847/
  002221110  001212210  012210120  001122120  001212210  001212210
  101210202  202201011  001202121  112202100  111200022  121202100
  110022012  210122001  120102012  210210012  220121001  210201021
  220101120  021100122  122101200  122001201  022021110  022101021
  122012001  120012201  210012210  000111222  100012221  010012212
  201121200  001221102  220121001  120122001  211102002  102121002
  012002211  122001210  012021201  012210210  122101200  102120210
  020210121  112120020  101020122  221020011  002220111  221020101
  211100022  210010122  201210012  201001122  210010122  210010122
      4373/      5329/      5885/      6726/      7567/      8506/
  001122210  002112210  001122201  001222011  001122201  001122021
  122200110  121200120  202201101  201112020  012221001  211201020
  020112012  120202011  210212010  220120011  210011022  120221010
  121100220  021101220  121100022  020120112  221020110  120010212
  102012021  012012012  010012212  120012201  100210221  021012102
  200221101  200121102  002221101  011201202  211202100  010212201
  012011202  112020201  212010210  112201200  002112210  212100201
  211220001  201220101  121120020  202011120  122100012  202120110
  210001122  210011022  120001122  112000122  120001122  102001122
      9445/     10232/     11019/     11545/     12467/     12987/
  000222111  001122201  001212012  001121202  001221012  000221112
  221112000  122100120  101202102  102220011  101212020  112202100
  100221012  210212010  220120011  210012012  210202011  210211020
```

221100012	022100121	120121020	221120100	220101102	121001022
100012221	010012212	210012210	100012221	120012201	120012201
012221100	101221002	202101201	221201100	021121200	002122101
012100221	212010210	112201200	012012210	112020210	202110210
222011100	201221001	021020121	112200021	202010121	221020011
111000222	120001122	012010122	020101122	012100122	011100222

13943/	14865/	15421/	16377/	16903/	17744/
001121202	001012122	001012122	010221012	101122020	101122200
112200120	121220001	122200011	021210120	102212100	002221011
210202011	210221010	220121010	110022021	210120012	210112002
122011200	220101021	011120220	220101120	120001212	221000112
001012122	010012212	120012201	122012001	021210102	100210221
220112001	102121200	200201112	201121200	010122201	011222100
112020210	212100210	212101200	102002211	012201210	022011210
201220011	122200101	112220001	201210102	221010021	112100022
020101122	001012122	001012122	012100212	202001121	220001121

18683/	19470/	20311/	21250/	22037/	22287/
101222010	100121220	100221120	100221120	102021201	101220201
001211022	221210001	221201010	121210020	002121102	002121120
210122001	010222011	100122012	210021012	210212001	210112002
220001112	221001012	112010202	012021201	211000122	122000112
120210201	001012122	021012102	021012102	210210210	210012210
011122200	012122100	020212011	120102012	001222110	011222001
122001210	112000212	012001221	012102210	122010210	022011210
002110122	122210100	212120100	202210101	021101022	201101022
212000121	200101221	201100221	201100221	120102021	120200121

22537/	22787/	23037/	23824/	24763/	25604/
101020221	100221201	100220112	100211202	100221102	100021122
221121000	221101020	202112100	202201011	201201021	122210100
020211012	020212011	210120021	210122001	210212010	210120012
101020212	011020122	021021012	011020122	121021002	112021200
212012010	212012010	120012201	221012100	010012212	001012122
010202121	001202112	012102102	001202112	022102101	220102011
012110202	112010202	102201210	122001210	212010210	012201210
222101100	202121100	221011020	112120020	102120120	221210001
100202121	120100221	011200221	020110221	021100221	001102221

26391/	27330/	28171/	28697/	29653/	30209/
100121022	100121022	201112200	200212110	202120110	200112120
121202010	121210020	102202101	201202011	102012120	211202001
020211012	110222010	210202011	120121002	110202012	120220011
211020201	220001112	021101022	021120012	021101202	021021012
201012120	021012102	010210212	100012221	120210201	000111222
120202110	011122200	002121102	012201102	020121102	012102102
012110202	212000211	112020210	022101201	012020211	112200201
212020101	202210101	121020021	112020120	201012021	122020110
002101221	002101221	220011120	211010220	211201020	201011220

31131/	32087/	32607/	33529/	34055/	34842/
200212110	200122110	200112210	200211120	200121201	200220111
012201021	122012100	121220010	121200021	202210101	202111020
210122001	110220012	120021012	110022012	210022011	110022012
021001122	021101022	021120021	022121100	011120022	021120102
120210201	100210221	001012122	010012212	021012102	120012201
001122102	002121102	102201102	221101002	002201112	021201102
122001210	012200211	012102201	012002211	112002210	012002211
102120012	221012001	212200101	102220101	121210020	202111020
211010220	211001220	210011220	201110220	120101220	111200220

35629/	36568/	37507/	38348/	39189/	39745/
200021121	200121201	200012121	200122011	200120112	200121012
212201100	212100120	221201001	021201120	102212100	102202110
110221002	110222001	120221010	120212010	110202012	110212002
112010202	022010112	011120022	121100022	221101002	221100102

```
001012122  001012122  010012212  010012212  100012221  100012221
020212011  011212002  002201112  002221101  022121100  021221100
022100211  122000211  212100201  212010201  012020211  022010211
221120010  201221010  122120100  201120102  221010021  211020021
101102220  120101220  101012220  112001220  011201220  012101220

   40701/     41227/     42183/     43105/     43631/     44553/
202010112  200011212  200112102  201002112  200012112  200120112
101220120  112220100  112200021  121212000  122200110  122010120
120102012  210221001  210122010  210201012  120121002  110222010
120121200  121001022  020021121  000121221  011120220  021101022
012012012  001012122  021210102  122210001  102012021  120012201
220101201  002122101  101102202  100101222  200201112  002121102
012021201  122100210  212001210  012120210  022101201  212000211
201200121  221200011  102220011  222010101  211220001  201212001
011212020  010112220  021011220  011022120  011012220  011201220
[Count of Latin Units = 45072]  OK!
```

We now have to examine about 22536 x 22535 x 22534 x 22533 pieces of our compositions, preparing memory arrays: tlu[45073][82] and mtc[45073][45073].

What a scale of job we have to have!

I am sorry to say I could not do that job with my little old machine, and I had no time enough to count them up to the last, either. I gave it up at last.

4. How about Multiple Type Including Nine 3x3 Little Squares within?

Is there any way to make such a rare and interesting object as 'Composite' type of order 4 or 8? Why don't we make 'Multiple' type, such as the one including nine 3x3 little squares within?

```
** Multiple 3*3 Conditions: **
 Block #0:
  n1+n2+n3=C    ...rw00;  n10+n11+n12=C ...rw01;  n19+n20+n21=C ...rw02;
  n1+n10+n19=C  ...cl00;  n2+n11+n20=C  ...cl01;  n3+n12+n21=C  ...cl02;
 Block #1:
  n4+n5+n6=C    ...rw10;  n13+n14+n15=C ...rw11;  n22+n23+n24=C ...rw12;
  n4+n13+n22=C  ...cl10;  n5+n14+n23=C  ...cl11;  n6+n15+n24=C  ...cl12;
 Block #2:
  n7+n8+n9=C    ...rw20;  n16+n17+n18=C ...rw21;  n25+n26+n27=C ...rw22;
  n7+n16+n25=C  ...cl20;  n8+n17+n26=C  ...cl21;  n9+n18+n27=C  ...cl22;

 ** Basic Form for Multiple & Simultaneous MS99 **
           .-------------------------.
  6  7  8  9| 1| 2| 3| 4| 5| 6| 7| 8| 9| 1  2  3  4  5
           |--+--+--|--+--+--|--+--+--|
 15 16 17 18|10|11|12|13|14|15|16|17|18|10 11 12 13 14
           |--+--+--|--+--+--|--+--+--|
 24 25 26 27|19|20|21|22|23|24|25|26|27|19 20 21 22 23
           |--------+--------+--------|
 33 34 35 36|28|29|30|31|32|33|34|35|36|28 29 30 31 32
           |--+--+--|--+--+--|--+--+--|
 42 43 44 45|37|38|39|40|41|42|43|44|45|37 38 39 40 41
           |--+--+--|--+--+--|--+--+--|
 51 52 53 54|46|47|48|49|50|51|52|53|54|46 47 48 49 50
           |--------+--------+--------|
 60 61 62 63|55|56|57|58|59|60|61|62|63|55 56 57 58 59
           |--+--+--|--+--+--|--+--+--|
 69 70 71 72|64|65|66|67|68|69|70|71|72|64 65 66 67 68
           |--+--+--|--+--+--|--+--+--|
 78 79 80 81|73|74|75|76|77|78|79|80|81|73 74 75 76 77
           '-------------------------'
```

```
 Block #3:
  n28+n29+n30=C ...rw30;  n37+n38+n39=C ...rw31;  n46+n47+n48=C ...rw32;
  n28+n37+n46=C ...cl30;  n29+n38+n47=C ...cl31;  n30+n39+n48=C ...cl32;
 Block #4:
  n31+n32+n33=C ...rw40;  n40+n41+n42=C ...rw41;  n49+n50+n51=C ...rw42;
  n31+n40+n49=C ...cl40;  n32+n41+n50=C ...cl41;  n33+n42+n51=C ...cl42;
 Block #5:
  n34+n35+n36=C ...rw50;  n43+n44+n45=C ...rw51;  n52+n53+n54=C ...rw52;
  n34+n43+n52=C ...cl50;  n35+n44+n53=C ...cl51;  n36+n45+n54=C ...cl52;
 Block #6:
  n55+n56+n57=C ...rw60;  n64+n65+n66=C ...rw61;  n73+n74+n75=C ...rw62;
  n55+n64+n73=C ...cl60;  n56+n65+n74=C ...cl61;  n57+n66+n75=C ...cl62;
 Block #7:
  n58+n59+n60=C ...rw70;  n67+n68+n69=C ...rw71;  n76+n77+n78=C ...rw72;
  n58+n67+n76=C ...cl70;  n59+n68+n77=C ...cl71;  n60+n69+n78=C ...cl72;
 Block #8:
  n61+n62+n63=C ...rw80;  n70+n71+n72=C ...rw81;  n79+n80+n81=C ...rw82;
  n61+n70+n79=C ...cl80;  n62+n71+n80=C ...cl81;  n63+n72+n81=C ...cl82;
```

Let's prepare the new basic form and equations, and substitute the old basic conditions with the new Multiple 3x3 Conditions as shown above, defining each block by six equations.

You should have as many flags as various conditions to watch if every line has {0, 1, and 2}.

You don't have to define two diagonals of each little square in advance, though you have to define the constant sum of every pan-diagonal of the whole square 9x9.

Let me show you a core part of my recent program for this object.

```
//** Multiple Type of Simultaneous Magic Squares of Order 9: **
//**  Self-complementary & Pan-diagonal; by 'New Euler's Method' **
//** File: 'CES9MSmlF.c' Dictated by Kanji Setsuda on  **
//**  Sep. 3, 2003 and May 15, 2006; Recompiled on **
//**  Feb.28, 2015 with MacOSX 10.10.2 and Xcode 6.1.1 **
//
//* Main Program *
//
int main(){
 short m,n;
  printf("\n");
  printf("** Multiple Type of Simultaneous Magic Squares of Order 9: **\n");
  printf("**  Self-complementary & Pan-diagonal; by 'New Euler's Method' **\n");
  for(n=0;n<82;n++){nm[n]=-1; flg[n]=0;}
  for(m=0;m<3;m++){
    for(n=0;n<3;n++){
      rw0[m][n]=0; rw1[m][n]=0; rw2[m][n]=0;
      cl0[m][n]=0; cl1[m][n]=0; cl2[m][n]=0;
      rw3[m][n]=0; rw4[m][n]=0; rw5[m][n]=0;
      cl3[m][n]=0; cl4[m][n]=0; cl5[m][n]=0;
      rw6[m][n]=0; rw7[m][n]=0; rw8[m][n]=0;
      cl6[m][n]=0; cl7[m][n]=0; cl8[m][n]=0;
    }}
  for(m=0;m<10;m++){
    for(n=0;n<3;n++){
      pd[m][n]=0; pb[m][n]=0;
    }}
  CC=2; cnt=0;
  nm[41]=1;
  rw4[1][1]=1; cl4[1][1]=1; pd[1][1]=1; pb[9][1]=1;
  stp01();
  printf("\n");
  printf(" [List of Latin Units]\n");
```

```
   pr9lu();
   printf("\n");
   printf(" [List of Compositions: Used Units//// Count[n1=1/Total]]\n");
   lcnt=cnt; cnt=0; cnt3=0; cnt4=0;
   cmbcmp(lcnt);
   if(cnt3==2){pr2ans();}
   if(cnt3==1){pr1ans();}
   printf(" [Count(n1=1/Total) = %d/%d]  OK!\n",cnt4,cnt);
   printf("\n");
   printf("** Recompiled and Listed by Kanji Setsuda on\n");
   printf("    Feb.28, 2015 with MacOSX 10.10.2 & Xcode 6.1.1 **\n");
   printf("\n");
 return 0;
}
//
//* Making Latin Squares *
//* Set n1 & n81 *
void stp01(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw0[0][a]==0)&&(cl0[0][a]==0)&&(pd[1][a]<3)&&(pb[1][a]<3)){
      if((rw8[2][b]==0)&&(cl8[2][b]==0)&&(pd[1][b]<3)&&(pb[8][b]<3)){
        nm[1]=a; nm[81]=b;
        rw0[0][a]=1; cl0[0][a]=1; pd[1][a]++; pb[1][a]++;
        rw8[2][b]=1; cl8[2][b]=1; pd[1][b]++; pb[8][b]++;
        stp02();
        rw0[0][a]=0; cl0[0][a]=0; pd[1][a]--; pb[1][a]--;
        rw8[2][b]=0; cl8[2][b]=0; pd[1][b]--; pb[8][b]--;
      }}
  }
}
//* Set n2 & n80 *
void stp02(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw0[0][a]==0)&&(cl0[1][a]==0)&&(pd[2][a]<3)&&(pb[2][a]<3)){
      if((rw8[2][b]==0)&&(cl8[1][b]==0)&&(pd[9][b]<3)&&(pb[7][b]<3)){
        nm[2]=a; nm[80]=b;
        rw0[0][a]=1; cl0[1][a]=1; pd[2][a]++; pb[2][a]++;
        rw8[2][b]=1; cl8[1][b]=1; pd[9][b]++; pb[7][b]++;
        stp03();
        rw0[0][a]=0; cl0[1][a]=0; pd[2][a]--; pb[2][a]--;
        rw8[2][b]=0; cl8[1][b]=0; pd[9][b]--; pb[7][b]--;
      }}
  }
}
//* Set n3 & n79 *
void stp03(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw0[0][a]==0)&&(cl0[2][a]==0)&&(pd[3][a]<3)&&(pb[3][a]<3)){
      if((rw8[2][b]==0)&&(cl8[0][b]==0)&&(pd[8][b]<3)&&(pb[6][b]<3)){
        nm[3]=a; nm[79]=b;
        rw0[0][a]=1; cl0[2][a]=1; pd[3][a]++; pb[3][a]++;
        rw8[2][b]=1; cl8[0][b]=1; pd[8][b]++; pb[6][b]++;
        stp04();
        rw0[0][a]=0; cl0[2][a]=0; pd[3][a]--; pb[3][a]--;
        rw8[2][b]=0; cl8[0][b]=0; pd[8][b]--; pb[6][b]--;
      }}
  }
}
//* Set n10 & n72 *
void stp04(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
```

```
    if((rw0[1][a]==0)&&(cl0[0][a]==0)&&(pd[9][a]<3)&&(pb[2][a]<3)){
      if((rw8[1][b]==0)&&(cl8[2][b]==0)&&(pd[2][b]<3)&&(pb[7][b]<3)){
        nm[10]=a; nm[72]=b;
        rw0[1][a]=1; cl0[0][a]=1; pd[9][a]++; pb[2][a]++;
        rw8[1][b]=1; cl8[2][b]=1; pd[2][b]++; pb[7][b]++;
        stp05();
        rw0[1][a]=0; cl0[0][a]=0; pd[9][a]--; pb[2][a]--;
        rw8[1][b]=0; cl8[2][b]=0; pd[2][b]--; pb[7][b]--;
      }}
  }
}
//* Set n19 & n63 *
void stp05(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw0[2][a]==0)&&(cl0[0][a]==0)&&(pd[8][a]<3)&&(pb[3][a]<3)){
      if((rw8[0][b]==0)&&(cl8[2][b]==0)&&(pd[3][b]<3)&&(pb[6][b]<3)){
        nm[19]=a; nm[63]=b;
        rw0[2][a]=1; cl0[0][a]=1; pd[8][a]++; pb[3][a]++;
        rw8[0][b]=1; cl8[2][b]=1; pd[3][b]++; pb[6][b]++;
        stp06();
        rw0[2][a]=0; cl0[0][a]=0; pd[8][a]--; pb[3][a]--;
        rw8[0][b]=0; cl8[2][b]=0; pd[3][b]--; pb[6][b]--;
      }}
  }
}
//* Set n11 & n71 *
void stp06(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw0[1][a]==0)&&(cl0[1][a]==0)&&(pd[1][a]<3)&&(pb[3][a]<3)){
      if((rw8[1][b]==0)&&(cl8[1][b]==0)&&(pd[1][b]<3)&&(pb[6][b]<3)){
        nm[11]=a; nm[71]=b;
        rw0[1][a]=1; cl0[1][a]=1; pd[1][a]++; pb[3][a]++;
        rw8[1][b]=1; cl8[1][b]=1; pd[1][b]++; pb[6][b]++;
        stp07();
        rw0[1][a]=0; cl0[1][a]=0; pd[1][a]--; pb[3][a]--;
        rw8[1][b]=0; cl8[1][b]=0; pd[1][b]--; pb[6][b]--;
      }}
  }
}
//* Set n12 & n70 *
void stp07(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw0[1][a]==0)&&(cl0[2][a]==0)&&(pd[2][a]<3)&&(pb[4][a]<3)){
      if((rw8[1][b]==0)&&(cl8[0][b]==0)&&(pd[9][b]<3)&&(pb[5][b]<3)){
        nm[12]=a; nm[70]=b;
        rw0[1][a]=1; cl0[2][a]=1; pd[2][a]++; pb[4][a]++;
        rw8[1][b]=1; cl8[0][b]=1; pd[9][b]++; pb[5][b]++;
        stp08();
        rw0[1][a]=0; cl0[2][a]=0; pd[2][a]--; pb[4][a]--;
        rw8[1][b]=0; cl8[0][b]=0; pd[9][b]--; pb[5][b]--;
      }}
  }
}
//* Set n20 & n62 *
void stp08(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw0[2][a]==0)&&(cl0[1][a]==0)&&(pd[9][a]<3)&&(pb[4][a]<3)){
      if((rw8[0][b]==0)&&(cl8[1][b]==0)&&(pd[2][b]<3)&&(pb[5][b]<3)){
        nm[20]=a; nm[62]=b;
        rw0[2][a]=1; cl0[1][a]=1; pd[9][a]++; pb[4][a]++;
        rw8[0][b]=1; cl8[1][b]=1; pd[2][b]++; pb[5][b]++;
```

```
          stp09();
          rw0[2][a]=0; cl0[1][a]=0; pd[9][a]--; pb[4][a]--;
          rw8[0][b]=0; cl8[1][b]=0; pd[2][b]--; pb[5][b]--;
        }}
    }
}
//* Set n21 & n61 *
void stp09(){
 short a,b;
   for(a=0;a<3;a++){b=CC-a;
     if((rw0[2][a]==0)&&(cl0[2][a]==0)&&(pd[1][a]<3)&&(pb[5][a]<3)){
       if((rw8[0][b]==0)&&(cl8[0][b]==0)&&(pd[1][b]<3)&&(pb[4][b]<3)){
         nm[21]=a; nm[61]=b;
         rw0[2][a]=1; cl0[2][a]=1; pd[1][a]++; pb[5][a]++;
         rw8[0][b]=1; cl8[0][b]=1; pd[1][b]++; pb[4][b]++;
         stp10();
         rw0[2][a]=0; cl0[2][a]=0; pd[1][a]--; pb[5][a]--;
         rw8[0][b]=0; cl8[0][b]=0; pd[1][b]--; pb[4][b]--;
       }}
   }
}
//* Level #2: *
//* Set n4 & n78 *
void stp10(){
 short a,b;
   for(a=0;a<3;a++){b=CC-a;
     if((rw1[0][a]==0)&&(cl1[0][a]==0)&&(pd[4][a]<3)&&(pb[4][a]<3)){
       if((rw7[2][b]==0)&&(cl7[2][b]==0)&&(pd[7][b]<3)&&(pb[5][b]<3)){
         nm[4]=a; nm[78]=b;
         rw1[0][a]=1; cl1[0][a]=1; pd[4][a]++; pb[4][a]++;
         rw7[2][b]=1; cl7[2][b]=1; pd[7][b]++; pb[5][b]++;
         stp11();
         rw1[0][a]=0; cl1[0][a]=0; pd[4][a]--; pb[4][a]--;
         rw7[2][b]=0; cl7[2][b]=0; pd[7][b]--; pb[5][b]--;
       }}
   }
}
//* Set n5 & n77 *
void stp11(){
 short a,b;
   for(a=0;a<3;a++){b=CC-a;
     if((rw1[0][a]==0)&&(cl1[1][a]==0)&&(pd[5][a]<3)&&(pb[5][a]<3)){
       if((rw7[2][b]==0)&&(cl7[1][b]==0)&&(pd[6][b]<3)&&(pb[4][b]<3)){
         nm[5]=a; nm[77]=b;
         rw1[0][a]=1; cl1[1][a]=1; pd[5][a]++; pb[5][a]++;
         rw7[2][b]=1; cl7[1][b]=1; pd[6][b]++; pb[4][b]++;
         stp12();
         rw1[0][a]=0; cl1[1][a]=0; pd[5][a]--; pb[5][a]--;
         rw7[2][b]=0; cl7[1][b]=0; pd[6][b]--; pb[4][b]--;
       }}
   }
}
//* Set n6 & n76 *
void stp12(){
 short a,b;
   for(a=0;a<3;a++){b=CC-a;
     if((rw1[0][a]==0)&&(cl1[2][a]==0)&&(pd[6][a]<3)&&(pb[6][a]<3)){
       if((rw7[2][b]==0)&&(cl7[0][b]==0)&&(pd[5][b]<3)&&(pb[3][b]<3)){
         nm[6]=a; nm[76]=b;
         rw1[0][a]=1; cl1[2][a]=1; pd[6][a]++; pb[6][a]++;
         rw7[2][b]=1; cl7[0][b]=1; pd[5][b]++; pb[3][b]++;
         stp13();
         rw1[0][a]=0; cl1[2][a]=0; pd[6][a]--; pb[6][a]--;
         rw7[2][b]=0; cl7[0][b]=0; pd[5][b]--; pb[3][b]--;
       }}
```

```
  }
}
//* Set n7 & n75 *
void stp13(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[0][a]==0)&&(cl2[0][a]==0)&&(pd[7][a]<3)&&(pb[7][a]<3)){
      if((rw6[2][b]==0)&&(cl6[2][b]==0)&&(pd[4][b]<3)&&(pb[2][b]<3)){
        nm[7]=a; nm[75]=b;
        rw2[0][a]=1; cl2[0][a]=1; pd[7][a]++; pb[7][a]++;
        rw6[2][b]=1; cl6[2][b]=1; pd[4][b]++; pb[2][b]++;
        stp14();
        rw2[0][a]=0; cl2[0][a]=0; pd[7][a]--; pb[7][a]--;
        rw6[2][b]=0; cl6[2][b]=0; pd[4][b]--; pb[2][b]--;
      }}
  }
}
//* Set n8 & n74 *
void stp14(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[0][a]==0)&&(cl2[1][a]==0)&&(pd[8][a]<3)&&(pb[8][a]<3)){
      if((rw6[2][b]==0)&&(cl6[1][b]==0)&&(pd[3][b]<3)&&(pb[1][b]<3)){
        nm[8]=a; nm[74]=b;
        rw2[0][a]=1; cl2[1][a]=1; pd[8][a]++; pb[8][a]++;
        rw6[2][b]=1; cl6[1][b]=1; pd[3][b]++; pb[1][b]++;
        stp15();
        rw2[0][a]=0; cl2[1][a]=0; pd[8][a]--; pb[8][a]--;
        rw6[2][b]=0; cl6[1][b]=0; pd[3][b]--; pb[1][b]--;
      }}
  }
}
//* Set n9 & n73 *
void stp15(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[0][a]==0)&&(cl2[2][a]==0)&&(pd[9][a]<3)&&(pb[9][a]<3)){
      if((rw6[2][b]==0)&&(cl6[0][b]==0)&&(pd[2][b]<3)&&(pb[9][b]<3)){
        nm[9]=a; nm[73]=b;
        rw2[0][a]=1; cl2[2][a]=1; pd[9][a]++; pb[9][a]++;
        rw6[2][b]=1; cl6[0][b]=1; pd[2][b]++; pb[9][b]++;
        stp16();
        rw2[0][a]=0; cl2[2][a]=0; pd[9][a]--; pb[9][a]--;
        rw6[2][b]=0; cl6[0][b]=0; pd[2][b]--; pb[9][b]--;
      }}
  }
}
//* Set n13 & n69 *
void stp16(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw1[1][a]==0)&&(cl1[0][a]==0)&&(pd[3][a]<3)&&(pb[5][a]<3)){
      if((rw7[1][b]==0)&&(cl7[2][b]==0)&&(pd[8][b]<3)&&(pb[4][b]<3)){
        nm[13]=a; nm[69]=b;
        rw1[1][a]=1; cl1[0][a]=1; pd[3][a]++; pb[5][a]++;
        rw7[1][b]=1; cl7[2][b]=1; pd[8][b]++; pb[4][b]++;
        stp17();
        rw1[1][a]=0; cl1[0][a]=0; pd[3][a]--; pb[5][a]--;
        rw7[1][b]=0; cl7[2][b]=0; pd[8][b]--; pb[4][b]--;
      }}
  }
}
//* Set n14 & n68 *
void stp17(){
 short a,b;
```

```
  for(a=0;a<3;a++){b=CC-a;
    if((rw1[1][a]==0)&&(cl1[1][a]==0)&&(pd[4][a]<3)&&(pb[6][a]<3)){
      if((rw7[1][b]==0)&&(cl7[1][b]==0)&&(pd[7][b]<3)&&(pb[3][b]<3)){
        nm[14]=a; nm[68]=b;
        rw1[1][a]=1; cl1[1][a]=1; pd[4][a]++; pb[6][a]++;
        rw7[1][b]=1; cl7[1][b]=1; pd[7][b]++; pb[3][b]++;
        stp18();
        rw1[1][a]=0; cl1[1][a]=0; pd[4][a]--; pb[6][a]--;
        rw7[1][b]=0; cl7[1][b]=0; pd[7][b]--; pb[3][b]--;
      }}
  }
}
//* Set n15 & n67 *
void stp18(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw1[1][a]==0)&&(cl1[2][a]==0)&&(pd[5][a]<3)&&(pb[7][a]<3)){
      if((rw7[1][b]==0)&&(cl7[0][b]==0)&&(pd[6][b]<3)&&(pb[2][b]<3)){
        nm[15]=a; nm[67]=b;
        rw1[1][a]=1; cl1[2][a]=1; pd[5][a]++; pb[7][a]++;
        rw7[1][b]=1; cl7[0][b]=1; pd[6][b]++; pb[2][b]++;
        stp19();
        rw1[1][a]=0; cl1[2][a]=0; pd[5][a]--; pb[7][a]--;
        rw7[1][b]=0; cl7[0][b]=0; pd[6][b]--; pb[2][b]--;
      }}
  }
}
//* Set n16 & n66 *
void stp19(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[1][a]==0)&&(cl2[0][a]==0)&&(pd[6][a]<3)&&(pb[8][a]<3)){
      if((rw6[1][b]==0)&&(cl6[2][b]==0)&&(pd[5][b]<3)&&(pb[1][b]<3)){
        nm[16]=a; nm[66]=b;
        rw2[1][a]=1; cl2[0][a]=1; pd[6][a]++; pb[8][a]++;
        rw6[1][b]=1; cl6[2][b]=1; pd[5][b]++; pb[1][b]++;
        stp20();
        rw2[1][a]=0; cl2[0][a]=0; pd[6][a]--; pb[8][a]--;
        rw6[1][b]=0; cl6[2][b]=0; pd[5][b]--; pb[1][b]--;
      }}
  }
}
//* Set n17 & n65 *
void stp20(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[1][a]==0)&&(cl2[1][a]==0)&&(pd[7][a]<3)&&(pb[9][a]<3)){
      if((rw6[1][b]==0)&&(cl6[1][b]==0)&&(pd[4][b]<3)&&(pb[9][b]<3)){
        nm[17]=a; nm[65]=b;
        rw2[1][a]=1; cl2[1][a]=1; pd[7][a]++; pb[9][a]++;
        rw6[1][b]=1; cl6[1][b]=1; pd[4][b]++; pb[9][b]++;
        stp21();
        rw2[1][a]=0; cl2[1][a]=0; pd[7][a]--; pb[9][a]--;
        rw6[1][b]=0; cl6[1][b]=0; pd[4][b]--; pb[9][b]--;
      }}
  }
}
//* Set n18 & n64 *
void stp21(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[1][a]==0)&&(cl2[2][a]==0)&&(pd[8][a]<3)&&(pb[1][a]<3)){
      if((rw6[1][b]==0)&&(cl6[0][b]==0)&&(pd[3][b]<3)&&(pb[8][b]<3)){
        nm[18]=a; nm[64]=b;
        rw2[1][a]=1; cl2[2][a]=1; pd[8][a]++; pb[1][a]++;
```

```
          rw6[1][b]=1; cl6[0][b]=1; pd[3][b]++; pb[8][b]++;
          stp22();
          rw2[1][a]=0; cl2[2][a]=0; pd[8][a]--; pb[1][a]--;
          rw6[1][b]=0; cl6[0][b]=0; pd[3][b]--; pb[8][b]--;
        }}
    }
}
//* Set n22 & n60 *
void stp22(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw1[2][a]==0)&&(cl1[0][a]==0)&&(pd[2][a]<3)&&(pb[6][a]<3)){
      if((rw7[0][b]==0)&&(cl7[2][b]==0)&&(pd[9][b]<3)&&(pb[3][b]<3)){
        nm[22]=a; nm[60]=b;
        rw1[2][a]=1; cl1[0][a]=1; pd[2][a]++; pb[6][a]++;
        rw7[0][b]=1; cl7[2][b]=1; pd[9][b]++; pb[3][b]++;
        stp23();
        rw1[2][a]=0; cl1[0][a]=0; pd[2][a]--; pb[6][a]--;
        rw7[0][b]=0; cl7[2][b]=0; pd[9][b]--; pb[3][b]--;
      }}
  }
}
//* Set n23 & n59 *
void stp23(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw1[2][a]==0)&&(cl1[1][a]==0)&&(pd[3][a]<3)&&(pb[7][a]<3)){
      if((rw7[0][b]==0)&&(cl7[1][b]==0)&&(pd[8][b]<3)&&(pb[2][b]<3)){
        nm[23]=a; nm[59]=b;
        rw1[2][a]=1; cl1[1][a]=1; pd[3][a]++; pb[7][a]++;
        rw7[0][b]=1; cl7[1][b]=1; pd[8][b]++; pb[2][b]++;
        stp24();
        rw1[2][a]=0; cl1[1][a]=0; pd[3][a]--; pb[7][a]--;
        rw7[0][b]=0; cl7[1][b]=0; pd[8][b]--; pb[2][b]--;
      }}
  }
}
//* Set n24 & n58 *
void stp24(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw1[2][a]==0)&&(cl1[2][a]==0)&&(pd[4][a]<3)&&(pb[8][a]<3)){
      if((rw7[0][b]==0)&&(cl7[0][b]==0)&&(pd[7][b]<3)&&(pb[1][b]<3)){
        nm[24]=a; nm[58]=b;
        rw1[2][a]=1; cl1[2][a]=1; pd[4][a]++; pb[8][a]++;
        rw7[0][b]=1; cl7[0][b]=1; pd[7][b]++; pb[1][b]++;
        stp25();
        rw1[2][a]=0; cl1[2][a]=0; pd[4][a]--; pb[8][a]--;
        rw7[0][b]=0; cl7[0][b]=0; pd[7][b]--; pb[1][b]--;
      }}
  }
}
//* Set n25 & n57 *
void stp25(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[2][a]==0)&&(cl2[0][a]==0)&&(pd[5][a]<3)&&(pb[9][a]<3)){
      if((rw6[0][b]==0)&&(cl6[2][b]==0)&&(pd[6][b]<3)&&(pb[9][b]<3)){
        nm[25]=a; nm[57]=b;
        rw2[2][a]=1; cl2[0][a]=1; pd[5][a]++; pb[9][a]++;
        rw6[0][b]=1; cl6[2][b]=1; pd[6][b]++; pb[9][b]++;
        stp26();
        rw2[2][a]=0; cl2[0][a]=0; pd[5][a]--; pb[9][a]--;
        rw6[0][b]=0; cl6[2][b]=0; pd[6][b]--; pb[9][b]--;
      }}
```

```
  }
}
//* Set n26 & n56 *
void stp26(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[2][a]==0)&&(cl2[1][a]==0)&&(pd[6][a]<3)&&(pb[1][a]<3)){
      if((rw6[0][b]==0)&&(cl6[1][b]==0)&&(pd[5][b]<3)&&(pb[8][b]<3)){
        nm[26]=a; nm[56]=b;
        rw2[2][a]=1; cl2[1][a]=1; pd[6][a]++; pb[1][a]++;
        rw6[0][b]=1; cl6[1][b]=1; pd[5][b]++; pb[8][b]++;
        stp27();
        rw2[2][a]=0; cl2[1][a]=0; pd[6][a]--; pb[1][a]--;
        rw6[0][b]=0; cl6[1][b]=0; pd[5][b]--; pb[8][b]--;
      }}
  }
}
//* Set n27 & n55 *
void stp27(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw2[2][a]==0)&&(cl2[2][a]==0)&&(pd[7][a]<3)&&(pb[2][a]<3)){
      if((rw6[0][b]==0)&&(cl6[0][b]==0)&&(pd[4][b]<3)&&(pb[7][b]<3)){
        nm[27]=a; nm[55]=b;
        rw2[2][a]=1; cl2[2][a]=1; pd[7][a]++; pb[2][a]++;
        rw6[0][b]=1; cl6[0][b]=1; pd[4][b]++; pb[7][b]++;
        stp28();
        rw2[2][a]=0; cl2[2][a]=0; pd[7][a]--; pb[2][a]--;
        rw6[0][b]=0; cl6[0][b]=0; pd[4][b]--; pb[7][b]--;
      }}
  }
}
//
//* ---(Skip)--- *
//
//* Set n40 & n42 *
void stp40(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((rw4[1][a]==0)&&(cl4[0][a]==0)&&(pd[9][a]<3)&&(pb[8][a]<3)){
      if((rw4[1][b]==0)&&(cl4[2][b]==0)&&(pd[2][b]<3)&&(pb[1][b]<3)){
        nm[40]=a; nm[42]=b;
        rw4[1][a]=1; cl4[0][a]=1; pd[9][a]++; pb[8][a]++;
        rw4[1][b]=1; cl4[2][b]=1; pd[2][b]++; pb[1][b]++;
        recordans();
        rw4[1][a]=0; cl4[0][a]=0; pd[9][a]--; pb[8][a]--;
        rw4[1][b]=0; cl4[2][b]=0; pd[2][b]--; pb[1][b]--;
      }}
  }
}
//
//* Record the Latin Squares *
void recordans(){
 short n;
  cnt++;
  tlu[cnt-1][0]=cnt;
  for(n=1;n<82;n++){tlu[cnt-1][n]=nm[n];}
}
//
//* Print the Latin Squares *
void pr9lu(){
 short m,l,l9,n,t;
  for(m=0;m<cnt;m=m+8){
    printf("%10d/%10d/%10d/%10d/%10d/%10d/%10d/%10d/\n",
      tlu[m][0],tlu[m+1][0],tlu[m+2][0],tlu[m+3][0],tlu[m+4][0],tlu[m+5][0],tlu[m+6][0],tlu[m+7][0]);
```

```
    for(l=0;l<9;l++){l9=l*9;
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m+1][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m+2][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m+3][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m+4][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m+5][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m+6][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[m+7][l9+n]);}
      printf("\n");
    }}
  printf(" [Count of Latin Units = %d]\n",cnt);
 //* Measure How Similar each Pair is. *
  for(m=0;m<cnt;m++){
    for(l=0;l<cnt;l++){t=0;
      for(n=1;n<82;n++){
        if(tlu[m][n]==tlu[l][n]){t++;}}
      mtc[m][l]=t;
    }}
 //* Print the Reference Table *
  printf("\n");
  printf(" [Reference Table for the Best Combination]\n");
  printf(" * |");
  for(n=1;n<=cnt;n++){printf("%3d",n);}
  printf("\n--+-----------------------------------------------------------------------------------------------\n");
  for(m=0;m<cnt;m++){
    printf("%3d|",m+1);
    for(n=0;n<cnt;n++){printf("%3d",mtc[m][n]);}
    printf("\n");
  }
}
//
//* Combine abd Compose *
void cmbcmp(short x){
 short n,d,md,fc,lmx;
  md=27; lmx=x;
  for(u1=0;u1<lmx;u1++){//cnt2=0;
    for(u2=0;u2<lmx;u2++){
      if(mtc[u2][u1]==md){cnt2=0;
        for(u3=0;u3<lmx;u3++){
          if((mtc[u3][u1]==md)&&(mtc[u3][u2]==md)){
            for(u4=0;u4<lmx;u4++){
              if((mtc[u4][u1]==md)&&(mtc[u4][u2]==md)&&(mtc[u4][u3]==md)){
                for(n=1;n<82;n++){flg[n]=0;}
                for(n=1;n<82;n++){
                  d=tlu[u1][n]*27+tlu[u2][n]*9+tlu[u3][n]*3+tlu[u4][n]+1;
                  nm[n]=d; flg[d]++;
                }
                fc=0;
                for(n=1;n<82;n++){
                  if(flg[n]==1){fc++;}else{break;}
                }
                if(fc==81){
                  if((nm[1]<nm[9])&&(nm[1]<nm[73])&&(nm[1]<nm[81])){
                    if(nm[2]>nm[10]){pr9ans();}}
                }
              }
```

```
              }
            }
          }
        }
      }
    }
}
//
//* Print 3 Forms *
void pr9ans(){
 short n;
  cnt++; cnt2++;
  if(nm[1]==1){cnt4++;}
  if(cnt2==1){
   if(nm[1]==1){
     cntr[0]=cntr[1]; cntr[1]=cntr[2]; cntr[2]=cnt;
     for(n=1;n<87;n++){anm[n]=bnm[n];}
     for(n=1;n<87;n++){bnm[n]=cnm[n];}
     for(n=1;n<82;n++){cnm[n]=nm[n];}
     cnm[82]=u1+1; cnm[83]=u2+1; cnm[84]=u3+1; cnm[85]=u4+1; cnm[86]=cnt4;
     cnt3++;
     if(cnt3==3){pr3ans(); cnt3=0;}
     //if(cnt3==1){pr1ans(); cnt3=0;}
   }}
}
//
//* Print 3 Answers *
void pr3ans(){
 short l,l9,n;
  printf("%4d/%2d/%2d/%2d/   [%3d/%5d]",
          anm[82],anm[83],anm[84],anm[85],anm[86],cntr[0]);
  printf("%5d/%2d/%2d/%2d/   [%3d/%5d]",
          bnm[82],bnm[83],bnm[84],bnm[85],bnm[86],cntr[1]);
  printf("%5d/%2d/%2d/%2d/   [%3d/%5d]\n",
          cnm[82],cnm[83],cnm[84],cnm[85],cnm[86],cntr[2]);
  for(l=0;l<9;l++){l9=l*9;
    printf(" ");
    for(n=1;n<10;n++){printf("%3d",anm[l9+n]);}
    printf("  ");
    for(n=1;n<10;n++){printf("%3d",bnm[l9+n]);}
    printf("  ");
    for(n=1;n<10;n++){printf("%3d",cnm[l9+n]);}
    printf("\n");
  }
  printf("\n");
}
//
//* Print 2 Answers *
void pr2ans(){
 short l,l9,n;
  printf("%4d/%2d/%2d/%2d/   [%3d/%5d]",
          bnm[82],bnm[83],bnm[84],bnm[85],bnm[86],cntr[1]);
  printf("%5d/%2d/%2d/%2d/   [%3d/%5d]\n",
          cnm[82],cnm[83],cnm[84],cnm[85],cnm[86],cntr[2]);
  for(l=0;l<9;l++){l9=l*9;
    printf(" ");
    for(n=1;n<10;n++){printf("%3d",bnm[l9+n]);}
    printf("  ");
    for(n=1;n<10;n++){printf("%3d",cnm[l9+n]);}
    printf("\n");
  }
  printf("\n");
}
//
//* Print 1 Answer *
```

```
void pr1ans(){
 short l,l9,n;
 short lu1,lu2,lu3,lu4;
  lu1=cnm[82]-1; lu2=cnm[83]-1; lu3=cnm[84]-1; lu4=cnm[85]-1;
  printf("%10d/%10d/%10d/%10d/%22d/%5d\n",
          lu1+1,lu2+1,lu3+1,lu4+1,cnm[86],cntr[2]);
  for(l=0;l<9;l++){l9=l*9;
    printf("  ");
    for(n=1;n<10;n++){printf("%d",tlu[lu1][l9+n]);}
    printf("  ");
    for(n=1;n<10;n++){printf("%d",tlu[lu2][l9+n]);}
    printf("  ");
    for(n=1;n<10;n++){printf("%d",tlu[lu3][l9+n]);}
    printf("  ");
    for(n=1;n<10;n++){printf("%d",tlu[lu4][l9+n]);}
    printf(" ");
    for(n=1;n<10;n++){printf("%3d",cnm[l9+n]);}
    printf("\n");
  }
  printf("\n");
}
//
```

The next list shows the result of my computation.

```
**  Multiple Type of Simultaneous Magic Squares of Order 9:  **
** Self-Complementary & Pan-Diagonal by "New Euler's Method" **

[List of Latin Units]
        1/        2/        3/        4/        5/        6/        7/        8/
  012012012 012012102 012201120 012201210 012012012 012021021 012102210 012120201
  120120120 120120021 120012201 120012102 201201201 201102102 201210021 201012120
  201201201 201201210 201120012 201120021 120120120 120210210 120021102 120201012
  201201201 201021201 012201120 012021120 120120120 102120102 021120210 012120201
  012012012 012210012 120012201 120210201 012012012 210012210 102012021 201012120
  120120120 120102120 201120012 201102012 201201201 021201021 210201102 120201012
  120120120 210120120 012201120 102201120 201201201 210210201 021102201 012120201
  201201201 102201201 120012201 021012201 120120120 021021120 102210120 201012120
  012012012 021012012 201120012 210120012 012012012 102102012 210021012 120201012
        9/       10/       11/       12/       13/       14/       15/       16/
  021012012 021021021 021102210 021120201 102021120 102021210 102102012 102102102
  102201201 102102102 102210021 102012120 021210201 021210102 021021120 021021021
  210120120 210210210 210021102 210201012 210102012 210102021 210210201 210210210
  120102120 102102102 021102210 012102201 102201210 102021210 021201021 021021021
  012210012 210210210 102210021 201210120 021012102 021210102 210012210 210210210
  201021201 021021021 210021102 120021012 210120021 210102021 102120102 102102102
  201201210 210210210 021102210 012120210 012021210 102021210 120210210 210210210
  120120021 021021021 102210021 201012021 120210102 021210102 201102102 102102102
  012012102 102102102 210021102 120201102 201102021 210102021 012021021 021021021
        17/       18/       19/       20/       21/       22/       23/       24/
  120120120 120120210 120201012 120201102 201102021 201120012 201201201 201210210
  201201201 201201102 201012120 201012021 120210102 120012201 120120120 120021021
  012012012 012012021 012120201 012120210 012021210 012201120 012012012 012102102
  201201201 201021201 120201012 120021012 210120021 201120012 120120120 102120102
  012012012 012210012 201012120 201210120 021012102 120012201 012012012 210012210
  120120120 120102120 012120201 012102201 102201210 012201120 201201201 021201021
  012012012 102012012 120201012 210201012 210102012 201120012 012012012 021021012
  120120120 021120120 201012120 102012120 021210201 120012201 201201201 102102201
  201201201 210201201 012120201 021120201 102021120 012201120 120120120 210210120
        25/       26/       27/       28/       29/       30/       31/       32/
  210102021 210120012 210201201 210210210 210021012 210021102 210210120 210210210
  021210102 021012201 021120120 021021021 102210120 102210021 102102201 102102102
  102021210 102201120 102012012 102102102 021102201 021102210 021021012 021021021
```

```
210102021  201102012  120102120  102102102  210201102  210021102  021201021  021021021
021210102  120210201  012210012  210210210  102012021  102210021  210012210  210210210
102021210  012021120  201021201  021021021  021120210  021102210  102120102  102102102
210102021  201120021  012012021  021021021  120021102  210021102  012102102  102102102
021210102  120012102  201201102  102102102  201210021  102210021  120021021  021021021
102021210  012201210  120120210  210210210  012102210  021102210  201210210  210210210
```

[Count of Latin Units = 32]

[List of **Compositions**: Used Units//// Count[n1=1/Total]]

```
        1/         3/         9/         6/                          1/    1
012012012  012201120  021012012  012021021   1 44 78 19 33 71 10 51 62
120120120  120012201  102201201  201102102  42 73  8 35 64 24 53 55 15
201201201  201120012  210120120  120210210  80  6 37 69 26 28 60 17 46
201201201  012201120  120102120  102120102  59 16 48 77  3 43 68 25 30
012012012  120012201  012210012  210012210  12 50 61  7 41 75 21 32 70
120120120  201120012  201021201  021201021  52 57 14 39 79  5 34 66 23
120120120  012201120  201201210  210210201  36 65 22 54 56 13 45 76  2
201201201  120012201  120120021  021021120  67 27 29 58 18 47 74  9 40
012012012  201120012  012012102  102102012  20 31 72 11 49 63  4 38 81
        1/         3/        10/         5/                          3/    9
012012012  012201120  021021021  012012012   1 44 78 19 35 69 10 53 60
120120120  120012201  102102102  201201201  42 73  8 33 64 26 51 55 17
201201201  201120012  210210210  120120120  80  6 37 71 24 28 62 15 46
201201201  012201120  102102102  120120120  59 12 52 77  3 43 68 21 34
012012012  120012201  210210210  012012012  16 50 57  7 41 75 25 32 66
120120120  201120012  021021021  201201201  48 61 14 39 79  5 30 70 23
120120120  012201120  210210210  201201201  36 67 20 54 58 11 45 76  2
201201201  120012201  021021021  120120120  65 27 31 56 18 49 74  9 40
012012012  201120012  102102102  012012012  22 29 72 13 47 63  4 38 81
        1/         3/        11/         5/                          5/   17
012012012  012201120  021102210  012012012   1 44 78 22 29 72 16 50 57
120120120  120012201  102210021  201201201  42 73  8 36 67 20 48 61 14
201201201  201120012  210021102  120120120  80  6 37 65 27 31 59 12 52
201201201  012201120  021102210  120120120  56 18 49 77  3 43 71 24 28
012012012  120012201  102210021  012012012  13 47 63  7 41 75 19 35 69
120120120  201120012  210021102  201201201  54 58 11 39 79  5 33 64 26
120120120  012201120  021102210  201201201  30 70 23 51 55 17 45 76  2
201201201  120012201  102210021  120120120  68 21 34 62 15 46 74  9 40
012012012  201120012  210021102  012012012  25 32 66 10 53 60  4 38 81
```

```
 1/ 3/12/ 6/   [  7/   25]    1/ 9/ 3/ 6/   [  9/   97]    1/ 9/ 6/ 3/   [ 11/  105]
 1 44 78 22 36 65 16 48 59     1 50 72  7 39 77  4 45 74     1 50 72  3 43 77  2 45 76
42 73  8 29 67 27 50 61 12    42 61 20 47 58 18 53 55 15    44 57 22 49 56 18 51 55 17
80  6 37 72 20 31 57 14 52    80 12 31 69 26 28 66 23 34    78 16 29 71 24 28 70 23 30
56 13 54 77  3 43 71 19 33    65 22 36 71  3 49 68 25 30    67 20 36 69  7 47 68 21 34
18 47 58  7 41 75 24 35 64     6 44 73 19 41 63  9 38 76     8 42 73 19 41 63  9 40 74
49 63 11 39 79  5 28 69 26    52 57 14 33 79 11 46 60 17    48 61 14 35 75 13 46 62 15
30 68 25 51 62 10 45 76  2    48 59 16 54 56 13 51 70  2    52 59 12 54 58 11 53 66  4
70 21 32 55 15 53 74  9 40    67 27 29 64 24 35 62 21 40    65 27 31 64 26 33 60 25 38
23 34 66 17 46 60  4 38 81     8 37 78  5 43 75 10 32 81     6 37 80  5 39 79 10 32 81

 1/ 9/ 7/ 3/   [ 14/  113]    1/ 9/12/ 6/   [ 17/  121]    1/10/ 3/ 5/   [ 19/  145]
 1 50 72  6 37 80  8 42 73     1 53 69  4 45 74  7 39 77     1 50 72  7 47 69  4 53 66
44 57 22 52 59 12 48 61 14    42 55 26 47 58 18 50 61 12    42 61 20 39 58 26 45 55 23
78 16 29 65 27 31 67 20 36    80 15 28 72 20 31 66 23 34    80 12 31 77 18 28 74 15 34
64 26 33 69  7 47 71 24 28    65 22 36 68  3 52 71 19 33    65  6 52 71  3 49 68  9 46
 5 39 79 19 41 63  3 43 77     9 38 76 25 41 57  6 44 73    22 44 57 19 41 63 25 38 60
54 58 11 35 75 13 49 56 18    49 63 11 30 79 14 46 60 17    36 73 14 33 79 11 30 76 17
46 62 15 51 55 17 53 66  4    48 59 16 51 62 10 54 67  2    48 67  8 54 64  5 51 70  2
68 21 34 70 23 30 60 25 38    70 21 32 64 24 35 56 27 40    59 27 37 56 24 43 62 21 40
 9 40 74  2 45 76 10 32 81     5 43 75  8 37 78 13 29 81    16 29 78 13 35 75 10 32 81
```

```
 1/10/ 5/ 3/   [ 21/  153]
 1 50 72  3 49 71  2 51 70
44 57 22 43 56 24 45 55 23
78 16 29 77 18 28 76 17 30
67  8 48 69  7 47 68  9 46
20 42 61 19 41 63 21 40 62
36 73 14 35 75 13 34 74 15
52 65  6 54 64  5 53 66  4
59 27 37 58 26 39 60 25 38
12 31 80 11 33 79 10 32 81

 1/10/ 8/ 3/   [ 24/  161]
 1 50 72  6 52 65  8 48 67
44 57 22 37 59 27 42 61 20
78 16 29 80 12 31 73 14 36
64  5 54 69  7 47 71  3 49
26 39 58 19 41 63 24 43 56
33 79 11 35 75 13 28 77 18
46 68  9 51 70  2 53 66  4
62 21 40 55 23 45 60 25 38
15 34 74 17 30 76 10 32 81

 1/10/11/ 5/   [ 27/  169]
 1 53 69  4 47 72  7 50 66
42 55 26 45 58 20 39 61 23
80 15 28 74 18 31 77 12 34
65  9 49 68  3 52 71  6 46
22 38 63 25 41 57 19 44 60
36 76 11 30 79 14 33 73 17
48 70  5 51 64  8 54 67  2
59 21 43 62 24 37 56 27 40
16 32 75 10 35 78 13 29 81

 1/11/ 3/ 5/   [ 29/  241]
 1 50 72 16 29 78 22 44 57
42 61 20 48 67  8 36 73 14
80 12 31 59 27 37 65  6 52
56 24 43 71  3 49 77 18 28
13 35 75 19 41 63  7 47 69
54 64  5 33 79 11 39 58 26
30 76 17 45 55 23 51 70  2
68  9 46 74 15 34 62 21 40
25 38 60  4 53 66 10 32 81

 1/12/ 3/ 6/   [ 39/  345]
 1 50 72 16 48 59 22 36 65
42 61 20 29 67 27 44 73  6
80 12 31 78  8 37 57 14 52
56 13 54 71  3 49 77  7 39
24 35 64 19 41 63 18 47 58
43 75  5 33 79 11 28 69 26
30 68 25 45 74  4 51 70  2
76  9 38 55 15 53 62 21 40
17 46 60 23 34 66 10 32 81

 2/ 4/ 9/ 6/   [ 49/  697]
 1 44 78 19 33 71 46 15 62
42 73  8 35 64 24 17 55 51
80  6 37 69 26 28 60 53 10
59 16 48  5 75 43 68 25 30
12 50 61 79 41  3 21 32 70
52 57 14 39  7 77 34 66 23
72 29 22 54 56 13 45 76  2
31 27 65 58 18 47 74  9 40
20 67 36 11 49 63  4 38 81

 2/ 9/ 4/ 6/   [ 57/  793]
 1 50 72  7 39 77 34 15 74
42 61 20 47 58 18 23 55 45
80 12 31 69 26 28 66 53  4
65 22 36 11 63 49 68 25 30
 6 44 73 79 41  3  9 38 76
52 57 14 33 19 71 46 60 17
78 29 16 54 56 13 51 70  2
37 27 59 64 24 35 62 21 40
 8 67 48  5 43 75 10 32 81

 2/10/ 4/ 5/   [ 67/  841]
 1 50 72  7 47 69 34 23 66
42 61 20 39 58 26 15 55 53
80 12 31 77 18 28 74 45  4
65  6 52 11 63 49 68  9 46
22 44 57 79 41  3 25 38 60
36 73 14 33 19 71 30 76 17
78 37  8 54 64  5 51 70  2
29 27 67 56 24 43 62 21 40
16 59 48 13 35 75 10 32 81

 2/11/ 4/ 5/   [ 77/  937]
 1 50 72 16 29 78 52 14 57
42 61 20 48 67  8  6 73 44
80 12 31 59 27 37 65 36 22
56 24 43 11 63 49 77 18 28
13 35 75 79 41  3  7 47 69
54 64  5 33 19 71 39 58 26
60 46 17 45 55 23 51 70  2
38  9 76 74 15 34 62 21 40
25 68 30  4 53 66 10 32 81

 2/12/ 4/ 6/   [ 87/ 1041]
 1 50 72 16 48 59 52  6 65
42 61 20 29 67 27 14 73 36
80 12 31 78  8 37 57 44 22
56 13 54 11 63 49 77  7 39
24 35 64 79 41  3 18 47 58
43 75  5 33 19 71 28 69 26
60 38 25 45 74  4 51 70  2
46  9 68 55 15 53 62 21 40
17 76 30 23 34 66 10 32 81

 3/ 1/ 9/ 6/   [ 97/ 1393]
 1 44 78 55 15 53 28 69 26
42 73  8 17 46 60 71 19 33
80  6 37 51 62 10 24 35 64
23 34 66 77  3 43 50 61 12
30 68 25  7 41 75 57 14 52
70 21 32 39 79  5 16 48 59
18 47 58 72 20 31 45 76  2
49 63 11 22 36 65 74  9 40
56 13 54 29 67 27  4 38 81

 3/ 9/ 1/ 6/   [105/ 1489]
 1 50 72 55 15 53 28 69 26
42 61 20 23 34 66 77  7 39
80 12 31 45 74  4 18 47 58
17 46 60 71  3 49 44 73  6
30 68 25 19 41 63 57 14 52
76  9 38 33 79 11 22 36 65
24 35 64 78  8 37 51 70  2
43 75  5 16 48 59 62 21 40
56 13 54 29 67 27 10 32 81

 3/10/ 1/ 5/   [115/ 1593]
 1 50 72 55 23 45 28 77 18
42 61 20 15 34 74 69  7 47
80 12 31 53 66  4 26 39 58
17 30 76 71  3 49 44 57 22
46 68  9 19 41 63 73 14 36
60 25 38 33 79 11  6 52 65
24 43 56 78 16 29 51 70  2
35 75 13  8 48 67 62 21 40
64  5 54 37 59 27 10 32 81

 3/11/ 1/ 5/   [125/ 1697]
 1 50 72 64  5 54 46 68  9
42 61 20 24 43 56 60 25 38
80 12 31 35 75 13 17 30 76
 8 48 67 71  3 49 53 66  4
37 59 27 19 41 63 55 23 45
78 16 29 33 79 11 15 34 74
 6 52 65 69  7 47 51 70  2
44 57 22 26 39 58 62 21 40
73 14 36 28 77 18 10 32 81

 3/12/ 1/ 6/   [135/ 1745]
 1 50 72 64 24 35 46 60 17
42 61 20  5 43 75 68 25 30
80 12 31 54 56 13  9 38 76
 8 37 78 71  3 49 53 55 15
48 59 16 19 41 63 66 23 34
67 27 29 33 79 11  4 45 74
 6 44 73 69 26 28 51 70  2
52 57 14  7 39 77 62 21 40
65 22 36 47 58 18 10 32 81

 4/ 2/ 9/ 6/   [145/ 2089]
 1 44 78 55 15 53 64 33 26
42 73  8 17 46 60 35 19 69
80  6 37 51 62 10 24 71 28
23 34 66  5 75 43 50 61 12
30 68 25 79 41  3 57 14 52
70 21 32 39  7 77 16 48 59
54 11 58 72 20 31 45 76  2
13 63 47 22 36 65 74  9 40
56 49 18 29 67 27  4 38 81

 9/ 1/ 3/ 6/   [193/ 2785]
 1 68 54  7 39 77  4 45 74
42 25 56 65 22 36 71 19 33
80 30 13 51 62 10 48 59 16
47 58 18 53  3 67 50 61 12
 6 44 73 55 41 27  9 38 76
70 21 32 15 79 29 64 24 35
66 23 34 72 20 31 69 52  2
49 63 11 46 60 17 26 57 40
 8 37 78  5 43 75 28 14 81

10/ 1/ 3/ 5/   [289/ 4177]
 1 68 54  7 65 51  4 71 48
42 25 56 39 22 62 45 19 59
80 30 13 77 36 10 74 33 16
47  6 70 53  3 67 50  9 64
58 44 21 55 41 27 61 38 24
18 73 32 15 79 29 12 76 35
66 49  8 72 46  5 69 52  2
23 63 37 20 60 43 26 57 40
34 11 78 31 17 75 28 14 81
```

```
 11/ 1/ 3/ 5/   [385/ 6961]   12/ 1/ 3/ 6/   [481/ 8353]
  1 68 54 34 11 78 58 44 21    1 68 54 34 66 23 58 18 47
 42 25 56 66 49  8 18 73 32   42 25 56 11 49 63 44 73  6
 80 30 13 23 63 37 47  6 70   80 30 13 78  8 37 21 32 70
 20 60 43 53  3 67 77 36 10   20 31 72 53  3 67 77  7 39
 31 17 75 55 41 27  7 65 51   60 17 46 55 41 27 36 65 22
 72 46  5 15 79 29 39 22 62   43 75  5 15 79 29 10 51 62
 12 76 35 45 19 59 69 52  2   12 50 61 45 74  4 69 52  2
 50  9 64 74 33 16 26 57 40   76  9 38 19 33 71 26 57 40
 61 38 24  4 71 48 28 14 81   35 64 24 59 16 48 28 14 81
[Count of Compositions(n1=1/Total) = 576/11136]

[Counts according to the Value of n1]
  1/576,  2/816,  3/576,  4/768,  5/352,  6/768,  7/496,  8/720,  9/496,
 10/544, 11/256, 12/544, 13/256, 14/  0, 15/256, 16/544, 17/256, 18/544,
 19/272, 20/368, 21/272, 22/320, 23/160, 24/320, 25/192, 26/272, 27/192,
 28/  0, 29/  0, 30/  0, 31/  0, 32/  0, 33/  0, 34/  0, 35/  0, 36/  0;
```

It is amazing that there are so many Multiple and Simultaneous type of Complete Euler Squares 9x9 as 11136. But these are really rare and precious jewels.

There are many 'non-Euler type' in reality, far more than 'Euler Squares'.

5. Can we make 4-Dimensional Extra-Cubic Objects of Order 3 by this Method?

I remember I once proposed you to compose 4-Dimensional Extra-Cubic Objects of order 3 and to get the Magic Squares of order 9 by dimension-converting from 4 down to 2. I found this transformation always makes 'Euler Square' by PNS of Base 3.

Now I want to make that object by our New Euler's Method using PNS of Base 3.

I expect I can count the solutions up to the last and know the total number of them.

Let's make Self-Complementary type of Magic Squares 9x9 finally, since I know we cannot compose any Simultaneous Euler Squares 9x9 of that type.

Let's prepare several basic Developed Figures and simultaneous equations defining what lines must add up to the same sum, the Magic Constant.

Every position of (n1, n2, n3, n4, ... , n80, n81) is necessarily defined by four lines which directly express four dimensions.

```
** Basic View-Forms for our Extra-Cubic Object 3^4 **

 [1]
   1-----2-----3       4-----5-----6       7-----8-----9
   |10     11    |12    |13     14    |15    |16     17    |18
  28  19-29--20-30--21  31  22-32--23-33--24  34  25-35--26-36--27
   |37 |   38    |39 |   |40 |   41    |42 |   |43 |   44    |45 |
  55--46-56--47-57  48  58--49-59--50-60  51  61--52-62--53-63  54
    64 |   65     66 |    67 |   68     69 |    70 |   71     72 |
      73-----74-----75      76-----77-----78      79-----80-----81

 [2]
   1-----2-----3       28----29-----30      55-----56-----57
   | 4       5     | 6   |31     32    |33    |58     59    |60
  10   7-11-- 8-12-- 9  37  34-38--35-39--36  64  61-65--62-66--63
   |13 |   14    |15 |   |40 |   41    |42 |   |67 |   68    |69 |
  19--16-20--17-21  18  46--43-47--44-48  45  73--70-74--71-75  72
    22 |   23     24 |    49 |   50     51 |    76 |   77     78 |
      25-----26-----27      52-----53-----54      79-----80-----81
```

[3] [Basic Positions]

1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27
28	29	30	31	32	33	34	35	36
37	38	39	40	41	42	43	44	45
46	47	48	49	50	51	52	53	54
55	56	57	58	59	60	61	62	63
64	65	66	67	68	69	70	71	72
73	74	75	76	77	78	79	80	81

[4] [Another Form]

1	2	3	10	11	12	19	20	21
4	5	6	13	14	15	22	23	24
7	8	9	16	17	18	25	26	27
28	29	30	37	38	39	46	47	48
31	32	33	40	41	42	49	50	51
34	35	36	43	44	45	52	53	54
55	56	57	64	65	66	73	74	75
58	59	60	67	68	69	76	77	78
61	62	63	70	71	72	79	80	81

** Conditions for 4-Dimensional Extra-Cubes of Order 3 **

n1+n2+n3=C...d101;	n1+n4+n7=C...d201;	n1+n10+n19=C...d301;	n1+n28+n55=C...d401;
n4+n5+n6=C...d102;	n2+n5+n8=C...d202;	n2+n11+n20=C...d302;	n2+n29+n56=C...d402;
n7+n8+n9=C...d103;	n3+n6+n9=C...d203;	n3+n12+n21=C...d303;	n3+n30+n57=C...d403;
n10+n11+n12=C...d104;	n10+n13+n16=C...d204;	n4+n13+n22=C...d304;	n4+n31+n58=C...d404;
n13+n14+n15=C...d105;	n11+n14+n17=C...d205;	n5+n14+n23=C...d305;	n5+n32+n59=C...d405;
n16+n17+n18=C...d106;	n12+n15+n18=C...d206;	n6+n15+n24=C...d306;	n6+n33+n60=C...d406;
n19+n20+n21=C...d107;	n19+n22+n25=C...d207;	n7+n16+n25=C...d307;	n7+n34+n61=C...d407;
n22+n23+n24=C...d108;	n20+n23+n26=C...d208;	n8+n17+n26=C...d308;	n8+n35+n62=C...d408;
n25+n26+n27=C...d109;	n21+n24+n27=C...d209;	n9+n18+n27=C...d309;	n9+n36+n63=C...d409;
n28+n29+n30=C...d110;	n28+n31+n34=C...d210;	n28+n37+n46=C...d310;	n10+n37+n64=C...d410;
n31+n32+n33=C...d111;	n29+n32+n35=C...d211;	n29+n38+n47=C...d311;	n11+n38+n65=C...d411;
n34+n35+n36=C...d112;	n30+n33+n36=C...d212;	n30+n39+n48=C...d312;	n12+n39+n66=C...d412;
n37+n38+n39=C...d113;	n37+n40+n43=C...d213;	n31+n40+n49=C...d313;	n13+n40+n67=C...d413;
n40+n41+n42=C...d114;	n38+n41+n44=C...d214;	n32+n41+n50=C...d314;	n14+n41+n68=C...d414;
n43+n44+n45=C...d115;	n39+n42+n45=C...d215;	n33+n42+n51=C...d315;	n15+n42+n69=C...d415;
n46+n47+n48=C...d116;	n46+n49+n52=C...d216;	n34+n43+n52=C...d316;	n16+n43+n70=C...d416;
n49+n50+n51=C...d117;	n47+n50+n53=C...d217;	n35+n44+n53=C...d317;	n17+n44+n71=C...d417;
n52+n53+n54=C...d118;	n48+n51+n54=C...d218;	n36+n45+n54=C...d318;	n18+n45+n72=C...d418;
n55+n56+n57=C...d119;	n55+n58+n61=C...d219;	n55+n64+n73=C...d319;	n19+n46+n73=C...d419;
n58+n59+n60=C...d120;	n56+n59+n62=C...d220;	n56+n65+n74=C...d320;	n20+n47+n74=C...d420;
n61+n62+n63=C...d121;	n57+n60+n63=C...d221;	n57+n66+n75=C...d321;	n21+n48+n75=C...d421;
n64+n65+n66=C...d122;	n64+n67+n70=C...d222;	n58+n67+n76=C...d322;	n22+n49+n76=C...d422;
n67+n68+n69=C...d123;	n65+n68+n71=C...d223;	n59+n68+n77=C...d323;	n23+n50+n77=C...d423;
n70+n71+n72=C...d124;	n66+n69+n72=C...d224;	n60+n69+n78=C...d324;	n24+n51+n78=C...d424;
n73+n74+n75=C...d125;	n73+n76+n79=C...d225;	n61+n70+n79=C...d325;	n25+n52+n79=C...d425;
n76+n77+n78=C...d126;	n74+n77+n80=C...d226;	n62+n71+n80=C...d326;	n26+n53+n80=C...d426;
n79+n80+n81=C...d127;	n75+n78+n81=C...d227;	n63+n72+n81=C...d327;	n27+n54+n81=C...d427;

** Self-Complementary Conditions: **

n1+n81=CC;	n2+n80=CC;	n3+n79=CC;	n4+n78=CC;
n5+n77=CC;	n6+n76=CC;	n7+n75=CC;	n8+n74=CC;
n9+n73=CC;	n10+n72=CC;	n11+n71=CC;	n12+n70=CC;
n13+n69=CC;	n14+n68=CC;	n15+n67=CC;	n16+n66=CC;
n17+n65=CC;	n18+n64=CC;	n19+n63=CC;	n20+n62=CC;
n21+n61=CC;	n22+n60=CC;	n23+n59=CC;	n24+n58=CC;
n25+n57=CC;	n26+n56=CC;	n27+n55=CC;	n28+n54=CC;
n29+n53=CC;	n30+n52=CC;	n31+n51=CC;	n32+n50=CC;
n33+n49=CC;	n34+n48=CC;	n35+n47=CC;	n36+n46=CC;
n37+n45=CC;	n38+n44=CC;	n39+n43=CC;	n40+n42=CC;

```
  n41+n41=CC;  n42+n40=CC;  ......;
```

On top of that the next two primary diagonals have to be specially defined.

```
  n1+n11+n21+n31+n41+n51+n61+n71+n81=LSM ...pd1;
  n9+n17+n25+n33+n41+n49+n57+n65+n73=LSM ...pb9;
```

Equal sums are already realized by the Complementary Pairs, but the decomposed patterns of Base 3 are still indefinite because two answers are possible:

```
  {0*3 + 1*3 + 2*3} or {1+1+1+1+1+1+1+1+1}
```

If you want to avoid the latter pattern, you should prepare the flag for each primary diagonal in advance to know how often each '0', '1' and '2' is used.

The values of Magic Constants are: `C=3; CC=2;` and `LSM=9;` here for Latin Units.

The next list gives you my recent program for this object.

```
//** Complete Euler Squares of Order 9 Composed by the 'New Euler's Method' **
//**  Down-converted from the 4-Dimensional Extra-Cubic Objects of Order 3: **
//**   Multiple 3x3 Type of Self-complementary Magic Squares of Order 9  **
//** File: 'Euler9CQ3C.c' Dictated by Kanji Setsuda **
//**  on Sep. 6, 2003; May 15, 2006; Recompiled on **
//**  Mar.10, 2015 with MacOSX 10.10.2 & Xcode 6.2 **
//
#include <stdio.h>
//
short int cnt, cnt1, cnt3;
short LSM, CC;
short u1, u2, u3, u4;
short tlu[9][82];
short mtc[9][9];
short nm[82], uflg[82];
short tnm[49][86];
//
short d1[28][3], d2[28][3], d3[28][3], d4[28][3];
short pd1[3], pb9[3];
//
//* Sub-Routines *
void stp01(void), stp02(void), stp03(void), stp04(void);
void stp05(void), stp06(void), stp07(void), stp08(void);
void stp09(void), stp10(void), stp11(void), stp12(void);
void stp13(void), stp14(void), 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), stp26(void), stp27(void), stp28(void);
void stp29(void), stp30(void), stp31(void), stp32(void);
void stp33(void), stp34(void), stp35(void), stp36(void);
void stp37(void), stp38(void), stp39(void), stp40(void);
void recordans(void), pr9lu(void), cmbcmp(void), cmprecord(void);
void srtans(void), exchng(short x);
void pr3ans(void), pr2ans(void), pr1ans(void);
//
//* Main Program *
int main(){
 short m,n;
  printf("\n");
  printf("** Complete Euler Squares of Order 9 Composed by the 'New Euler's Method' **\n");
  printf("**  Down-converted from the 4-Dimensional Extra-Cubic Objects of Order 3: **\n");
  printf("**   Multiple 3x3 Type of Self-complementary Magic Squares of Order 9   **\n");
  for(n=0;n<82;n++){nm[n]=-1; uflg[n]=0;}
  for(m=0;m<28;m++){
    for(n=0;n<3;n++){
      d1[m][n]=0; d2[m][n]=0; d3[m][n]=0; d4[m][n]=0;
    }}
  for(n=0;n<3;n++){pd1[n]=0; pb9[n]=0;}
```

```
  LSM=9; CC=2; cnt=0;
  nm[41]=1; pd1[1]=1; pb9[1]=1;
  d1[14][1]=1; d2[14][1]=1; d3[14][1]=1; d4[14][1]=1;
  printf("\n");
  printf(" [List of Latin Units]\n");
  stp01();
  pr9lu();
  printf("\n");
  printf("** List of the Standard Solutions: Used Units//// Sol_Number# **\n");
  cnt=0; cnt1=0; cnt3=0;
  cmbcmp();
  srtans();
  //pr1ans();
  pr3ans();
  printf("\n");
  printf(" [Solution Counts(n1=1/Total) = %d/%d]  OK!\n",cnt1,cnt);
  printf("\n");
  printf("** Recompiled and Listed by Kanji Setsuda on\n");
  printf("    Mar.10, 2015 with MacOSX 10.10.2 & Xcode 6.2 **\n");
  printf("\n");
 return 0;
}
//
//* Making Latin Units *
//* Set n1 & n81 *
void stp01(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[1][a]==0)&&(d2[1][a]==0)&&(d3[1][a]==0)&&(d4[1][a]==0)){
      if((d1[27][b]==0)&&(d2[27][b]==0)&&(d3[27][b]==0)&&(d4[27][b]==0)){
       if((pd1[a]<3)&&(pd1[b]<3)){
         nm[1]=a; nm[81]=b;
         d1[1][a]=1; d2[1][a]=1; d3[1][a]=1; d4[1][a]=1; pd1[a]++;
         d1[27][b]=1; d2[27][b]=1; d3[27][b]=1; d4[27][b]=1; pd1[b]++;
         stp02();
         d1[1][a]=0; d2[1][a]=0; d3[1][a]=0; d4[1][a]=0; pd1[a]--;
         d1[27][b]=0; d2[27][b]=0; d3[27][b]=0; d4[27][b]=0; pd1[b]--;
       }}}
  }
}
/* Set n2 & n80 */
void stp02(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[1][a]==0)&&(d2[2][a]==0)&&(d3[2][a]==0)&&(d4[2][a]==0)){
      if((d1[27][b]==0)&&(d2[26][b]==0)&&(d3[26][b]==0)&&(d4[26][b]==0)){
        nm[2]=a; nm[80]=b;
        d1[1][a]=1; d2[2][a]=1; d3[2][a]=1; d4[2][a]=1;
        d1[27][b]=1; d2[26][b]=1; d3[26][b]=1; d4[26][b]=1;
        stp03();
        d1[1][a]=0; d2[2][a]=0; d3[2][a]=0; d4[2][a]=0;
        d1[27][b]=0; d2[26][b]=0; d3[26][b]=0; d4[26][b]=0;
      }}
  }
}
/* Set n3 & n79 */
void stp03(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[1][a]==0)&&(d2[3][a]==0)&&(d3[3][a]==0)&&(d4[3][a]==0)){
      if((d1[27][b]==0)&&(d2[25][b]==0)&&(d3[25][b]==0)&&(d4[25][b]==0)){
        nm[3]=a; nm[79]=b;
        d1[1][a]=1; d2[3][a]=1; d3[3][a]=1; d4[3][a]=1;
        d1[27][b]=1; d2[25][b]=1; d3[25][b]=1; d4[25][b]=1;
        stp04();
```

```
        d1[1][a]=0; d2[3][a]=0; d3[3][a]=0; d4[3][a]=0;
        d1[27][b]=0; d2[25][b]=0; d3[25][b]=0; d4[25][b]=0;
      }}
  }
}
/* Set n10 & n72 */
void stp04(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[4][a]==0)&&(d2[4][a]==0)&&(d3[1][a]==0)&&(d4[10][a]==0)){
      if((d1[24][b]==0)&&(d2[24][b]==0)&&(d3[27][b]==0)&&(d4[18][b]==0)){
        nm[10]=a; nm[72]=b;
        d1[4][a]=1; d2[4][a]=1; d3[1][a]=1; d4[10][a]=1;
        d1[24][b]=1; d2[24][b]=1; d3[27][b]=1; d4[18][b]=1;
        stp05();
        d1[4][a]=0; d2[4][a]=0; d3[1][a]=0; d4[10][a]=0;
        d1[24][b]=0; d2[24][b]=0; d3[27][b]=0; d4[18][b]=0;
      }}
  }
}
/* Set n19 & n63 */
void stp05(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[7][a]==0)&&(d2[7][a]==0)&&(d3[1][a]==0)&&(d4[19][a]==0)){
      if((d1[21][b]==0)&&(d2[21][b]==0)&&(d3[27][b]==0)&&(d4[9][b]==0)){
        nm[19]=a; nm[63]=b;
        d1[7][a]=1; d2[7][a]=1; d3[1][a]=1; d4[19][a]=1;
        d1[21][b]=1; d2[21][b]=1; d3[27][b]=1; d4[9][b]=1;
        stp06();
        d1[7][a]=0; d2[7][a]=0; d3[1][a]=0; d4[19][a]=0;
        d1[21][b]=0; d2[21][b]=0; d3[27][b]=0; d4[9][b]=0;
      }}
  }
}
/* Set n11 & n71 */
void stp06(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[4][a]==0)&&(d2[5][a]==0)&&(d3[2][a]==0)&&(d4[11][a]==0)){
      if((d1[24][b]==0)&&(d2[23][b]==0)&&(d3[26][b]==0)&&(d4[17][b]==0)){
       if((pd1[a]<3)&&(pd1[b]<3)){
         nm[11]=a; nm[71]=b;
         d1[4][a]=1; d2[5][a]=1; d3[2][a]=1; d4[11][a]=1; pd1[a]++;
         d1[24][b]=1; d2[23][b]=1; d3[26][b]=1; d4[17][b]=1; pd1[b]++;
         stp07();
         d1[4][a]=0; d2[5][a]=0; d3[2][a]=0; d4[11][a]=0; pd1[a]--;
         d1[24][b]=0; d2[23][b]=0; d3[26][b]=0; d4[17][b]=0; pd1[b]--;
       }}}
  }
}
/* Set n12 & n70 */
void stp07(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[4][a]==0)&&(d2[6][a]==0)&&(d3[3][a]==0)&&(d4[12][a]==0)){
      if((d1[24][b]==0)&&(d2[22][b]==0)&&(d3[25][b]==0)&&(d4[16][b]==0)){
        nm[12]=a; nm[70]=b;
        d1[4][a]=1; d2[6][a]=1; d3[3][a]=1; d4[12][a]=1;
        d1[24][b]=1; d2[22][b]=1; d3[25][b]=1; d4[16][b]=1;
        stp08();
        d1[4][a]=0; d2[6][a]=0; d3[3][a]=0; d4[12][a]=0;
        d1[24][b]=0; d2[22][b]=0; d3[25][b]=0; d4[16][b]=0;
      }}
  }
```

```
}
/* Set n20 & n62 */
void stp08(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[7][a]==0)&&(d2[8][a]==0)&&(d3[2][a]==0)&&(d4[20][a]==0)){
      if((d1[21][b]==0)&&(d2[20][b]==0)&&(d3[26][b]==0)&&(d4[8][b]==0)){
        nm[20]=a; nm[62]=b;
        d1[7][a]=1; d2[8][a]=1; d3[2][a]=1; d4[20][a]=1;
        d1[21][b]=1; d2[20][b]=1; d3[26][b]=1; d4[8][b]=1;
        stp09();
        d1[7][a]=0; d2[8][a]=0; d3[2][a]=0; d4[20][a]=0;
        d1[21][b]=0; d2[20][b]=0; d3[26][b]=0; d4[8][b]=0;
      }}
  }
}
/* Set n21 & n61 */
void stp09(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[7][a]==0)&&(d2[9][a]==0)&&(d3[3][a]==0)&&(d4[21][a]==0)){
      if((d1[21][b]==0)&&(d2[19][b]==0)&&(d3[25][b]==0)&&(d4[7][b]==0)){
       if((pd1[a]<3)&&(pd1[b]<3)){
         nm[21]=a; nm[61]=b;
         d1[7][a]=1; d2[9][a]=1; d3[3][a]=1; d4[21][a]=1; pd1[a]++;
         d1[21][b]=1; d2[19][b]=1; d3[25][b]=1; d4[7][b]=1; pd1[b]++;
         stp10();
         d1[7][a]=0; d2[9][a]=0; d3[3][a]=0; d4[21][a]=0; pd1[a]--;
         d1[21][b]=0; d2[19][b]=0; d3[25][b]=0; d4[7][b]=0; pd1[b]--;
       }}}
  }
}
//* Level #2: *
//* Set n4 & n78 *
void stp10(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[2][a]==0)&&(d2[1][a]==0)&&(d3[4][a]==0)&&(d4[4][a]==0)){
      if((d1[26][b]==0)&&(d2[27][b]==0)&&(d3[24][b]==0)&&(d4[24][b]==0)){
        nm[4]=a; nm[78]=b;
        d1[2][a]=1; d2[1][a]=1; d3[4][a]=1; d4[4][a]=1;
        d1[26][b]=1; d2[27][b]=1; d3[24][b]=1; d4[24][b]=1;
        stp11();
        d1[2][a]=0; d2[1][a]=0; d3[4][a]=0; d4[4][a]=0;
        d1[26][b]=0; d2[27][b]=0; d3[24][b]=0; d4[24][b]=0;
      }}
  }
}
/* Set n5 & n77 */
void stp11(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[2][a]==0)&&(d2[2][a]==0)&&(d3[5][a]==0)&&(d4[5][a]==0)){
      if((d1[26][b]==0)&&(d2[26][b]==0)&&(d3[23][b]==0)&&(d4[23][b]==0)){
        nm[5]=a; nm[77]=b;
        d1[2][a]=1; d2[2][a]=1; d3[5][a]=1; d4[5][a]=1;
        d1[26][b]=1; d2[26][b]=1; d3[23][b]=1; d4[23][b]=1;
        stp12();
        d1[2][a]=0; d2[2][a]=0; d3[5][a]=0; d4[5][a]=0;
        d1[26][b]=0; d2[26][b]=0; d3[23][b]=0; d4[23][b]=0;
      }}
  }
}
/* Set n6 & n76 */
void stp12(){
```

```
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[2][a]==0)&&(d2[3][a]==0)&&(d3[6][a]==0)&&(d4[6][a]==0)){
      if((d1[26][b]==0)&&(d2[25][b]==0)&&(d3[22][b]==0)&&(d4[22][b]==0)){
        nm[6]=a; nm[76]=b;
        d1[2][a]=1; d2[3][a]=1; d3[6][a]=1; d4[6][a]=1;
        d1[26][b]=1; d2[25][b]=1; d3[22][b]=1; d4[22][b]=1;
        stp13();
        d1[2][a]=0; d2[3][a]=0; d3[6][a]=0; d4[6][a]=0;
        d1[26][b]=0; d2[25][b]=0; d3[22][b]=0; d4[22][b]=0;
      }}
  }
}
/* Set n7 & n75 */
void stp13(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[3][a]==0)&&(d2[1][a]==0)&&(d3[7][a]==0)&&(d4[7][a]==0)){
      if((d1[25][b]==0)&&(d2[27][b]==0)&&(d3[21][b]==0)&&(d4[21][b]==0)){
        nm[7]=a; nm[75]=b;
        d1[3][a]=1; d2[1][a]=1; d3[7][a]=1; d4[7][a]=1;
        d1[25][b]=1; d2[27][b]=1; d3[21][b]=1; d4[21][b]=1;
        stp14();
        d1[3][a]=0; d2[1][a]=0; d3[7][a]=0; d4[7][a]=0;
        d1[25][b]=0; d2[27][b]=0; d3[21][b]=0; d4[21][b]=0;
      }}
  }
}
/* Set n8 & n74 */
void stp14(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[3][a]==0)&&(d2[2][a]==0)&&(d3[8][a]==0)&&(d4[8][a]==0)){
      if((d1[25][b]==0)&&(d2[26][b]==0)&&(d3[20][b]==0)&&(d4[20][b]==0)){
        nm[8]=a; nm[74]=b;
        d1[3][a]=1; d2[2][a]=1; d3[8][a]=1; d4[8][a]=1;
        d1[25][b]=1; d2[26][b]=1; d3[20][b]=1; d4[20][b]=1;
        stp15();
        d1[3][a]=0; d2[2][a]=0; d3[8][a]=0; d4[8][a]=0;
        d1[25][b]=0; d2[26][b]=0; d3[20][b]=0; d4[20][b]=0;
      }}
  }
}
/* Set n9 & n73 */
void stp15(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[3][a]==0)&&(d2[3][a]==0)&&(d3[9][a]==0)&&(d4[9][a]==0)){
      if((d1[25][b]==0)&&(d2[25][b]==0)&&(d3[19][b]==0)&&(d4[19][b]==0)){
       if((pb9[a]<3)&&(pb9[b]<3)){
         nm[9]=a; nm[73]=b;
         d1[3][a]=1; d2[3][a]=1; d3[9][a]=1; d4[9][a]=1; pb9[a]++;
         d1[25][b]=1; d2[25][b]=1; d3[19][b]=1; d4[19][b]=1; pb9[b]++;
         stp16();
         d1[3][a]=0; d2[3][a]=0; d3[9][a]=0; d4[9][a]=0; pb9[a]--;
         d1[25][b]=0; d2[25][b]=0; d3[19][b]=0; d4[19][b]=0; pb9[b]--;
       }}}
  }
}
/* Set n13 & n69 */
void stp16(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[5][a]==0)&&(d2[4][a]==0)&&(d3[4][a]==0)&&(d4[13][a]==0)){
      if((d1[23][b]==0)&&(d2[24][b]==0)&&(d3[24][b]==0)&&(d4[15][b]==0)){
```

```
          nm[13]=a; nm[69]=b;
          d1[5][a]=1; d2[4][a]=1; d3[4][a]=1; d4[13][a]=1;
          d1[23][b]=1; d2[24][b]=1; d3[24][b]=1; d4[15][b]=1;
          stp17();
          d1[5][a]=0; d2[4][a]=0; d3[4][a]=0; d4[13][a]=0;
          d1[23][b]=0; d2[24][b]=0; d3[24][b]=0; d4[15][b]=0;
        }}
  }
}
/* Set n14 & n68 */
void stp17(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[5][a]==0)&&(d2[5][a]==0)&&(d3[5][a]==0)&&(d4[14][a]==0)){
      if((d1[23][b]==0)&&(d2[23][b]==0)&&(d3[23][b]==0)&&(d4[14][b]==0)){
        nm[14]=a; nm[68]=b;
        d1[5][a]=1; d2[5][a]=1; d3[5][a]=1; d4[14][a]=1;
        d1[23][b]=1; d2[23][b]=1; d3[23][b]=1; d4[14][b]=1;
        stp18();
        d1[5][a]=0; d2[5][a]=0; d3[5][a]=0; d4[14][a]=0;
        d1[23][b]=0; d2[23][b]=0; d3[23][b]=0; d4[14][b]=0;
      }}
  }
}
/* Set n15 & n67 */
void stp18(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[5][a]==0)&&(d2[6][a]==0)&&(d3[6][a]==0)&&(d4[15][a]==0)){
      if((d1[23][b]==0)&&(d2[22][b]==0)&&(d3[22][b]==0)&&(d4[13][b]==0)){
        nm[15]=a; nm[67]=b;
        d1[5][a]=1; d2[6][a]=1; d3[6][a]=1; d4[15][a]=1;
        d1[23][b]=1; d2[22][b]=1; d3[22][b]=1; d4[13][b]=1;
        stp19();
        d1[5][a]=0; d2[6][a]=0; d3[6][a]=0; d4[15][a]=0;
        d1[23][b]=0; d2[22][b]=0; d3[22][b]=0; d4[13][b]=0;
      }}
  }
}
/* Set n16 & n66 */
void stp19(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[6][a]==0)&&(d2[4][a]==0)&&(d3[7][a]==0)&&(d4[16][a]==0)){
      if((d1[22][b]==0)&&(d2[24][b]==0)&&(d3[21][b]==0)&&(d4[12][b]==0)){
        nm[16]=a; nm[66]=b;
        d1[6][a]=1; d2[4][a]=1; d3[7][a]=1; d4[16][a]=1;
        d1[22][b]=1; d2[24][b]=1; d3[21][b]=1; d4[12][b]=1;
        stp20();
        d1[6][a]=0; d2[4][a]=0; d3[7][a]=0; d4[16][a]=0;
        d1[22][b]=0; d2[24][b]=0; d3[21][b]=0; d4[12][b]=0;
      }}
  }
}
/* Set n17 & n65 */
void stp20(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[6][a]==0)&&(d2[5][a]==0)&&(d3[8][a]==0)&&(d4[17][a]==0)){
      if((d1[22][b]==0)&&(d2[23][b]==0)&&(d3[20][b]==0)&&(d4[11][b]==0)){
       if((pb9[a]<3)&&(pb9[b]<3)){
          nm[17]=a; nm[65]=b;
          d1[6][a]=1; d2[5][a]=1; d3[8][a]=1; d4[17][a]=1; pb9[a]++;
          d1[22][b]=1; d2[23][b]=1; d3[20][b]=1; d4[11][b]=1; pb9[b]++;
          stp21();
```

```
          d1[6][a]=0; d2[5][a]=0; d3[8][a]=0; d4[17][a]=0; pb9[a]--;
          d1[22][b]=0; d2[23][b]=0; d3[20][b]=0; d4[11][b]=0; pb9[b]--;
        }}}
  }
}
/* Set n18 & n64 */
void stp21(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[6][a]==0)&&(d2[6][a]==0)&&(d3[9][a]==0)&&(d4[18][a]==0)){
      if((d1[22][b]==0)&&(d2[22][b]==0)&&(d3[19][b]==0)&&(d4[10][b]==0)){
        nm[18]=a; nm[64]=b;
        d1[6][a]=1; d2[6][a]=1; d3[9][a]=1; d4[18][a]=1;
        d1[22][b]=1; d2[22][b]=1; d3[19][b]=1; d4[10][b]=1;
        stp22();
        d1[6][a]=0; d2[6][a]=0; d3[9][a]=0; d4[18][a]=0;
        d1[22][b]=0; d2[22][b]=0; d3[19][b]=0; d4[10][b]=0;
      }}
  }
}
/* Set n22 & n60 */
void stp22(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[8][a]==0)&&(d2[7][a]==0)&&(d3[4][a]==0)&&(d4[22][a]==0)){
      if((d1[20][b]==0)&&(d2[21][b]==0)&&(d3[24][b]==0)&&(d4[6][b]==0)){
        nm[22]=a; nm[60]=b;
        d1[8][a]=1; d2[7][a]=1; d3[4][a]=1; d4[22][a]=1;
        d1[20][b]=1; d2[21][b]=1; d3[24][b]=1; d4[6][b]=1;
        stp23();
        d1[8][a]=0; d2[7][a]=0; d3[4][a]=0; d4[22][a]=0;
        d1[20][b]=0; d2[21][b]=0; d3[24][b]=0; d4[6][b]=0;
      }}
  }
}
/* Set n23 & n59 */
void stp23(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[8][a]==0)&&(d2[8][a]==0)&&(d3[5][a]==0)&&(d4[23][a]==0)){
      if((d1[20][b]==0)&&(d2[20][b]==0)&&(d3[23][b]==0)&&(d4[5][b]==0)){
        nm[23]=a; nm[59]=b;
        d1[8][a]=1; d2[8][a]=1; d3[5][a]=1; d4[23][a]=1;
        d1[20][b]=1; d2[20][b]=1; d3[23][b]=1; d4[5][b]=1;
        stp24();
        d1[8][a]=0; d2[8][a]=0; d3[5][a]=0; d4[23][a]=0;
        d1[20][b]=0; d2[20][b]=0; d3[23][b]=0; d4[5][b]=0;
      }}
  }
}
/* Set n24 & n58 */
void stp24(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[8][a]==0)&&(d2[9][a]==0)&&(d3[6][a]==0)&&(d4[24][a]==0)){
      if((d1[20][b]==0)&&(d2[19][b]==0)&&(d3[22][b]==0)&&(d4[4][b]==0)){
        nm[24]=a; nm[58]=b;
        d1[8][a]=1; d2[9][a]=1; d3[6][a]=1; d4[24][a]=1;
        d1[20][b]=1; d2[19][b]=1; d3[22][b]=1; d4[4][b]=1;
        stp25();
        d1[8][a]=0; d2[9][a]=0; d3[6][a]=0; d4[24][a]=0;
        d1[20][b]=0; d2[19][b]=0; d3[22][b]=0; d4[4][b]=0;
      }}
  }
}
```

```
/* Set n25 & n57 */
void stp25(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[9][a]==0)&&(d2[7][a]==0)&&(d3[7][a]==0)&&(d4[25][a]==0)){
      if((d1[19][b]==0)&&(d2[21][b]==0)&&(d3[21][b]==0)&&(d4[3][b]==0)){
        nm[25]=a; nm[57]=b;
        d1[9][a]=1; d2[7][a]=1; d3[7][a]=1; d4[25][a]=1;
        d1[19][b]=1; d2[21][b]=1; d3[21][b]=1; d4[3][b]=1;
        stp26();
        d1[9][a]=0; d2[7][a]=0; d3[7][a]=0; d4[25][a]=0;
        d1[19][b]=0; d2[21][b]=0; d3[21][b]=0; d4[3][b]=0;
      }}
  }
}
/* Set n26 & n56 */
void stp26(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[9][a]==0)&&(d2[8][a]==0)&&(d3[8][a]==0)&&(d4[26][a]==0)){
      if((d1[19][b]==0)&&(d2[20][b]==0)&&(d3[20][b]==0)&&(d4[2][b]==0)){
        nm[26]=a; nm[56]=b;
        d1[9][a]=1; d2[8][a]=1; d3[8][a]=1; d4[26][a]=1;
        d1[19][b]=1; d2[20][b]=1; d3[20][b]=1; d4[2][b]=1;
        stp27();
        d1[9][a]=0; d2[8][a]=0; d3[8][a]=0; d4[26][a]=0;
        d1[19][b]=0; d2[20][b]=0; d3[20][b]=0; d4[2][b]=0;
      }}
  }
}
/* Set n27 & n55 */
void stp27(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[9][a]==0)&&(d2[9][a]==0)&&(d3[9][a]==0)&&(d4[27][a]==0)){
      if((d1[19][b]==0)&&(d2[19][b]==0)&&(d3[19][b]==0)&&(d4[1][b]==0)){
        nm[27]=a; nm[55]=b;
        d1[9][a]=1; d2[9][a]=1; d3[9][a]=1; d4[27][a]=1;
        d1[19][b]=1; d2[19][b]=1; d3[19][b]=1; d4[1][b]=1;
        stp28();
        d1[9][a]=0; d2[9][a]=0; d3[9][a]=0; d4[27][a]=0;
        d1[19][b]=0; d2[19][b]=0; d3[19][b]=0; d4[1][b]=0;
      }}
  }
}
//* Level #3: *
//* Set n28 & n54 *
void stp28(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[10][a]==0)&&(d2[10][a]==0)&&(d3[10][a]==0)&&(d4[1][a]==0)){
      if((d1[18][b]==0)&&(d2[18][b]==0)&&(d3[18][b]==0)&&(d4[27][b]==0)){
        nm[28]=a; nm[54]=b;
        d1[10][a]=1; d2[10][a]=1; d3[10][a]=1; d4[1][a]=1;
        d1[18][b]=1; d2[18][b]=1; d3[18][b]=1; d4[27][b]=1;
        stp29();
        d1[10][a]=0; d2[10][a]=0; d3[10][a]=0; d4[1][a]=0;
        d1[18][b]=0; d2[18][b]=0; d3[18][b]=0; d4[27][b]=0;
      }}
  }
}
/* Set n37 & n45 */
void stp29(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
```

```
    if((d1[13][a]==0)&&(d2[13][a]==0)&&(d3[10][a]==0)&&(d4[10][a]==0)){
      if((d1[15][b]==0)&&(d2[15][b]==0)&&(d3[18][b]==0)&&(d4[18][b]==0)){
        nm[37]=a; nm[45]=b;
        d1[13][a]=1; d2[13][a]=1; d3[10][a]=1; d4[10][a]=1;
        d1[15][b]=1; d2[15][b]=1; d3[18][b]=1; d4[18][b]=1;
        stp30();
        d1[13][a]=0; d2[13][a]=0; d3[10][a]=0; d4[10][a]=0;
        d1[15][b]=0; d2[15][b]=0; d3[18][b]=0; d4[18][b]=0;
      }}
  }
}
/* Set n46 & n36 */
void stp30(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[16][a]==0)&&(d2[16][a]==0)&&(d3[10][a]==0)&&(d4[19][a]==0)){
      if((d1[12][b]==0)&&(d2[12][b]==0)&&(d3[18][b]==0)&&(d4[9][b]==0)){
        nm[46]=a; nm[36]=b;
        d1[16][a]=1; d2[16][a]=1; d3[10][a]=1; d4[19][a]=1;
        d1[12][b]=1; d2[12][b]=1; d3[18][b]=1; d4[9][b]=1;
        stp31();
        d1[16][a]=0; d2[16][a]=0; d3[10][a]=0; d4[19][a]=0;
        d1[12][b]=0; d2[12][b]=0; d3[18][b]=0; d4[9][b]=0;
      }}
  }
}
/* Set n29 & n53 */
void stp31(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[10][a]==0)&&(d2[11][a]==0)&&(d3[11][a]==0)&&(d4[2][a]==0)){
      if((d1[18][b]==0)&&(d2[17][b]==0)&&(d3[17][b]==0)&&(d4[26][b]==0)){
        nm[29]=a; nm[53]=b;
        d1[10][a]=1; d2[11][a]=1; d3[11][a]=1; d4[2][a]=1;
        d1[18][b]=1; d2[17][b]=1; d3[17][b]=1; d4[26][b]=1;
        stp32();
        d1[10][a]=0; d2[11][a]=0; d3[11][a]=0; d4[2][a]=0;
        d1[18][b]=0; d2[17][b]=0; d3[17][b]=0; d4[26][b]=0;
      }}
  }
}
/* Set n38 & n44 */
void stp32(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[13][a]==0)&&(d2[14][a]==0)&&(d3[11][a]==0)&&(d4[11][a]==0)){
      if((d1[15][b]==0)&&(d2[14][b]==0)&&(d3[17][b]==0)&&(d4[17][b]==0)){
        nm[38]=a; nm[44]=b;
        d1[13][a]=1; d2[14][a]=1; d3[11][a]=1; d4[11][a]=1;
        d1[15][b]=1; d2[14][b]=1; d3[17][b]=1; d4[17][b]=1;
        stp33();
        d1[13][a]=0; d2[14][a]=0; d3[11][a]=0; d4[11][a]=0;
        d1[15][b]=0; d2[14][b]=0; d3[17][b]=0; d4[17][b]=0;
      }}
  }
}
/* Set n47 & n35 */
void stp33(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[16][a]==0)&&(d2[17][a]==0)&&(d3[11][a]==0)&&(d4[20][a]==0)){
      if((d1[12][b]==0)&&(d2[11][b]==0)&&(d3[17][b]==0)&&(d4[8][b]==0)){
        nm[47]=a; nm[35]=b;
        d1[16][a]=1; d2[17][a]=1; d3[11][a]=1; d4[20][a]=1;
        d1[12][b]=1; d2[11][b]=1; d3[17][b]=1; d4[8][b]=1;
```

```
          stp34();
          d1[16][a]=0; d2[17][a]=0; d3[11][a]=0; d4[20][a]=0;
          d1[12][b]=0; d2[11][b]=0; d3[17][b]=0; d4[8][b]=0;
        }}
  }
}
/* Set n30 & n52 */
void stp34(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[10][a]==0)&&(d2[12][a]==0)&&(d3[12][a]==0)&&(d4[3][a]==0)){
      if((d1[18][b]==0)&&(d2[16][b]==0)&&(d3[16][b]==0)&&(d4[25][b]==0)){
        nm[30]=a; nm[52]=b;
        d1[10][a]=1; d2[12][a]=1; d3[12][a]=1; d4[3][a]=1;
        d1[18][b]=1; d2[16][b]=1; d3[16][b]=1; d4[25][b]=1;
        stp35();
        d1[10][a]=0; d2[12][a]=0; d3[12][a]=0; d4[3][a]=0;
        d1[18][b]=0; d2[16][b]=0; d3[16][b]=0; d4[25][b]=0;
      }}
  }
}
/* Set n39 & n43 */
void stp35(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[13][a]==0)&&(d2[15][a]==0)&&(d3[12][a]==0)&&(d4[12][a]==0)){
      if((d1[15][b]==0)&&(d2[13][b]==0)&&(d3[16][b]==0)&&(d4[16][b]==0)){
        nm[39]=a; nm[43]=b;
        d1[13][a]=1; d2[15][a]=1; d3[12][a]=1; d4[12][a]=1;
        d1[15][b]=1; d2[13][b]=1; d3[16][b]=1; d4[16][b]=1;
        stp36();
        d1[13][a]=0; d2[15][a]=0; d3[12][a]=0; d4[12][a]=0;
        d1[15][b]=0; d2[13][b]=0; d3[16][b]=0; d4[16][b]=0;
      }}
  }
}
/* Set n48 & n34 */
void stp36(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[16][a]==0)&&(d2[18][a]==0)&&(d3[12][a]==0)&&(d4[21][a]==0)){
      if((d1[12][b]==0)&&(d2[10][b]==0)&&(d3[16][b]==0)&&(d4[7][b]==0)){
        nm[48]=a; nm[34]=b;
        d1[16][a]=1; d2[18][a]=1; d3[12][a]=1; d4[21][a]=1;
        d1[12][b]=1; d2[10][b]=1; d3[16][b]=1; d4[7][b]=1;
        stp37();
        d1[16][a]=0; d2[18][a]=0; d3[12][a]=0; d4[21][a]=0;
        d1[12][b]=0; d2[10][b]=0; d3[16][b]=0; d4[7][b]=0;
      }}
  }
}
//* Level #4: *
//* Set n31 & n51 *
void stp37(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[11][a]==0)&&(d2[10][a]==0)&&(d3[13][a]==0)&&(d4[4][a]==0)){
      if((d1[17][b]==0)&&(d2[18][b]==0)&&(d3[15][b]==0)&&(d4[24][b]==0)){
       if((pd1[a]<3)&&(pd1[b]<3)){
        nm[31]=a; nm[51]=b;
        d1[11][a]=1; d2[10][a]=1; d3[13][a]=1; d4[4][a]=1; pd1[a]++;
        d1[17][b]=1; d2[18][b]=1; d3[15][b]=1; d4[24][b]=1; pd1[b]++;
        stp38();
        d1[11][a]=0; d2[10][a]=0; d3[13][a]=0; d4[4][a]=0; pd1[a]--;
        d1[17][b]=0; d2[18][b]=0; d3[15][b]=0; d4[24][b]=0; pd1[b]--;
```

```
        }}}
  }
}
/* Set n32 & n50 */
void stp38(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[11][a]==0)&&(d2[11][a]==0)&&(d3[14][a]==0)&&(d4[5][a]==0)){
      if((d1[17][b]==0)&&(d2[17][b]==0)&&(d3[14][b]==0)&&(d4[23][b]==0)){
        nm[32]=a; nm[50]=b;
        d1[11][a]=1; d2[11][a]=1; d3[14][a]=1; d4[5][a]=1;
        d1[17][b]=1; d2[17][b]=1; d3[14][b]=1; d4[23][b]=1;
        stp39();
        d1[11][a]=0; d2[11][a]=0; d3[14][a]=0; d4[5][a]=0;
        d1[17][b]=0; d2[17][b]=0; d3[14][b]=0; d4[23][b]=0;
      }}
  }
}
/* Set n33 & n49 */
void stp39(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[11][a]==0)&&(d2[12][a]==0)&&(d3[15][a]==0)&&(d4[6][a]==0)){
      if((d1[17][b]==0)&&(d2[16][b]==0)&&(d3[13][b]==0)&&(d4[22][b]==0)){
       if((pb9[a]<3)&&(pb9[b]<3)){
         nm[33]=a; nm[49]=b;
         d1[11][a]=1; d2[12][a]=1; d3[15][a]=1; d4[6][a]=1; pb9[a]++;
         d1[17][b]=1; d2[16][b]=1; d3[13][b]=1; d4[22][b]=1; pb9[b]++;
         stp40();
         d1[11][a]=0; d2[12][a]=0; d3[15][a]=0; d4[6][a]=0; pb9[a]--;
         d1[17][b]=0; d2[16][b]=0; d3[13][b]=0; d4[22][b]=0; pb9[b]--;
       }}}
  }
}
/* Set n40 & n42 */
void stp40(){
 short a,b;
  for(a=0;a<3;a++){b=CC-a;
    if((d1[14][a]==0)&&(d2[13][a]==0)&&(d3[13][a]==0)&&(d4[13][a]==0)){
      if((d1[14][b]==0)&&(d2[15][b]==0)&&(d3[15][b]==0)&&(d4[15][b]==0)){
        nm[40]=a; nm[42]=b;
        d1[14][a]=1; d2[13][a]=1; d3[13][a]=1; d4[13][a]=1;
        d1[14][b]=1; d2[15][b]=1; d3[15][b]=1; d4[15][b]=1;
        recordans();
        d1[14][a]=0; d2[13][a]=0; d3[13][a]=0; d4[13][a]=0;
        d1[14][b]=0; d2[15][b]=0; d3[15][b]=0; d4[15][b]=0;
      }}
  }
}
//
//* Record the Latin Units *
void recordans(){
 short n;
  cnt++;
  tlu[cnt-1][0]=cnt;
  for(n=1;n<82;n++){tlu[cnt-1][n]=nm[n];}
}
//
//* Print the Latin Units *
void pr9lu(){
 short m,l,l9,n,t;
  for(m=0;m<cnt;m=m+4){
    printf("%18d/%19d/%19d/%19d/\n",
           tlu[m][0],tlu[m+1][0],tlu[m+2][0],tlu[m+3][0]);
    for(l=0;l<9;l++){l9=l*9;
```

```
        printf(" ");
        for(n=1;n<10;n++){printf("%2d",tlu[m][l9+n]);}
        printf("  ");
        for(n=1;n<10;n++){printf("%2d",tlu[m+1][l9+n]);}
        printf("  ");
        for(n=1;n<10;n++){printf("%2d",tlu[m+2][l9+n]);}
        printf("  ");
        for(n=1;n<10;n++){printf("%2d",tlu[m+3][l9+n]);}
        printf("\n");
      }}
    printf(" [Count of Latin Units = %d]\n",cnt);
    /* Measure How Similar each Pair is. */
    for(m=0;m<cnt;m++){
      for(l=0;l<cnt;l++){t=0;
        for(n=1;n<82;n++){
          if(tlu[m][n]==tlu[l][n]){t++;}}
        mtc[m][l]=t;
      }}
    /* Print the Reference Table */
    printf("\n");
    printf(" [Reference Table for the Best Combination]\n");
    printf(" * |");
    for(n=1;n<=cnt;n++){printf("%3d",n);}
    printf("\n --+------------------------\n");
    for(m=0;m<cnt;m++){
      printf("%3d|",m+1);
      for(n=0;n<cnt;n++){printf("%3d",mtc[m][n]);}
      printf("\n");
    }
}
//
//* Combine abd Compose *
void cmbcmp(){
 short n,d,md,fc;
   md=27;
   for(u1=0;u1<8;u1++){
     for(u2=0;u2<8;u2++){
       if(mtc[u2][u1]==md){
         for(u3=0;u3<8;u3++){
           if((mtc[u3][u1]==md)&&(mtc[u3][u2]==md)){
             for(u4=0;u4<8;u4++){
               if((mtc[u4][u1]==md)&&(mtc[u4][u2]==md)&&(mtc[u4][u3]==md)){
                 for(n=1;n<82;n++){uflg[n]=0;}
                 for(n=1;n<82;n++){
                   d=tlu[u1][n]*27+tlu[u2][n]*9+tlu[u3][n]*3+tlu[u4][n]+1;
                   nm[n]=d; uflg[d]++;
                 }
                 fc=0;
                 for(n=1;n<82;n++){
                   if(uflg[n]==1){fc++;}else{break;}
                 }
                 if(fc==81){
                   if((nm[1]<nm[81])&&(nm[1]<nm[9])&&(nm[1]<nm[73])){
                     if(nm[2]>nm[10]){cmprecord();}
                   }}
               }
             }
           }
         }
       }
     }
   }
}
//
//* Save the Compositions to the Table *
```

```
void cmprecord(){
 short n;
  tnm[cnt][0]=cnt+1;
  for(n=1;n<82;n++){tnm[cnt][n]=nm[n];}
  tnm[cnt][82]=u1+1; tnm[cnt][83]=u2+1; tnm[cnt][84]=u3+1; tnm[cnt][85]=u4+1;
  cnt++; if(nm[1]==1){cnt1++;}
}
//
//* Sort the Composed Data *
void srtans(){
 short mx,n,f;
 short d1,d2,d3,d4,d5,d6,d7,d8,d9,d10;
  mx=cnt-1;
  do{f=0;
    for(n=0;n<mx;n++){
      d1=tnm[n][1]; d2=tnm[n+1][1];
      d3=tnm[n][2]*82+tnm[n][3]; d4=tnm[n+1][2]*82+tnm[n+1][3];
      d5=tnm[n][10]*82-tnm[n][19]; d6=tnm[n+1][10]*82-tnm[n+1][19];
      d7=tnm[n][11]*82+tnm[n][21]; d8=tnm[n+1][11]*82+tnm[n+1][21];
      d9=tnm[n][4]*82+tnm[n][28]; d10=tnm[n+1][4]*82+tnm[n+1][28];
      if(d1>d2){excsol(n); f=1;}
      else if((d1==d2)&&(d3<d4)){excsol(n); f=1;}
      else if((d1==d2)&&(d3==d4)&&(d5<d6)){excsol(n); f=1;}
      else if((d1==d2)&&(d3==d4)&&(d5==d6)&&(d7<d8)){excsol(n); f=1;}
      else if((d1==d2)&&(d3==d4)&&(d5==d6)&&(d7==d8)&&(d9>d10)){excsol(n); f=1;}
    }
    mx--;
  }while(f>0);
}
//
void excsol(short x){
 short n,d;
  for(n=1;n<86;n++){d=tnm[x][n]; tnm[x][n]=tnm[x+1][n]; tnm[x+1][n]=d;}
}
//
//* Print 3 Answers *
void pr3ans(){
 short m,n,l,l9,p;
  for(m=0;m<cnt;m=m+3){
    for(n=0;n<3;n++){
      printf("%3d/%2d/%2d/%2d/%13d#",
              tnm[m+n][82],tnm[m+n][83],tnm[m+n][84],tnm[m+n][85],tnm[m+n][0]);
      if(n<2){printf("  ");}
    }
    printf("\n");
    for(l=0;l<9;l++){l9=l*9;
      for(n=0;n<3;n++){
        for(p=1;p<10;p++){printf("%3d",tnm[m+n][l9+p]);}
        if(n<2){printf("  ");}
      }
      printf("\n");
    }
  }
}
//
//* Print 1 Answer *
void pr1ans(){
 short m,l,l9,n;
 short lu1,lu2,lu3,lu4;
  for(m=0;m<cnt;m++){
    lu1=tnm[m][82]-1; lu2=tnm[m][83]-1; lu3=tnm[m][84]-1; lu4=tnm[m][85]-1;
    printf("%10d/%10d/%10d/%10d/%27d#\n",
            lu1+1,lu2+1,lu3+1,lu4+1,tnm[m][0]);
    for(l=0;l<9;l++){l9=l*9;
      printf("  ");
```

```
      for(n=1;n<10;n++){printf("%d",tlu[lu1][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[lu2][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[lu3][l9+n]);}
      printf("  ");
      for(n=1;n<10;n++){printf("%d",tlu[lu4][l9+n]);}
      printf(" ");
      for(n=1;n<10;n++){printf("%3d",tnm[m][l9+n]);}
      printf("\n");
    }
  }
}
```

The next list demonstrates what our job could have produced.

```
** Complete Euler Squares of Order 9 Composed by the 'New Euler's Method' **
**  Down-converted from the 4-Dimensional Extra-Cubic Objects of Order 3: **
**   Multiple 3x3 Type of Self-complementary Magic Squares of Order 9  **

 [List of Latin Units]
                1/                  2/                  3/                  4/
 0 1 2 2 0 1 1 2 0   0 2 1 2 1 0 1 0 2   0 2 1 1 0 2 2 1 0   0 2 1 2 1 0 1 0 2
 2 0 1 1 2 0 0 1 2   1 0 2 0 2 1 2 1 0   2 1 0 0 2 1 1 0 2   2 1 0 1 0 2 0 2 1
 1 2 0 0 1 2 2 0 1   2 1 0 1 0 2 0 2 1   1 0 2 2 1 0 0 2 1   1 0 2 0 2 1 2 1 0
 2 0 1 1 2 0 0 1 2   2 1 0 1 0 2 0 2 1   2 1 0 0 2 1 1 0 2   1 0 2 0 2 1 2 1 0
 1 2 0 0 1 2 2 0 1   0 2 1 2 1 0 1 0 2   1 0 2 2 1 0 0 2 1   0 2 1 2 1 0 1 0 2
 0 1 2 2 0 1 1 2 0   1 0 2 0 2 1 2 1 0   0 2 1 1 0 2 2 1 0   2 1 0 1 0 2 0 2 1
 1 2 0 0 1 2 2 0 1   1 0 2 0 2 1 2 1 0   1 0 2 2 1 0 0 2 1   2 1 0 1 0 2 0 2 1
 0 1 2 2 0 1 1 2 0   2 1 0 1 0 2 0 2 1   0 2 1 1 0 2 2 1 0   1 0 2 0 2 1 2 1 0
 2 0 1 1 2 0 0 1 2   0 2 1 2 1 0 1 0 2   2 1 0 0 2 1 1 0 2   0 2 1 2 1 0 1 0 2
                5/                  6/                  7/                  8/
 2 0 1 0 1 2 1 2 0   2 0 1 1 2 0 0 1 2   2 0 1 0 1 2 1 2 0   2 1 0 0 2 1 1 0 2
 0 1 2 1 2 0 2 0 1   0 1 2 2 0 1 1 2 0   1 2 0 2 0 1 0 1 2   0 2 1 1 0 2 2 1 0
 1 2 0 2 0 1 0 1 2   1 2 0 0 1 2 2 0 1   0 1 2 1 2 0 2 0 1   1 0 2 2 1 0 0 2 1
 1 2 0 2 0 1 0 1 2   0 1 2 2 0 1 1 2 0   0 1 2 1 2 0 2 0 1   0 2 1 1 0 2 2 1 0
 2 0 1 0 1 2 1 2 0   1 2 0 0 1 2 2 0 1   2 0 1 0 1 2 1 2 0   1 0 2 2 1 0 0 2 1
 0 1 2 1 2 0 2 0 1   2 0 1 1 2 0 0 1 2   1 2 0 2 0 1 0 1 2   2 1 0 0 2 1 1 0 2
 0 1 2 1 2 0 2 0 1   1 2 0 0 1 2 2 0 1   1 2 0 2 0 1 0 1 2   1 0 2 2 1 0 0 2 1
 1 2 0 2 0 1 0 1 2   2 0 1 1 2 0 0 1 2   0 1 2 1 2 0 2 0 1   2 1 0 0 2 1 1 0 2
 2 0 1 0 1 2 1 2 0   0 1 2 2 0 1 1 2 0   2 0 1 0 1 2 1 2 0   0 2 1 1 0 2 2 1 0
 [Count of Latin Units = 8]

 [Reference Table for the Best Combination]
 * |  1  2  3  4  5  6  7  8
 --+------------------------
  1| 81 27 27 27 27 27 27 27
  2| 27 81 27 27 27 27 27 27
  3| 27 27 81 27 27 27 27 27
  4| 27 27 27 81 27 27 27 27
  5| 27 27 27 27 81 27 27 27
  6| 27 27 27 27 27 81 27 27
  7| 27 27 27 27 27 27 81 27
  8| 27 27 27 27 27 27 27 81

** List of the Standard Solutions: Used Units Sol_Number# **
        3/         4/         2/         1/                          1#
  021102210  021210102  021210102  012201120   1 80 42 54 13 56 68 30 25
  210021102  210102021  102021210  201120012  78 37  8 11 63 49 34 23 66
  102210021  102021210  210102021  120012201  44  6 73 58 47 18 21 70 32
  210021102  102021210  210102021  201120012  72 31 20  5 75 43 46 17 60
  102210021  021210102  021210102  120012201  29 27 67 79 41  3 15 55 53
  021102210  210102021  102021210  012201120  22 65 36 39  7 77 62 51 10
```

```
102210021  210102021  102021210  120012201  50 12 61 64 35 24  9 76 38
021102210  102021210  210102021  012201120  16 59 48 33 19 71 74 45  4
210021102  021210102  021210102  201120012  57 52 14 26 69 28 40  2 81
       4/         3/         2/         1/                        2#
021210102  021102210  021210102  012201120   1 80 42 72 31 20 50 12 61
210102021  210021102  102021210  201120012  78 37  8 29 27 67 16 59 48
102021210  102210021  210102021  120012201  44  6 73 22 65 36 57 52 14
102021210  210021102  210102021  201120012  54 13 56  5 75 43 64 35 24
021210102  102210021  021210102  120012201  11 63 49 79 41  3 33 19 71
210102021  021102210  102021210  012201120  58 47 18 39  7 77 26 69 28
210102021  102210021  102021210  120012201  68 30 25 46 17 60  9 76 38
102021210  021102210  210102021  012201120  34 23 66 15 55 53 74 45  4
021210102  210021102  021210102  201120012  21 70 32 62 51 10 40  2 81
       3/         2/         4/         1/                        3#
021102210  021210102  021210102  012201120   1 80 42 54 13 56 68 30 25
210021102  102021210  210102021  201120012  72 31 20  5 75 43 46 17 60
102210021  210102021  102021210  120012201  50 12 61 64 35 24  9 76 38
210021102  210102021  102021210  201120012  78 37  8 11 63 49 34 23 66
102210021  021210102  021210102  120012201  29 27 67 79 41  3 15 55 53
021102210  102021210  210102021  012201120  16 59 48 33 19 71 74 45  4
102210021  102021210  210102021  120012201  44  6 73 58 47 18 21 70 32
021102210  210102021  102021210  012201120  22 65 36 39  7 77 62 51 10
210021102  021210102  021210102  201120012  57 52 14 26 69 28 40  2 81
       4/         2/         3/         1/                        4#
021210102  021210102  021102210  012201120   1 80 42 78 37  8 44  6 73
210102021  102021210  210021102  201120012  72 31 20 29 27 67 22 65 36
102021210  210102021  102210021  120012201  50 12 61 16 59 48 57 52 14
102021210  210102021  210021102  201120012  54 13 56 11 63 49 58 47 18
021210102  021210102  102210021  120012201   5 75 43 79 41  3 39  7 77
210102021  102021210  021102210  012201120  64 35 24 33 19 71 26 69 28
210102021  102021210  102210021  120012201  68 30 25 34 23 66 21 70 32
102021210  210102021  021102210  012201120  46 17 60 15 55 53 62 51 10
021210102  021210102  210021102  201120012   9 76 38 74 45  4 40  2 81
       2/         3/         4/         1/                        5#
021210102  021102210  021210102  012201120   1 80 42 72 31 20 50 12 61
102021210  210021102  210102021  201120012  54 13 56  5 75 43 64 35 24
210102021  102210021  102021210  120012201  68 30 25 46 17 60  9 76 38
210102021  210021102  102021210  201120012  78 37  8 29 27 67 16 59 48
021210102  102210021  021210102  120012201  11 63 49 79 41  3 33 19 71
102021210  021102210  210102021  012201120  34 23 66 15 55 53 74 45  4
102021210  102210021  210102021  120012201  44  6 73 22 65 36 57 52 14
210102021  021102210  102021210  012201120  58 47 18 39  7 77 26 69 28
021210102  210021102  021210102  201120012  21 70 32 62 51 10 40  2 81
       2/         4/         3/         1/                        6#
021210102  021210102  021102210  012201120   1 80 42 78 37  8 44  6 73
102021210  210102021  210021102  201120012  54 13 56 11 63 49 58 47 18
210102021  102021210  102210021  120012201  68 30 25 34 23 66 21 70 32
210102021  102021210  210021102  201120012  72 31 20 29 27 67 22 65 36
021210102  021210102  102210021  120012201   5 75 43 79 41  3 39  7 77
102021210  210102021  021102210  012201120  46 17 60 15 55 53 62 51 10
102021210  210102021  102210021  120012201  50 12 61 16 59 48 57 52 14
210102021  102021210  021102210  012201120  64 35 24 33 19 71 26 69 28
021210102  021210102  210021102  201120012   9 76 38 74 45  4 40  2 81
       3/         2/         1/         4/                        7#
021102210  021210102  012201120  021210102   1 78 44 54 11 58 68 34 21
210021102  102021210  201120012  210102021  72 29 22  5 79 39 46 15 62
102210021  210102021  120012201  102021210  50 16 57 64 33 26  9 74 40
210021102  210102021  201120012  102021210  80 37  6 13 63 47 30 23 70
102210021  021210102  120012201  021210102  31 27 65 75 41  7 17 55 51
021102210  102021210  012201120  210102021  12 59 52 35 19 69 76 45  2
102210021  102021210  120012201  210102021  42  8 73 56 49 18 25 66 32
021102210  210102021  012201120  102021210  20 67 36 43  3 77 60 53 10
210021102  021210102  201120012  021210102  61 48 14 24 71 28 38  4 81
```

```
       4/         2/         1/         3/                              8#
021210102  021210102  012201120  021102210   1 78 44 80 37  6 42  8 73
210102021  102021210  201120012  210021102  72 29 22 31 27 65 20 67 36
102021210  210102021  120012201  102210021  50 16 57 12 59 52 61 48 14
102021210  210102021  201120012  210021102  54 11 58 13 63 47 56 49 18
021210102  021210102  120012201  102210021   5 79 39 75 41  7 43  3 77
210102021  102021210  012201120  021102210  64 33 26 35 19 69 24 71 28
210102021  102021210  120012201  102210021  68 34 21 30 23 70 25 66 32
102021210  210102021  012201120  021102210  46 15 62 17 55 51 60 53 10
021210102  021210102  201120012  210021102   9 74 40 76 45  2 38  4 81
       2/         3/         1/         4/                              9#
021210102  021102210  012201120  021210102   1 78 44 72 29 22 50 16 57
102021210  210021102  201120012  210102021  54 11 58  5 79 39 64 33 26
210102021  102210021  120012201  102021210  68 34 21 46 15 62  9 74 40
210102021  210021102  201120012  102021210  80 37  6 31 27 65 12 59 52
021210102  102210021  120012201  021210102  13 63 47 75 41  7 35 19 69
102021210  021102210  012201120  210102021  30 23 70 17 55 51 76 45  2
102021210  102210021  120012201  210102021  42  8 73 20 67 36 61 48 14
210102021  021102210  012201120  102021210  56 49 18 43  3 77 24 71 28
021210102  210021102  201120012  021210102  25 66 32 60 53 10 38  4 81
       2/         4/         1/         3/                             10#
021210102  021210102  012201120  021102210   1 78 44 80 37  6 42  8 73
102021210  210102021  201120012  210021102  54 11 58 13 63 47 56 49 18
210102021  102021210  120012201  102210021  68 34 21 30 23 70 25 66 32
210102021  102021210  201120012  210021102  72 29 22 31 27 65 20 67 36
021210102  021210102  120012201  102210021   5 79 39 75 41  7 43  3 77
102021210  210102021  012201120  021102210  46 15 62 17 55 51 60 53 10
102021210  210102021  120012201  102210021  50 16 57 12 59 52 61 48 14
210102021  102021210  012201120  021102210  64 33 26 35 19 69 24 71 28
021210102  021210102  201120012  210021102   9 74 40 76 45  2 38  4 81
       2/         1/         3/         4/                             11#
021210102  012201120  021102210  021210102   1 72 50 78 29 16 44 22 57
102021210  201120012  210021102  210102021  54  5 64 11 79 33 58 39 26
210102021  120012201  102210021  102021210  68 46  9 34 15 74 21 62 40
210102021  201120012  210021102  102021210  80 31 12 37 27 59  6 65 52
021210102  120012201  102210021  021210102  13 75 35 63 41 19 47  7 69
102021210  012201120  021102210  210102021  30 17 76 23 55 45 70 51  2
102021210  120012201  102210021  210102021  42 20 61  8 67 48 73 36 14
210102021  012201120  021102210  102021210  56 43 24 49  3 71 18 77 28
021210102  201120012  210021102  021210102  25 60 38 66 53  4 32 10 81
       2/         1/         4/         3/                             12#
021210102  012201120  021210102  021102210   1 72 50 80 31 12 42 20 61
102021210  201120012  210102021  210021102  54  5 64 13 75 35 56 43 24
210102021  120012201  102021210  102210021  68 46  9 30 17 76 25 60 38
210102021  201120012  102021210  210021102  78 29 16 37 27 59  8 67 48
021210102  120012201  021210102  102210021  11 79 33 63 41 19 49  3 71
102021210  012201120  210102021  021102210  34 15 74 23 55 45 66 53  4
102021210  120012201  210102021  102210021  44 22 57  6 65 52 73 36 14
210102021  012201120  102021210  021102210  58 39 26 47  7 69 18 77 28
021210102  201120012  021210102  210021102  21 62 40 70 51  2 32 10 81
 3/ 4/ 2/ 8/           13#    4/ 3/ 2/ 8/           14#    3/ 2/ 4/ 8/           15#
 3 80 40 52 15 56 68 28 27    3 80 40 70 33 20 50 10 63    3 80 40 52 15 56 68 28 27
76 39  8 11 61 51 36 23 64   76 39  8 29 25 69 18 59 46   70 33 20  5 73 45 48 17 58
44  4 75 60 47 16 19 72 32   44  4 75 24 65 34 55 54 14   50 10 63 66 35 22  7 78 38
70 33 20  5 73 45 48 17 58   52 15 56  5 73 45 66 35 22   76 39  8 11 61 51 36 23 64
29 25 69 81 41  1 13 57 53   11 61 51 81 41  1 31 21 71   29 25 69 81 41  1 13 57 53
24 65 34 37  9 77 62 49 12   60 47 16 37  9 77 26 67 30   18 59 46 31 21 71 74 43  6
50 10 63 66 35 22  7 78 38   68 28 27 48 17 58  7 78 38   44  4 75 60 47 16 19 72 32
18 59 46 31 21 71 74 43  6   36 23 64 13 57 53 74 43  6   24 65 34 37  9 77 62 49 12
55 54 14 26 67 30 42  2 79   19 72 32 62 49 12 42  2 79   55 54 14 26 67 30 42  2 79
 4/ 2/ 3/ 8/           16#    2/ 3/ 4/ 8/           17#    2/ 4/ 3/ 8/           18#
 3 80 40 76 39  8 44  4 75    3 80 40 70 33 20 50 10 63    3 80 40 76 39  8 44  4 75
70 33 20 29 25 69 24 65 34   52 15 56  5 73 45 66 35 22   52 15 56 11 61 51 60 47 16
```

```
50 10 63 18 59 46 55 54 14
52 15 56 11 61 51 60 47 16
 5 73 45 81 41  1 37  9 77
66 35 22 31 21 71 26 67 30
68 28 27 36 23 64 19 72 32
48 17 58 13 57 53 62 49 12
 7 78 38 74 43  6 42  2 79
```

```
 3/ 2/ 1/ 5/           19#
 3 76 44 52 11 60 68 36 19
70 29 24  5 81 37 48 13 62
50 18 55 66 31 26  7 74 42
80 39  4 15 61 47 28 23 72
33 25 65 73 41  9 17 57 49
10 59 54 35 21 67 78 43  2
40  8 75 56 51 16 27 64 32
20 69 34 45  1 77 58 53 12
63 46 14 22 71 30 38  6 79
```

```
 4/ 2/ 1/ 6/           20#
 3 76 44 80 39  4 40  8 75
70 29 24 33 25 65 20 69 34
50 18 55 10 59 54 63 46 14
52 11 60 15 61 47 56 51 16
 5 81 37 73 41  9 45  1 77
66 31 26 35 21 67 22 71 30
68 36 19 28 23 72 27 64 32
48 13 62 17 57 49 58 53 12
 7 74 42 78 43  2 38  6 79
```

```
 2/ 3/ 1/ 5/           21#
 3 76 44 70 29 24 50 18 55
52 11 60  5 81 37 66 31 26
68 36 19 48 13 62  7 74 42
80 39  4 33 25 65 10 59 54
15 61 47 73 41  9 35 21 67
28 23 72 17 57 49 78 43  2
40  8 75 20 69 34 63 46 14
56 51 16 45  1 77 22 71 30
27 64 32 58 53 12 38  6 79
```

```
 2/ 4/ 1/ 6/           22#
 3 76 44 80 39  4 40  8 75
52 11 60 15 61 47 56 51 16
68 36 19 28 23 72 27 64 32
70 29 24 33 25 65 20 69 34
 5 81 37 73 41  9 45  1 77
48 13 62 17 57 49 58 53 12
50 18 55 10 59 54 63 46 14
66 31 26 35 21 67 22 71 30
 7 74 42 78 43  2 38  6 79
```

```
 2/ 1/ 3/ 5/           23#
 3 70 50 76 29 18 44 24 55
52  5 66 11 81 31 60 37 26
68 48  7 36 13 74 19 62 42
80 33 10 39 25 59  4 65 54
15 73 35 61 41 21 47  9 67
28 17 78 23 57 43 72 49  2
40 20 63  8 69 46 75 34 14
56 45 22 51  1 71 16 77 30
27 58 38 64 53  6 32 12 79
```

```
 2/ 1/ 4/ 6/           24#
 3 70 50 80 33 10 40 20 63
52  5 66 15 73 35 56 45 22
68 48  7 28 17 78 27 58 38
76 29 18 39 25 59  8 69 46
11 81 31 61 41 21 51  1 71
36 13 74 23 57 43 64 53  6
44 24 55  4 65 54 75 34 14
60 37 26 47  9 67 16 77 30
19 62 42 72 49  2 32 12 79
```

```
 3/ 4/ 8/ 2/           25#
 7 78 38 48 17 58 68 28 27
74 43  6 13 57 53 36 23 64
42  2 79 62 49 12 19 72 32
66 35 22  5 73 45 52 15 56
31 21 71 81 41  1 11 61 51
26 67 30 37  9 77 60 47 16
50 10 63 70 33 20  3 80 40
18 59 46 29 25 69 76 39  8
55 54 14 24 65 34 44  4 75
```

```
 4/ 3/ 8/ 2/           26#
 7 78 38 66 35 22 50 10 63
74 43  6 31 21 71 18 59 46
42  2 79 26 67 30 55 54 14
48 17 58  5 73 45 70 33 20
13 57 53 81 41  1 29 25 69
62 49 12 37  9 77 24 65 34
68 28 27 52 15 56  3 80 40
36 23 64 11 61 51 76 39  8
19 72 32 60 47 16 44  4 75
```

```
 3/ 2/ 8/ 4/           27#
 7 78 38 48 17 58 68 28 27
66 35 22  5 73 45 52 15 56
50 10 63 70 33 20  3 80 40
74 43  6 13 57 53 36 23 64
31 21 71 81 41  1 11 61 51
18 59 46 29 25 69 76 39  8
42  2 79 62 49 12 19 72 32
26 67 30 37  9 77 60 47 16
55 54 14 24 65 34 44  4 75
```

```
 2/ 3/ 8/ 4/           28#
 7 78 38 66 35 22 50 10 63
48 17 58  5 73 45 70 33 20
68 28 27 52 15 56  3 80 40
74 43  6 31 21 71 18 59 46
13 57 53 81 41  1 29 25 69
36 23 64 11 61 51 76 39  8
42  2 79 26 67 30 55 54 14
62 49 12 37  9 77 24 65 34
19 72 32 60 47 16 44  4 75
```

```
 3/ 2/ 5/ 1/           29#
 7 74 42 48 13 62 68 36 19
66 31 26  5 81 37 52 11 60
50 18 55 70 29 24  3 76 44
78 43  2 17 57 49 28 23 72
35 21 67 73 41  9 15 61 47
10 59 54 33 25 65 80 39  4
38  6 79 58 53 12 27 64 32
22 71 30 45  1 77 56 51 16
63 46 14 20 69 34 40  8 75
```

```
 2/ 3/ 5/ 1/           30#
 7 74 42 66 31 26 50 18 55
48 13 62  5 81 37 70 29 24
68 36 19 52 11 60  3 76 44
78 43  2 35 21 67 10 59 54
17 57 49 73 41  9 33 25 65
28 23 72 15 61 47 80 39  4
38  6 79 22 71 30 63 46 14
58 53 12 45  1 77 20 69 34
27 64 32 56 51 16 40  8 75
```

```
 2/ 1/ 5/ 3/           31#
 7 66 50 74 31 18 42 26 55
48  5 70 13 81 29 62 37 24
68 52  3 36 11 76 19 60 44
78 35 10 43 21 59  2 67 54
17 73 33 57 41 25 49  9 65
28 15 80 23 61 39 72 47  4
38 22 63  6 71 46 79 30 14
58 45 20 53  1 69 12 77 34
27 56 40 64 51  8 32 16 75
```

```
 2/ 1/ 6/ 4/           32#
 7 66 50 78 35 10 38 22 63
48  5 70 17 73 33 58 45 20
68 52  3 28 15 80 27 56 40
74 31 18 43 21 59  6 71 46
13 81 29 57 41 25 53  1 69
36 11 76 23 61 39 64 51  8
42 26 55  2 67 54 79 30 14
62 37 24 49  9 65 12 77 34
19 60 44 72 47  4 32 16 75
```

```
 3/ 4/ 8/ 7/           33#
 9 76 38 46 17 60 68 30 25
74 45  4 15 55 53 34 23 66
40  2 81 62 51 10 21 70 32
64 35 24  5 75 43 54 13 56
33 19 71 79 41  3 11 63 49
26 69 28 39  7 77 58 47 18
50 12 61 72 31 20  1 80 42
16 59 48 29 27 67 78 37  8
57 52 14 22 65 36 44  6 73
```

```
 4/ 3/ 8/ 7/           34#
 9 76 38 64 35 24 50 12 61
74 45  4 33 19 71 16 59 48
40  2 81 26 69 28 57 52 14
46 17 60  5 75 43 72 31 20
15 55 53 79 41  3 29 27 67
```

```
 3/ 2/ 8/ 5/           35#
 9 76 38 46 17 60 68 30 25
64 35 24  5 75 43 54 13 56
50 12 61 72 31 20  1 80 42
74 45  4 15 55 53 34 23 66
33 19 71 79 41  3 11 63 49
```

```
 2/ 3/ 8/ 5/           36#
 9 76 38 64 35 24 50 12 61
46 17 60  5 75 43 72 31 20
68 30 25 54 13 56  1 80 42
74 45  4 33 19 71 16 59 48
15 55 53 79 41  3 29 27 67
```

```
 62 51 10 39  7 77 22 65 36    16 59 48 29 27 67 78 37  8    34 23 66 11 63 49 78 37  8
 68 30 25 54 13 56  1 80 42    40  2 81 62 51 10 21 70 32    40  2 81 26 69 28 57 52 14
 34 23 66 11 63 49 78 37  8    26 69 28 39  7 77 58 47 18    62 51 10 39  7 77 22 65 36
 21 70 32 58 47 18 44  6 73    57 52 14 22 65 36 44  6 73    21 70 32 58 47 18 44  6 73
  3/ 2/ 5/ 8/           37#     2/ 3/ 5/ 8/           38#     2/ 1/ 5/ 6/           39#
 9 74 40 46 15 62 68 34 21     9 74 40 64 33 26 50 16 57     9 64 50 74 33 16 40 26 57
64 33 26  5 79 39 54 11 58    46 15 62  5 79 39 72 29 22    46  5 72 15 79 29 62 39 22
50 16 57 72 29 22  1 78 44    68 34 21 54 11 58  1 78 44    68 54  1 34 11 78 21 58 44
76 45  2 17 55 51 30 23 70    76 45  2 35 19 69 12 59 52    76 35 12 45 19 59  2 69 52
35 19 69 75 41  7 13 63 47    17 55 51 75 41  7 31 27 65    17 75 31 55 41 27 51  7 65
12 59 52 31 27 65 80 37  6    30 23 70 13 63 47 80 37  6    30 13 80 23 63 37 70 47  6
38  4 81 60 53 10 25 66 32    38  4 81 24 71 28 61 48 14    38 24 61  4 71 48 81 28 14
24 71 28 43  3 77 56 49 18    60 53 10 43  3 77 20 67 36    60 43 20 53  3 67 10 77 36
61 48 14 20 67 36 42  8 73    25 66 32 56 49 18 42  8 73    25 56 42 66 49  8 32 18 73
  2/ 1/ 6/ 5/           40#     3/ 8/ 4/ 2/           41#     3/ 8/ 2/ 4/           42#
 9 64 50 76 35 12 38 24 61    19 72 32 36 23 64 68 28 27    19 72 32 36 23 64 68 28 27
46  5 72 17 75 31 60 43 20    62 49 12 13 57 53 48 17 58    60 47 16 11 61 51 52 15 56
68 54  1 30 13 80 25 56 42    42  2 79 74 43  6  7 78 38    44  4 75 76 39  8  3 80 40
74 33 16 45 19 59  4 71 48    60 47 16 11 61 51 52 15 56    62 49 12 13 57 53 48 17 58
15 79 29 55 41 27 53  3 67    37  9 77 81 41  1  5 73 45    37  9 77 81 41  1  5 73 45
34 11 78 23 63 37 66 49  8    26 67 30 31 21 71 66 35 22    24 65 34 29 25 69 70 33 20
40 26 57  2 69 52 81 28 14    44  4 75 76 39  8  3 80 40    42  2 79 74 43  6  7 78 38
62 39 22 51  7 65 10 77 36    24 65 34 29 25 69 70 33 20    26 67 30 31 21 71 66 35 22
21 58 44 70 47  6 32 18 73    55 54 14 18 59 46 50 10 63    55 54 14 18 59 46 50 10 63
  2/ 5/ 3/ 1/           43#     2/ 5/ 1/ 3/           44#     3/ 8/ 4/ 7/           45#
19 62 42 60 37 26 44 24 55    19 60 44 62 37 24 42 26 55    21 70 32 34 23 66 68 30 25
36 13 74 11 81 31 76 29 18    36 11 76 13 81 29 74 31 18    62 51 10 15 55 53 46 17 60
68 48  7 52  5 66  3 70 50    68 52  3 48  5 70  7 66 50    40  2 81 74 45  4  9 76 38
72 49  2 47  9 67  4 65 54    72 47  4 49  9 65  2 67 54    58 47 18 11 63 49 54 13 56
23 57 43 61 41 21 39 25 59    23 61 39 57 41 25 43 21 59    39  7 77 79 41  3  5 75 43
28 17 78 15 73 35 80 33 10    28 15 80 17 73 33 78 35 10    26 69 28 33 19 71 64 35 24
32 12 79 16 77 30 75 34 14    32 16 75 12 77 34 79 30 14    44  6 73 78 37  8  1 80 42
64 53  6 51  1 71  8 69 46    64 51  8 53  1 69  6 71 46    22 65 36 29 27 67 72 31 20
27 58 38 56 45 22 40 20 63    27 56 40 58 45 20 38 22 63    57 52 14 16 59 48 50 12 61
  3/ 8/ 2/ 5/           46#     2/ 5/ 3/ 8/           47#     2/ 5/ 1/ 6/           48#
21 70 32 34 23 66 68 30 25    21 62 40 58 39 26 44 22 57    21 58 44 62 39 22 40 26 57
58 47 18 11 63 49 54 13 56    34 15 74 11 79 33 78 29 16    34 11 78 15 79 29 74 33 16
44  6 73 78 37  8  1 80 42    68 46  9 54  5 64  1 72 50    68 54  1 46  5 72  9 64 50
62 51 10 15 55 53 46 17 60    70 51  2 47  7 69  6 65 52    70 47  6 51  7 65  2 69 52
39  7 77 79 41  3  5 75 43    23 55 45 63 41 19 37 27 59    23 63 37 55 41 27 45 19 59
22 65 36 29 27 67 72 31 20    30 17 76 13 75 35 80 31 12    30 13 80 17 75 31 76 35 12
40  2 81 74 45  4  9 76 38    32 10 81 18 77 28 73 36 14    32 18 73 10 77 36 81 28 14
26 69 28 33 19 71 64 35 24    66 53  4 49  3 71  8 67 48    66 49  8 53  3 67  4 71 48
57 52 14 16 59 48 50 12 61    25 60 38 56 43 24 42 20 61    25 56 42 60 43 20 38 24 61

 [Solution Counts(n1=1/Total) = 12/48]  OK!

** Recompiled and Listed by Kanji Setsuda on
    Mar.10, 2015 with MacOSX 10.10.2 & Xcode 6.2 **
```

I would say this is the most rare set of precious solutions of Magic Squares of order 9. You cannot make it by any other methods without using PNS of Base 3.

6. Can we make it without using Positional Number System of Base 3?

I tried to make the same type of magic squares of order 9 by such an ordinary program as usual without using PNS of Base 3.

The next list demonstrates the result of my recent research. I defined the equal sum of every line, but did not define any number pattern of all.

```
*** Multiple Type of Self-Complementary Magic Squares ***
**  of Order 9: Down-Converted from 4-D ECO of Order 3 **
```

```
                          1/  /D3i   27*         9*         3*         1*
 1 80 42 78 37  8 44  6 73  021210102 021210102 021102210 012201120
72 31 20 29 27 67 22 65 36  210102021 102021210 210021102 201120012
50 12 61 16 59 48 57 52 14  102021210 210102021 102210021 120012201
54 13 56 11 63 49 58 47 18  102021210 210102021 210021102 201120012
 5 75 43 79 41  3 39  7 77  021210102 021210102 102210021 120012201
64 35 24 33 19 71 26 69 28  210102021 102021210 021102210 012201120
68 30 25 34 23 66 21 70 32  210102021 102021210 102210021 120012201
46 17 60 15 55 53 62 51 10  102021210 210102021 021102210 012201120
 9 76 38 74 45  4 40  2 81  021210102 021210102 210021102 201120012
                         13/  /D3i   27*         9*         3*         1*
 1 78 44 80 37  6 42  8 73  021210102 021210102 012201120 021102210
72 29 22 31 27 65 20 67 36  210102021 102021210 201120012 210021102
50 16 57 12 59 52 61 48 14  102021210 210102021 120012201 102210021
54 11 58 13 63 47 56 49 18  102021210 210102021 201120012 210021102
 5 79 39 75 41  7 43  3 77  021210102 021210102 120012201 102210021
64 33 26 35 19 69 24 71 28  210102021 102021210 012201120 021102210
68 34 21 30 23 70 25 66 32  210102021 102021210 120012201 102210021
46 15 62 17 55 51 60 53 10  102021210 210102021 012201120 021102210
 9 74 40 76 45  2 38  4 81  021210102 021210102 201120012 210021102
                         23/  /D3i   27*         9*         3*         1*
 1 77 45 80 39  4 42  7 74  021210102 021210102 012201120 012120201
71 30 22 33 25 65 19 68 36  210102021 102021210 201120012 120201012
51 16 56 10 59 54 62 48 13  102021210 210102021 120012201 201012120
53 12 58 15 61 47 55 50 18  102021210 210102021 201120012 120201012
 6 79 38 73 41  9 44  3 76  021210102 021210102 120012201 201012120
64 32 27 35 21 67 24 70 29  210102021 102021210 012201120 012120201
69 34 20 28 23 72 26 66 31  210102021 102021210 120012201 201012120
46 14 63 17 57 49 60 52 11  102021210 210102021 012201120 012120201
 8 75 40 78 43  2 37  5 81  021210102 021210102 201120012 120201012
                         31/  /D3i   27*         9*         3*         1*
 1 72 50 80 31 12 42 20 61  021210102 012201120 021210102 021102210
54  5 64 13 75 35 56 43 24  102021210 201120012 210102021 210021102
68 46  9 30 17 76 25 60 38  210102021 120012201 102021210 102210021
78 29 16 37 27 59  8 67 48  210102021 201120012 102021210 210021102
11 79 33 63 41 19 49  3 71  021210102 120012201 021210102 102210021
34 15 74 23 55 45 66 53  4  102021210 012201120 210102021 021102210
44 22 57  6 65 52 73 36 14  102021210 120012201 210102021 102210021
58 39 26 47  7 69 18 77 28  210102021 012201120 102021210 021102210
21 62 40 70 51  2 32 10 81  021210102 201120012 021210102 210021102
                         39/  /D3i   27*         9*         3*         1*
 1 71 51 80 33 10 42 19 62  021210102 012201120 021210102 012120201
53  6 64 15 73 35 55 44 24  102021210 201120012 210102021 120201012
69 46  8 28 17 78 26 60 37  210102021 120012201 102021210 201012120
77 30 16 39 25 59  7 68 48  210102021 201120012 102021210 120201012
12 79 32 61 41 21 50  3 70  021210102 120012201 021210102 201012120
34 14 75 23 57 43 66 52  5  102021210 012201120 210102021 012120201
45 22 56  4 65 54 74 36 13  102021210 120012201 210102021 201012120
58 38 27 47  9 67 18 76 29  210102021 012201120 102021210 012120201
20 63 40 72 49  2 31 11 81  021210102 201120012 021210102 120201012
                         45/  /D3i   27*         9*         3*         1*
 1 69 53 78 35 10 44 19 60  021210102 012201120 012120201 021210102
51  8 64 17 73 33 55 42 26  102021210 201120012 120201012 210102021
71 46  6 28 15 80 24 62 37  210102021 120012201 201012120 102021210
77 34 12 43 21 59  3 68 52  210102021 201120012 120201012 102021210
16 75 32 57 41 25 50  7 66  021210102 120012201 201012120 021210102
30 14 79 23 61 39 70 48  5  102021210 012201120 012120201 210102021
45 20 58  2 67 54 76 36 11  102021210 120012201 201012120 210102021
56 40 27 49  9 65 18 74 31  210102021 012201120 012120201 102021210
22 63 38 72 47  4 29 13 81  021210102 201120012 120201012 021210102
                         49/  /D3i   27*         9*         3*         1*
 1 54 68 53 67  3 69  2 52  012120201 021210102 021210102 021102210
51 65  7 64  9 50  8 49 66  120201012 210102021 102021210 210021102
```

```
71  4 48  6 47 70 46 72  5  201012120 102021210 210102021 102210021
45 59 19 58 21 44 20 43 60  120201012 102021210 210102021 210021102
56 25 42 27 41 55 40 57 26  201012120 021210102 021210102 102210021
22 39 62 38 61 24 63 23 37  012120201 210102021 102021210 021102210
77 10 36 12 35 76 34 78 11  201012120 210102021 102021210 102210021
16 33 74 32 73 18 75 17 31  012120201 102021210 210102021 021102210
30 80 13 79 15 29 14 28 81  120201012 021210102 021210102 210021102
                      55/  /D3i  27*        9*        3*        1*
 1 53 69 54 67  2 68  3 52  012120201 021210102 021210102 012201120
51 64  8 65  9 49  7 50 66  120201012 210102021 102021210 201120012
71  6 46  4 47 72 48 70  5  201012120 102021210 210102021 120012201
45 58 20 59 21 43 19 44 60  120201012 102021210 210102021 201120012
56 27 40 25 41 57 42 55 26  201012120 021210102 021210102 120012201
22 38 63 39 61 23 62 24 37  012120201 210102021 102021210 012201120
77 12 34 10 35 78 36 76 11  201012120 210102021 102021210 120012201
16 32 75 33 73 17 74 18 31  012120201 102021210 210102021 012201120
30 79 14 80 15 28 13 29 81  120201012 021210102 021210102 201120012
                      59/  /D3i  27*        9*        3*        1*
 1 51 71 54 65  4 68  7 48  012120201 021210102 012201120 021210102
45 56 22 59 25 39 19 42 62  120201012 102021210 201120012 210102021
77 16 30 10 33 80 36 74 13  201012120 210102021 120012201 102021210
53 64  6 67  9 47  3 50 70  120201012 210102021 201120012 102021210
58 27 38 21 41 61 44 55 24  201012120 021210102 120012201 021210102
12 32 79 35 73 15 76 18 29  012120201 102021210 012201120 210102021
69  8 46  2 49 72 52 66  5  201012120 102021210 120012201 210102021
20 40 63 43 57 23 60 26 37  012120201 210102021 012201120 102021210
34 75 14 78 17 28 11 31 81  120201012 021210102 201120012 021210102
                      61/                       79/                       97/
 2 81 40 78 37  8 43  5 75    2 79 42 78 38  7 43  6 74    2 78 43 81 37  5 40  8 75
70 32 21 29 27 67 24 64 35   72 32 19 28 27 68 23 64 36   70 29 24 32 27 64 21 67 35
51 10 62 16 59 48 56 54 13   49 12 62 17 58 48 57 53 13   51 16 56 10 59 54 62 48 13
52 14 57 11 63 49 60 46 17   52 15 56 11 61 51 60 47 16   52 11 60 14 63 46 57 49 17
 6 73 44 79 41  3 38  9 76    5 73 45 81 41  1 37  9 77    6 79 38 73 41  9 44  3 76
65 36 22 33 19 71 25 68 30   66 35 22 31 21 71 26 67 30   65 33 25 36 19 68 22 71 30
69 28 26 34 23 66 20 72 31   69 29 25 34 24 65 20 70 33   69 34 20 28 23 72 26 66 31
47 18 58 15 55 53 61 50 12   46 18 59 14 55 54 63 50 10   47 15 61 18 55 50 58 53 12
 7 77 39 74 45  4 42  1 80    8 76 39 75 44  4 40  3 80    7 74 42 77 45  1 39  4 80
                     121/                      141/                      159/
 2 76 45 81 38  4 40  9 74    2 72 49 81 31 11 40 20 63    2 70 51 81 32 10 40 21 62
72 29 22 31 27 65 20 67 36   52  5 66 14 75 34 57 43 23   54  5 64 13 75 35 56 43 24
49 18 56 11 58 54 63 47 13   69 46  8 28 17 78 26 60 37   67 48  8 29 16 78 27 59 37
52 12 59 14 61 48 57 50 16   76 29 18 38 27 58  9 67 47   76 30 17 38 25 60  9 68 46
 5 79 39 75 41  7 43  3 77   12 79 32 61 41 21 50  3 70   11 79 33 63 41 19 49  3 71
66 32 25 34 21 68 23 70 30   35 15 73 24 55 44 64 53  6   36 14 73 22 57 44 65 52  6
69 35 19 28 24 71 26 64 33   45 22 56  4 65 54 74 36 13   45 23 55  4 66 53 74 34 15
46 15 62 17 55 51 60 53 10   59 39 25 48  7 68 16 77 30   58 39 26 47  7 69 18 77 28
 8 73 42 78 44  1 37  6 80   19 62 42 71 51  1 33 10 80   20 61 42 72 50  1 31 12 80
                     173/                      177/                      181/
 2 69 52 78 34 11 43 20 60    2 67 54 78 35 10 43 21 59    2 54 67 52 68  3 69  1 53
49  8 66 17 75 31 57 40 26   51  8 64 16 75 32 56 40 27   51 64  8 65  9 49  7 50 66
72 46  5 28 14 81 23 63 37   70 48  5 29 13 81 24 62 37   70  5 48  6 46 71 47 72  4
76 35 12 44 21 58  3 67 53   76 36 11 44 19 60  3 68 52   43 59 21 60 19 44 20 45 58
18 73 32 55 41 27 50  9 64   17 73 33 57 41 25 49  9 65   56 27 40 25 41 57 42 55 26
29 15 79 24 61 38 70 47  6   30 14 79 22 63 38 71 46  6   24 37 62 38 63 22 61 23 39
45 19 59  1 68 54 77 36 10   45 20 58  1 69 53 77 34 12   78 10 35 11 36 76 34 77 12
56 42 25 51  7 65 16 74 33   55 42 26 50  7 66 18 74 31   16 32 75 33 73 17 74 18 31
22 62 39 71 48  4 30 13 80   23 61 39 72 47  4 28 15 80   29 81 13 79 14 30 15 28 80
                     193/                      201/                      203/
 2 52 69 54 68  1 67  3 53    2 51 70 54 64  5 67  8 48    2 49 72 54 65  4 67  9 47
51 65  7 64  9 50  8 49 66   43 56 24 59 27 37 21 40 62   45 56 22 58 27 38 20 40 63
70  6 47  5 46 72 48 71  4   78 16 29 10 32 81 35 75 13   76 18 29 11 31 81 36 74 13
43 60 20 59 19 45 21 44 58   52 65  6 68  9 46  3 49 71   52 66  5 68  7 48  3 50 70
56 25 42 27 41 55 40 57 26   60 25 38 19 41 63 44 57 22   59 25 39 21 41 61 43 57 23
```

```
24 38 61 37 63 23 62 22 39
78 11 34 10 36 77 35 76 12
16 33 74 32 73 18 75 17 31
29 79 15 81 14 28 13 30 80
                     205/
 3 80 40 77 37  9 43  6 74
71 31 21 28 27 68 24 65 34
49 12 62 18 59 46 56 52 15
53 13 57 10 63 50 60 47 16
 4 75 44 81 41  1 38  7 78
66 35 22 32 19 72 25 69 29
67 30 26 36 23 64 20 70 33
48 17 58 14 55 54 61 51 11
 8 76 39 73 45  5 42  2 79
                     235/
 3 71 49 80 31 12 40 21 62
53  4 66 13 75 35 57 44 22
67 48  8 30 17 76 26 58 39
77 28 18 37 27 59  9 68 46
10 81 32 63 41 19 50  1 72
36 14 73 23 55 45 64 54  5
43 24 56  6 65 52 74 34 15
60 38 25 47  7 69 16 78 29
20 61 42 70 51  2 33 11 79
                     253/
 3 53 67 52 69  2 68  1 54
49 66  8 65  7 51  9 50 64
71  4 48  6 47 70 46 72  5
43 60 20 59 19 45 21 44 58
56 25 42 27 41 55 40 57 26
24 38 61 37 63 23 62 22 39
77 10 36 12 35 76 34 78 11
18 32 73 31 75 17 74 16 33
28 81 14 80 13 30 15 29 79
                     265/
 4 81 38 74 40  9 45  2 76
72 29 22 31 27 65 20 67 36
47 13 63 18 56 49 58 54 11
48 14 61 16 57 50 59 52 12
 5 79 39 75 41  7 43  3 77
70 30 23 32 25 66 21 68 34
71 28 24 33 26 64 19 69 35
46 15 62 17 55 51 60 53 10
 6 80 37 73 42  8 44  1 78
                     697/
 7 78 38 69 29 25 47 16 60
77 37  9 28 27 68 18 59 46
39  8 76 26 67 30 58 48 17
51 11 61  2 79 42 70 33 20
10 63 50 81 41  1 32 19 72
62 49 12 40  3 80 21 71 31
65 34 24 52 15 56  6 74 43
36 23 64 14 55 54 73 45  5
22 66 35 57 53 13 44  4 75
                     905/
10 81 32 62 40 21 51  2 70
78 29 16 37 27 59  8 67 48
35 13 75 24 56 43 64 54  5
36 14 73 22 57 44 65 52  6
11 79 33 63 41 19 49  3 71
76 30 17 38 25 60  9 68 46
77 28 18 39 26 58  7 69 47
34 15 74 23 55 45 66 53  4
12 80 31 61 42 20 50  1 72

11 33 79 36 73 14 76 17 30
69  7 47  1 50 72 53 66  4
20 42 61 45 55 23 58 26 39
34 74 15 77 18 28 12 31 80
                     217/
 3 77 43 80 37  6 40  9 74
71 28 24 31 27 65 21 68 34
49 18 56 12 59 52 62 46 15
53 10 60 13 63 47 57 50 16
 4 81 38 75 41  7 44  1 78
66 32 25 35 19 69 22 72 29
67 36 20 30 23 70 26 64 33
48 14 61 17 55 51 58 54 11
 8 73 42 76 45  2 39  5 79
                     243/
 3 70 50 80 33 10 40 20 63
52  5 66 15 73 35 56 45 22
68 48  7 28 17 78 27 58 38
76 29 18 39 25 59  8 69 46
11 81 31 61 41 21 51  1 71
36 13 74 23 57 43 64 53  6
44 24 55  4 65 54 75 34 14
60 37 26 47  9 67 16 77 30
19 62 42 72 49  2 32 12 79
                     259/
 3 52 68 53 69  1 67  2 54
49 65  9 66  7 50  8 51 64
71  6 46  4 47 72 48 70  5
43 59 21 60 19 44 20 45 58
56 27 40 25 41 57 42 55 26
24 37 62 38 63 22 61 23 39
77 12 34 10 35 78 36 76 11
18 31 74 32 75 16 73 17 33
28 80 15 81 13 29 14 30 79
                     405/
 5 81 37 70 29 24 48 13 62
75 40  8 32 27 64 16 56 51
43  2 78 21 67 35 59 54 10
46 14 63  6 79 38 71 30 22
17 57 49 73 41  9 33 25 65
60 52 11 44  3 76 19 68 36
72 28 23 47 15 61  4 80 39
31 26 66 18 55 50 74 42  7
20 69 34 58 53 12 45  1 77
                     745/
 8 78 37 69 28 26 46 17 60
76 38  9 29 27 67 18 58 47
39  7 77 25 68 30 59 48 16
49 11 63  2 81 40 72 31 20
12 61 50 79 41  3 32 21 70
62 51 10 42  1 80 19 71 33
66 34 23 52 14 57  5 75 43
35 24 64 15 55 53 73 44  6
22 65 36 56 54 13 45  4 74
                    1029/
11 81 31 60 37 26 52  5 66
79 32 12 38 27 58  6 64 53
33 10 80 25 59 39 65 54  4
34 14 75 20 63 40 69 46  8
15 73 35 61 41 21 47  9 67
74 36 13 42 19 62  7 68 48
78 28 17 43 23 57  2 72 49
29 18 76 24 55 44 70 50  3
16 77 30 56 45 22 51  1 71

12 32 79 34 75 14 77 16 30
69  8 46  1 51 71 53 64  6
19 42 62 44 55 24 60 26 37
35 73 15 78 17 28 10 33 80
                     227/
 3 76 44 80 39  4 40  8 75
70 29 24 33 25 65 20 69 34
50 18 55 10 59 54 63 46 14
52 11 60 15 61 47 56 51 16
 5 81 37 73 41  9 45  1 77
66 31 26 35 21 67 22 71 30
68 36 19 28 23 72 27 64 32
48 13 62 17 57 49 58 53 12
 7 74 42 78 43  2 38  6 79
                     249/
 3 67 53 77 36 10 43 20 60
49  8 66 18 73 32 56 42 25
71 48  4 28 14 81 24 61 38
76 35 12 45 19 59  2 69 52
17 75 31 55 41 27 51  7 65
30 13 80 23 63 37 70 47  6
44 21 58  1 68 54 78 34 11
57 40 26 50  9 64 16 74 33
22 62 39 72 46  5 29 15 79
                     263/
 3 49 71 53 66  4 67  8 48
43 56 24 60 25 38 20 42 61
77 18 28 10 32 81 36 73 14
52 65  6 69  7 47  2 51 70
59 27 37 19 41 63 45 55 23
12 31 80 35 75 13 76 17 30
68  9 46  1 50 72 54 64  5
21 40 62 44 57 22 58 26 39
34 74 15 78 16 29 11 33 79
                     581/
 6 80 37 71 28 24 46 15 62
74 40  9 31 27 65 18 56 49
43  3 77 21 68 34 59 52 12
47 13 63  4 81 38 72 29 22
16 57 50 75 41  7 32 25 66
60 53 10 44  1 78 19 69 35
70 30 23 48 14 61  5 79 39
33 26 64 17 55 51 73 42  8
20 67 36 58 54 11 45  2 76
                     857/
 9 77 37 67 30 26 47 16 60
76 39  8 29 25 69 18 59 46
38  7 78 27 68 28 58 48 17
49 12 62  2 79 42 72 32 19
11 61 51 81 41  1 31 21 71
63 50 10 40  3 80 20 70 33
65 34 24 54 14 55  4 75 44
36 23 64 13 57 53 74 43  6
22 66 35 56 52 15 45  5 73
                    1189/
12 80 31 59 37 27 52  6 65
62 40 21 46  9 68 15 74 34
49  3 71 18 77 28 56 43 24
53  4 66 10 81 32 60 38 25
13 75 35 63 41 19 47  7 69
57 44 22 50  1 72 16 78 29
58 39 26 54  5 64 11 79 33
48  8 67 14 73 36 61 42 20
17 76 30 55 45 23 51  2 70
```

1289/

13 81 29 56 40 27 54 2 67
75 32 16 43 21 59 5 70 48
35 10 78 24 62 37 64 51 8
36 11 76 22 63 38 65 49 9
14 79 30 57 41 25 52 3 68
73 33 17 44 19 60 6 71 46
74 31 18 45 20 58 4 72 47
34 12 77 23 61 39 66 50 7
15 80 28 55 42 26 53 1 69

1433/

14 81 28 46 5 72 63 37 23
61 38 24 15 79 29 47 6 70
48 4 71 62 39 22 13 80 30
57 40 26 17 75 31 49 8 66
50 9 64 55 41 27 18 73 32
16 74 33 51 7 65 56 42 25
52 2 69 60 43 20 11 78 34
12 76 35 53 3 67 58 44 21
59 45 19 10 77 36 54 1 68

1505/

15 80 28 47 4 72 61 39 23
62 37 24 13 81 29 48 5 70
46 6 71 63 38 22 14 79 30
56 40 27 16 75 32 51 8 64
49 9 65 57 41 25 17 73 33
18 74 31 50 7 66 55 42 26
52 3 68 60 44 19 11 76 36
12 77 34 53 1 69 58 45 20
59 43 21 10 78 35 54 2 67

1601/

16 78 29 51 2 70 56 43 24
60 38 25 11 79 33 52 6 65
47 7 69 61 42 20 15 74 34
59 37 27 10 81 32 54 5 64
46 9 68 63 41 19 14 73 36
18 77 28 50 1 72 55 45 23
48 8 67 62 40 21 13 75 35
17 76 30 49 3 71 57 44 22
58 39 26 12 80 31 53 4 66

1669/

17 78 28 49 2 72 57 43 23
60 37 26 11 81 31 52 5 66
46 8 69 63 40 20 14 75 34
58 38 27 12 79 32 53 6 64
47 9 67 61 41 21 15 73 35
18 76 29 50 3 70 55 44 24
48 7 68 62 42 19 13 74 36
16 77 30 51 1 71 56 45 22
59 39 25 10 80 33 54 4 65

1749/

18 76 29 49 2 72 56 45 22
59 39 25 12 79 32 52 5 66
46 8 69 62 42 19 15 73 35
58 38 27 11 81 31 54 4 65
48 7 68 61 41 21 14 75 34
17 78 28 51 1 71 55 44 24
47 9 67 63 40 20 13 74 36
16 77 30 50 3 70 57 43 23
60 37 26 10 80 33 53 6 64

1793/

19 72 32 36 23 64 68 28 27
62 49 12 13 57 53 48 17 58
42 2 79 74 43 6 7 78 38
60 47 16 11 61 51 52 15 56
37 9 77 81 41 1 5 73 45
26 67 30 31 21 71 66 35 22
44 4 75 76 39 8 3 80 40
24 65 34 29 25 69 70 33 20
55 54 14 18 59 46 50 10 63

1817/

20 72 31 36 22 65 67 29 27
61 50 12 14 57 52 48 16 59
42 1 80 73 44 6 8 78 37
58 47 18 11 63 49 54 13 56
39 7 77 79 41 3 5 75 43
26 69 28 33 19 71 64 35 24
45 4 74 76 38 9 2 81 40
23 66 34 30 25 68 70 32 21
55 53 15 17 60 46 51 10 62

1873/

21 71 31 35 22 66 67 30 26
62 49 12 13 57 53 48 17 58
40 3 80 75 44 4 8 76 39
59 46 18 10 63 50 54 14 55
37 9 77 81 41 1 5 73 45
27 68 28 32 19 72 64 36 23
43 6 74 78 38 7 2 79 42
24 65 34 29 25 69 70 33 20
56 52 15 16 60 47 51 11 61

1897/

22 72 29 36 20 67 65 31 27
57 50 16 14 61 48 52 12 59
44 1 78 73 42 8 6 80 37
56 49 18 13 63 47 54 11 58
43 3 77 75 41 7 5 79 39
24 71 28 35 19 69 64 33 26
45 2 76 74 40 9 4 81 38
23 70 30 34 21 68 66 32 25
55 51 17 15 62 46 53 10 60

1949/

23 72 28 34 20 69 66 31 26
57 49 17 14 63 46 52 11 60
43 2 78 75 40 8 5 81 37
55 50 18 15 61 47 53 12 58
44 3 76 73 41 9 6 79 38
24 70 29 35 21 67 64 32 27
45 1 77 74 42 7 4 80 39
22 71 30 36 19 68 65 33 25
56 51 16 13 62 48 54 10 59

2013/

24 70 29 34 20 69 65 33 25
56 51 16 15 61 47 52 11 60
43 2 78 74 42 7 6 79 38
55 50 18 14 63 46 54 10 59
45 1 77 73 41 9 5 81 37
23 72 28 36 19 68 64 32 27
44 3 76 75 40 8 4 80 39
22 71 30 35 21 67 66 31 26
57 49 17 13 62 48 53 12 58

2041/

25 39 59 33 80 10 65 4 54
30 77 16 71 1 51 22 45 56
68 7 48 19 42 62 36 74 13
29 76 18 70 3 50 24 44 55
67 9 47 21 41 61 35 73 15
27 38 58 32 79 12 64 6 53
69 8 46 20 40 63 34 75 14
26 37 60 31 81 11 66 5 52
28 78 17 72 2 49 23 43 57

2053/

26 39 58 31 80 12 66 4 53
30 76 17 71 3 49 22 44 57
67 8 48 21 40 62 35 75 13
28 77 18 72 1 50 23 45 55
68 9 46 19 41 63 36 73 14
27 37 59 32 81 10 64 5 54
69 7 47 20 42 61 34 74 15
25 38 60 33 79 11 65 6 52
29 78 16 70 2 51 24 43 56

2077/

27 37 59 31 80 12 65 6 52
29 78 16 72 1 50 22 44 57
67 8 48 20 42 61 36 73 14
28 77 18 71 3 49 24 43 56
69 7 47 19 41 63 35 75 13
26 39 58 33 79 11 64 5 54
68 9 46 21 40 62 34 74 15
25 38 60 32 81 10 66 4 53
30 76 17 70 2 51 23 45 55

2089/

28 81 14 26 40 57 69 2 52
78 11 34 37 63 23 8 49 66
17 31 75 60 20 43 46 72 5
18 32 73 58 21 44 47 70 6
29 79 15 27 41 55 67 3 53
76 12 35 38 61 24 9 50 64
77 10 36 39 62 22 7 51 65
16 33 74 59 19 45 48 71 4
30 80 13 25 42 56 68 1 54

2185/

29 81 13 24 37 62 70 5 48
79 14 30 38 63 22 6 46 71
15 28 80 61 23 39 47 72 4
16 32 75 56 27 40 51 64 8
33 73 17 25 41 57 65 9 49
74 18 31 42 55 26 7 50 66
78 10 35 43 59 21 2 54 67
11 36 76 60 19 44 52 68 3
34 77 12 20 45 58 69 1 53

2305/

30 80 13 23 37 63 70 6 47
71 4 48 28 81 14 24 38 61
22 39 62 72 5 46 29 79 15
26 40 57 64 9 50 33 74 16
31 75 17 27 41 55 65 7 51
66 8 49 32 73 18 25 42 56
67 3 53 36 77 10 20 43 60
21 44 58 68 1 54 34 78 11
35 76 12 19 45 59 69 2 52

2377/

31 81 11 20 40 63 72 2 49
75 14 34 43 57 23 5 52 66

2481/

32 81 10 16 29 78 75 13 35
73 14 36 33 79 11 17 30 76

2529/

33 80 10 17 28 78 73 15 35
74 13 36 31 81 11 18 29 76

```
 17 28 78 60 26 37 46 69  8    18 28 77 74 15 34 31 80 12    16 30 77 75 14 34 32 79 12
 18 29 76 58 27 38 47 67  9    21 40 62 59 27 37 43 56 24    20 40 63 58 27 38 45 56 22
 32 79 12 21 41 61 70  3 50    44 57 22 19 41 63 60 25 38    43 57 23 21 41 61 59 25 39
 73 15 35 44 55 24  6 53 64    58 26 39 45 55 23 20 42 61    60 26 37 44 55 24 19 42 62
 74 13 36 45 56 22  4 54 65    70  2 51 48 67  8  5 54 64    70  3 50 48 68  7  5 52 66
 16 30 77 59 25 39 48 68  7     6 52 65 71  3 49 46 68  9     6 53 64 71  1 51 46 69  8
 33 80 10 19 42 62 71  1 51    47 69  7  4 53 66 72  1 50    47 67  9  4 54 65 72  2 49
                   2585/                         2625/                         2665/
 34 78 11 12 35 76 77 10 36    35 78 10 12 34 77 76 11 36    36 67 20 11 54 58 76  2 45
 75 17 31 32 73 18 16 33 74    73 17 33 32 75 16 18 31 74    64 26 33 51 55 17  8 42 73
 14 28 81 79 15 29 30 80 13    15 28 80 79 14 30 29 81 13    23 30 70 61 14 48 39 79  5
 20 43 60 58 21 44 45 59 19    19 44 60 59 21 43 45 58 20    10 53 60 78  1 44 35 69 19
 40 57 26 27 41 55 56 25 42    42 55 26 25 41 57 56 27 40    50 57 16  7 41 75 66 25 32
 63 23 37 38 61 24 22 39 62    62 24 37 39 61 23 22 38 63    63 13 47 38 81  4 22 29 72
 69  2 52 53 67  3  1 54 68    69  1 53 52 68  3  2 54 67    77  3 43 34 68 21 12 52 59
  8 49 66 64  9 50 51 65  7     8 51 64 66  7 50 49 65  9     9 40 74 65 27 31 49 56 18
 46 72  5  6 47 70 71  4 48    46 71  6  5 48 70 72  4 47    37 80  6 24 28 71 62 15 46
                   2681/                         2745/                         2769/
 37 81  5  8 40 75 78  2 43    38 81  4  6 37 80 79  5 39    39 77  7  4 45 74 80  1 42
 63 14 46 49 57 17 11 52 60    61 14 48 47 63 13 15 46 62    58 18 47 53 55 15 12 50 61
 23 28 72 66 26 31 34 69 20    24 28 71 70 23 30 29 72 22    26 28 69 66 23 34 31 72 20
 24 29 70 64 27 32 35 67 21    25 32 66 65 27 31 33 64 26    22 36 65 71 19 33 30 68 25
 38 79  6  9 41 73 76  3 44    42 73  8  7 41 75 74  9 40    44 73  6  3 41 79 76  9 38
 61 15 47 50 55 18 12 53 58    56 18 49 51 55 17 16 50 57    57 14 52 49 63 11 17 46 60
 62 13 48 51 56 16 10 54 59    60 10 53 52 59 12 11 54 58    62 10 51 48 59 16 13 54 56
 22 30 71 65 25 33 36 68 19    20 36 67 69 19 35 34 68 21    21 32 70 67 27 29 35 64 24
 39 80  4  7 42 74 77  1 45    43 77  3  2 45 76 78  1 44    40 81  2  8 37 78 75  5 43
[Count = 2784]  OK!
```

At a glance it seems we could make almost all beautiful Euler Squares 9x9. But examine those several samples above with your careful watch, and you will find some certainly contain the number pattern 1+1+1+1+1+1+1+1+1 in any primary diagonal in any layer. I could not avoid it at all.

The total count of solutions is 2784. The former 48 solution set is certainly included in this larger solution set.

You can compose this smaller solution set even by our New Euler's Method. You may well eliminate all dictations about pd1 and pb9 from the previous program list.

New Euler's Method could pass every hard examination and could build almost all objects we have wanted to. I think it could have successfully demonstrated its power and its wide range of applicability. How wonderful it is!

But why is it so? Why could we make almost everything by this method?

I would say, almost every type of Pan-magic Squares and Cubes of any order must have the beautiful structure in common, what we call "Complete Euler Squares".

It is the most amazing thing of all, don't you think so?

Of course, there are many 'Non-Euler Type', maybe far more than 'Euler ones', and we cannot make any one of them by this method at all! But, it bothers us no longer.

Any 'Non-Euler type' makes only a 'background', but the 'Euler type' certainly plays the 'leading figure' in the world of magic things.

It implies there is certainly a 'noble family'.

```
(The Original was written on Sep.30, 2003 with MacOS9 & MWCW;
  Retyped and recalculated on May 17, 2006 with MacOSX & Xcode 2.2;
    Recompiled and listed on Mar.10, 2015 with MacOSX 10.10.2 & Xcode 6.2)
-------------------------------------------------------------------------------------------
      Kanji Setsuda: E-Mail Address <jag12001@nifty.com>
```