

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

Section 3-4: The Most Fundamental Solution and Transformation Process of 'C&C' Magic Squares of order 8

#1. In the former sections we studied about 6-dimensional extra-cubic magic forms of order 2 and their decompositions. As a result we have obtained 15 fundamental solutions of 'Composite and Complete' Magic Cubes of order 4 and 10 fundamental solutions of 'C&C' Magic Squares of order 8 by dimension-converting from the origin. In the last section we could reconstruct all 90 solutions of C&C Magic Squares of order 8 from the Fundamental 10 by our original transformation process.

But I am not yet satisfied with it. My intellectual imagination is growing wilder. Couldn't we find the only one 'Most Fundamental Solution' of all? Couldn't we invent any transformation process for the only one Fundamental making into all 10 solutions of 'C&C' Magic Squares 8x8 just as we have done in C&C Magic Cubes 4x4x4?

#2. The Fundamental Ten are originally taken out of the 6-dimensional extra-cubic magic forms of order 2 by dimensional down-conversion. But this origin has essentially only one solution in itself. It comes to appear with 10 faces in the 2-dimensional world. I would say F10 are all children of the same Origin in other words. They must have some common memory of the 'Parent', or of 'brothers and sisters'. I want to find it.

First of all let's make any possible pair of solutions among F10, and collect only similar ones as many as possible. We want 'similar' pairs to have the same common 32(half of all) elements at least in each.

The next list shows 12 'similar pairs' we have wanted.

*** Find Similar Pairs among The Fundamental Ten ***
*** of 'Composite & Complete' Magic Squares 8x8 ***

```
[ 1]                2/O  <-- Unmatched -->    1/F
  *  *  *  * 22 44 17 47    *  *  *  * 14 52  9 55
  *  *  *  * 41 23 46 20    *  *  *  * 49 15 54 12
11 53 16 50  *  *  *  *    19 45 24 42  *  *  *  *
56 10 51 13  *  *  *  *    48 18 43 21  *  *  *  *
43 21 48 18  *  *  *  *    51 13 56 10  *  *  *  *
24 42 19 45  *  *  *  *    16 50 11 53  *  *  *  *
  *  *  *  * 54 12 49 15    *  *  *  * 46 20 41 23
  *  *  *  *  9 55 14 52    *  *  *  * 17 47 22 44

[ 2]                7/O  <-- Unmatched -->    1/F
  *  *  4 62 12 54  *  *    *  *  6 60 14 52  *  *
60  6  *  *  *  * 52 14    62  4  *  *  *  * 54 12
21 43  *  *  *  * 29 35    19 45  *  *  *  * 27 37
  *  * 45 19 37 27  *  *    *  * 43 21 35 29  *  *
53 11  *  *  *  * 61  3    51 13  *  *  *  * 59  5
  *  * 13 51  5 59  *  *    *  * 11 53  3 61  *  *
  *  * 36 30 44 22  *  *    *  * 38 28 46 20  *  *
28 38  *  *  *  * 20 46    30 36  *  *  *  * 22 44
```



```
[ 9]                6/O  <-- Unmatched -->    5/F
 * * 18 48 50 16 * *   * * 10 56 42 24 * *
 * * 45 19 13 51 * *   * * 53 11 21 43 * *
 * * 24 42 56 10 * *   * * 16 50 48 18 * *
 * * 43 21 11 53 * *   * * 51 13 19 45 * *
15 49 * * * * 47 17   23 41 * * * * 55 9
52 14 * * * * 20 46   44 22 * * * * 12 54
 9 55 * * * * 41 23   17 47 * * * * 49 15
54 12 * * * * 22 44   46 20 * * * * 14 52
```

```
[10]                8/O  <-- Unmatched -->    7/F
 * * * * 20 46 17 47   * * * * 12 54 9 55
 * * * * 41 23 44 22   * * * * 49 15 52 14
13 51 16 50 * * * *   21 43 24 42 * * * *
56 10 53 11 * * * *   48 18 45 19 * * * *
45 19 48 18 * * * *   53 11 56 10 * * * *
24 42 21 43 * * * *   16 50 13 51 * * * *
 * * * * 52 14 49 15   * * * * 44 22 41 23
 * * * * 9 55 12 54   * * * * 17 47 20 46
```

```
[11]                10/O <-- Unmatched -->    7/F
 * * * * 8 58 5 59   * * * * 12 54 9 55
56 10 53 11 * * * *   60 6 57 7 * * * *
25 39 28 38 * * * *   21 43 24 42 * * * *
 * * * * 41 23 44 22   * * * * 37 27 40 26
57 7 60 6 * * * *   53 11 56 10 * * * *
 * * * * 9 55 12 54   * * * * 5 59 8 58
 * * * * 40 26 37 27   * * * * 44 22 41 23
24 42 21 43 * * * *   28 38 25 39 * * * *
```

```
[12]                9/O  <-- Unmatched -->    8/F
 * * * * 36 30 33 31   * * * * 20 46 17 47
 * * * * 25 39 28 38   * * * * 41 23 44 22
 * * * * 48 18 45 19   * * * * 32 34 29 35
 * * * * 21 43 24 42   * * * * 37 27 40 26
29 35 32 34 * * * *   45 19 48 18 * * * *
40 26 37 27 * * * *   24 42 21 43 * * * *
17 47 20 46 * * * *   33 31 36 30 * * * *
44 22 41 23 * * * *   28 38 25 39 * * * *
```

#3. The next job we have to do is to collect similar pairs whose 'Unmatched' patterns are the same and common. Classify them into several groups.

```
** Collect Similar Pairs whose 'Unmatched' patterns **
*** are the same among F10 of C&C Magic Squares 8x8 **
```

Group of **Type 1**

```
[ 2]                7/   <-- Unmatched -->    1/F
 * * 4 62 12 54 * *   * * 6 60 14 52 * *
60 6 * * * * 52 14   62 4 * * * * 54 12
21 43 * * * * 29 35   19 45 * * * * 27 37
 * * 45 19 37 27 * *   * * 43 21 35 29 * *
53 11 * * * * 61 3    51 13 * * * * 59 5
 * * 13 51 5 59 * *   * * 11 53 3 61 * *
 * * 36 30 44 22 * *   * * 38 28 46 20 * *
28 38 * * * * 20 46   30 36 * * * * 22 44
```

```
[ 5]                                8/  <-- Unmatched -->    2/F
 * * 4 62 20 46 * * * * 6 60 22 44 * *
60 6 * * * * 44 22 62 4 * * * * 46 20
13 51 * * * * 29 35 11 53 * * * * 27 37
 * * 53 11 37 27 * * * * 51 13 35 29 * *
45 19 * * * * 61 3 43 21 * * * * 59 5
 * * 21 43 5 59 * * * * 19 45 3 61 * *
 * * 36 30 52 14 * * * * 38 28 54 12 * *
28 38 * * * * 12 54 30 36 * * * * 14 52
```

```
[ 7]                                9/  <-- Unmatched -->    3/F
 * * 4 62 36 30 * * * * 6 60 38 28 * *
60 6 * * * * 28 38 62 4 * * * * 30 36
13 51 * * * * 45 19 11 53 * * * * 43 21
 * * 53 11 21 43 * * * * 51 13 19 45 * *
29 35 * * * * 61 3 27 37 * * * * 59 5
 * * 37 27 5 59 * * * * 35 29 3 61 * *
 * * 20 46 52 14 * * * * 22 44 54 12 * *
44 22 * * * * 12 54 46 20 * * * * 14 52
```

Group of **Type 2**

```
[ 3]                                3/  <-- Unmatched -->    2/F
 * * * * 38 28 33 31 * * * * 22 44 17 47
 * * * * 25 39 30 36 * * * * 41 23 46 20
 * * * * 48 18 43 21 * * * * 32 34 27 37
 * * * * 19 45 24 42 * * * * 35 29 40 26
27 37 32 34 * * * * 43 21 48 18 * * * *
40 26 35 29 * * * * 24 42 19 45 * * * *
17 47 22 44 * * * * 33 31 38 28 * * * *
46 20 41 23 * * * * 30 36 25 39 * * * *
```

```
[ 8]                                5/  <-- Unmatched -->    4/F
 * * * * 42 24 33 31 * * * * 26 40 17 47
 * * * * 21 43 30 36 * * * * 37 27 46 20
 * * * * 48 18 39 25 * * * * 32 34 23 41
 * * * * 19 45 28 38 * * * * 35 29 44 22
23 41 32 34 * * * * 39 25 48 18 * * * *
44 22 35 29 * * * * 28 38 19 45 * * * *
17 47 26 40 * * * * 33 31 42 24 * * * *
46 20 37 27 * * * * 30 36 21 43 * * * *
```

```
[12]                                9/  <-- Unmatched -->    8/F
 * * * * 36 30 33 31 * * * * 20 46 17 47
 * * * * 25 39 28 38 * * * * 41 23 44 22
 * * * * 48 18 45 19 * * * * 32 34 29 35
 * * * * 21 43 24 42 * * * * 37 27 40 26
29 35 32 34 * * * * 45 19 48 18 * * * *
40 26 37 27 * * * * 24 42 21 43 * * * *
17 47 20 46 * * * * 33 31 36 30 * * * *
44 22 41 23 * * * * 28 38 25 39 * * * *
```

Group of **Type 3**

```
[ 1]                                2/  <-- Unmatched -->    1/F
 * * * * 22 44 17 47 * * * * 14 52 9 55
 * * * * 41 23 46 20 * * * * 49 15 54 12
11 53 16 50 * * * * 19 45 24 42 * * * *
56 10 51 13 * * * * 48 18 43 21 * * * *
43 21 48 18 * * * * 51 13 56 10 * * * *
24 42 19 45 * * * * 16 50 11 53 * * * *
 * * * * 54 12 49 15 * * * * 46 20 41 23
 * * * * 9 55 14 52 * * * * 17 47 22 44
```

```
[10]                8/  <-- Unmatched -->    7/F
  * * * * 20 46 17 47   * * * * 12 54  9 55
  * * * * 41 23 44 22   * * * * 49 15 52 14
 13 51 16 50 * * * *   21 43 24 42 * * * *
 56 10 53 11 * * * *   48 18 45 19 * * * *
 45 19 48 18 * * * *   53 11 56 10 * * * *
 24 42 21 43 * * * *   16 50 13 51 * * * *
  * * * * 52 14 49 15   * * * * 44 22 41 23
  * * * *  9 55 12 54   * * * * 17 47 20 46
```

Group of **Type 4**

```
[ 4]                4/  <-- Unmatched -->    2/F
  * * 10 56 26 40 * *   * *  6 60 22 44 * *
  * * 53 11 37 27 * *   * * 57  7 41 23 * *
  7 57 * * * * 23 41   11 53 * * * * 27 37
 60  6 * * * * 44 22   56 10 * * * * 40 26
 39 25 * * * * 55  9   43 21 * * * * 59  5
 28 38 * * * * 12 54   24 42 * * * *  8 58
  * * 42 24 58  8 * *   * * 38 28 54 12 * *
  * * 21 43  5 59 * *   * * 25 39  9 55 * *
```

```
[ 6]                5/  <-- Unmatched -->    3/F
  * * 10 56 42 24 * *   * *  6 60 38 28 * *
  * * 53 11 21 43 * *   * * 57  7 25 39 * *
  7 57 * * * * 39 25   11 53 * * * * 43 21
 60  6 * * * * 28 38   56 10 * * * * 24 42
 23 41 * * * * 55  9   27 37 * * * * 59  5
 44 22 * * * * 12 54   40 26 * * * *  8 58
  * * 26 40 58  8 * *   * * 22 44 54 12 * *
  * * 37 27  5 59 * *   * * 41 23  9 55 * *
```

Group of **Type 5**

```
[ 9]                6/  <-- Unmatched -->    5/F
  * * 18 48 50 16 * *   * * 10 56 42 24 * *
  * * 45 19 13 51 * *   * * 53 11 21 43 * *
  * * 24 42 56 10 * *   * * 16 50 48 18 * *
  * * 43 21 11 53 * *   * * 51 13 19 45 * *
 15 49 * * * * 47 17   23 41 * * * * 55  9
 52 14 * * * * 20 46   44 22 * * * * 12 54
  9 55 * * * * 41 23   17 47 * * * * 49 15
 54 12 * * * * 22 44   46 20 * * * * 14 52
```

Group of **Type 6**

```
[11]                10/ <-- Unmatched -->    7/F
  * * * *  8 58  5 59   * * * * 12 54  9 55
 56 10 53 11 * * * *   60  6 57  7 * * * *
 25 39 28 38 * * * *   21 43 24 42 * * * *
  * * * * 41 23 44 22   * * * * 37 27 40 26
 57  7 60  6 * * * *   53 11 56 10 * * * *
  * * * *  9 55 12 54   * * * *  5 59  8 58
  * * * * 40 26 37 27   * * * * 44 22 41 23
 24 42 21 43 * * * *   28 38 25 39 * * * *
```

#4. There are 6 groups of similar pairs whose 'Unmatched' patterns are the same. Could you imagine something common like a transformation method for each group? I found it for each group, and dictated the program codes as follows:

```
/**/
/* Program Codes for 6 Types of Transformations */
```

```

/**/
/* Transformations: Type 1~6 */
void trans1(){
    short n;
    for(n=0; n<65; n++){ dn[n]=cn[n]; }
    dn[3]=cn[10]; dn[4]=cn[9]; dn[5]=cn[16]; dn[6]=cn[15];
    dn[9]=cn[4]; dn[10]=cn[3]; dn[15]=cn[6]; dn[16]=cn[5];
    dn[17]=cn[28]; dn[18]=cn[27]; dn[23]=cn[30]; dn[24]=cn[29];
    dn[27]=cn[18]; dn[28]=cn[17]; dn[29]=cn[24]; dn[30]=cn[23];
    dn[33]=cn[44]; dn[34]=cn[43]; dn[39]=cn[46]; dn[40]=cn[45];
    dn[43]=cn[34]; dn[44]=cn[33]; dn[45]=cn[40]; dn[46]=cn[39];
    dn[51]=cn[58]; dn[52]=cn[57]; dn[53]=cn[64]; dn[54]=cn[63];
    dn[57]=cn[52]; dn[58]=cn[51]; dn[63]=cn[54]; dn[64]=cn[53];
}
/**/
void trans2(){
    short n;
    for(n=0; n<65; n++){ dn[n]=cn[n]; }
    dn[5]=cn[51]; dn[6]=cn[52]; dn[7]=cn[49]; dn[8]=cn[50];
    dn[13]=cn[59]; dn[14]=cn[60]; dn[15]=cn[57]; dn[16]=cn[58];
    dn[21]=cn[35]; dn[22]=cn[36]; dn[23]=cn[33]; dn[24]=cn[34];
    dn[29]=cn[43]; dn[30]=cn[44]; dn[31]=cn[41]; dn[32]=cn[42];
    dn[33]=cn[23]; dn[34]=cn[24]; dn[35]=cn[21]; dn[36]=cn[22];
    dn[41]=cn[31]; dn[42]=cn[32]; dn[43]=cn[29]; dn[44]=cn[30];
    dn[49]=cn[7]; dn[50]=cn[8]; dn[51]=cn[5]; dn[52]=cn[6];
    dn[57]=cn[15]; dn[58]=cn[16]; dn[59]=cn[13]; dn[60]=cn[14];
}
/**/
void trans3(){
    short n;
    for(n=0; n<65; n++){ dn[n]=cn[n]; }
    dn[5]=cn[63]; dn[6]=cn[64]; dn[7]=cn[61]; dn[8]=cn[62];
    dn[13]=cn[55]; dn[14]=cn[56]; dn[15]=cn[53]; dn[16]=cn[54];
    dn[17]=cn[43]; dn[18]=cn[44]; dn[19]=cn[41]; dn[20]=cn[42];
    dn[25]=cn[35]; dn[26]=cn[36]; dn[27]=cn[33]; dn[28]=cn[34];
    dn[33]=cn[27]; dn[34]=cn[28]; dn[35]=cn[25]; dn[36]=cn[26];
    dn[41]=cn[19]; dn[42]=cn[20]; dn[43]=cn[17]; dn[44]=cn[18];
    dn[53]=cn[15]; dn[54]=cn[16]; dn[55]=cn[13]; dn[56]=cn[14];
    dn[61]=cn[7]; dn[62]=cn[8]; dn[63]=cn[5]; dn[64]=cn[6];
}
/**/
void trans4(){
    short n;
    for(n=0; n<65; n++){ dn[n]=cn[n]; }
    dn[3]=cn[26]; dn[4]=cn[25]; dn[5]=cn[32]; dn[6]=cn[31];
    dn[11]=cn[18]; dn[12]=cn[17]; dn[13]=cn[24]; dn[14]=cn[23];
    dn[17]=cn[12]; dn[18]=cn[11]; dn[23]=cn[14]; dn[24]=cn[13];
    dn[25]=cn[4]; dn[26]=cn[3]; dn[31]=cn[6]; dn[32]=cn[5];
    dn[33]=cn[60]; dn[34]=cn[59]; dn[39]=cn[62]; dn[40]=cn[61];
    dn[41]=cn[52]; dn[42]=cn[51]; dn[47]=cn[54]; dn[48]=cn[53];
    dn[51]=cn[42]; dn[52]=cn[41]; dn[53]=cn[48]; dn[54]=cn[47];
    dn[59]=cn[34]; dn[60]=cn[33]; dn[61]=cn[40]; dn[62]=cn[39];
}

```

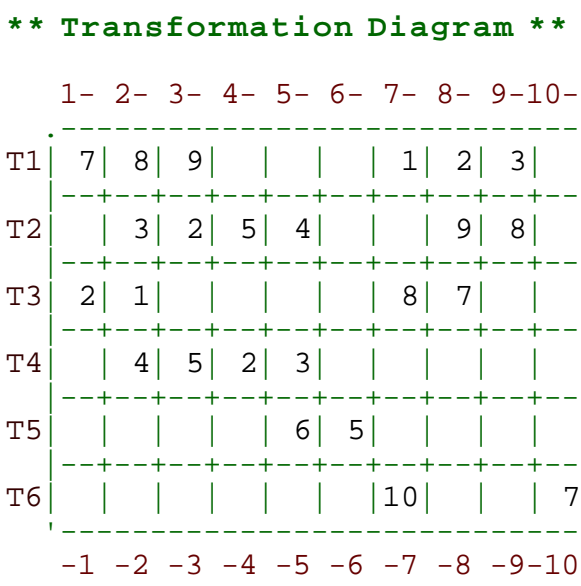
```

/**/
void trans5(){
  short n;
  for(n=0; n<65; n++) { dn[n]=cn[n]; }
  dn[3]=cn[22];  dn[4]=cn[21];  dn[5]=cn[20];    dn[6]=cn[19];
  dn[11]=cn[30]; dn[12]=cn[29]; dn[13]=cn[28];  dn[14]=cn[27];
  dn[19]=cn[6];  dn[20]=cn[5];  dn[21]=cn[4];   dn[22]=cn[3];
  dn[27]=cn[14]; dn[28]=cn[13]; dn[29]=cn[12];  dn[30]=cn[11];
  dn[33]=cn[56]; dn[34]=cn[55]; dn[39]=cn[50];  dn[40]=cn[49];
  dn[41]=cn[64]; dn[42]=cn[63]; dn[47]=cn[58];  dn[48]=cn[57];
  dn[49]=cn[40]; dn[50]=cn[39]; dn[55]=cn[34];  dn[56]=cn[33];
  dn[57]=cn[48]; dn[58]=cn[47]; dn[63]=cn[42];  dn[64]=cn[41];
}
/**/
void trans6(){
  short n;
  for(n=0; n<65; n++) { dn[n]=cn[n]; }
  dn[5]=cn[47];  dn[6]=cn[48];  dn[7]=cn[45];    dn[8]=cn[46];
  dn[9]=cn[35];  dn[10]=cn[36]; dn[11]=cn[33];   dn[12]=cn[34];
  dn[17]=cn[59]; dn[18]=cn[60]; dn[19]=cn[57];   dn[20]=cn[58];
  dn[29]=cn[55]; dn[30]=cn[56]; dn[31]=cn[53];   dn[32]=cn[54];
  dn[33]=cn[11]; dn[34]=cn[12]; dn[35]=cn[9];    dn[36]=cn[10];
  dn[45]=cn[7];  dn[46]=cn[8];  dn[47]=cn[5];    dn[48]=cn[6];
  dn[53]=cn[31]; dn[54]=cn[32]; dn[55]=cn[29];   dn[56]=cn[30];
  dn[57]=cn[19]; dn[58]=cn[20]; dn[59]=cn[17];   dn[60]=cn[18];
}
/**/

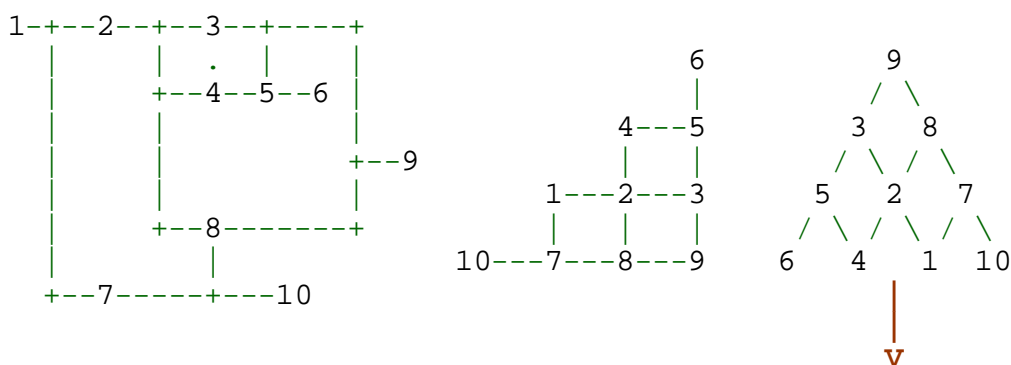
```

#5. How do you use these transformation procedures? How do you make the only one into the other 9 fundamental solutions?

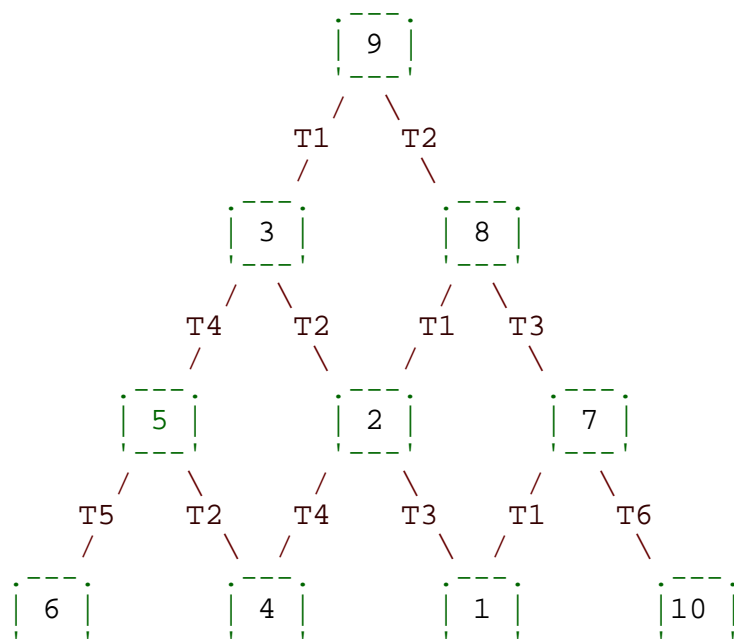
It is useful for you to make such a quick-view diagram as follows:



There are several good ways of making your job, but let me introduce you of my latest invention. I put it in the “Transition Diagram of Transformation Process” as follows:



**** Transition Diagram of Transformation Process ****



Let's start with the solution No.9 at the top of this diagram. Transform it by Type 1, and you will have No.3. Transform No.9 into No.8 by Type 2. Transform these new results No.3 into No.5 by Type 4 and No. 8 into No.7 by Type 3.

Do the same thing for one after another, you will finally have all F10 solutions. And you will know No.9 is really the 'Most Fundamental Solution' of all.

But is only No.9 the 'MFS' of all? No, it isn't.

Anyone could become the "MFS" of all the rest.

Watch the diagram carefully and find you can start with anyone anywhere. You can get to anyone elsewhere, since every transformation method is essentially reversible. You can go back and upward in the diagram and get to everyone everywhere.

I should say anyone of F10 solutions could become the MFS as a representative of all.

Each of them has inherited the common memory of its 'Parent' and proves to belong to the only one family. We found nothing else at last.

(Revised Edition written on December 27, 2002 by Kanji Setsuda)

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

↓
v (See the next list.)

*** Check the Result of Making the Most Fundamental Solution ***
 * into the 10 Fundamental Solutions of 'C&C' Magic Squares 8x8 *

1/T								<-- Unmatched -->	1/F														
1	63	6	60	14	52	9	55	*	*	*	*	*	*	*	*	1	63	6	60	14	52	9	55
62	4	57	7	49	15	54	12	*	*	*	*	*	*	*	*	62	4	57	7	49	15	54	12
19	45	24	42	32	34	27	37	*	*	*	*	*	*	*	*	19	45	24	42	32	34	27	37
48	18	43	21	35	29	40	26	*	*	*	*	*	*	*	*	48	18	43	21	35	29	40	26
51	13	56	10	64	2	59	5	*	*	*	*	*	*	*	*	51	13	56	10	64	2	59	5
16	50	11	53	3	61	8	58	*	*	*	*	*	*	*	*	16	50	11	53	3	61	8	58
33	31	38	28	46	20	41	23	*	*	*	*	*	*	*	*	33	31	38	28	46	20	41	23
30	36	25	39	17	47	22	44	*	*	*	*	*	*	*	*	30	36	25	39	17	47	22	44

2/T								<-- Unmatched -->	2/F													
1	63	6	60	22	44	17	47	*	*	*	*	*	*	*	1	63	6	60	22	44	17	47
62	4	57	7	41	23	46	20	*	*	*	*	*	*	*	62	4	57	7	41	23	46	20
11	53	16	50	32	34	27	37	*	*	*	*	*	*	*	11	53	16	50	32	34	27	37
56	10	51	13	35	29	40	26	*	*	*	*	*	*	*	56	10	51	13	35	29	40	26
43	21	48	18	64	2	59	5	*	*	*	*	*	*	*	43	21	48	18	64	2	59	5
24	42	19	45	3	61	8	58	*	*	*	*	*	*	*	24	42	19	45	3	61	8	58
33	31	38	28	54	12	49	15	*	*	*	*	*	*	*	33	31	38	28	54	12	49	15
30	36	25	39	9	55	14	52	*	*	*	*	*	*	*	30	36	25	39	9	55	14	52

3/T								<-- Unmatched -->	3/F													
1	63	6	60	38	28	33	31	*	*	*	*	*	*	*	1	63	6	60	38	28	33	31
62	4	57	7	25	39	30	36	*	*	*	*	*	*	*	62	4	57	7	25	39	30	36
11	53	16	50	48	18	43	21	*	*	*	*	*	*	*	11	53	16	50	48	18	43	21
56	10	51	13	19	45	24	42	*	*	*	*	*	*	*	56	10	51	13	19	45	24	42
27	37	32	34	64	2	59	5	*	*	*	*	*	*	*	27	37	32	34	64	2	59	5
40	26	35	29	3	61	8	58	*	*	*	*	*	*	*	40	26	35	29	3	61	8	58
17	47	22	44	54	12	49	15	*	*	*	*	*	*	*	17	47	22	44	54	12	49	15
46	20	41	23	9	55	14	52	*	*	*	*	*	*	*	46	20	41	23	9	55	14	52

4/T								<-- Unmatched -->	4/F													
1	63	10	56	26	40	17	47	*	*	*	*	*	*	*	1	63	10	56	26	40	17	47
62	4	53	11	37	27	46	20	*	*	*	*	*	*	*	62	4	53	11	37	27	46	20
7	57	16	50	32	34	23	41	*	*	*	*	*	*	*	7	57	16	50	32	34	23	41
60	6	51	13	35	29	44	22	*	*	*	*	*	*	*	60	6	51	13	35	29	44	22
39	25	48	18	64	2	55	9	*	*	*	*	*	*	*	39	25	48	18	64	2	55	9
28	38	19	45	3	61	12	54	*	*	*	*	*	*	*	28	38	19	45	3	61	12	54
33	31	42	24	58	8	49	15	*	*	*	*	*	*	*	33	31	42	24	58	8	49	15
30	36	21	43	5	59	14	52	*	*	*	*	*	*	*	30	36	21	43	5	59	14	52

5/T								<-- Unmatched -->	5/F													
1	63	10	56	42	24	33	31	*	*	*	*	*	*	*	1	63	10	56	42	24	33	31
62	4	53	11	21	43	30	36	*	*	*	*	*	*	*	62	4	53	11	21	43	30	36
7	57	16	50	48	18	39	25	*	*	*	*	*	*	*	7	57	16	50	48	18	39	25
60	6	51	13	19	45	28	38	*	*	*	*	*	*	*	60	6	51	13	19	45	28	38
23	41	32	34	64	2	55	9	*	*	*	*	*	*	*	23	41	32	34	64	2	55	9
44	22	35	29	3	61	12	54	*	*	*	*	*	*	*	44	22	35	29	3	61	12	54
17	47	26	40	58	8	49	15	*	*	*	*	*	*	*	17	47	26	40	58	8	49	15
46	20	37	27	5	59	14	52	*	*	*	*	*	*	*	46	20	37	27	5	59	14	52

6/T								<-- Unmatched -->	6/F													
1	63	18	48	50	16	33	31	*	*	*	*	*	*	*	1	63	18	48	50	16	33	31
62	4	45	19	13	51	30	36	*	*	*	*	*	*	*	62	4	45	19	13	51	30	36
7	57	24	42	56	10	39	25	*	*	*	*	*	*	*	7	57	24	42	56	10	39	25
60	6	43	21	11	53	28	38	*	*	*	*	*	*	*	60	6	43	21	11	53	28	38
15	49	32	34	64	2	47	17	*	*	*	*	*	*	*	15	49	32	34	64	2	47	17
52	14	35	29	3	61	20	46	*	*	*	*	*	*	*	52	14	35	29	3	61	20	46
9	55	26	40	58	8	41	23	*	*	*	*	*	*	*	9	55	26	40	58	8	41	23
54	12	37	27	5	59	22	44	*	*	*	*	*	*	*	54	12	37	27	5	59	22	44

	7/T	<-- Unmatched -->	7/F																				
1	63	4	62	12	54	9	55	*	*	*	*	*	*	*	*	1	63	4	62	12	54	9	55
60	6	57	7	49	15	52	14	*	*	*	*	*	*	*	*	60	6	57	7	49	15	52	14
21	43	24	42	32	34	29	35	*	*	*	*	*	*	*	*	21	43	24	42	32	34	29	35
48	18	45	19	37	27	40	26	*	*	*	*	*	*	*	*	48	18	45	19	37	27	40	26
53	11	56	10	64	2	61	3	*	*	*	*	*	*	*	*	53	11	56	10	64	2	61	3
16	50	13	51	5	59	8	58	*	*	*	*	*	*	*	*	16	50	13	51	5	59	8	58
33	31	36	30	44	22	41	23	*	*	*	*	*	*	*	*	33	31	36	30	44	22	41	23
28	38	25	39	17	47	20	46	*	*	*	*	*	*	*	*	28	38	25	39	17	47	20	46
	8/T	<-- Unmatched -->	8/F																				
1	63	4	62	20	46	17	47	*	*	*	*	*	*	*	*	1	63	4	62	20	46	17	47
60	6	57	7	41	23	44	22	*	*	*	*	*	*	*	*	60	6	57	7	41	23	44	22
13	51	16	50	32	34	29	35	*	*	*	*	*	*	*	*	13	51	16	50	32	34	29	35
56	10	53	11	37	27	40	26	*	*	*	*	*	*	*	*	56	10	53	11	37	27	40	26
45	19	48	18	64	2	61	3	*	*	*	*	*	*	*	*	45	19	48	18	64	2	61	3
24	42	21	43	5	59	8	58	*	*	*	*	*	*	*	*	24	42	21	43	5	59	8	58
33	31	36	30	52	14	49	15	*	*	*	*	*	*	*	*	33	31	36	30	52	14	49	15
28	38	25	39	9	55	12	54	*	*	*	*	*	*	*	*	28	38	25	39	9	55	12	54
	9/T	<-- Unmatched -->	9/F																				
1	63	4	62	36	30	33	31	*	*	*	*	*	*	*	*	1	63	4	62	36	30	33	31
60	6	57	7	25	39	28	38	*	*	*	*	*	*	*	*	60	6	57	7	25	39	28	38
13	51	16	50	48	18	45	19	*	*	*	*	*	*	*	*	13	51	16	50	48	18	45	19
56	10	53	11	21	43	24	42	*	*	*	*	*	*	*	*	56	10	53	11	21	43	24	42
29	35	32	34	64	2	61	3	*	*	*	*	*	*	*	*	29	35	32	34	64	2	61	3
40	26	37	27	5	59	8	58	*	*	*	*	*	*	*	*	40	26	37	27	5	59	8	58
17	47	20	46	52	14	49	15	*	*	*	*	*	*	*	*	17	47	20	46	52	14	49	15
44	22	41	23	9	55	12	54	*	*	*	*	*	*	*	*	44	22	41	23	9	55	12	54
	10/T	<-- Unmatched -->	10/F																				
1	63	4	62	8	58	5	59	*	*	*	*	*	*	*	*	1	63	4	62	8	58	5	59
56	10	53	11	49	15	52	14	*	*	*	*	*	*	*	*	56	10	53	11	49	15	52	14
25	39	28	38	32	34	29	35	*	*	*	*	*	*	*	*	25	39	28	38	32	34	29	35
48	18	45	19	41	23	44	22	*	*	*	*	*	*	*	*	48	18	45	19	41	23	44	22
57	7	60	6	64	2	61	3	*	*	*	*	*	*	*	*	57	7	60	6	64	2	61	3
16	50	13	51	9	55	12	54	*	*	*	*	*	*	*	*	16	50	13	51	9	55	12	54
33	31	36	30	40	26	37	27	*	*	*	*	*	*	*	*	33	31	36	30	40	26	37	27
24	42	21	43	17	47	20	46	*	*	*	*	*	*	*	*	24	42	21	43	17	47	20	46

All Right!