Cube Synthesizer Numbering Scheme

Introduction

One of the most important part of the Cube Synthesizer program is the database of basic algorithms. For each type of cube, there is a table of basic algorithms, which can be looked up by computing an algorithm index. The index is a simple function of the cube order (N), the number of the facelet of origin (Origin), the number of the facelet of destination on the front face (Destination) and, for odd-order cubes, the front face center facelet orientation (Orientation).

The ordering of algorithms, numbering of indexes and cube faces have all been designed for providing scalability ie. for being applicable to *any* cube of order N.

Basic Numbers

Number of Stickers per Cube: $6.N^2$						
2x2x2	**3x3x3**	**4x4x4**	**5x5x5**	**6x6x6**	**7x7x7**	**8x8x8**
24	54	96	150	216	294	384
Number of Basic Algorithms per Cube: $24 + 24.N^2$						
2x2x2	**3x3x3**	**4x4x4**	**5x5x5**	**6x6x6**	**7x7x7**	**8x8x8**
110	240	408	624	888	1200	1560
Number of Numbers per Cube (States): $144.N^2 + 144.N^4$						
2x2x2	**3x3x3**	**4x4x4**	**5x5x5**	**6x6x6**	**7x7x7**	**8x8x8**
2880	12 960	39 168	93 600	191 808	352 800	599 040

Numbered Cubes

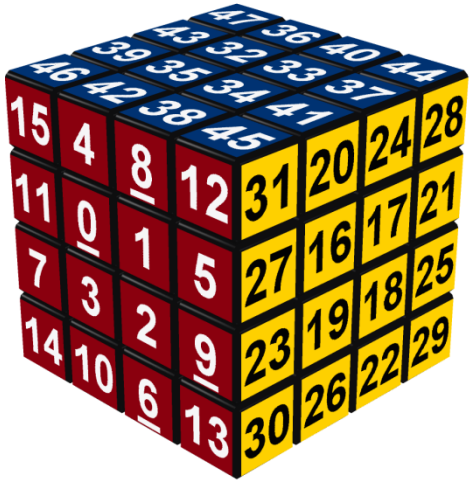
Numbered Cubes are used to check basic algorithms and fill in lookup tables with numbers indicating the cube state. A Basic Algorithm is one of the shortest length algorithms used for moving a numbered cubie from any face to the front face.

NxNxN Numbered Cubes

2x2x2

3x3x3

4x4x4



Cube Rotation

Algorithms for rotating the cube are considered as 'zero-move' sequences. Therefore, by rotating the cube for placing a center cubie on the front face at the very beginning, we get a 'zero-length' basic algorithm and thus the length of the complete algorithm will be shorter.

The first step towards solving the front face is then to rotate the cube for placing a center cubie with the right orientation in location '0'. For odd-order cubes, the center cubie orientation should be specified. For even-order cubes, there is no need to specify any orientation, because it is already implicitely derived from the Origin cubie number as follows:

Orientation = Origin modulo N^2 (even-order cubes only)

NxNxN Cube Numbering Scheme – Cube Rotation – 24 Cases	
AlgorithmIndex Bounds (N even or odd)	
AlgorithmIndexMin = 0	AlgorithmIndexMax = 23
AlgorithmIndex (General Formula for N even)	
AlgorithmIndex = Origin modulo N^2 + 4.integer(Origin/N^2)	
AlgorithmIndex (General Formula for N odd)	
AlgorithmIndex = Orientation + 4.integer(Origin/N^2)	

Center Cubie Orientation

The front face center cubie is identified by number '0' for even- and odd-order cubes. The front face center cubie orientation can take 4 values. The new cubie orientation is obtained by rotating a center cubie shown on the texture in –90° steps ie. CounterClockWise (CCW).

Center Cubie Orientation			
0	**1**	**2**	**3**
–0° or +0°	–90° or +270°	–180° or +180°	–270° or +90°
North	East	South	West

Block Indexing

It may be useful to divide the list of algorithms in blocks indicating where cubies come from. Block #0 is related to cube rotation algorithms whereas Blocks #1 to 6 are related to individual cubie moves from each of the 6 faces to the front face.

Block Indexing				
Block Number	**Comment**	**To Face F ← From Face X**	**Lower AlgorithmIndex**	**Upper AlgorithmIndex**
0	Cube rotation	**F ← FRULDB**	0	23
1	Cubie move	**F ← F**	24	24 + 4.N^2 – 1
2	Cubie move	**F ← R**	24 + 4.N^2	24 + 8.N^2 – 1
3	Cubie move	**F ← U**	24 + 8.N^2	24 + 12.N^2 – 1
4	Cubie move	**F ← L**	24 + 12.N^2	24 + 16.N^2 – 1
5	Cubie move	**F ← D**	24 + 16.N^2	24 + 20.N^2 – 1
6	Cubie move	**F ← B**	24 + 20.N^2	24 + 24.N^2 – 1
Total Number of Algorithms = 24 + 24.N^2				

Cube State Indexing

A cube state is defined as a row of 6.N^2 numbers, from 0 up to 6.N^2 – 1, in any order that is a valid case. There are N^2 numbers per cube face. In the Initial state, the row of numbers is a set of linearly increasing numbers, beginning with number '0', as follows:

[Face **F**, Face **R**, Face **U**, Face **L**, Face **D**, face **B**] = [0,1,2,3,4,5,6,7,8,….., 6.N^2 – 1]

In any state other than the Initial state, numbers are ordered differently and the row reflects the cube state. Numbers are then sorted in the row as they would be displayed after applying the row associated basic algorithm to a numbered cube. So, the row indexing scheme is exactly the same as of the algorithm indexing scheme. Any facelet number can then be accessed separately by using a row sub-index varying from 0 to 6.N^2 – 1.

2x2x2 Cube

2x2x2 Cube Numbering Scheme – Sequence FRULDB																								
Face F								Face R								Face U							
			0	1							4	5							8	9			
			3	2							7	6							11	10			
Face L								Face D								Face B							
			12	13							16	17							20	21			
			15	14							19	18							23	22			

AlgorithmIndex (General Formula for N even)
AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + $(4.N^2)$.integer(Origin/N^2)
AlgorithmIndex (Formula for N = 2)
AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + 16.integer(Origin/4)

3x3x3 Cube

3x3x3 Cube Numbering Scheme – Sequence FRULDB																								
Face F								Face R								Face U							
		8	1	5						17	10	14						26	19	23			
		4	**0**	2						13	**9**	11						22	**18**	20			
		7	3	6						16	12	15						25	21	24			
Face L								Face D								Face B							
		35	28	32						44	37	41						53	46	50			
		31	**27**	29						40	**36**	38						49	**45**	47			
		34	30	33						43	39	42						52	48	51			

AlgorithmIndex (General Formula for N odd)

Case 1: Destination = 0

AlgorithmIndex = 24 + Orientation + $(4.N^2)$.integer(Origin/N^2)

Case 2: Destination <> 0

AlgorithmIndex = 24 + ((Origin modulo N^2) – 1) modulo 4 + 4.Destination + $(4.N^2)$.integer(Origin/N^2)

AlgorithmIndex (Formula for N = 3)

Case 1: Destination = 0

AlgorithmIndex = 24 + Orientation + 36.integer(Origin/9)

Case 2: Destination <> 0

AlgorithmIndex = 24 + ((Origin modulo 9) – 1) modulo 4 + 4.Destination + 36.integer(Origin/9)

3x3x3 Cube Numbering Scheme – Algorithm Indexing (0 – 119)

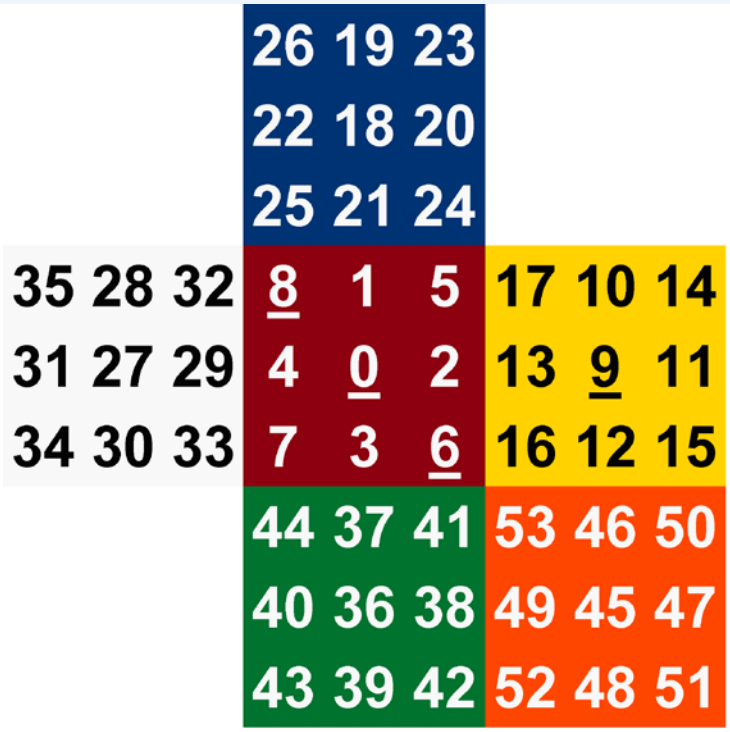
Destination	Origin	Orientation	Index	Faces	Destination	Origin	Orientation	Index	Faces
0	0	0	0	F ← F	0	9	0	60	F ← R
0	0	1	1	F ← F	0	9	1	61	F ← R
0	0	2	2	F ← F	0	9	2	62	F ← R
0	0	3	3	F ← F	0	9	3	63	F ← R
0	9	0	4	F ← R	1	10		64	F ← R
0	9	1	5	F ← R	1	11		65	F ← R
0	9	2	6	F ← R	1	12		66	F ← R
0	9	3	7	F ← R	1	13		67	F ← R
0	18	0	8	F ← U	2	10		68	F ← R
0	18	1	9	F ← U	2	11		69	F ← R
0	18	2	10	F ← U	2	12		70	F ← R
0	18	3	11	F ← U	2	13		71	F ← R
0	27	0	12	F ← L	3	10		72	F ← R
0	27	1	13	F ← L	3	11		73	F ← R
0	27	2	14	F ← L	3	12		74	F ← R
0	27	3	15	F ← L	3	13		75	F ← R
0	36	0	16	F ← D	4	10		76	F ← R
0	36	1	17	F ← D	4	11		77	F ← R
0	36	2	18	F ← D	4	12		78	F ← R
0	36	3	19	F ← D	4	13		79	F ← R
0	45	0	20	F ← B	5	14		80	F ← R
0	45	1	21	F ← B	5	15		81	F ← R
0	45	2	22	F ← B	5	16		82	F ← R
0	45	3	23	F ← B	5	17		83	F ← R
0	0	0	24	F ← F	6	14		84	F ← R
0	0	1	25	F ← F	6	15		85	F ← R
0	0	2	26	F ← F	6	16		86	F ← R
0	0	3	27	F ← F	6	17		87	F ← R
1	1		28	F ← F	7	14		88	F ← R
1	2		29	F ← F	7	15		89	F ← R
1	3		30	F ← F	7	16		90	F ← R
1	4		31	F ← F	7	17		91	F ← R
2	1		32	F ← F	8	14		92	F ← R
2	2		33	F ← F	8	15		93	F ← R
2	3		34	F ← F	8	16		94	F ← R
2	4		35	F ← F	8	17		95	F ← U
3	1		36	F ← F	0	18	0	96	F ← U
3	2		37	F ← F	0	18	1	97	F ← U
3	3		38	F ← F	0	18	2	98	F ← U
3	4		39	F ← F	0	18	3	99	F ← U
4	1		40	F ← F	1	19		100	F ← U
4	2		41	F ← F	1	20		101	F ← U
4	3		42	F ← F	1	21		102	F ← U
4	4		43	F ← F	1	22		103	F ← U
5	5		44	F ← F	2	19		104	F ← U
5	6		45	F ← F	2	20		105	F ← U
5	7		46	F ← F	2	21		106	F ← U
5	8		47	F ← F	2	22		107	F ← U
6	5		48	F ← F	3	19		108	F ← U
6	6		49	F ← F	3	20		109	F ← U
6	7		50	F ← F	3	21		110	F ← U
6	8		51	F ← F	3	22		111	F ← U
7	5		52	F ← F	4	19		112	F ← U
7	6		53	F ← F	4	20		113	F ← U
7	7		54	F ← F	4	21		114	F ← U
7	8		55	F ← F	4	22		115	F ← U
8	5		56	F ← F	5	23		116	F ← U
8	6		57	F ← F	5	24		117	F ← U
8	7		58	F ← F	5	25		118	F ← U
8	8		59	F ← F	5	26		119	F ← U

3x3x3 Cube Numbering Scheme – Algorithm Indexing (120 – 239)

Destination	Origin	Orientation	Index	Faces	Destination	Origin	Orientation	Index	Faces
6	23		120	F ← U	3	37		180	F ← D
6	24		121	F ← U	3	38		181	F ← D
6	25		122	F ← U	3	39		182	F ← D
6	26		123	F ← U	3	40		183	F ← D
7	23		124	F ← U	4	37		184	F ← D
7	24		125	F ← U	4	38		185	F ← D
7	25		126	F ← U	4	39		186	F ← D
7	26		127	F ← U	4	40		187	F ← D
8	23		128	F ← U	5	41		188	F ← D
8	24		129	F ← U	5	42		189	F ← D
8	25		130	F ← U	5	43		190	F ← D
8	26		131	F ← U	5	44		191	F ← D
0	27	0	132	F ← L	6	41		192	F ← D
0	27	1	133	F ← L	6	42		193	F ← D
0	27	2	134	F ← L	6	43		194	F ← D
0	27	3	135	F ← L	6	44		195	F ← D
1	28		136	F ← L	7	41		196	F ← D
1	29		137	F ← L	7	42		197	F ← D
1	30		138	F ← L	7	43		198	F ← D
1	31		139	F ← L	7	44		199	F ← D
2	28		140	F ← L	8	41		200	F ← D
2	29		141	F ← L	8	42		201	F ← D
2	30		142	F ← L	8	43		202	F ← D
2	31		143	F ← L	8	44		203	F ← D
3	28		144	F ← L	0	45	0	204	F ← B
3	29		145	F ← L	0	45	1	205	F ← B
3	30		146	F ← L	0	45	2	206	F ← B
3	31		147	F ← L	0	45	3	207	F ← B
4	28		148	F ← L	1	46		208	F ← B
4	29		149	F ← L	1	47		209	F ← B
4	30		150	F ← L	1	48		210	F ← B
4	31		151	F ← L	1	49		211	F ← B
5	32		152	F ← L	2	46		212	F ← B
5	33		153	F ← L	2	47		213	F ← B
5	34		154	F ← L	2	48		214	F ← B
5	35		155	F ← L	2	49		215	F ← B
6	32		156	F ← L	3	46		216	F ← B
6	33		157	F ← L	3	47		217	F ← B
6	34		158	F ← L	3	48		218	F ← B
6	35		159	F ← L	3	49		219	F ← B
7	32		160	F ← L	4	46		220	F ← B
7	33		161	F ← L	4	47		221	F ← B
7	34		162	F ← L	4	48		222	F ← B
7	35		163	F ← L	4	49		223	F ← B
8	32		164	F ← L	5	50		224	F ← B
8	33		165	F ← L	5	51		225	F ← B
8	34		166	F ← L	5	52		226	F ← B
8	35		167	F ← L	5	53		227	F ← B
0	36	0	168	F ← D	6	50		228	F ← B
0	36	1	169	F ← D	6	51		229	F ← B
0	36	2	170	F ← D	6	52		230	F ← B
0	36	3	171	F ← D	6	53		231	F ← B
1	37		172	F ← D	7	50		232	F ← B
1	38		173	F ← D	7	51		233	F ← B
1	39		174	F ← D	7	52		234	F ← B
1	40		175	F ← D	7	53		235	F ← B
2	37		176	F ← D	8	50		236	F ← B
2	38		177	F ← D	8	51		237	F ← B
2	39		178	F ← D	8	52		238	F ← B
2	40		179	F ← D	8	53		239	F ← B

3x3x3 Numbered Cube

Texture



Virtual Cube

3x3x3 Numbered Cube – Examples of Basic Algorithms

Number '**19**' is brought from face **U** to face **F**

Number '**14**' is brought from face **R** to face **F**

Basic Algorithms

B' MR B MR'

B2 R' B R

A 3x3x3 Numbered Cube is used to check basic algorithms and fill in lookup tables with numbers indicating the cube state. A basic algorithm is one of the shortest length algorithms that can be found on a cube. Basic algorithms that do not change front face cubies other than the origin and destination cubies are generally built with commutators ([A,B] = A B' A' B).

4x4x4 Cube

4x4x4 Cube Numbering Scheme – Sequence FRULDB																							
Face F								**Face R**								**Face U**							
		15	4	8	12					31	20	24	28					47	36	40	44		
		11	**0**	**1**	5					27	**16**	**17**	21					43	**32**	**33**	37		
		7	**3**	**2**	9					23	**19**	**18**	25					39	**35**	**34**	41		
		14	10	6	13					30	26	22	29					46	42	38	45		
Face L								**Face D**								**Face B**							
		63	52	56	60					79	68	72	76					95	84	88	92		
		59	**48**	**49**	53					75	**64**	**65**	69					91	**80**	**81**	85		
		55	**51**	**50**	57					71	**67**	**66**	73					87	**83**	**82**	89		
		62	58	54	61					78	74	70	77					94	90	86	93		

AlgorithmIndex (General Formula for N even)

AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + $(4.N^2)$.integer(Origin/N^2)

AlgorithmIndex (Formula for N = 4)

AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + 64.integer(Origin/16)

5x5x5 Cube

5x5x5 Cube Numbering Scheme – Sequence FRULDB																									
Face F								**Face R**								**Face U**									
	24	9	13	17	21				49	34	38	42	46				74	59	63	67	71				
	20	8	1	5	10				45	33	26	30	35				70	58	51	55	60				
	16	4	**0**	2	14				41	29	**25**	27	39				66	54	**50**	52	64				
	12	7	3	6	18				37	32	28	31	43				62	57	53	56	68				
	23	19	15	11	22				48	44	40	36	47				73	69	65	61	72				
Face L								**Face D**								**Face B**									
	99	84	88	92	96				124	109	113	117	121				149	134	138	142	146				
	95	83	76	80	85				120	108	101	105	110				145	133	126	130	135				
	91	79	**75**	77	89				116	104	**100**	102	114				141	129	**125**	127	139				
	87	82	78	81	93				112	107	103	106	118				137	132	128	131	143				
	98	94	90	86	97				123	119	115	111	122				148	144	140	136	147				

AlgorithmIndex (General Formula for N odd)
Case 1: Destination = 0
AlgorithmIndex = 24 + Orientation + $(4.N^2)$.integer(Origin/N^2)
Case 2: Destination <> 0
AlgorithmIndex = 24 + ((Origin modulo N^2) – 1) modulo 4 + 4.Destination + $(4.N^2)$.integer(Origin/N^2)
AlgorithmIndex (Formula for N = 5)
Case 1: Destination = 0
AlgorithmIndex = 24 + Orientation + 100.integer(Origin/25)
Case 2: Destination <> 0
AlgorithmIndex = 24 + ((Origin modulo 25) – 1) modulo 4 + 4.Destination + 100.integer(Origin/25)

6x6x6 Cube

6x6x6 Cube Numbering Scheme – Sequence FRULDB																								
Face F								**Face R**								**Face U**							
	35	16	20	24	28	32			71	52	56	60	64	68			107	88	92	96	100	104	
	31	15	4	8	12	17			67	51	40	44	48	53			103	87	76	80	84	89	
	27	11	**0**	**1**	5	21			63	47	**36**	**37**	41	57			99	83	**72**	**73**	77	93	
	23	7	**3**	**2**	9	25			59	43	**39**	**38**	45	61			95	79	**75**	**74**	81	97	
	19	14	10	6	13	29			55	50	46	42	49	65			91	86	82	78	85	101	
	34	30	26	22	18	33			70	66	62	58	54	69			106	102	98	94	90	105	
Face L								**Face D**								**Face B**							
	143	124	128	132	136	140			179	160	164	168	172	176			215	196	200	204	208	212	
	139	123	112	116	120	125			175	159	148	152	156	161			211	195	184	188	192	197	
	135	119	**108**	**109**	113	129			171	155	**144**	**145**	149	165			207	191	**180**	**181**	185	201	
	131	115	**111**	**110**	117	133			167	151	**147**	**146**	153	169			203	187	**183**	**182**	189	205	
	127	122	118	114	121	137			163	158	154	150	157	173			199	194	190	186	193	209	
	142	138	134	130	126	141			178	174	170	166	162	177			214	210	206	202	198	213	

AlgorithmIndex (General Formula for N even)

AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + $(4.N^2)$.integer(Origin/N^2)

AlgorithmIndex (Formula for N = 6)

AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + 144.integer(Origin/36)

7x7x7 Cube

7x7x7 Cube Numbering Scheme – Sequence FRULDB																															
Face F								Face R								Face U															
48	25	29	33	37	41	45		97	74	78	82	86	90	94		146	123	127	131	135	139	143									
44	24	9	13	17	21	26		93	73	58	62	66	70	75		142	122	107	111	115	119	124									
40	20	8	1	5	10	30		89	69	57	50	54	59	79		138	118	106	99	103	108	128									
36	16	4	**0**	2	14	34		85	65	53	**49**	51	63	83		134	114	102	**98**	100	112	132									
32	12	7	3	6	18	38		81	61	56	52	55	67	87		130	110	105	101	104	116	136									
28	23	19	15	11	22	42		77	72	68	64	60	71	91		126	121	117	113	109	120	140									
47	43	39	35	31	27	46		96	92	88	84	80	76	95		145	141	137	133	129	125	144									
Face L								Face D								Face B															
195	172	176	180	184	188	192		244	221	225	229	233	237	241		293	270	274	278	282	286	290									
191	171	156	160	164	168	173		240	220	205	209	213	217	222		289	269	254	258	262	266	271									
187	167	155	148	152	157	177		236	216	204	197	201	206	226		285	265	253	246	250	255	275									
183	163	151	**147**	149	161	181		232	212	200	**196**	198	210	230		281	261	249	**245**	247	259	279									
179	159	154	150	153	165	185		228	208	203	199	202	214	234		277	257	252	248	251	263	283									
175	170	166	162	158	169	189		224	219	215	211	207	218	238		273	268	264	260	256	267	287									
194	190	186	182	178	174	193		243	239	235	231	227	223	242		292	288	284	280	276	272	291									

AlgorithmIndex (General Formula for N odd)
Case 1: Destination = 0
AlgorithmIndex = 24 + Orientation + $(4.N^2)$.integer(Origin/N^2)
Case 2: Destination <> 0
AlgorithmIndex = 24 + ((Origin modulo N^2) – 1) modulo 4 + 4.Destination + $(4.N^2)$.integer(Origin/N^2)
AlgorithmIndex (Formula for N = 7)
Case 1: Destination = 0
AlgorithmIndex = 24 + Orientation + 196.integer(Origin/49)
Case 2: Destination <> 0
AlgorithmIndex = 24 + ((Origin modulo 49) – 1) modulo 4 + 4.Destination + 196.integer(Origin/49)

8x8x8 Cube

8x8x8 Cube Numbering Scheme – Sequence FRULDB																									
Face F								Face R								Face U							
63	36	40	44	48	52	56	60	127	100	104	108	112	116	120	124	191	164	168	172	176	180	184	188
59	35	16	20	24	28	32	37	123	99	80	84	88	92	96	101	187	163	144	148	152	156	160	165
55	31	15	4	8	12	17	41	119	95	79	68	72	76	81	105	183	159	143	132	136	140	145	169
51	27	11	**0**	**1**	5	21	45	115	91	75	**64**	**65**	69	85	109	179	155	139	**128**	**129**	133	149	173
47	23	7	**3**	**2**	9	25	49	111	87	71	**67**	**66**	73	89	113	175	151	135	**131**	**130**	137	153	177
43	19	14	10	6	13	29	53	107	83	78	74	70	77	93	117	171	147	142	138	134	141	157	181
39	34	30	26	22	18	33	57	103	98	94	90	86	82	97	121	167	162	158	154	150	146	161	185
62	58	54	50	46	42	38	61	126	122	118	114	110	106	102	125	190	186	182	178	174	170	166	189
Face L								Face D								Face B							
255	228	232	236	240	244	248	252	319	292	296	300	304	308	312	316	383	356	360	364	368	372	376	380
251	227	208	212	216	220	224	229	315	291	272	276	280	284	288	293	379	355	336	340	344	348	352	357
247	223	207	196	200	204	209	233	311	287	271	260	264	268	273	297	375	351	335	324	328	332	337	361
243	219	203	**192**	**193**	197	213	237	307	283	267	**256**	**257**	261	277	301	371	347	331	**320**	**321**	325	341	365
239	215	199	**195**	**194**	201	217	241	303	279	263	**259**	**258**	265	281	305	367	343	327	**323**	**322**	329	345	369
235	211	206	202	198	205	221	245	299	275	270	266	262	269	285	309	363	339	334	330	326	333	349	373
231	226	222	218	214	210	225	249	295	290	286	282	278	274	289	313	359	354	350	346	342	338	353	377
254	250	246	242	238	234	230	253	318	314	310	306	302	298	294	317	382	378	374	370	366	362	358	381
AlgorithmIndex (General Formula for N even)																									
AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + $(4.N^2)$.integer(Origin/N^2)																									
AlgorithmIndex (Formula for N = 8)																									
AlgorithmIndex = 24 + Origin modulo 4 + 4.Destination + 256.integer(Origin/64)																									