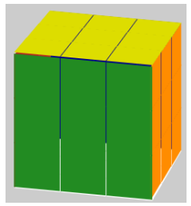
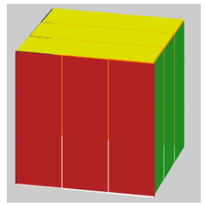
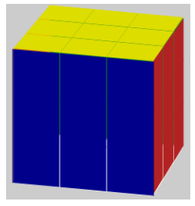
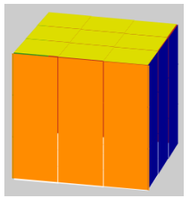
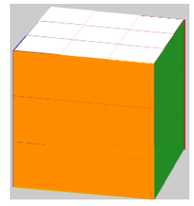
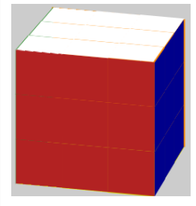
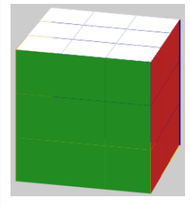
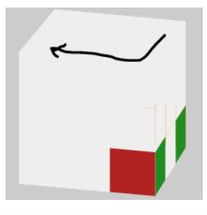
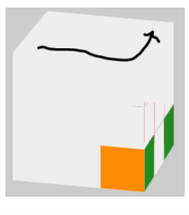
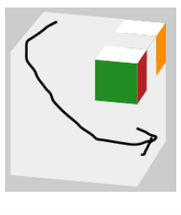
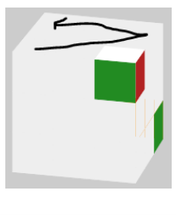
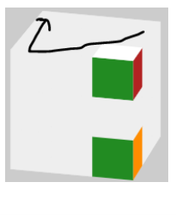
LL permutation

Assign names to each corner piece (assuming DRB = white-blue-orange).
LL corners are types of civilizations (clockwise LL permutation follows progression of civilizations); D layer corners are Star Wars characters.
Name of each corner is supposed to roughly represent its colors.

							
Hunter/gatherer	Agricultural	Industrial	Space-faring		Yoda	Palpatine	Anakin

LL prediction

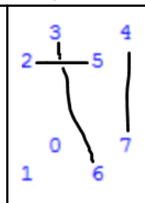
Read out the ordered permutation of 3 LL corners based on where Yoda and Anakin are (if Palpatine is unsolved, pretend he is whatever corner is currently at DLF). Figure out what swap would be necessary to make this permutation match clockwise LL corner permutation.

								
Both solved - any 3 corners on U in clockwise order	Both swapped: any 3 corners on U in counterclockwise order	Anakin-Yoda on U: the remaining 2 corners on U in clockwise order + DRF	Yoda-Anakin on U: the remaining 2 corners on U in counterclockwise order + DRB	Diag on U: DRF + the corner on U immediately clockwise of Yoda + DRB	Solved Anakin: beginning diagonally across from Yoda, 3 LL corners in counterclockwise order	Confused Anakin: beginning diagonally across from Yoda, 3 LL corners in clockwise order	Solved Yoda: beginning clockwise of Anakin, 3 LL corners in counterclockwise order	Confused Yoda: beginning counterclockwise of Anakin, 3 LL corners in clockwise order

Note that for the (solved, confused) x (Yoda, Anakin) cases: any type of Anakin permutation begins diagonally across from Yoda on the U layer, and any type of Yoda permutation begins opposite of the direction the permutation is traced out (so if the permutation goes clockwise, you begin directly counterclockwise of Anakin, and vice versa). Also note that solved cases will always go counterclockwise while confused cases will go clockwise.

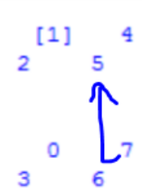
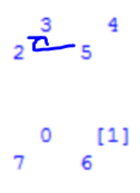
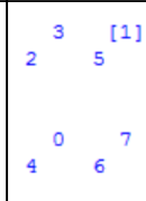
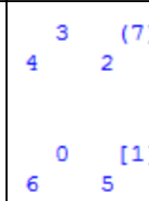
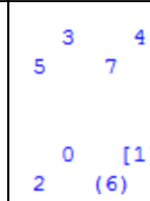
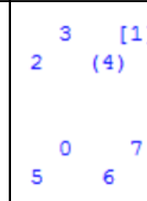
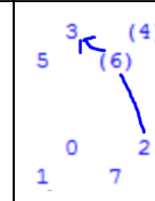
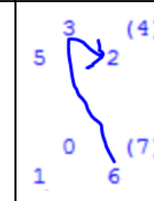
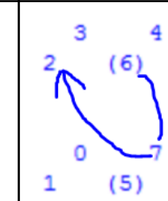
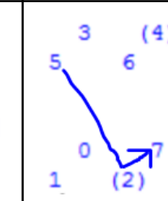
Swap standardization

If Palpatine is unsolved, the predicted swap above must be rewritten into one that involves him. If Palpatine is solved, the predicted swap above must be rewritten into a swap of two corners on R. (Without this step, tracking becomes much more difficult.)
All (6 nCr 2) = 15 possible swaps can be captured by 5 triplets below. A swap in a triplet can be safely rewritten as any of the other two. If the swap you have is not in standard form, look for it in the table below and find a rewrite that is. The mnemonics given are ways of generating two of the 3 swaps in each triplet.

				
(R U R’ U’)3	U2	(R’ U’ R U)3	R2 U2 R2	R2

Tracking F

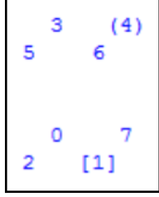
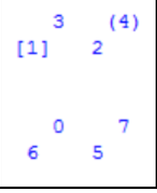
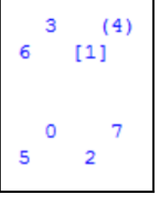
Memorize 15 different 4-cycles to learn what F moves do. Each cell below represents a 4-cycle. The diagram represents start state; arrow shows which swaps are cycled between when an F move is applied (since Palpatine just moves around in response to the F move, the arrows track the swap piece).

Paths (DLF not on F): when DLF @ ULB (and it's mirror across UR, DRB), cycling between 3 swap states + no-swap. When swap is URB-DRB, cycling between 3 other swaps.				**Singularities**: unaffected by F moves.				**Creatures** (DLF on F): cycling between 3 swap states + an initial swap on R. You can use "where in the arrow" a piece is to determine where DLF would have to be, since all the diagrams start with DLF solved. E.g. Seahorse's 3 in the middle of the arrow means that DLF will be an F2 away from solved; i.e., at URF.				
Launch	Cloud	Point	Spaz	Sun	Obelisk	Chevron	Spacecraft	Tripod	Seahorse	Horseshoe	Twister	Ground
												

[ULB]-URB, [ULB]-ULF are also singularities

Reduction Strategy

Making use of the Point path above, use <R,U> to move swap pieces around, and then finish reduction with one the following based on DLF's orientation:

F	F'	F2
		

Intermediate: learn optimal solutions for no-swap and DLF-solved cases (these tend to be pretty annoying). 3 CO * 7 CP no-swaps + 3 CO * 6 CP DLF-solved = 39 cases.

Algorithms for special cases

df

Annotated Solves (2x2)

z2 y' F2 R' U' R F' R2 F2 R2 U
z2 y' F' R' U R' F' R' F U2 R' U'
z2 y' U F U2 R F2 R2 U R2 U
z2 y' U2 R' U' F2 U2 F' U F R2
z2 y' F' R2 U' R2 U2 R2 U' F' R'

Example Solves (3x3)

Click here for running statistics on movecount.

1: R' D' L' D F2 D' B' U F D2 B' L' U2 F' U2 F' U2 F L D2
x y' // insp
r' U R F' u' r' U' f2 // EOPair 8/8
r F' r' // 2GLine 3/11
U' R E2 R' U' r2 // Block 6/17
R' U2 R U R' U2 R U' R U R U R // F2L 13/30
R' U2 R U R' U R U R U R' U R U2 R' U2 // 2GLL 16-3/**43**

2: R2 F2 L2 B' R2 B L2 F' L2 R2 F2 D' L R2 B' L R F L2 D' R'
x y // insp
U' F' U2 R' u' r u R' f2 // EOPair 9/9
U2 r' F r // 2GLine 4/13
U u2 r2 R' u' R2 u // Block 7/20
U' R U' R' U R' U R' U2 R U2 R' U R // F2L 14/34
U R U2 R' U' R U' R' U R U R' U R U2 R' U' // 2GLL 17/**51**

3: F2 D2 U2 F2 L' R2 F2 R' F2 U2 R2 F' L2 U' B F2 U R U' F R2
x' // insp
R' u' F' u r u' R S2 // EOPair 8/8
R2 U' R U' F2 // 2GLine 5/13
R' u' R2 u' M2 U2 R U r2 // Block 9/22
U2 R2 U' R U2 R U R U R // F2L 10/32
M2 U M2 U2 M2 U M2 U' // 2GLL 8/**40**

4: B2 U L B2 R U' B U2 B2 L2 F2 L' F R U2 R2 B L U2 R
z' y' // insp
F' r u' F' r U' f' r // EOPair 8/8
R' F2 D R' D' // 2GLine 5/13
R u R2 u' U' r2 U r2 // Block 8/21
U R U' R2' U' R U R2 U R2 // F2L 10/31
U2 R U' R' U' R U2 R' U2 R U R' U2 R U R' U' // 2GLL 17/**48**

5: F U B D2 L' U2 D2 B2 L' U' L2 D2 B2 U2 D2 F' R2 B' R2 U2 F2
x2 y' // insp
R u R' f R2 F U f' u f2 // EOPair 10/10
R' U' F2 // 2GLine 3/13
U' r2 U' R u2 M2 U' R2 U' r2 // Block 10/23
U R' U2 R U R' U2' R U R' U' R U z' // F2L 13/36
U' R U2 R' U' R U R' U' R U R' U' R U' R' U // 2GLL 17/**53**

6: U2 F R L' D L2 B' R F B2 R L2 B2 D2 R D2 R L2 B2
z y' // insp
f U' R F U r f2 // EOPair 7/7

E' F2 R' U R' F2 // 2GLine 6/13
M2 U' R' u2 R' r2 U2 R' U M2 // Block 10/23
U' R' U2 R' U R U' R' U R' U' R U R' // F2L 15/37
U R2 F2 R' U2 R' U' R U' R F2 R2 U2 // 2GLL 13/**50**

7: R2 D2 L2 D F2 U' R2 U F2 U' F' U2 B D R' F2 R D L2 F
x' // insp
R U2 R' f r U' f2 // EOPair 7/7
R U R' L' U' L // 2GLine 6/13
r2 U2 R U' r2 R u2 R' u2 // Block 9/22
U R2 U R' U' R U z' // F2L 7/29
U2 R' U' F R' F' R2 U' R' U F' U F R U2 // 2GLL 15/**44**

8: F2 R2 B2 L2 F2 D L2 D2 F2 U R2 F' U F' D R F L F L2 U2
z y // insp
U u2 r' f r f // EOPair 6/6
u r F r // 2GLine 4/10
U' M2 E2 // Block 3/13
R2 U2 R' U' R2 U R2 U' R' U z' // F2L 10/23
R U2 R' U' R U R' U' R U' R' // 2GLL 11/**34**

9: B' R' D F2 L' F2 R' B D' L2 B2 R' D2 B' D' F L2 D2 F2 L
x2 z' // insp
f' R E r U' r f2 // EOPair 7/7
U R2 U' F2 // 2GLine 4/11
E2 R U' r2 U R u' R2 u' // Block 9/20
R' U' R U2 R' U' R' U z' // F2L 8/28
U R' U2 R2 U R U R U' R' U' R2 U R U' // 2GLL 15/**43**

10: D2 L' D' F2 D B2 L B U2 R F R U2 L2 B U' F' D2 L2 D'
x y // insp
U2 r u r' U' f2 // EOPair 6/6
r' F r // 2GLine 3/9
R U' M2 U R' u2 // Block 6/15
R U' R U R U R U' R' U R // F2L 11/26
U' R U R' U R U' R' U R U2 R' U' R2 // 2GLL+R2 14/**40**

11: U2 B2 U R2 F' U2 R D2 R' U' R B2 L' D F' D' R2 U F D'
x2 y' // insp
R r f R r2 U f' // EOPair 7/7
r U2 F2 // 2GLine 3/10
u2 R' u' r2' u R' U r' U2 r' // Block 10/20
R U' R' U R U R' U' R' U2 R // F2L 11/31
U' R2 U' R U' R2 U R2 U' R' U R2 U R2 U2 // 2GLL 15/**46**

12: F' D F' L' D R' F2 L F U' L F L2 B' D' F R2 U2 F2 D2

x2 // insp
R U F r U2 f' r // EOPair 7/7
R2' U' R F2 // 2GLine 4/11
R' u r2 u r2' U R U r2 // Block 9/20
R' U' R2 U' R U' R U R' U' R' U z' // F2L 12/32
U2 R U2 R' U2 R' U2 R' U R U' R U2 R2 U2 R' U2 // 2GLL 19/**49**

13: F2 D2 L2 D2 R' B' D2 R2 D B' R2 U2 B2 R2 F' L F' U2 R' D
x2 z' // insp
M f R F' f2 // EOPair 5/5
U' M' F M // 2GLine 4/9
U R' U r2 U' R' u2 // Block 7/16
U' R' U R' U R' U2 R2 U R' U R U' R' U' R // F2L 16/32
U' R' U' R U' R' U2 R U2 // 2GLL 9/**41**

14: B L F L2 D2 F L U L D L' F' U' F L' F2 D2 F2 L U
z2 // insp
l' U' M' U M' x' u R f2 // EOPair 8/8
U' R2 U R' U F2 // CPLine 6/14
M2 U' R u r2 E M2 U' M2 // Block 9/23
U R' U' R U' R2 U' R' U2 R U' R' U' R // F2L 14/37
U' R' U2 R U R' U R2 U2' R' U' R U' R' U // 2GLL 15/**52**

15: D R D2 F D R2 B L D2 F2 U' B' L F2 L' B' D' B L2 U
x2 // insp
R f R U' F u f' // EOPair 7/7
R' U2 F // CPLine 3/10
U2 r' u2 r U2 r // Block 6/16
R U' R U' R' U R U R2' U R // F2L 11/27
L U' L' U L2 U2 L' U' L U' L2 U L U' L' E // 2GLL 16/**43**

16: F L2 B L' U L2 D' B2 L' U2 F D F' U L' B2 D L' D2 R'
y2 z' // insp
U F M' F r' f // EOPair 6/6
u R2' U' F2 // CPLine 4/10
u2 U r u2 r U' M U2 M // Block 9/19
R U R U R' U2 R U2 R' // F2L+forceAS 9/28
U2 R' U' R U' R2 U' R' U' R2 U R U R2 U2 // 2GLL 15/**43**

17: U2 F D L2 U R D L B U2 R2 U2 B L2 D' B L' U' F' R2
x2 z // insp
M f U M' U r' U' f2 // EOPair 8/8
U R U F2 // CPLine 4/12
M2 U R' u R2 u R2 U r2 // Block 9/21
U2 R2 U2 R U' R U R2 U2' R2' // F2L 10/31
U2 R' U R2 U R' U R U2 R U2 R U R' U R2 U // 2GLL 17/**48**

18: R U2 L2 F R2 F2 U F2 L D B' R' D' R D F' U' F2 U' R
z2 // insp
r' R2 u M' U F R' f // EOPair 8/8
F2 U F u // CPLine 4/12
R U2 r2' U' r2 // Block 5/17
U R U2 R' U2 R2 U' R' U' R U R2' U' R2 // F2L 14/31
M2 U M U2 M' U M2 u2 // 2GLL 8/**39**

19: B2 L' U L' F2 L2 D2 F2 L' F D2 L U2 L' F2 R B' L D2 B
z y // insp
r' f M F u' r' S2 // EOPair 7/7
R' u' F // CPLine 3/10
r2 u' r' E2 r U' r2' // Block 7/17
R' U' R' U R2 U' R' U' R U R U z' // F2L 12/29
U' x' U' R' D R U R2 U2 R D' R' U2 R2 F // 2GLL 14/**43**

20: F R F R' U' F2 L' U2 R F' D L' U B U2 F2 L U L' F
x2 // insp
R' U2 S U f // EOPair 5/5
u' U' R' U F2 // CPLine 5/10
R u r2 u' M2 U2 R U' r2 // Block 9/19
U R2 U' R' U2 R' U2 z' // F2L 7/26
U R' U2 R U R' U R2 U2 R' U' R U' R' U' r2 // 2GLL 16/**42**

21: U F D2 R' D' R' U2 F' U2 R F R2 F L' F' R D' F2 R' B
z' // insp
r' y' R U' r U2' r2' // EOPair 6/6
U2 R2 y // CPLine 2/8
u2 U M' U2 r R2' u2 r2 // Block 8/16
U2 R' U' R U' R U' R U R2' U2 R U z' // F2L 13/29
R U R' U R U2 R' U' R U2 R' U' R U' R' r2 // 2GLL 16/**45**

22: B D2 F' L F' L' F' R B D' F U2 B2 R U' L' F L' F' R'
// insp
U' r2' u' y' R U r2' // EOPair 6/6
y R2 U F2 // CPLine 3/9
R2 u2 U R U' r2 // Block 6/15
R U2 R2 U' R2 U' z' // F2L 6/21
U2 R U R2 U' R2 U' R2 U2 R2 U2 R' U R2 // 2GLL 14/**35**

23: L2 F U F2 U' F2 D L F2 U' B' R B L B' R U F' L2 U
z' // insp
r U' R2' y' r U M2' // EOPair 6/6
y U' r F r' // CPLine 4/10
U2 r2 U r2' U R u2 // Block 7/17

U2 R' U' R' U2 R2 U2 z' // F2L 7/24
M2 U M' U2 M U M2 U' // 2GLL 8/**32**

24: F L2 D2 L B' U L F R' F' U' F D' B2 U' F R B D2 R'
y2 x // insp
F' r E' r' M U F R f2 // EOPair 9/9
R' U2 r' F r // CPLine 5/14
U' r' U2 r u2 U' R u2 // Block 8/22
R U2 z' // F2L-E 2/24
y2 U R U R' U2 R U' R' U' R U2 R' U2 R2 // L5E 14/**38**

25: B D2 L2 B D' F2 U' R B' R D F R' D' F' D B2 L2 U2 F2
x2 // insp
U' r' u' R2 y' r U2' r2' // EOPair 7/7
y r F M' U F2 // CPLine 5/12
R U' R' U' r2 R' U' R u2 // Block 9/21
R2 U' R U' R2' U' R' U R2 U' R2' U2 R2 // F2L 13/34
U' R' U' R U' R' U2 R U R' U2' R U R' U R // 2GLL 16/**50**

26: R' D' L' B U B L' F' R F' U2 R' B' L2 F2 U' B U L' U'
y' z' // insp
y' r2 U' r' R2 U' R' // EOPair 6/6
R2 U' R2' y // CPLine 3-1/8
r R u2 M' E r2' u // Block 7/15
R2 U' R2' U2 R U R2' U' R U z' // F2L 10/25
U' R U R' U R U' R' U R U' R' U R U2' R' U2 r2 // 2GLL 18/**43**

27: R2 B2 L2 D B' D2 R' F2 D F D2 R' B R' U' F2 R' D' L U'
y' // insp
R' F R u' U2 f' U f' // EOPair 8/8
E' r' F' r // CPLine 4/12
r2 U R' U r2 R' u R2 u' // Block 9/21
R U' R U R' U2 R U' R' // F2L 9/30
U2 R U' R' U2 R U R' U2 R U R' U R U' R' U2 R2 // 2GLL 18/**48**

28: D B D B2 D R U L F2 D2 F' L2 B U' F' L2 B L' U' L2
x' // insp
y' E R B' r' U M2 // EOPair 6/6
y U L' U' L // CPLine 4/10
r2 U' R2' U' M2 U2 R' u2 // Block 8/18
U R' U2 R U R2' // F2L 6/24
M2 U M2 U M' U2 M2 U2 M' U // 2GLL 11/**34**

29: B U' B2 R' B R' B2 L2 F' U2 L F2 D F2 D2 R2 D' F L2 U2
x y2 // insp
R2 U2 R U u r' // EOPair 7/7

y r' U r F' r' // CPLine 5/12
U R' u R2 u R U r2' // Block 8/20
U' R2' U' R2 U R U R // F2L 8/27
U R' U2 R U R' U' R U R' U R' // 2GLL 12/**39**

30: U L2 F' R2 F2 U2 L2 F' U2 B' D2 B' U' R' U' L U' L2 F R2
x' // insp
f u' r' u' R f' r U S2 // EOPair 9/9
R' u F u' // CPLine 3/13
R2 U R u2' R' U' r2 // Block 7/20
U R U' R' U R2 U2 R' U' R U R2' // F2L 12/32
U' R U' R' U2 R U' R U R' U R U2 R' U2 R2 // 2GLL 16/**48**

31: R U' L2 F2 D' L' D R' B' U' F' D2 L B U R2 D' B D B2
z' // insp
U' F2 R u r U' f2 // EOPair 7/7
u R2 F' u' // CPLine 4/11
u' M2 u U2 r2 U r2' // Block 7/18
R' U R U' R' U R U R' // F2L 9/27
u2 R U R' U R U2 R' U R' U' R U' R' U2 R // 2GLL 16/**43**

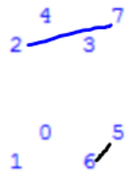
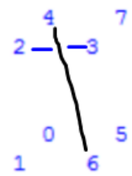
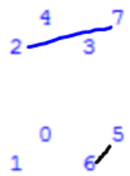
32: U' L F' D R2 D' L2 U2 L' D2 L2 B' R U R B' U2 B D2 L
x2 // insp
R U2 R' u' R' y' r2 // EOPair 6/6
U' R2' y // CPLine 2/8
U R2 u2 R' u2 R U' M2' // Block 8/16
U2 R U R U2' R U2 R U' R' // F2L 10/26
U' R U R' U R U2 R' U' R U2 R' U' R U' R' U' // 2GLL 17/**43**

33: L B' U R' F' U L' D' L2 U F' R2 F' U F' L' B' R' D2 L'
x2 // insp
F2 U2 r F' R U R' f' // EOPair 8/8
r' F2 // CPLine 2/10
R r2 u2 R' u M2 u R u R2' u // Block 11/21
R2 U R U' R U' R2 U R2' U' R U z' // F2L 12/33
U2 R' U' R U' R' U2 R U' R U R' U R U2 R' U // 2GLL 17/**50**

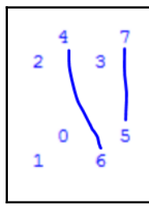
Scratch work

Correction swaps

If only A or only B is defined, look for it in the table below. C = the blue swap.

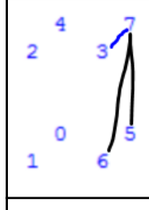
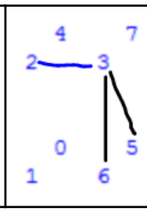
A only				
				
B only				
				

Else if A and B don't overlap and are one of the configurations shown below, C = no-swap.

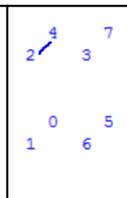
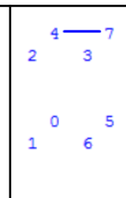
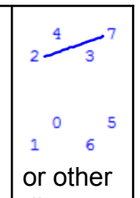
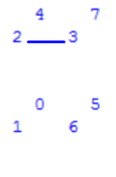
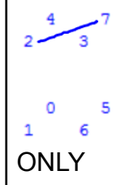
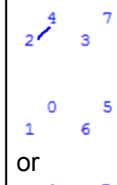
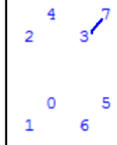
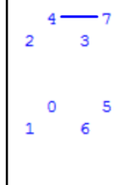
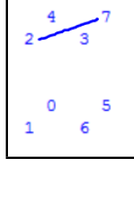
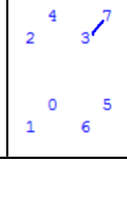
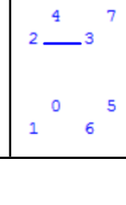
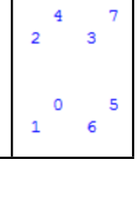
				

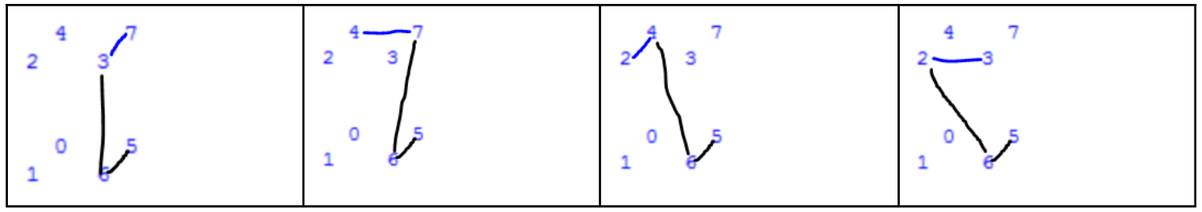
Else (if A and B don't overlap), C = the only swap s.t. none of A, B, C overlap.

Else if A and B overlap, there are 8 possibilities for C, which fall into two categories as shown below. C = the blue swap.

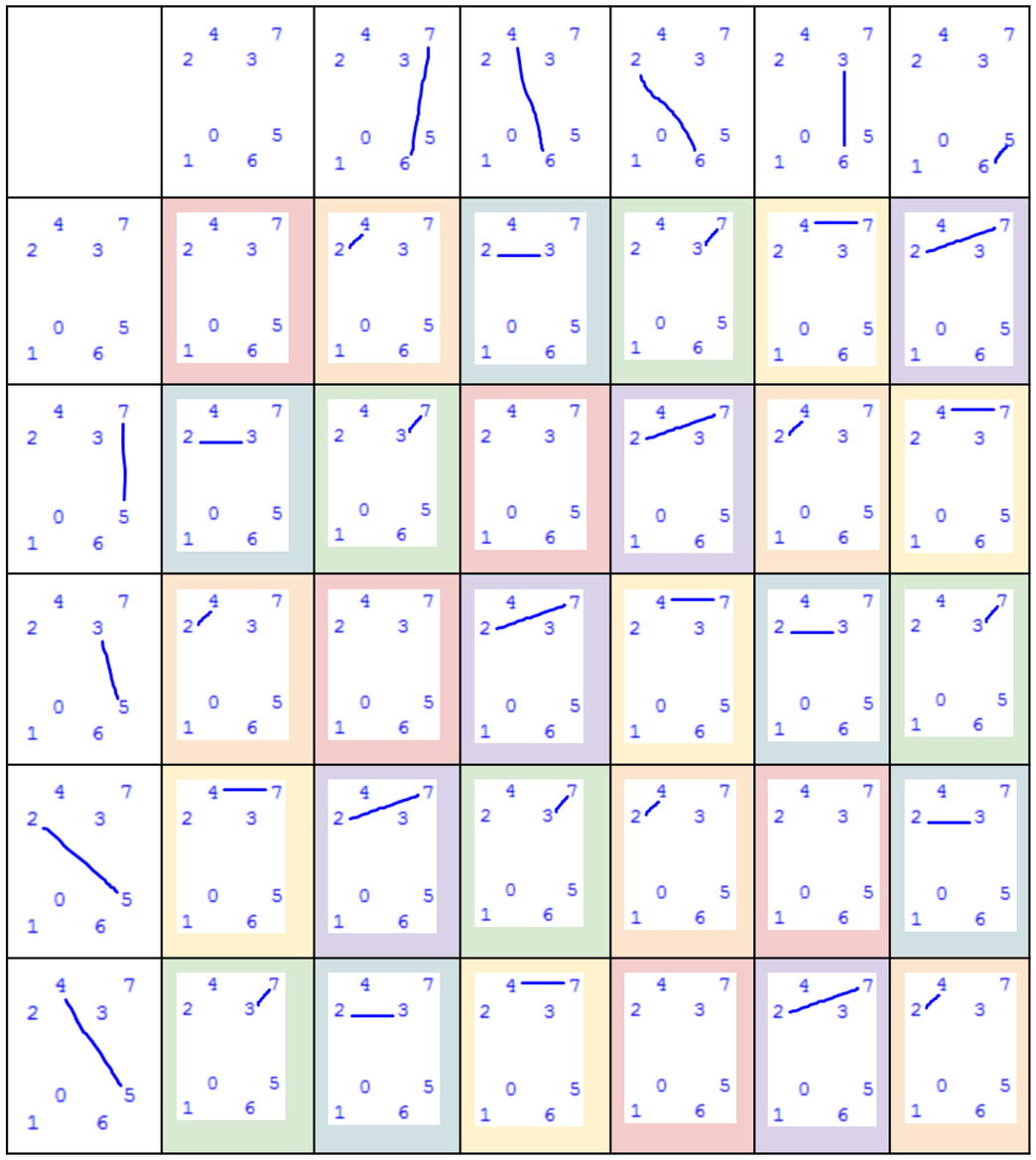
Clockwise			
			
Counterclockwise			

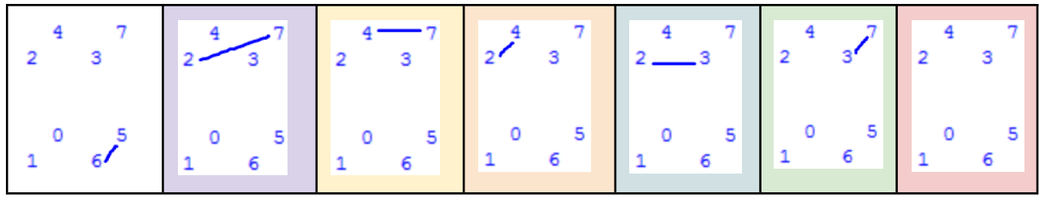
Applying corrections (LL rows, C in columns)

			 or other diag
	 ONLY	 or 	
			

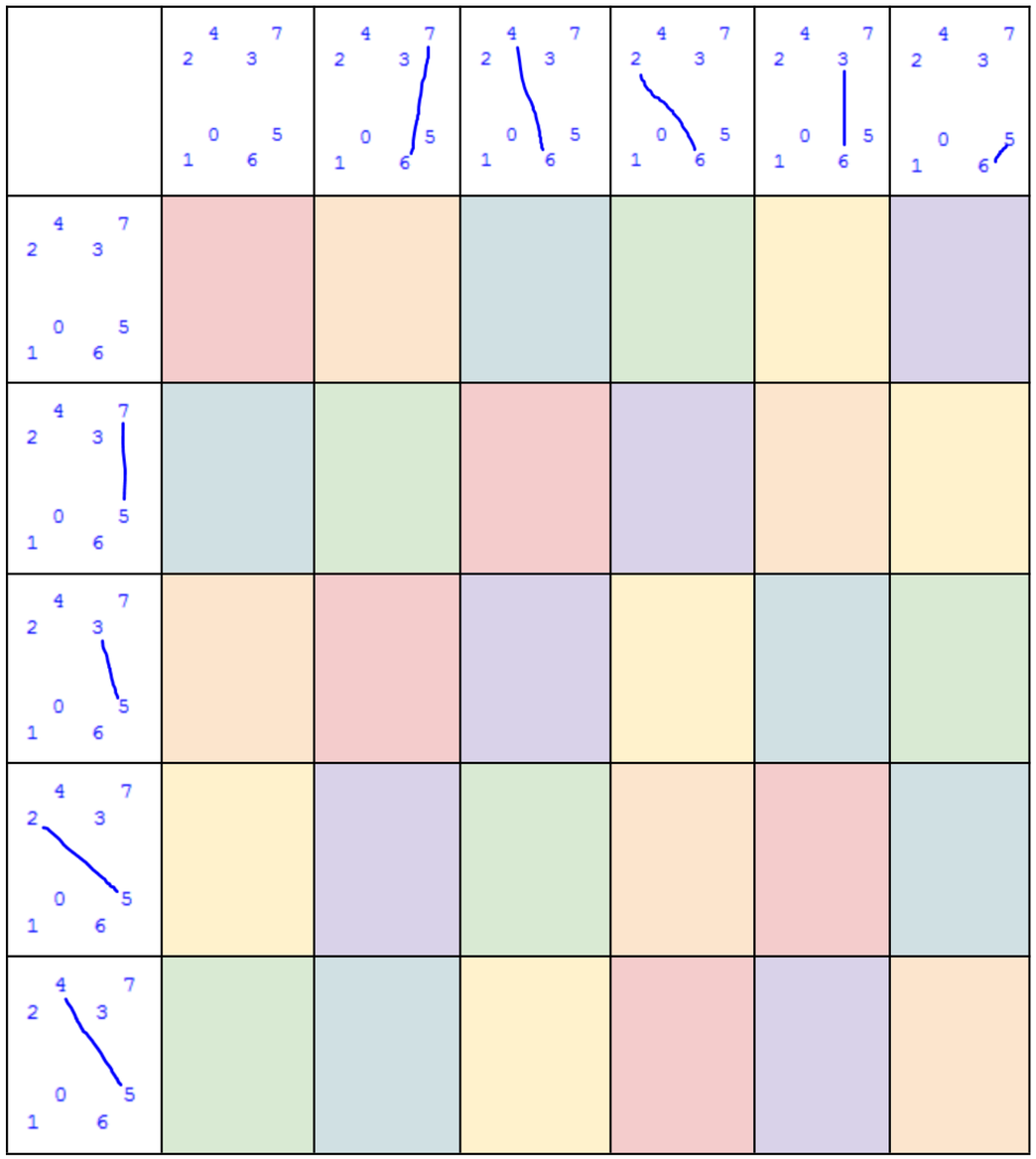


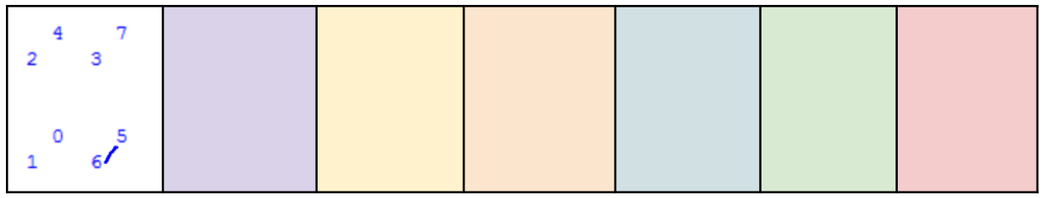
Corrections (DBR swap in rows, DFR swap in columns)





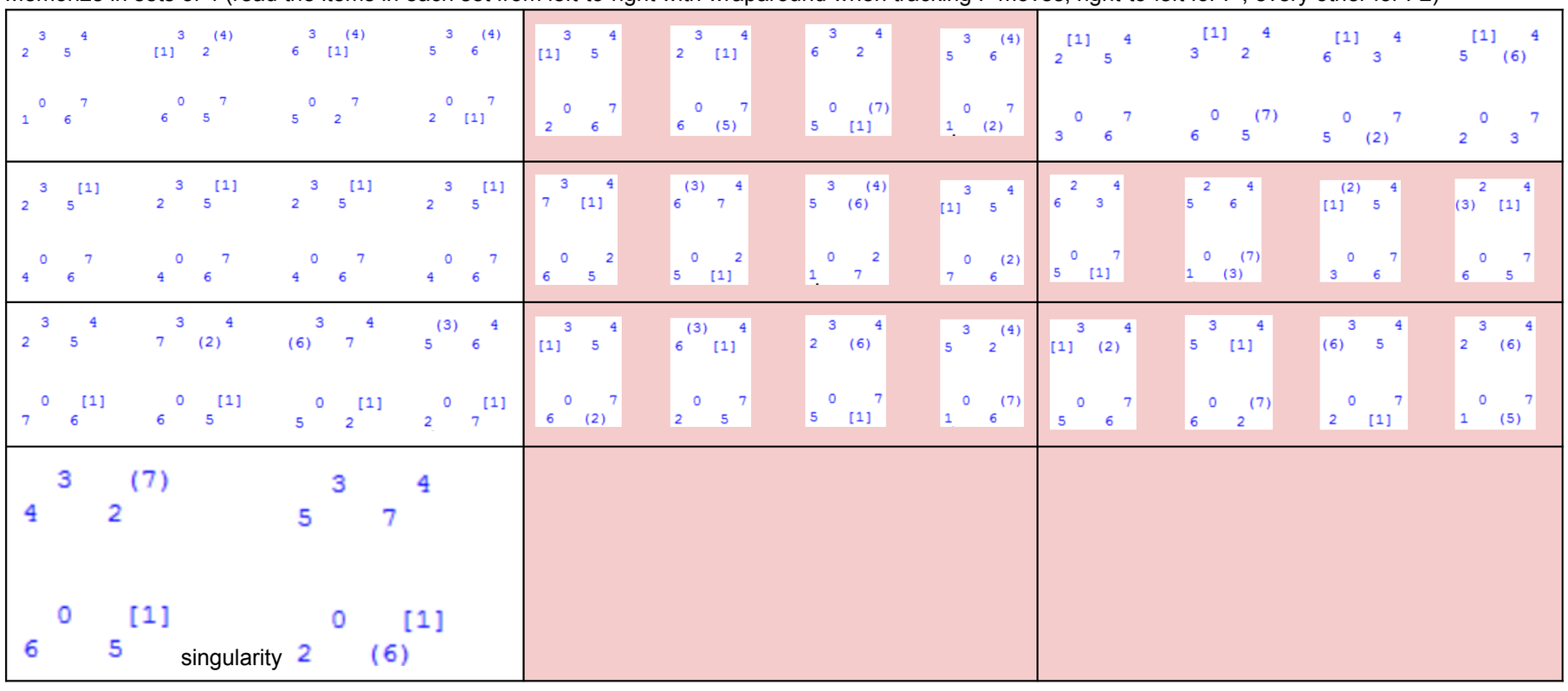
Corrections (DBR swap in rows, DFR swap in columns)

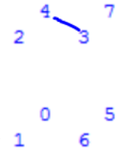




Tracking F

Memorize in sets of 4 (read the items in each set from left-to-right with wraparound when tracking F moves, right-to-left for F', every other for F2)





or

Algorithms

http://www.jaapsch.net/puzzles/subgroup.htm

DFL sticker	-ULF	-ULB	-URB	-URF	-DRF	-DRB

URF	R’ U’ R F2	U R’ U’ F2	**F2**	D R U R U R’ D’	U R U’ F2	U R2 U’ F2
FUR	R U2 R' U' F2	D R’ D’				

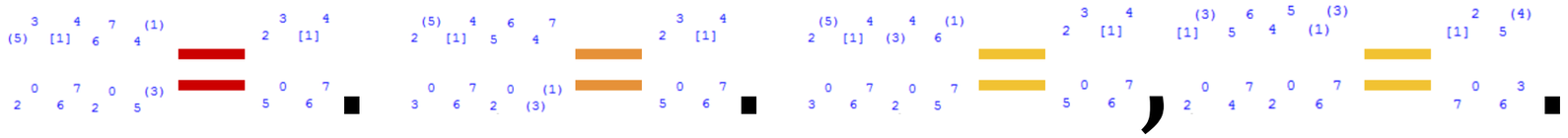
Tracing, table form

DLF @ ULF															

DLF @ URF															

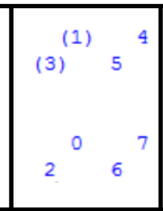
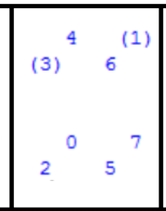
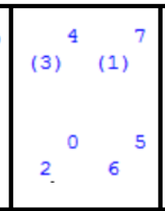
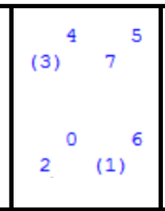
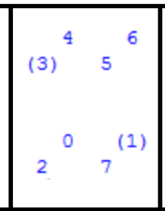
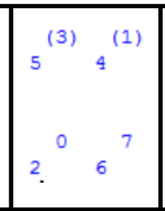
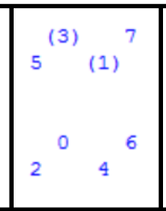
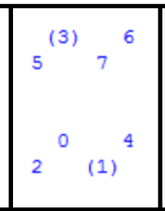
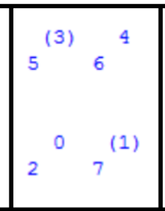
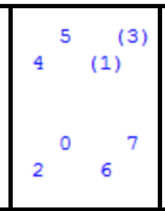
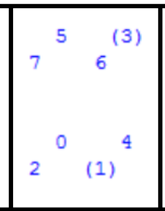
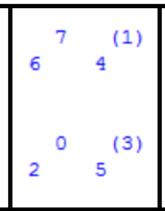
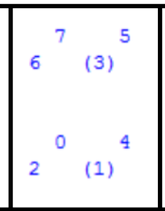
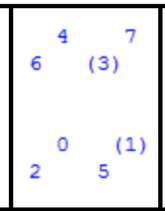
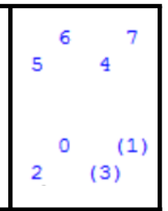
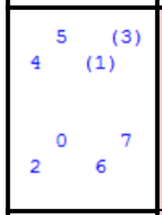
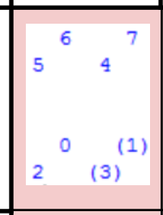
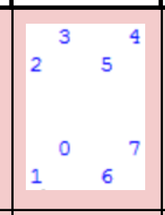
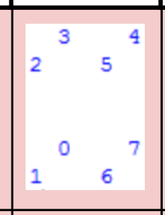
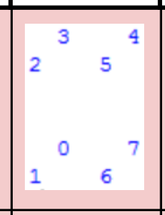
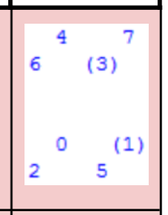
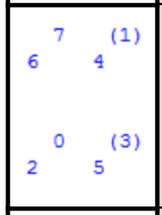
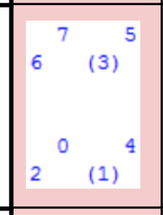
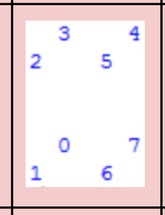
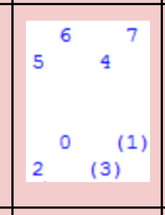
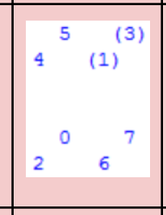
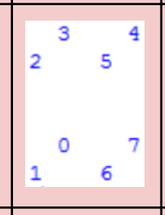
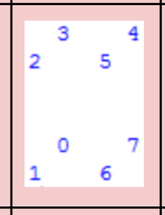
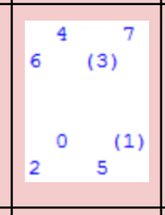
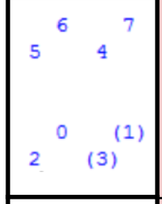
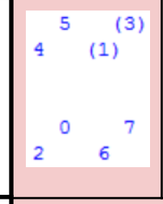
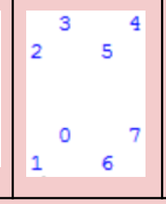
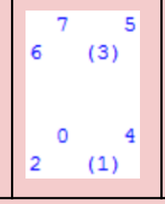
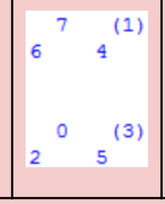
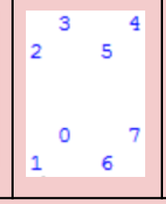
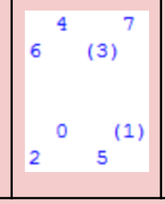
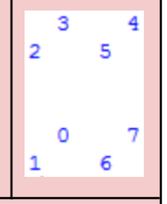
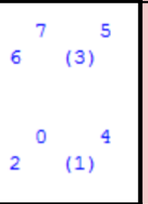
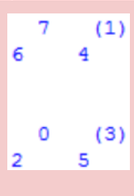
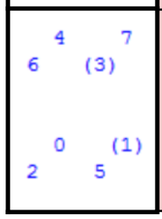
3 (5) 2 [1] 0 7 4 6	(3) 4 6 [1] 0 7 2 5	5 4 3 [1] 0 7 2 (6)	3 5 4 [1] 0 (7) 2 6	3 4 2 [1] 0 7 5 6	3 4 (5) [1] 0 7 2 6	3 5 4 [1] 0 (7) 2 6	3 4 (5) [1] 0 7 2 6	5 4 3 [1] 0 7 2 (6)	3 4 2 [1] 0 7 5 6	3 4 2 [1] 0 7 5 6	3 4 (5) [1] 0 7 2 6	(4) 3 5 [1] 0 7 2 6	(4) 3 5 [1] 0 7 2 6	5 4 3 [1] 0 7 2 (6)	3 5 4 [1] 0 (7) 2 6

Tracing, rule form (colors of ‘=’ correspond to those in table, fns are left-associative)



$$(\mathrm{R}^{n})=\mathrm{R}^{n},\quad (\mathrm{R}^{n})=\mathrm{R}^{n}.$$

Tracing (DLF unsolved)

DLF @ URF															
															
															
															
	mirror  across S														
															

DLF @ ULF (full)

	(1) 4 (3) 5 0 7 2 6	4 (1) (3) 6 0 7 2 5	4 7 (3) (1) 0 5 2 6	4 5 (3) 7 0 6 2 (1)	4 6 (3) 5 0 (1) 2 7	(3) (1) 5 4 0 7 2 6	(3) 7 5 (1) 0 6 2 4	(3) 6 5 7 0 4 2 (1)	(3) 4 5 6 0 (1) 2 7	5 (3) 4 (1) 0 7 2 6	5 (3) 7 6 0 4 2 (1)	7 (1) 6 4 0 (3) 2 5	7 5 6 (3) 0 4 2 (1)	4 7 6 (3) 0 (1) 2 5	6 7 5 4 0 (1) 2 (3)
(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	2 3 [1] (5) 0 7 4 6	2 4 [1] 3 0 7 5 (6)	2 4 [1] 5 0 (7) 6 3	2 (4) [1] 5 0 3 7 6	4 2 [1] 5 0 7 3 (6)	5 4 [1] 2 0 (7) 3 6	6 (4) [1] 5 0 7 3 2	7 4 [1] (5) 0 2 3 6	2 (5) [1] 4 0 7 3 6	3 4 [1] 5 0 7 2 6	2 7 [1] 5 0 (4) 3 6	2 4 [1] (6) 0 7 3 5	3 4 [1] 5 0 7 2 6	2 4 [1] 5 0 6 3 (7)
3 (4) [1] 5 0 6 2 7	4 2 [1] 5 0 7 3 (6)	3 4 [1] 5 0 7 2 6	2 4 [1] 5 0 (7) 6 3	2 3 [1] (5) 0 7 4 6	(3) 6 [1] 5 0 7 2 4	2 3 [1] (5) 0 7 4 6	3 4 [1] 5 0 7 2 6	(3) 6 [1] 5 0 7 2 4	5 4 [1] 2 0 (7) 3 6	(3) 6 [1] 5 0 7 2 4	5 4 [1] 2 0 (7) 3 6	4 2 [1] 5 0 7 3 (6)	4 2 [1] 5 0 7 3 (6)	7 4 [1] (5) 0 2 3 6	3 4 [1] 5 0 7 2 6
3 7 [1] (5) 0 4 2 6	2 4 [1] 5 0 (7) 6 3	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	6 (4) [1] 5 0 7 3 2	4 2 [1] 5 0 7 3 (6)	2 (4) [1] 5 0 3 7 6	4 2 [1] 5 0 7 3 (6)	3 4 [1] 5 0 7 2 6	(3) 5 [1] 5 0 7 2 4	2 4 [1] 5 0 (7) 6 3	4 2 [1] 5 0 7 3 6	3 4 [1] 5 0 7 2 6	(3) 6 [1] 5 0 7 2 4	6 (4) [1] 5 0 7 3 2	4 2 [1] 5 0 7 3 (6)
3 5 [1] 4 0 7 2 (6)	6 (4) [1] 5 0 7 3 2	2 4 [1] 5 0 (7) 6 3	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	2 3 [1] (5) 0 7 4 6	(3) 6 [1] 5 0 7 2 4	2 3 [1] (5) 0 7 4 6	2 4 [1] 5 0 (7) 6 3	3 4 [1] 5 0 7 2 6	3 4 [1] 5 0 7 2 6	2 3 [1] (5) 0 7 4 6	2 (4) [1] 5 0 3 7 6	2 (4) [1] 5 0 7 3 6	2 7 [1] 5 0 (4) 3 6	(3) 6 [1] 5 0 7 2 4
3 4 [1] 6 0 (7) 2 5	2 3 [1] (5) 0 7 4 6	4 2 [1] 5 0 7 3 (6)	2 (4) [1] 5 0 3 7 6	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	3 4 [1] 5 0 7 2 6	2 (4) [1] 5 0 3 7 6	4 2 [1] 5 0 7 3 (6)	2 (4) [1] 5 0 3 7 6	2 3 [1] (5) 0 7 4 6	2 (4) [1] 5 0 3 7 6	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	4 2 [1] 5 0 7 3 (6)	2 3 [1] (5) 0 7 4 6

DLF @ URF

	(1) 4 (3) 5 0 7 2 6	4 (1) (3) 6 0 7 2 5	4 7 (3) (1) 0 5 2 6	4 5 (3) 7 0 6 2 (1)	4 6 (3) 5 0 (1) 2 7	(3) (1) 5 4 0 7 2 6	(3) 7 5 (1) 0 6 2 4	(3) 6 5 7 0 4 2 (1)	(3) 4 5 6 0 (1) 2 7	5 (3) 4 (1) 0 7 2 6	5 (3) 7 6 0 4 2 (1)	7 (1) 6 4 0 (3) 2 5	7 5 6 (3) 0 4 2 (1)	4 7 6 (3) 0 (1) 2 5	6 7 5 4 0 (1) 2 (3)
(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	2 3 [1] (5) 0 7 4 6	2 4 [1] 3 0 7 5 (6)	2 4 [1] 5 0 (7) 6 3	2 (4) [1] 5 0 3 7 6	4 2 [1] 5 0 7 3 (6)	5 4 [1] 2 0 (7) 3 6	6 (4) [1] 5 0 7 3 2	7 4 [1] (5) 0 2 3 6	2 (5) [1] 4 0 7 3 6	3 4 [1] 5 0 7 2 6	2 7 [1] 5 0 (4) 3 6	2 4 [1] (6) 0 7 3 5	3 4 [1] 5 0 7 2 6	2 4 [1] 5 0 6 3 (7)
3 (4) [1] 5 0 6 2 7	4 2 [1] 5 0 7 3 (6)	3 4 [1] 5 0 7 2 6	2 4 [1] 5 0 (7) 6 3	2 3 [1] (5) 0 7 4 6	(3) 6 [1] 5 0 7 2 4	2 3 [1] (5) 0 7 4 6	3 4 [1] 5 0 7 2 6	(3) 6 [1] 5 0 7 2 4	5 4 [1] 2 0 (7) 3 6	(3) 6 [1] 5 0 7 2 4	5 4 [1] 2 0 (7) 3 6	4 2 [1] 5 0 7 3 (6)	4 2 [1] 5 0 7 3 (6)	7 4 [1] (5) 0 2 3 6	3 4 [1] 5 0 7 2 6
3 7 [1] (5) 0 4 2 6	2 4 [1] 5 0 (7) 6 3	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	6 (4) [1] 5 0 7 3 2	4 2 [1] 5 0 7 3 (6)	2 (4) [1] 5 0 3 7 6	4 2 [1] 5 0 7 3 (6)	3 4 [1] 5 0 7 2 6	(3) 6 [1] 5 0 7 2 4	2 4 [1] 5 0 (7) 6 3	4 2 [1] 5 0 7 3 (6)	3 4 [1] 5 0 7 2 6	(3) 6 [1] 5 0 7 2 4	6 (4) [1] 5 0 7 3 2	4 2 [1] 5 0 7 3 (6)
3 5 [1] 4 0 7 2 (6)	6 (4) [1] 5 0 7 3 2	2 4 [1] 5 0 (7) 6 3	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	2 3 [1] (5) 0 7 4 6	(3) 6 [1] 5 0 7 2 4	2 3 [1] (5) 0 7 4 6	2 4 [1] 5 0 (7) 6 3	3 4 [1] 5 0 7 2 6	3 4 [1] 5 0 7 2 6	2 3 [1] (5) 0 7 4 6	2 (4) [1] 5 0 3 7 6	2 (4) [1] 5 0 7 3 6	2 7 [1] 5 0 (4) 3 6	(3) 6 [1] 5 0 7 2 4
3 4 [1] 6 0 (7) 2 5	2 3 [1] (5) 0 7 4 6	4 2 [1] 5 0 7 3 (6)	2 (4) [1] 5 0 3 7 6	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	3 4 [1] 5 0 7 2 6	2 (4) [1] 5 0 3 7 6	4 2 [1] 5 0 7 3 (6)	2 (4) [1] 5 0 3 7 6	2 3 [1] (5) 0 7 4 6	2 (4) [1] 5 0 3 7 6	(3) 6 [1] 5 0 7 2 4	3 4 [1] 5 0 7 2 6	4 2 [1] 5 0 7 3 (6)	2 3 [1] (5) 0 7 4 6