In[49]:=
```
(*
Input:   a vector v in {-1, 1}^n;
Output:  a unique integer via Sum_{i=0}^n 2^(i-1)(v_i+1)
*)
encodeAsInteger[v_] := Dot[(v + 1) / 2 , 2^(Range[0, Length[v] - 1])]

(*
Input:
 n:   the number of ligand sites (an integer);
 int: the label of a particular configuration (0 ≤ int ≤ 2^n - 1);
Output:
 a vector in {-1, 1}^n (the opposite transformation from encodeAsInteger)
*)
decodeInteger[n_, int_] := PadRight[Reverse[IntegerDigits[int, 2]], n] * 2 - 1

(*
Input:   a list of vectors in {-1, 1}^n;
Output:
 a representative from the list that does not depend on the order of the list;
Note:    here we take the one with the minimum encodeAsInteger value
*)
uniqueRepresentative[list_] := MinimalBy[list, encodeAsInteger][[1]]

(*
Input:
 generators: a list of signed permutation matrices;
Output:  the subgroup generated by those signed permutation
  matrices (all possible applications in all possible orders)
*)
generateSubgroup[generatorMatrices_] := Module[{groupSoFar , groupSize = -1},
  groupSoFar = DeleteDuplicates @
    Append[generatorMatrices , IdentityMatrix[Length[generatorMatrices[[1]]]]];
  While[Length[groupSoFar] ≠ groupSize ,
   groupSize = Length[groupSoFar];
   groupSoFar = DeleteDuplicates[(Dot @@ #) &/@ Tuples[groupSoFar , 2]];
  ];
  SortBy[groupSoFar , Minus]
 ]

(*
Input:
 v: a vector v in {-1, 1}^n;
 group: a subgroup of the n x n signed permutations matrices;
Output:
 the set { G.v } taken over all G in the subgroup [the orbit of v under G]
*)
```

```
groupOrbit[v_, group_] := DeleteDuplicates[#.v & /@ group]

(*
Input:
 group: a subgroup of the n x n signed permutations matrices;
Output: all possible group orbits
*)
allGroupOrbits[group_] :=
 groupOrbit[#, group] & /@ Tuples[{-1, 1}, Dimensions[group][[-1]]]

(*
Input:
 group: a subgroup of the n x n signed permutations matrices;
Output: the representatives of all orbits,
indexed by the length of the orbit
*)
repsByLength[group_] := Module[{groupedByLength},
  groupedByLength = Map[encodeAsInteger[uniqueRepresentative[#]] &,
    GroupBy[allGroupOrbits[group], Length], {2}];
  Sort /@ DeleteDuplicates /@ groupedByLength
 ]

(*
Input:
 signedPermutation : a signed permutation;
Output: the signed permutation matrix corresponding to it
*)
toMatrix[signedPermutation_] := Module[{toReturn, n, i},
  n = Length[signedPermutation];
  toReturn = ConstantArray[0, {n, n}];
  For[i = 1, i ≤ n, ++i,
   toReturn[[i, Abs[signedPermutation[[i]]]]] = Sign[signedPermutation[[i]]];
  ];
  toReturn
 ]

(*
Input:
 group: a subgroup of the n x n signed permutation
  matrices (presumably generated from the rigid motions);
 reflectionMatrix : a signed permutation matrix representing a reflection;
Output: a list of chiral pairs
*)
chiralPairs[group_, reflectionMatrix_] := Module[{group2, set1, set2, diff},
  group2 = generateSubgroup[Append[group, reflectionMatrix]];
```

```
  set1 = DeleteDuplicates[uniqueRepresentative /@ allGroupOrbits[group]];
  set2 = DeleteDuplicates[uniqueRepresentative /@ allGroupOrbits[group2]];
  diff = Complement[set1, set2];
  {#, uniqueRepresentative[groupOrbit[reflectionMatrix.#, group]]} & /@ diff
 ]

(*
Input:
 group: a subgroup of the n x n signed permutation
  matrices (presumably generated from the rigid motions);
 reflectionMatrix : a signed permutation matrix representing a reflection;
Output: the representatives of all orbits,
indexed by the length of the orbit and grouped into chiral pairs
*)
repsByLengthWithChiral[group_, reflectionMatrix_] :=
 Module[{combine, chiralPairList},
  combine = Function[{singletons, pairs}, Join[Complement[singletons,
      Flatten[pairs]], Select[pairs, MemberQ[singletons, #[[1]]] &]]];
  chiralPairList = Sort[Sort /@ Map[encodeAsInteger,
      chiralPairs[group, reflectionMatrix], {2}]];
  combine[#, chiralPairList] & /@ repsByLength[group]
 ]
```

In[28]:=
```
(* 3 x M and 6 x L *)
(* M's: a, b, c (clockwise) *)
(* L's:
  1: {a → b, top}, 2: {a → c, top}, 3: {b → c, top},
4: {a → b, bot}, 5: {a → c, bot}, 6: {b → c, bot}
*)
(* rotate in the plane: a → b → c → a; bot, top stay same *)
t1 = toMatrix[{3, -1, -2, 6, -4, -5}];
(* rotate around axis through c: a ↔ b and top ↔ bot; c stays same *)
t2 = toMatrix[{-4, 6, 5, -1, 3, 2}];
(* reflect across plane through all M's: top ↔ bottom; rest stay same *)
t3 = toMatrix[{4, 5, 6, 1, 2, 3}];

g1 = generateSubgroup[{t1, t2}];
reps = repsByLengthWithChiral[g1, t3]
Total[Length[Flatten[#]] & /@ reps]
(* Total number of isomers (distinct with respect to rotation) *)
Total[Length[Select[#, ListQ]] & /@ reps](* Total number of chiral pairs *)
```

Out[32]=
```
<|6 → {0, {2, 13}, {3, 6}, {5, 10}}, 3 → {{1, 8}, {4, 11}, {7, 14}}, 2 → {18}, 1 → {{21, 42}}|>
```

Out[33]=
```
16
```

Out[34]= 7

In[35]:=
```
(* 4 x M and 8 x L *)
(* Vertices a (bottom left), b (bottom back), c (bottom right), d (top) *)
(* 1: {a → b, top}, 2: {a → b, bot}, 3: {a → c}, 4: {b → c},
5: {a → d}, 6: {b → d}, 7: {c → d, left}, 8: { c → d, right} *)

(* rotate around axis through midpoints of 3 and 6: a ↔ c,
b ↔ d, bot ↔ left, top ↔ right *)
t1 = toMatrix[{8, 7, -3, -5, -4, -6, 2, 1}];
(* rotate around axis through midpoints of 4 and 5: a ↔ d,
b ↔ c, top ↔ left, bot ↔ right *)
t2 = toMatrix[{-7, -8, -6, -4, -5, -3, -1, -2}];
(* reflection over plane slicing (1, 2) in half and separating (7, 8): a ↔ b,
left ↔ right; everything else remains same. *)
t3 = toMatrix[{-1, -2, 4, 3, 6, 5, 8, 7}];

g1 = generateSubgroup [{t1, t2}];
reps = repsByLengthWithChiral [g1, t3]
Total[Length[Flatten[#]] & /@ reps]
(* Total number of isomers (distinct with respect to rotation) *)
Total[Length[Select[#, ListQ]] & /@ reps](* Total number of chiral pairs *)
```

Out[39]=
```
<| 4 → {{0, 3}, {1, 2}, {4, 11}, {5, 10}, {6, 9}, {7, 8}, {12, 15},
    {13, 14}, {16, 35}, {17, 34}, {18, 33}, {19, 28}, {20, 43}, {21, 42}, {22, 41},
    {24, 27}, {25, 38}, {26, 37}, {29, 46}, {30, 45}, {49, 50}, {53, 58}, {54, 57},
    {61, 62}, {69, 109}, {70, 137}, {73, 93}, {74, 133}, {77, 85}, {78, 141}},
 2 → {89, 101, {65, 125}, {66, 129}, {90, 153}}|>
```

Out[40]= 68

Out[41]= 33

In[42]:=
```
(* 6 x M and 12 x L *)
(* Vertices M1, M2, ..., M6 oriented like a 6-sided die *)
(* 1: M1 → M2, 2: M1 → M3, 3: M1 → M4, 4: M1 → M5, 5: M2 → M3, 6: M2 → M4,
   7: M2 → M6, 8: M3 → M5, 9: M3 → M6, 10: M4 → M5, 11: M4 → M6, 12: M5 → M6
*)
(* Rotate M1 → M2 → M6 → M5 → M1; M3, M4 stay fixed *)
t1 = toMatrix[{7, 5, 6, -1, -9, -11, -12, -2, 8, -3, 10, 4}];
(* Rotate M1 → M3 → M6 → M4 → M1; M2, M5 stay fixed *)
t2 = toMatrix[{-5, 9, -2, 8, 7, -1, 6, -12, -11, 4, 3, -10}];
(* reflection over plane that goes through M1, M2, M6, M5 (so M3 ↔ M4 *)
t3 = toMatrix[{1, 3, 2, 4, 6, 5, 7, 10, 11, 8, 9, 12}];

g1 = generateSubgroup [{t1, t2}];
reps = repsByLengthWithChiral [g1, t3]
Total[Length[Flatten[#]] & /@ reps]
(* Total number of isomers (distinct with respect to rotation) *)
Total[Length[Select[#, ListQ]] & /@ reps](* Total number of chiral pairs *)
```

Out[46]=
```
<| 24 → {0, 1, 6, 7, 8, 9, 14, 31, 49, 50, 55, 59, 64, 65, 70, 71, 72, 73, 94, 206, 215,
    222, 238, 301, 313, {2, 4}, {3, 5}, {10, 12}, {11, 13}, {16, 32}, {17, 33}, {18, 36},
    {19, 37}, {20, 34}, {21, 35}, {22, 38}, {23, 39}, {24, 40}, {25, 41}, {26, 44}, {27, 45},
    {28, 42}, {29, 43}, {30, 46}, {51, 53}, {66, 68}, {67, 69}, {74, 76}, {75, 77}, {80, 96},
    {81, 97}, {82, 100}, {83, 101}, {84, 98}, {85, 99}, {86, 102}, {87, 103}, {88, 104},
    {89, 105}, {90, 108}, {91, 109}, {92, 106}, {93, 107}, {145, 453}, {147, 449},
    {151, 223}, {192, 365}, {193, 220}, {194, 361}, {195, 450}, {196, 433}, {197, 212},
    {198, 582}, {199, 443}, {200, 333}, {202, 329}, {204, 462}, {208, 229}, {209, 323},
    {210, 291}, {211, 355}, {213, 339}, {214, 307}, {216, 225}, {217, 322}, {218, 290},
    {219, 221}, {224, 436}, {227, 458}, {228, 432}, {230, 316}, {297, 312}, {298, 435}},
 12 → {15, 48, 54, 57, 78, 315, 317, 330, {150, 550}, {231, 599}},
 6 → {63, 119, 159, {144, 485}},
 8 → {467, {232, 600}, {289, 440}, {345, 838}}, 2 → {1337}|>
```

Out[47]= 186

Out[48]= 74

In[59]:=
```
12 * 7 + 6 * 6 + 4 + 4
```

Out[59]= 128