GAP_Examples

```
9^9;
```
```
387420489
```
```
a := 8*8;
```
```
64
```
```
a^2;
```
```
4096
```
```
a = 63;
```
```
false
```
```
list := [2,4,5];
```
```
[ 2, 4, 5 ]
```
```
Length(list);
```
```
3
```
```
2*list;
```
```
[ 4, 8, 10 ]
```
```
list^2;
```
```
45
```
```
Add(list, 7);
```

```
list;
```
```
[ 2, 4, 5, 7 ]
```
```
Append(list, list);
```

```
list;
```
```
[ 2, 4, 5, 7, 2, 4, 5, 7 ]
```
```
Product(list);
```
```
78400
```
```
Sort(list);
```

```
list;
```
```
[ 2, 2, 4, 4, 5, 5, 7, 7 ]
```
```
list := [5,3,8,7];;
```

```
Sortex(list);
```
```
(1,2)(3,4)
```
```
p := (1,2,3);
```

```
    (1,2,3)
```

```
2^p;
```

```
    3
```

```
p*(1,2);
```

```
    (2,3)
```

```
p^-1;
```

```
    (1,3,2)
```

```
p*last;
```

```
    ()
```

```
(2,3,1)
```

```
    (1,2,3)
```

```
a := (1,2); b := (1,2);
```

```
    (1,2)
    (1,2)
```

```
IsIdenticalObj(a,b);
```

```
    true
```

```
b := a;
```

```
    (1,2)
```

```
b := (1,3);
```

```
    (1,3)
```

```
a;
```

```
    (1,2)
```

```
S6 := SymmetricGroup(6);
```

```
    Sym( [ 1 .. 6 ] )
```

```
Size(S6);
```

```
    720
```

```
s6 := Group([(1,2),(1,2,3,4,5,6)]);
```

```
    Group([ (1,2), (1,2,3,4,5,6) ])
```

```
Size(s6);
```

```
    720
```

```
S6 = s6;
```

```
    true
```

```
IsIdenticalObj(S6,s6);
```

```
    false
```

```
IsAbelian(S6);
```

```
    false
```

```
IsAssociative(s6);
```

```
    true
```

```
IsCyclic(S6);
```

```
    false
```

```
c6 := Subgroup(s6, [(1,2,3,4,5,6)] );
```

```
    Group([ (1,2,3,4,5,6) ])
```

```
IsSubgroup(s6,c6);
```

```
    true
```

```
AsList(c6);
```

```
    [ (), (1,2,3,4,5,6), (1,3,5)(2,4,6), (1,4)(2,5)(3,6),
    (1,5,3)(2,6,4),
      (1,6,5,4,3,2) ]
```

```
Index(s6,c6);
```

```
    120
```

```
Index(s6, SymmetricGroup(4));
```

```
    30
```

```
d6 := DerivedSubgroup(s6);
```

```
    Group([ (1,2,3), (2,3,4), (2,4)(3,5), (2,6,4) ])
```

```
Size(d6);
```

```
    360
```

```
DerivedSubgroup(c6);
```

```
    Group(())
```

```
IsAbelian(c6);
```

```
    true
```

```
DerivedSubgroup(d6);
```

```
    Group([ (1,4)(2,3), (1,5,4,2,3), (2,5)(3,4), (2,3,6) ])
```

```
last = d6;
```

```
    true
```

```
IsPerfect(d6);
```

```
    true
```

```
IsPerfect(s6);
```

```
    false
```

```
A6 := AlternatingGroup(6);
```

```
    Alt( [ 1 .. 6 ] )
```

```
A6 = d6;
```

```
    true
```

```
SignPerm( (1,2)(3,4) );
```

```
1
```

```
SignPerm((1,2));
```

```
-1
```

```
List(AsList(d6), SignPerm);
```

```
[ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1,
  1, 1, 1, 1, 1, 1, 1, 1, 1, 1 ]
```

```
fg := FreeGroup("a","b");
```

```
<free group on the generators [ a, b ]>
```

```
x := fg.1; y := fg.2;
```

```
a
b
```

```
x^2*y*x;
```

```
a^2*b*a
```

```
Size(fg);
```

```
    infinity
```

```
IsAbelian(fg);
```

```
    false
```

```
rels := [x^2, y^3, x*y*x*y];
```

```
    [ a^2, b^3, a*b*a*b ]
```

```
fg1 := fg / rels;
```

```
    <fp group on the generators [ a, b ]>
```

```
Size(fg1);
```

```
    6
```

```
AsList(fg1);
```

```
    [ <identity ...>, a, b, a*b, a*b*a, b*a ]
```

```
x := fg1.1; y := fg1.2;
```

```
    a
    b
```

```
y^2 = x*y*x;
```

```
    true
```

```
IsCyclic(fg1);
```

```
    false
```

```
MultiplicationTable(fg1);
```

```
    [ [ 1, 2, 3, 4, 5, 6 ], [ 2, 1, 4, 3, 6, 5 ], [ 3, 6, 5, 2, 1, 4 ],
      [ 4, 5, 6, 1, 2, 3 ], [ 5, 4, 1, 6, 3, 2 ], [ 6, 3, 2, 5, 4, 1 ] ]
```

```
V := VectorSpace(Rationals, [[1,1,1], [1,0,2]]);
```

```
    <vector space over Rationals, with 2 generators>
```

```
[2,1,3] in V;
```

```
    true
```

```
Dimension(V);
```

```
    2
```

```
IsVectorSpace(Rationals);
```

```
    true
```

```
Basis(V);
```

```
    SemiEchelonBasis( <vector space of dimension 2 over Rationals>,
    [ [ 1, 1, 1 ], [ 0, 1, -1 ] ] )
```

```
BasisVectors(Basis(V));
```

```
    [ [ 1, 1, 1 ], [ 0, 1, -1 ] ]
```

```
V := GF(5)^3;
```

```
    ( GF(5)^3 )
```

```
AsList(V);
```

```
[ [ 0*Z(5), 0*Z(5), 0*Z(5) ], [ 0*Z(5), 0*Z(5), Z(5)^0 ],
  [ 0*Z(5), 0*Z(5), Z(5) ], [ 0*Z(5), 0*Z(5), Z(5)^2 ],
  [ 0*Z(5), 0*Z(5), Z(5)^3 ], [ 0*Z(5), Z(5)^0, 0*Z(5) ],
  [ 0*Z(5), Z(5)^0, Z(5)^0 ], [ 0*Z(5), Z(5)^0, Z(5) ],
  [ 0*Z(5), Z(5)^0, Z(5)^2 ], [ 0*Z(5), Z(5)^0, Z(5)^3 ],
  [ 0*Z(5), Z(5), 0*Z(5) ], [ 0*Z(5), Z(5), Z(5)^0 ], [ 0*Z(5),
Z(5), Z(5) ],
  [ 0*Z(5), Z(5), Z(5)^2 ], [ 0*Z(5), Z(5), Z(5)^3 ],
  [ 0*Z(5), Z(5)^2, 0*Z(5) ], [ 0*Z(5), Z(5)^2, Z(5)^0 ],
  [ 0*Z(5), Z(5)^2, Z(5) ], [ 0*Z(5), Z(5)^2, Z(5)^2 ],
  [ 0*Z(5), Z(5)^2, Z(5)^3 ], [ 0*Z(5), Z(5)^3, 0*Z(5) ],
  [ 0*Z(5), Z(5)^3, Z(5)^0 ], [ 0*Z(5), Z(5)^3, Z(5) ],
  [ 0*Z(5), Z(5)^3, Z(5)^2 ], [ 0*Z(5), Z(5)^3, Z(5)^3 ],
  [ Z(5)^0, 0*Z(5), 0*Z(5) ], [ Z(5)^0, 0*Z(5), Z(5)^0 ],
  [ Z(5)^0, 0*Z(5), Z(5) ], [ Z(5)^0, 0*Z(5), Z(5)^2 ],
  [ Z(5)^0, 0*Z(5), Z(5)^3 ], [ Z(5)^0, Z(5)^0, 0*Z(5) ],
  [ Z(5)^0, Z(5)^0, Z(5)^0 ], [ Z(5)^0, Z(5)^0, Z(5) ],
  [ Z(5)^0, Z(5)^0, Z(5)^2 ], [ Z(5)^0, Z(5)^0, Z(5)^3 ],
  [ Z(5)^0, Z(5), 0*Z(5) ], [ Z(5)^0, Z(5), Z(5)^0 ], [ Z(5)^0,
Z(5), Z(5) ],
  [ Z(5)^0, Z(5), Z(5)^2 ], [ Z(5)^0, Z(5), Z(5)^3 ],
  [ Z(5)^0, Z(5)^2, 0*Z(5) ], [ Z(5)^0, Z(5)^2, Z(5)^0 ],
  [ Z(5)^0, Z(5)^2, Z(5) ], [ Z(5)^0, Z(5)^2, Z(5)^2 ],
  [ Z(5)^0, Z(5)^2, Z(5)^3 ], [ Z(5)^0, Z(5)^3, 0*Z(5) ],
  [ Z(5)^0, Z(5)^3, Z(5)^0 ], [ Z(5)^0, Z(5)^3, Z(5) ],
  [ Z(5)^0, Z(5)^3, Z(5)^2 ], [ Z(5)^0, Z(5)^3, Z(5)^3 ],
  [ Z(5), 0*Z(5), 0*Z(5) ], [ Z(5), 0*Z(5), Z(5)^0 ], [ Z(5),
0*Z(5), Z(5) ],
  [ Z(5), 0*Z(5), Z(5)^2 ], [ Z(5), 0*Z(5), Z(5)^3 ],
  [ Z(5), Z(5)^0, 0*Z(5) ], [ Z(5), Z(5)^0, Z(5)^0 ], [ Z(5),
Z(5)^0, Z(5) ],
  [ Z(5), Z(5)^0, Z(5)^2 ], [ Z(5), Z(5)^0, Z(5)^3 ], [ Z(5), Z(5),
0*Z(5) ],
  [ Z(5), Z(5), Z(5)^0 ], [ Z(5), Z(5), Z(5) ], [ Z(5), Z(5), Z(5)^2
],
  [ Z(5), Z(5), Z(5)^3 ], [ Z(5), Z(5)^2, 0*Z(5) ], [ Z(5), Z(5)^2,
Z(5)^0 ],
  [ Z(5), Z(5)^2, Z(5) ], [ Z(5), Z(5)^2, Z(5)^2 ], [ Z(5), Z(5)^2,
Z(5)^3 ],
  [ Z(5), Z(5)^3, 0*Z(5) ], [ Z(5), Z(5)^3, Z(5)^0 ], [ Z(5),
Z(5)^3, Z(5) ],
  [ Z(5), Z(5)^3, Z(5)^2 ], [ Z(5), Z(5)^3, Z(5)^3 ],
  [ Z(5)^2, 0*Z(5), 0*Z(5) ], [ Z(5)^2, 0*Z(5), Z(5)^0 ],
  [ Z(5)^2, 0*Z(5), Z(5) ], [ Z(5)^2, 0*Z(5), Z(5)^2 ],
  [ Z(5)^2, 0*Z(5), Z(5)^3 ], [ Z(5)^2, Z(5)^0, 0*Z(5) ],
  [ Z(5)^2, Z(5)^0, Z(5)^0 ], [ Z(5)^2, Z(5)^0, Z(5) ],
```

```
  [ Z(5)^2, Z(5)^0, Z(5)^2 ], [ Z(5)^2, Z(5)^0, Z(5)^3 ],
  [ Z(5)^2, Z(5), 0*Z(5) ], [ Z(5)^2, Z(5), Z(5)^0 ], [ Z(5)^2,
Z(5), Z(5) ],
  [ Z(5)^2, Z(5), Z(5)^2 ], [ Z(5)^2, Z(5), Z(5)^3 ],
  [ Z(5)^2, Z(5)^2, 0*Z(5) ], [ Z(5)^2, Z(5)^2, Z(5)^0 ],
  [ Z(5)^2, Z(5)^2, Z(5) ], [ Z(5)^2, Z(5)^2, Z(5)^2 ],
  [ Z(5)^2, Z(5)^2, Z(5)^3 ], [ Z(5)^2, Z(5)^3, 0*Z(5) ],
  [ Z(5)^2, Z(5)^3, Z(5)^0 ], [ Z(5)^2, Z(5)^3, Z(5) ],
  [ Z(5)^2, Z(5)^3, Z(5)^2 ], [ Z(5)^2, Z(5)^3, Z(5)^3 ],
  [ Z(5)^3, 0*Z(5), 0*Z(5) ], [ Z(5)^3, 0*Z(5), Z(5)^0 ],
  [ Z(5)^3, 0*Z(5), Z(5) ], [ Z(5)^3, 0*Z(5), Z(5)^2 ],
  [ Z(5)^3, 0*Z(5), Z(5)^3 ], [ Z(5)^3, Z(5)^0, 0*Z(5) ],
  [ Z(5)^3, Z(5)^0, Z(5)^0 ], [ Z(5)^3, Z(5)^0, Z(5) ],
  [ Z(5)^3, Z(5)^0, Z(5)^2 ], [ Z(5)^3, Z(5)^0, Z(5)^3 ],
  [ Z(5)^3, Z(5), 0*Z(5) ], [ Z(5)^3, Z(5), Z(5)^0 ], [ Z(5)^3,
Z(5), Z(5) ],
  [ Z(5)^3, Z(5), Z(5)^2 ], [ Z(5)^3, Z(5), Z(5)^3 ],
  [ Z(5)^3, Z(5)^2, 0*Z(5) ], [ Z(5)^3, Z(5)^2, Z(5)^0 ],
  [ Z(5)^3, Z(5)^2, Z(5) ], [ Z(5)^3, Z(5)^2, Z(5)^2 ],
  [ Z(5)^3, Z(5)^2, Z(5)^3 ], [ Z(5)^3, Z(5)^3, 0*Z(5) ],
  [ Z(5)^3, Z(5)^3, Z(5)^0 ], [ Z(5)^3, Z(5)^3, Z(5) ],
  [ Z(5)^3, Z(5)^3, Z(5)^2 ], [ Z(5)^3, Z(5)^3, Z(5)^3 ] ]
```

```
Length(last);
```

```
125
```

```
vorne := (1,3,5,7)(2,4,6,8)(23,9,37,47)(22,16,38,48)(21,15,39,41);
hinten:= (25,27,29,31)(26,28,30,32)(17,11,35,45)(18,12,34,44)
(19,13,33,43);
oben  := (17,19,21,23)(18,20,22,24)(1,43,27,9)(2,42,26,10)
(3,41,25,11);
unten := (33,35,37,39)(34,36,38,40)(5,47,31,13)(6,46,30,14)
(7,45,29,15);
rechts:= (9,11,13,15)(10,12,14,16)(3,19,29,37)(4,20,28,36)
(5,21,27,35);
links := (41,43,45,47)(42,44,46,48)(1,17,31,39)(8,24,32,40)
(7,23,25,33);
```

```
rubik := Group(vorne,hinten,oben,unten,rechts,links);
```

```
<permutation group with 6 generators>
```

```
Size(rubik);
```

```
43252003274489856000
```

```
Factorial(8)*Factorial(12)*3^8*2^12;
```

```
519024039293878272000
```

```
last / Size(rubik);
```

```
    12
```

```
(3,4) in rubik;
```

```
    false
```

```
(3,9,21) in rubik;
```

```
    false
```

```
(2,22) in rubik;
```

```
    false
```

```
gens := List(GeneratorsOfGroup(rubik));
```

```
    [ (1,3,5,7)(2,4,6,8)(9,37,47,23)(15,39,41,21)(16,38,48,22),
      (11,35,45,17)(12,34,44,18)(13,33,43,19)(25,27,29,31)(26,28,30,32),

      (1,43,27,9)(2,42,26,10)(3,41,25,11)(17,19,21,23)(18,20,22,24),
      (5,47,31,13)(6,46,30,14)(7,45,29,15)(33,35,37,39)(34,36,38,40),
      (3,19,29,37)(4,20,28,36)(5,21,27,35)(9,11,13,15)(10,12,14,16),
      (1,17,31,39)(7,23,25,33)(8,24,32,40)(41,43,45,47)(42,44,46,48) ]
```

```
Append(gens, [(2,22), (3,9,21), (21,19)(3,11)(9,27)]);
```

```
Size(Group(gens));
```

```
    519024039293878272000
```

```
(20,22)(2,10) in rubik;
```

```
    false
```

```
(20,22)(2,10) in Group(gens);
```

```
    true
```

```
DerivedSeries(rubik);
```

```
    [ <permutation group of size 43252003274489856000 with 6
    generators>,
      <permutation group of size 2162600163724492800 with 5
    generators> ]
```

```
Centre(rubik);
```

```
    Group([
    (2,22)(4,16)(6,38)(8,48)(10,20)(12,28)(14,36)(18,26)(24,42)(30,34)(3\
    2,
        44)(40,46) ])
```

```
Size(last);
```

```
    2
```

```
Random(rubik);
```

```
    (1,33,3,39,15,17,19,13)(2,22)(4,32,6,18,12,40,30,20,24,8,16,44,38,26\
    ,28,46,34,
    10,42,48)(5,25,11,35,23,45,9,7)(21,47,37,43,27,29,41,31)
```

```
cent := Centralizer(rubik, Group((1,2),
(1,2,3,4,5,6,7,8,25,26,27,28,29,30,31,32)));
```

```
    <permutation group with 3 generators>
```

```
Size(cent);
```

```
    96
```

```
GeneratorsOfGroup(cent);
```

```
    [ (24,42)(40,46), (14,24,40)(36,42,46), (10,14)(20,36)(24,40)(42,46)
    ]
```

```
norm := Normalizer(rubik, Group((1,2),
(1,2,3,4,5,6,7,8,25,26,27,28,29,30,31,32)));
```

```
    <permutation group with 17 generators>
```

```
stab := Stabilizer(rubik, Group((1,2),
(1,2,3,4,5,6,7,8,25,26,27,28,29,30,31,32)));
```

```
    <permutation group with 17 generators>
```

```
List([norm, stab], Size);
```

```
    [ 156067430400, 156067430400 ]
```

```
norm = stab;
```

```
    true
```

```
IsCyclic(rubik);
```

```
    false
```

```
SmallGeneratingSet(rubik);
```

```
    [
    (1,17,13,19,39,23,25,35,11,7,41,43,29,27,47)(2,32,10,48,30,6,26,12,4\
    2,22,44,

    20,8,34,38,18,28,24)(3,37)(4,16)(5,21)(9,15)(14,40)(31,45,33)(36,46)\
    ,

    (1,41,23)(2,38,24,8,16,22,6,42,48,4)(3,35,5,47)(7,21,13,15)(9,29,37,\
    39)(10,

    44,26,30,36,40,12,20,32,18,34,14,46,28)(11,19,27)(17,31,43,45,25,33)
    ]
```

```
Length(last);
```

```
    2
```

```

```