

# Simple Groups of Small Order

## 1 Synopsis

The purpose of this project is to determine values of  $n$ ,  $1 \leq n < 250$ , for which  $n$  is the order of a simple group.

## 2 Some **GAP** commands you might find helpful

For further information about these and other commands, see the documentation.

- **Group**( *gens* )  
returns the permutation group that is generated by the list *gens*.
- **Subgroup**( *grp*, *gens* )  
returns the subgroup of *grp* generated by the list *gens*.
- **IsAbelian**( *grp* )  
determines whether or not *grp* is commutative.
- **Elements**( *C* )  
determines the elements in *C*.
- **Size**( *C* )  
determines the number of elements in the list or collection *C*.
- **Order**( *elm* )  
determines the order of the group element *elm*.
- **List**( *L*, *elm*  $\rightarrow$   $P(\textit{elm})$  )  
determines a list whose elements are of the form  $P(\textit{elm})$  where *elm* ranges over *L* and *P* is a function whose domain is *L*. An element can occur several times in a list.
- **Set**( *L*, *elm*  $\rightarrow$   $P(\textit{elm})$  )  
determines a set whose elements are of the form  $P(\textit{elm})$  where *elm* ranges over *L* and *P* is a function whose domain is *L*. An element can only occur once in a set. This can be used to find sets of the form  $\{P(x):x \in L\}$ , where *L* is a set and  $P(x)$  is a function whose domain is *L*.

*Example: To form the symmetric group of degree 4, compute a subgroup, A4, generated by two 3-cycles, determine that A4 is non-abelian, display the elements of A4, find its order, form a list containing the orders of the elements, and form a set containing the orders of the elements.*

```
gap> S4 := Group([(1,2),(2,3),(3,4)]);
Group([ (1,2), (2,3), (3,4) ])
gap> A4 := Subgroup(S4, [(1,2,3), (2,3,4)]);
Group([ (1,2,3), (2,4) ])
gap> IsAbelian(A4);
false
gap> Elements(A4);
[ (), (2,3,4), (2,4,3), (1,2)(3,4), (1,2,3), (1,2,4),
  (1,3,2), (1,3,4), (1,3)(2,4), (1,4,2), (1,4,3),
  (1,4)(2,3) ]
gap> Size(A4);
12
gap> List( A4, g -> Order(g));
[ 1, 3, 3, 2, 3, 3, 3, 3, 2, 3, 3, 2 ]
gap> Set( A4, g -> Order(g));
[1, 2, 3]
```

- `Union( C1, C2 )`

returns the union of sets  $C1$  and  $C2$ . You can specify more than two parameters.

- `Intersection( C1, C2 )`

returns the intersection of sets  $C1$  and  $C2$ . You can specify more than two parameters.

- `Difference( C1, C2 )`

returns the set of elements in  $C1$  which are not in  $C2$ .

*Example: To find the set of all elements of the symmetric group of degree 4 whose sixth power is the identity but whose square is not the identity.*

```
gap> S4 := SymmetricGroup(4);
Sym( [ 1 .. 4 ] )
gap> T := Filtered( S4, g -> g^6 = () );
[ (), (1,3)(2,4), (1,4)(2,3), (1,2)(3,4), (2,3,4), (1,3,2),
  (1,4,3), (1,2,4), (2,4,3), (1,3,4), (1,4,2), (1,2,3), (3,4),
  (1,2), (2,3), (1,4), (2,4), (1,3) ]
gap> V := Filtered( S4, g -> g^2 = () );
[ (), (1,3)(2,4), (1,4)(2,3), (1,2)(3,4), (3,4), (1,2), (2,3),
  (1,4), (2,4), (1,3) ]
gap> Difference(T, V);
[ (2,3,4), (2,4,3), (1,2,3), (1,2,4), (1,3,2), (1,3,4),
  (1,4,2), (1,4,3) ]
```

- `AddSet( S, elm )`

changes the set  $S$  by adjoining  $elm$  to it. Since GAP has a special ordering for sets,  $elm$  may not be placed “at the end” of the set.

- `Add( L, elm )`

changes the list  $L$  by placing  $elm$  at the end of it. Since a set is also a list, `Add` can be used when  $L$  is a set.

*Example: To add 2 to the list of odd numbers less than 20, first using `Addset`, then using `Add`.*

```
gap> L := Filtered([1..20], x -> x mod 2 = 1);
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19 ]
gap> AddSet(L,2);
gap> L;
[ 1, 2, 3, 5, 7, 9, 11, 13, 15, 17, 19 ]
gap> L := Filtered([1..20], x -> x mod 2 = 1);
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19 ]
gap> Add(L,2);
gap> L;
[ 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 2 ]
```

- functions

In writing a function in GAP it is necessary to specify the name of the function, the parameters of the function, local variables if any, and the object to be returned, if any. Variables which are not specified as local are assumed to be global. Thus, an error message will result if a variable which is neither previously defined nor declared as local is used in a function.

Example: To find the set of elements of a group which are  $n$ -th powers of elements.

```
gap> NthPowers := function(G, n)
  local Y, g;
  Y := [];
  for y in Elements(G) do
    AddSet(Y, y^n);
  od;
  return Y;
end;
```

### 3 Description

**Definition:** A finite group is said to be **simple** if the only normal subgroups of  $G$  are 1 and  $G$ .

In 1981, the classification of the finite simple groups was completed. As stated in **Finite Simple Groups**, by Daniel Gorenstein, the classification was

... one of the most remarkable achievements in the history of mathematics. Involving the combined efforts of several hundred mathematicians from around the world over a period of 30 years, the full proof covered something between 5,000 and 10,000 journal pages, spread over 300 to 500 individual papers.

Most of the tools used in the classification involve very deep results. Even the problem of determining which numbers less than 1000 can be the orders of simple groups is difficult. (Proving that there is no simple group of order 720 is particularly challenging.)

Your task in this project is to use the Sylow Theorems and other basic results to find the smallest possible set  $A$ ,  $A \subseteq \{1, 2, \dots, 250\}$ , for which you can prove the following theorem:

**Theorem 1** *If  $G$  is a simple, non-abelian group and  $|G| < 250$ , then  $|G| \in A$ .*

The following result determines which prime powers can be the orders of simple groups.

**Theorem 2** *If  $p$  is a prime and  $G$  is a  $p$ -group, then  $G$  is simple if and only if  $G$  has order 1 or  $p$ .*

Thus, the set of numbers for which you are looking for does not contain any prime powers. The following GAP command will return the set of all non-prime powers less than or equal to 250.

```
Filtered([2..250], n -> Size(Set(Factors(n))) > 1);
```

Perhaps the most basic tools for the study of finite groups are the Sylow Theorems.

**Definition:** If  $n$  is a positive integer and  $p$  is a prime, then the **p-part** of  $n$  is the largest power of  $p$  which divides  $n$ .

**Definition:** Let  $G$  be a finite group and  $p$  be a prime. A **Sylow  $p$ -subgroup** of  $G$  is a subgroup,  $S$ , of  $G$  such that  $|S|$  is the  $p$ -part of  $|G|$ . The set of all Sylow  $p$ -subgroups of  $G$  will be denoted  $Syl_p(G)$ .

The Sylow Theorems can be proved using various group actions, but you may use them without proof in this project.

**Theorem 3 (Sylow)** *Let  $G$  be a finite group and  $p$  be a prime.*

1. *There exists a Sylow  $p$ -subgroup of  $G$ .*
2. *If  $S, T \in Syl_p(G)$ , there exists  $g \in G$  such that  $S^g = T$ .*
3. *If  $S \in Syl_p(G)$ , then  $|G : \mathbb{N}_G(S)| = |Syl_p(G)|$ .*
4.  $|Syl_p(G)| \equiv 1 \pmod{p}$ .

The following GAP function gives, for a given  $n$  and prime  $p$ , the set of numbers of the form  $1 + kp$  which divide  $n$ .

```
gap> Sylow := function(n, p)
>   # Purpose : To find the possible numbers of Sylow p-groups
>   #           of a group of order n.
>   # Input  : n, p - positive integers
>   # Output : S - a list
>   local M, S;
>   M := List([0..Int((n-1)/p)], k -> 1 + k*p);
>   S := Filtered(M, m -> (n mod m) = 0);
>   return S;
> end;
```

**Example:** *There is no simple group of order 84.*

Suppose  $G$  is a group of order 84. Then from the following we see that  $84 = 2^2 \cdot 3 \cdot 7$ ;  $|Syl_2(G)|$  is either 1, 3, 7 or 21;  $|Syl_3(G)| = 1, 4, 7, \text{ or } 28$ ; and  $G$  has a unique Sylow 7-subgroup. This means that  $G$  has a normal subgroup of order 7 and is thus not simple.

```
gap> Factors(84);
[ 2, 2, 3, 7 ]
gap> Sylow(84,2);
[ 1, 3, 7, 21 ]
gap> Sylow(84, 3);
[ 1, 4, 7, 28 ]
gap> Sylow(84, 7);
[ 1 ]
```

You should be able to write a function which accepts as a parameter a number  $n$ , returns true if there a prime  $p$  which divides  $n$  for which `Sylow(n, p)` contains only one element, and returns false otherwise. Suppose that such a function, say `HasNormalSylow()`, has been written. If  $n$  is the order of a non-abelian simple group, then  $n$  has more than one Sylow  $p$ -subgroup for every prime dividing  $p$ . Thus, `HasNormalSylow(n)` is false for such a value of  $n$ .

Assuming the function `HasNormalSylow()` has been written, the following GAP session would return the list of 26 numbers between 1 and 250 which are not prime powers and for which `HasNormalSylow()` is false.

```
W := Filtered([1..250], n -> Size(Set(Factors(n))) > 1);
Y := Filtered(W, n -> HasNormalSylow(n) = false);
```

Of these 26 numbers, 60 and 168 are the orders of simple groups. To eliminate the other group orders you may use any of the following results that you can prove.

**Theorem 4** *If  $G$  is a finite group and  $p$  is a prime such that  $p$  is largest power of  $p$  which divides  $|G|$ .*

1. *If  $S, T \in Syl_p(G)$  then  $S \cap T = 1$ .*
2. *There are  $(p-1)|Syl_p(G)|$  elements of order  $p$  in  $G$ .*

**Theorem 5** *If  $G$  is a finite simple group and  $H \leq G$  with  $|G| = n$  and  $|H| = k$ , then  $n|(n/k)!$ . (Hint: Consider the homomorphism induced by that action of  $G$  on the left cosets of  $H$  in  $G$ .)*

**Theorem 6** *If  $G$  is a group  $|G| = 2n$ , where  $n$  is odd, then there is a subgroup  $H$  of  $G$  such that  $|G : H| = 2$  and  $H \triangleleft G$ . (Hint: Consider the homomorphism used in the proof of Cayley's theorem and show that the image of an element of order 2 in  $G$  under this homomorphism is an odd permutation.)*

Here is an example of how you might use these results, once you have proven them.

**Example:** There is no simple group of order 30.

Suppose  $G$  is a simple group of order 30. The following session shows that  $30 = 2 \cdot 3 \cdot 5$ ,  $|Syl_3(G)| = 10$ , and  $|Syl_5(G)| = 6$ . Therefore the hypothesis of Theorem 4 are satisfied for both the primes 3 and 5 and, thus,  $G$  has  $2 \cdot 10 = 20$  elements of order 3 and  $4 \cdot 6 = 24$  elements of order 5. This is a contradiction since  $G$  has only 30 elements.

```
gap> Factors(30);
[ 2, 3, 5 ]
gap> Sylow(30,3);
[ 1, 10 ]
gap> Sylow(30,5);
[ 1, 6 ]
```

## 4 What to hand in

1. Prove Theorem 2.
2. Find the smallest possible set,  $A$ , which is described above.

Notes:

- (a) You will want to write the function `HasNormalSylow()`.
  - (b) The numbers 120, 144, 180, and 240 are the most difficult. You may not be able to show that these cannot be the orders of simple groups.
3. Write a 2 page report on the classification of the finite simple groups. In your report, include a brief description of the simple groups of order 60 and of order 168.