ABSTRACT ALGEBRA WITH GAP

Julianne G. Rainbolt
and
Joseph A. Gallian

ABSTRACT ALGEBRA WITH GAP

TABLE OF CONTENTS

ABSTRACT ALGEBRA WITH GAP
Julianne G. Rainbolt and Joseph A. Gallian

PREFACE

There is a growing interest in the use of discovery based instruction in undergraduate mathematics courses. Although abstract algebra is primarily a course that emphasizes theory and proofs, students can benefit from the many computational aspects of the core concepts of groups and rings. In this volume and accompanying website we provide exercises that are to be done using the software GAP. These exercises are designed to provide students with a convenient way to generate data with the intent of increasing understanding and enabling them to make and test conjectures. Most of the exercises do not require programming on the part of the students. We have included some exercises where students are asked to write their own programs modeled after ones that are given in the manual.

This manual grew out of lab exercises created by one of the authors, Julianne Rainbolt, to supplement the textbook "Contemporary Abstract Algebra" by Joseph A. Gallian. The chapters of this manual correspond to the eighth edition of this text. However, this manual has been revised in such a way that it does not have to be used with this text. It can instead be used to supplement most introductory courses in abstract algebra. There are a few exceptions to this. Chapters 28, 30 and 31 assume the students are familiar with the notation and content of the corresponding chapters in Gallian's book. Also Exercises 6 - 9 in Chapter 0 of the manual use material from Chapter 0 of Gallian's book. These four exercises and three chapters of the manual can be skipped if Gallian's book is not the textbook being used. For the situation where Gallian's text is being used we have included references to this text. These references are provided within square brackets at the end of certain exercises, theorems and examples. We believe that "Abstract Algebra with GAP" offers students a convenient way to explore the basic concepts of groups and rings. No experience with GAP is assumed.

The authors would like to thank those who have provided us with ideas and suggestions for this manual. In particular, major parts of Chapters -1, 2, 3 and 5 come from Loren Larson at Saint Olaf College (retired) and Russell Blyth at Saint Louis University. The content of Chapter 25, is based on course material written by Christine Stevens at Saint Louis University. We would particularly like to thank Alexander Hulpke for his technical knowledge and helpful responses on questions we had about GAP. He also provided many suggestions which we have incorporated into this manual. In addition, comments made by Russell Blyth, Peter Brooksbank, Allen Hibbard, David Jackson, and Charles Wright have led to many improvements in this manual.

ABSTRACT ALGEBRA WITH GAP
Julianne G. Rainbolt and Joseph A. Gallian

NOTE TO INSTRUCTOR

This note to instructors is intended to provide an overview of how the authors view the possible uses of this manual. Using software in an upper division mathematics course can have its place if it somehow provides a way for the student to better understand the material. There are at least five ways GAP can be a useful pedagogical tool:
1) as a fancy calculator,
2) as a way to provide large or complicated examples,
3) as a way for students to write simple computer algorithms,
4) as a way for producing large amounts of data so that the student can formulate a conjecture and
5) as a means for students to work in collaboration.

GAP can be used as a fancy calculator and thus eliminate some repetitive hand calculations that the student may have otherwise had to do. For example, once a student knows how to find all the conjugacy classes of a group of small order, GAP can be used to provide all the conjugacy classes of a group of order 80.

GAP has many built in functions, operations, and algebraic structures. Thus GAP can be used to quickly provide numerous examples, many more and of more complexity than could be done by hand. For example, all fourteen groups (and their properties) of order 16 could be easily examined during a small portion of a single class period, using GAP.

The students can begin writing simple computer algorithms using GAP commands. Writing an algorithm causes the student to solidify a new concept. For example, in order to write an algorithm that finds all the nilpotent elements in a group, the student has to be able to very precisely write code for checking nilpotency.

GAP also provides a means for producing large amounts of data quickly. A student can then look for patterns and formulate conjectures. When a student "discovers" a theorem before it is proved he may be more likely to remember and understand it. Also theorems that may be beyond the scope of the course could be introduced in this way. In some cases the patterns that develop help students understand the proof of the theorem or provide them with a possible approach to try to prove the theorem. In other cases the patterns that develop can be explained geometrically, and thus provide students with another way to understand the concept. Also in some cases, deceptive patterns can lead to incorrect conjectures and thus initiate an informative discussion on how to reformulate and test a new conjecture.

Doing projects on a computer lends itself to group work. For example, if a large amount of data is being produced in order to formulate a conjecture about a group of prime power order, students in

a group could each pick a certain number of primes to test and then compare their results as a group.

All five of the above pedagogical approaches are used in the exercises in this manual. The third, however, is not emphasized as much as the other four. The intent of this manual is to provide a supplement to a more traditional way of teaching abstract algebra. A course where the the main focus is to use abstract algebra concepts, learn how to prove abstract algebra theorems and understand abstract algebra structures, is assumed. We want to keep to a minimum the amount of time a student spends learning software code. Only an extremely small portion of the power of GAP is introduced to the students. In fact, some of the built in functions are purposely not introduced, as we do not want GAP to do too much of the work for the student. In addition, only the GAP commands that are going to be specifically used in a particular chapter are introduced.

The exercises in the later chapters of this lab manual do not assume that the student has worked all previous lab manual exercises. Thus exercises can be picked or skipped as the instructor views appropriate. On the other hand, the GAP material in Chapters -1 to Chapter 22 assumes that the GAP commands introduced in the text of the previous chapters has been covered. Thus we recommend that the student read each chapter of the lab manual as the corresponding material from the text is covered, even if no computer exercises are assigned. After Chapter 22, the GAP content no longer builds from chapter to chapter, and so the lab manual chapters can be done in any order from Chapters 23 - 33. We suggest instructors use the *Instructor Solution Manual* that accompanies this lab manual. It contains additional GAP commands and context which instructors will find useful when teaching with this software.

-1 Chapter: An Introduction to GAP

This chapter provides the instructions needed to use this manual as well as the very basics on using GAP. Much of the material in this chapter is based on material written by Russell Blyth at Saint Louis University. GAP is a free software program. This manual assumes you have installed one of the 4.6 versions of GAP. If you do not have this version already, you can download a copy from:

`http://www.gap-system.org/`

A reference manual and extensive tutorial for the software GAP are also available at this website.

Some Commands That You Will Use Often

1) To exit from GAP type `quit;`
2) Type <`ctl`>`-p` to redisplay the previous command
3) If an error causes GAP to enter a loop (and gives you the prompt `brk>`) you can exit the loop by typing <`ctl`>`-D` or `quit;`.
4) Type `?` followed by a subject name to get help about that subject.

The `?` command, listed above, is particularly useful. This can be used whenever you forget the exact command for something or when you wonder if GAP has a command for a particular operation. For example, suppose you want to know how to denote multiplication in GAP. Type

```
gap> ?multiplication
```

and GAP returns a list of subtopics on multiplying different algebraic structures.

Careful: GAP distinguishes between upper and lower case letters.
Careful: Always put a semicolon at the end of a command. Two semicolons at the end of a command will cause GAP to execute the command but not list the output of the command.

Getting Use to GAP

We will now do some exercises to get you comfortable with GAP.

At the gap prompt type $(5+3)*9$; then hit the enter key. Your screen should look like this:

```
gap > (5+3)*9;
72
```

At the prompt type $(5+3)*9$ (without the semicolon) and then hit enter.

```
gap> (5+3)*9

>
```

Notice nothing happens. GAP does not know you have finished the command because there is no semicolon. Now type ; and then hit return. You should now get the 72.

```
gap> (5 + 3) * 9
>;
72
gap>
```

Now type (5 + 3 * 9; and hit enter. Your screen should look like

```
gap> (5 + 3 * 9;
Syntax error:  )  expected
```

An error message occurred because the number of left parentheses and right parentheses does not match. Type <ctl>-p to redisplay the previous command. Then use your arrow keys to insert the needed parenthesis.

```
gap> (5 + 3 * 9;
Syntax error:  )  expected
gap> (5 + 3) * 9;
72
gap>
```

GAP can be used to test the equality of two values. At the GAP prompt type 6=9; and then hit enter.

```
gap> 6=9;
false
```

GAP returned the value of false since 6 is not equal to 9. Now use GAP to complete the following exercises. Don't forget to put a semicolon at the end of a command.

-1.1 Compute 3^{121}.
-1.2 Determine if $2^{25} + (45 * 51,777)$ is greater than 34 million.

You can assign values to variables in GAP by using :=. This allows you to refer to an object with a name. The name of the variable is called an identifier. In the following example a is an identifier.

```
gap> a := (10 + 7) * (9 - 6);
51
gap> a;
51
gap> a * (a-1);
2550
gap> a:=14;;
gap> a * (a-1);
```

```
182
```

Notice when an identifier is assigned a value that value is echoed on the next line. Because two semicolons occurred in the line `a:= 14;;` the value of `a` was not echoed. Almost any sequence of letters and digits containing at least one letter can be used as an identifier. Use GAP to complete the following exercises.

-1.3 Assign the value 4 to the identifier `b`.
-1.4 Assign the value 522456 to the identifier `bignumber`.
-1.5 Compute `b + bignumber`.

This completes the computer exercises for Chapter -1. Exit GAP by typing `quit;` at the gap prompt.

0 Chapter: Preliminaries

Properties of Integers

The software GAP contains many predefined functions. Functions in GAP begin with a capital letter. The function `Gcd` gives the *greatest common divisor* of two nonzero integers. The function `Lcm` gives the *least common multiple* of two nonzero integers. Examples:

```
gap> Gcd(123, 456);
3
gap> Lcm(123,456);
18696
```

Theorem: For any nonzero integers a and b, there exist integers s and t such that the greatest common divisor of a and b equals $as + bt$. [Gallian, Chapter 0, Theorem 0.2]

The function `Gcdex` provides the numbers s and t in this theorem. An example:

```
gap> Gcdex(4,15);
rec( coeff1 := 4, coeff2 := -1, coeff3 := -15, coeff4 := 4, gcd := 1 )
gap>
```

The above output tells us that the gcd of 4 and 15 is 1 and that $\gcd(4, 15) = 4 * 4 + (-1) * 15$. [Gallian, Chapter 0, Example 2] That is, `coeff1` and `coeff2` are integers s and t such that $\gcd(a, b) = as + bt$. (The output `coeff3` and `coeff4` are integers m and n such that $0 = am + bn$.)

Exercises
Don't forget functions begin with capital letters.

0.1 Compute the gcd of 8701 and 10057.
0.2 Compute the lcm of 25 and 80.
0.3 Use `Gcdex` to find integers s and t so that $\gcd(8701, 10057) = 8701s + 10057t$.

Modular Arithmetic

GAP can perform modular arithmetic. For example

```
gap> 23 mod 6;
5
gap> 5 mod 6 + 11 mod 6;
10
gap> 10 mod 6;
4
gap> (5 mod 6 + 11 mod 6) mod 6;
```

```
4
```

Careful: Blank spaces are needed around `mod`. (`23mod6` will be viewed as an identifier by GAP.)

Functions (Mappings)

You can create your own functions within GAP. One way to do this is to use the *maps-to* operator −>. This is a minus sign and a greater than sign with no blank space between. The following is an example of creating a function called square which takes a number and squares it.

```
gap> square:=x -> x^2;
function( x ) ... end
```

Now we can use this function:

```
gap> square(2);
4
gap> square(5);
25
```

Exercises
Use GAP to do the following exercise.

0.4 Define a function `SumFirstnInt` which takes as input a positive integer n and outputs the sum $1 + 2 + 3 + \ldots + n$. Then use this function to find this sum for $n = 100$ and $n = 987$. (Hint: Use the fact that the sum of the first n positive integers equals $n(n + 1)/2$.)

On the website for this manual, there is a file of subroutines that will be used in some of the exercises. For example, suppose we wanted to use the function $f(t) = t^3 + 3 - 2t$ while in GAP. This function is called `newfunction` in the subroutine folder. Place this subroutine in the folder where you have GAP installed. If you open the subroutine you will see it contains the following.

```
newfunction:= function(t)
local x;
x:= t^3 + 3 - 2*t;
return x;
end;
```

Now whenever we run GAP we can use this function by first reading it in or by copying the above text and pasting it into GAP:

```
gap> Read("newfunction");
gap> newfunction(6);
207
```

We can also easily edit this function by just opening and editing the file "newfunction". If for

example we want another function, called `newfunction2`, to be $f(t) = t^3 + 3 - 5t$. Open up "newfunction." Change `newfunction` to `newfunction2` and the 2 to a 5:

```
newfunction2:= function(t)
local x;
x:= t^3 + 3 - 5*t;
return x;
end;
```

Then save the file as "newfunction2". In GAP we can now use this function:

```
gap> Read("newfunction2");
gap> newfunction2(3);
15
```

The remainder of this chapter assumes you are using the text "Contemporary Abstract Algebra" by Joseph Gallian. The remainder of the material in this chapter is not needed to understand the remaining chapters in this manual.

Read through [Gallian, Chapter 0, Example 4]. Type in

```
gap> 3953988164 mod 9;
2
```

This verifies that 39539881642 could be a valid postal service money order number. In contrast, if we enter

```
gap> 3955988164 mod 9;
4
```

we see that 39559881642 is not a valid money order number. Here is another way to do this using GAP:

```
gap> 3953988164 mod 9 = 2;
true
gap> 3955988164 mod 9 = 2;
false
```

Exercises
Use GAP to do the following exercises.

0.5 Define a function that calculates the United States Postal Service check digit of a 10 digit number. [Gallian, Chapter 0, Example 4]

0.6 Use your function from Exercise 0.5 to verify that 3953981642 is a valid United States Postal Service money order number. Now make one digit incorrect. Does your function detect the error?

Enter the number with the 9 in position 2 replaced with a 0. Was the error detected? Explain why or why not. Enter the number with two digits transposed. Was the error detected? Explain why or why not. [Gallian, Chapter 0, Computer Exercise 1]

0.7 Write a GAP function that will test the validity of a UPC number. (See [Gallian, Chapter 0, Example 5].) Save this function as a file that you can use again later. Use it to verify that 090146003386 is valid. Now enter the same number with one digit incorrect. Was the error detected? Enter the number with two consecutive digits transposed. Was the error detected? Enter the number with the 3 and the 8 transposed. Was the error detected? Explain why or why not. Enter the number with the 9 and the 1 transposed. Was the error detected? Explain why it was or was not. [Gallian, Chapter 0, Computer Exercise 2]

0.8 Write a GAP function that will test the validity of a UPS number. Use it to verify that 8733456723 is valid. Now enter the same number with one digit incorrect. Was the error detected? Enter the number with two consecutive digits transposed. Was the error detected? Enter the number with the 8 replaced by 1. Was the error detected? Explain why or why not. [Gallian, Chapter 0, Computer Exercise 3]

0.9 Write a GAP function that will test the validity of an `ISBN_10` number. (See [Gallian, Chapter 0, Exercise 49].) Use it to verify that the 0395872456 is valid. Now enter the same number with one digit incorrect. Enter the number with two digits transposed (they need not be consecutive). Was the error detected? [Gallian, Chapter 0, Computer Exercise 5]

1 Chapter: Introduction to Groups

The symmetries of a regular n-gon is called the dihedral group of order $2n$. We will denote this group by D_n. Consider the square with vertices labeled as in Figure 1.1 of this manual. (See the last page of this chapter for figures.) The way GAP denotes the element in D_4 that is a rotation by 90 degrees is $(1,2,3,4)$. This notation means vertex 1 goes to vertex 2, vertex 2 to 3, vertex 3 to 4 and 4 to 1. (Chapter 5 will give more details on this notation). Similarly, the horizontal reflection is denoted by $(1,2)(3,4)$. (See Figure 1.2.)

The command in GAP for the dihedral group D_n is `DihedralGroup(IsPermGroup,2n)`. For example to get D_4 we type:

```
gap> d4:= DihedralGroup(IsPermGroup,8);
Group([ (1,2,3,4), (2,4) ])
gap> Elements(d4);
[ (), (2,4), (1,2)(3,4), (1,2,3,4), (1,3), (1,3)(2,4), (1,4,3,2), (1,4)(2,3) ]
```

The command `Elements` listed the elements in the group. The identity is denoted by `()`. The command `Size` gives the number of elements in the group (that is, the order of the group).

```
gap> Size(d4);
8
```

The command `Size` is also useful to find the number of elements in a set. Elements can be multiplied (the operation is functional composition):

```
gap> (1,4)(2,3)*(2,4);
(1,2,3,4)
```

Careful: GAP multiplies these elements from left to right whereas many textbooks (including Gallian's) multiply these elements from right to left.

Exercises

1.1 Explain geometrically why a reflection followed by a reflection is a rotation. [Gallian, Chapter 1, Exercise 6] Using GAP take a reflection in D_4 and multiply it by a reflection in D_4. What rotation do you get?

1.2 Make a conjecture about what a rotation followed by a reflection is for **any** dihedral group. What about a reflection followed by a rotation? Test your conjecture by using GAP to compute the product of a reflection followed by a rotation for several pairs of reflections and rotations. You may want to draw a picture of the n-gon to help you determine which rotation or reflection you are getting.

1.3 Let r_1, r_2, r_3 represent rotations in D_n and let f_1, f_2, and f_3 represent reflections in D_n. Determine whether $r_1 r_2 f_1 r_3 f_2 f_3 r_3$ is a rotation or a reflection. [Gallian, Chapter 1, Exercise 10]

1.4 Using GAP, find elements A, B and C in D_5 such that $AB = BC$ but $A \neq C$. [Gallian, Chapter 1, Exercise 11]

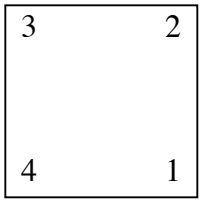


Figure 1.1

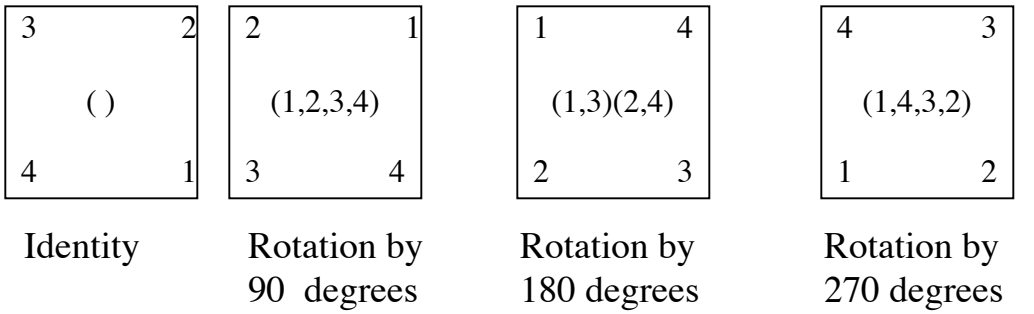


Figure 1.2

2 Chapter: Groups

Let $U(n)$ be the set of all positive integers less than n and relatively prime to n. The set $U(n)$ is a group under multiplication modulo n.

For this chapter you will need the file named "ulist". (Thanks goes to Alexander Hulpke for pointing out this revised version of "ulist".) Fetch the file "ulist" from the website for this manual and place it in the folder in which you have GAP stored. The appendix at the end of this chapter contains a print out of the file "ulist." To use the file "ulist" while in GAP type:

```
gap> Read("ulist");
```

This command reads in a copy of the file "ulist". (Alternatively, you can just copy and paste the contents of this file into GAP.)

Careful: If you exit GAP and then reenter GAP you will need to read in the file "ulist" again, if you want to continue to use it. For any n the file "ulist" contains a function, also called `ulist`, which will list the elements of $U(n)$. For example:

```
gap> ulist(100);
[ ZmodnZObj( 1, 100 ), ZmodnZObj( 3, 100 ), ZmodnZObj( 7, 100 ),
  ZmodnZObj( 9, 100 ), ZmodnZObj( 11, 100 ), ZmodnZObj( 13, 100 ),
  ZmodnZObj( 17, 100 ), ZmodnZObj( 19, 100 ), ZmodnZObj( 21, 100 ),
  ZmodnZObj( 23, 100 ), ZmodnZObj( 27, 100 ), ZmodnZObj( 29, 100 ),
  ZmodnZObj( 31, 100 ), ZmodnZObj( 33, 100 ), ZmodnZObj( 37, 100 ),
  ZmodnZObj( 39, 100 ), ZmodnZObj( 41, 100 ), ZmodnZObj( 43, 100 ),
  ZmodnZObj( 47, 100 ), ZmodnZObj( 49, 100 ), ZmodnZObj( 51, 100 ),
  ZmodnZObj( 53, 100 ), ZmodnZObj( 57, 100 ), ZmodnZObj( 59, 100 ),
  ZmodnZObj( 61, 100 ), ZmodnZObj( 63, 100 ), ZmodnZObj( 67, 100 ),
  ZmodnZObj( 69, 100 ), ZmodnZObj( 71, 100 ), ZmodnZObj( 73, 100 ),
  ZmodnZObj( 77, 100 ), ZmodnZObj( 79, 100 ), ZmodnZObj( 81, 100 ),
  ZmodnZObj( 83, 100 ), ZmodnZObj( 87, 100 ), ZmodnZObj( 89, 100 ),
  ZmodnZObj( 91, 100 ), ZmodnZObj( 93, 100 ), ZmodnZObj( 97, 100 ),
  ZmodnZObj( 99, 100 ) ]
```

The output `ZmodnZObj( 3, 100 )`, for example, means the element 3 mod 100.

Exercises

2.1 Using GAP determine the size of $U(n)$ for $n = 9, 27, 81, 243, 5, 25, 125$. Make a conjecture about the size of $U(p^k)$ where p is a prime not equal to 2 and k is a positive integer. Do not count the elements in $U(n)$, instead use `Size` to make GAP count the elements for you! [Gallian, Chapter 2, Computer Exercise 2]

2.2 Using GAP determine the size of $U(n)$ for $n = 18, 54, 162, 486, 50, 250, 98, 242$. Make a conjecture about the relationship between the size of $U(2p^k)$ and the size of $U(p^k)$ where p is a prime not equal to 2. [Gallian, Chapter 2, Computer Exercise 2]

2.3 Let r and s be relatively prime integers. Use GAP to help you make a conjecture about the size of $U(rs)$ in terms of the sizes of $U(r)$ and $U(s)$.

2.4 Recall from Chapter 0 how to do modular arithmetic in GAP. Use the function `Gcdex` to find the inverses of the elements in $U(100)$. For example, to find the inverse of 3 in $U(100)$ use `Gcdex(3,100)`.

The command `GL(n,p)` creates the general linear group of $n \times n$ invertible matrices with entries in $\mathbf{Z}_p$ and `SL(n,p)` creates the special linear group of $n \times n$ invertible matrices with entries in $\mathbf{Z}_p$ and determinate equal to one. For example the following creates $\mathrm{GL}(3, \mathbf{Z}_5)$ and $\mathrm{SL}(3, \mathbf{Z}_5)$:

```
gap> g:= GL(3,5);
GL(3,5)
gap> s:= SL(3,5);
SL(3,5)
```

We can use our `Size` command to find the number of elements (the order) in these groups:

```
gap> Size(g);
1488000
gap> Size(s);
372000
```

Exercises

2.5 Find the number of elements in $\mathrm{GL}(2, \mathbf{Z}_p)$ and $\mathrm{SL}(2, \mathbf{Z}_p)$ for $p = 3, 5, 7$ and 11. What relationship do you see between the orders of $\mathrm{GL}(2, \mathbf{Z}_p)$ and $\mathrm{SL}(2, \mathbf{Z}_p)$ and $p-1$? Does this relationship hold for $p = 2$? Based on these examples does it appear that p always divides the order of $\mathrm{SL}(2, \mathbf{Z}_p)$? What about $p-1$? What about $p+1$? Guess a formula for the order of $\mathrm{SL}(2, \mathbf{Z}_p)$. Guess a formula for the order of $\mathrm{GL}(2, \mathbf{Z}_p)$. [Gallian, Chapter 2, Computer Exercise 4].

Appendix for Chapter 2
The following is the file "ulist" which is used in this chapter:

```
ulist:= function(n)
local s,i,o;
o:= One(Integers mod n);
s:= n-> Filtered([1..n-1], i -> Gcd(i,n) = 1);
return s(n)*o;
end;
```

3 Chapter: Finite Groups; Subgroups

In order to do the exercises in this section you will need to read into GAP the files "ulist" and "cyclic." (Or you can copy and paste the contents of these files into GAP.) The file "cyclic" was written by Loren Larson at St. Olaf College and then revised per comments from Alexander Hulpke. The file "ulist" should already be in your GAP folder. Fetch the file "cyclic" from the website for this manual and place the file in your GAP folder. The appendix at the end of this chapter contains a print out of the file "cyclic."

GAP has many useful features that allow you to examine subgroups:

```
gap> G:= DihedralGroup(IsPermGroup, 16);
Group([ (1,2,3,4,5,6,7,8), (2,8)(3,7)(4,6) ])
gap> Elements(G);
[ (), (2,8)(3,7)(4,6), (1,2)(3,8)(4,7)(5,6), (1,2,3,4,5,6,7,8), (1,3)(4,8)(5,7),
(1,3,5,7)(2,4,6,8), (1,4)(2,3)(5,8)(6,7), (1,4,7,2,5,8,3,6), (1,5)(2,4)(6,8),
(1,5)(2,6)(3,7)(4,8), (1,6)(2,5)(3,4)(7,8), (1,6,3,8,5,2,7,4), (1,7)(2,6)(3,5),
(1,7,5,3)(2,8,6,4), (1,8,7,6,5,4,3,2), (1,8)(2,7)(3,6)(4,5) ]
gap> a:= G.1;
(1,2,3,4,5,6,7,8)
gap> b:= G.2;
(2,8)(3,7)(4,6)
gap> H:= Subgroup(G, [a]);
Group([ (1,2,3,4,5,6,7,8) ])
gap> Elements(H);
[(), (1,2,3,4,5,6,7,8), (1,3,5,7)(2,4,6,8), (1,4,7,2,5,8,3,6),
(1,5)(2,6)(3,7)(4,8), (1,6,3,8,5,2,7,4), (1,7,5,3)(2,8,6,4),
(1,8,7,6,5,4,3,2)]
```

The first command above assigns the name `G` to the dihedral group of order 16 (the group of symmetries of a regular 8-gon). From the next command listing the elements in `G`, we can see that both $(1,2,3,4,5,6,7,8)$ and $(1,8)(2,7)(3,6)(4,5)$ are elements of G. More generally, $(1,2,3,\ldots,n)$ represents a rotation of $360/n$ degrees; for n even, $(1,n)(2,n-1)(3,n-2)...(\frac{n}{2},\frac{n}{2}+1)$ is a reflection; and for n odd, $(1,n)(2,n-1)...(\frac{n-1}{2},\frac{n-1}{2}+2)$ is a reflection. The next two commands assign the names `a` and `b` to two generators of `G`. The next command assigns the name `H` to the cyclic subgroup of `G` generated by `a`. Notice that `H` is a subgroup of `G` of order 8.

```
gap> K:= Subgroup(G, [a,b]);
Group([ (1,2,3,4,5,6,7,8), (2,8)(3,7)(4,6) ])
gap> Size(K);
16
```

This first command above assigns the name `K` to the subgroup of `G` generated by `a` and `b`. That is, `K` is the subgroup of `G` obtained by taking all possible finite strings of `a`'s and `b`'s. Notice that after this command GAP echoes the generators of the subgroup. By typing in `Elements(K)` or `Size(K)`

we can see that `K = G`.

In the line `gap> K:= Subgroup(G, [a,b]);` the use of `[a,b]` is a list in GAP. In general, square brackets enclose lists. In this case, we are listing the generators of `K`.

```
gap> c:= (1,5)(2,6)(3,7)(4,8);;
gap> L:= Subgroup(G, [a,c]);
Group([ (1,2,3,4,5,6,7,8), (1,5)(2,6)(3,7)(4,8) ])
gap> Size(L);
8
```

Notice that the subgroup `L` is a proper subgroup of `G`. In fact, c is a power of a, as we can see from the list of elements of H. Thus L is a subgroup of H. But L and H have the same order, so $L = H$. Also notice the use of a double semicolon at the end of the line defining c. This causes GAP not to echo the definition of c on the next line. (Compare the lines defining a and b with the line defining c.)

```
gap> M:= Subgroup(G, [a^2,b]);
Group([ (1,3,5,7)(2,4,6,8), (2,8)(3,7)(4,6) ])
gap> Elements(M);
[ (), (2,8)(3,7)(4,6), (1,3)(4,8)(5,7), (1,3,5,7)(2,4,6,8), (1,5)(2,4)(6,8),
(1,5)(2,6)(3,7)(4,8), (1,7)(2,6)(3,5), (1,7,5,3)(2,8,6,4) ]
```

Notice M is another subgroup of G of order 8, not equal to H:

```
gap> M=H;
false
```

The function `cyclic(n,a)` which is in the file "cyclic" produces the list of elements in the cyclic subgroup of $U(n)$ generated by the element `a` in $U(n)$. For example,

```
gap> cyclic(15,7);
[ ZmodnZObj( 7, 15 ), ZmodnZObj( 4, 15 ), ZmodnZObj( 13, 15 ),
ZmodnZObj( 1, 15 ) ]
```

gives the subgroup of $U(15)$ generated by 7. (The output `ZmodnZObj( 7, 15 )`, for example, means the element 7 mod 15 in $U(15)$.) If you use this function incorrectly and try to generated a subgroup generated by `a` when `a` is not in $U(n)$, this function will return empty brackets:

```
gap> cyclic(15,3);
[ ]
```

The following is a list of some other commands that you might find useful.

1) The command to find the center of the group G is `Center(G)`.

2) The command to find the centralizer of an element a in a group G is `Centralizer(G,a)`.

3) The command to find the order of an element a in a group G is `Order(a)`.

4) The command `IsAbelian(G)` tells you whether or not the group G is Abelian.

5) The command `IsCyclic(G)` tells you whether or not the group G is cyclic.

There is no need for you to memorize a large collection of GAP commands. Just type in ? followed by a key word describing what you want GAP to do, and the software will provide helpful comments and examples on using commands. For example, say we want to find the order of an element in a group and wonder exactly how to type this in GAP. Type:

```
gap> ?order
```

GAP then provides many possible help topics:

```
Help:  several entries match this topic - type ?2 to get match [2]
[1] Reference:  Order
[2] AutomGrp (not loaded):  Order
[3] kbmag (not loaded):  Order
[4] Reference:  Ordering of Booleans
[5] Reference:  Orderings
[6] Reference:  Orderings on families of associative words
[7] Reference:  order of the prime residue group
[8] Reference:  OrderMod
[9] Reference:  OrderedPartitions
[10] Reference:  ordering booleans
[11] Reference:  ordering of records
[12] Reference:  order of a list, collection or domain
[13] Reference:  OrderingsFamily
[14] Reference:  OrderingByLessThanFunctionNC
[15] Reference:  OrderingByLessThanOrEqualFunctionNC
[16] Reference:  OrderingOnGenerators
[17] Reference:  OrderOfRewritingSystem
[18] Reference:  OrderingOfRewritingSystem
[19] Reference:  order of a group
[20] Reference:  OrdersTom
[21] Reference:  OrdersClassRepresentatives
[22] Reference:  Order (for a class function)
[23] Reference:  ordered partitions internal representation
[24] AutomGrp (not loaded):  OrderUsingSections
[25] Citrus (not loaded):  OrderEndomorphisms (monoid of order preserving transformations)
[26] CTblLib:  Ordering of Characters and Classes
```

```
[27] FR (not loaded):  Order of FR elements
[28] FR (not loaded):  Order of groups
[29] GRAPE (not loaded):  OrderGraph
[30] GRAPE (not loaded):  OrderGraph
[31] GUAVA (not loaded):  order of polynomial
[32] hecke (not loaded):  OrderOfQ
[33] kbmag (not loaded):  OrderingOfKBMAGRewritingSystem
[34] kbmag (not loaded):  OrderingOfRewritingSystem
[35] RCWA (not loaded):  Orders of commutators
[36] RCWA (not loaded):  Order of an rcwa permutation
[37] RCWA (not loaded):  Order of an rcwa mapping of Z x Z
[38] RDS (not loaded):  Ordered signatures by quotient images
```

It looks like the nineteenth one is the one we want so type:

```
gap> ?19
```

GAP then provides a description of the `Order` command and examples.

Exercises
Use GAP to help you work the following exercises.

3.1 Determine whether the group $U(n)$ is cyclic for $n = 3, 9, 27, 5, 25, 125, 7, 49, 343$. (Use the function `cyclic` discussed above but not the command `IsCyclic`.) Make a conjecture. Test your conjecture for other values of n.

3.2 Determine whether the group $U(n)$ is cyclic for $n = 6, 18, 54, 10, 50, 250, 14, 98, 686$. Make a conjecture.

3.3 Determine whether the group $U(n)$ is cyclic for $n = 8, 16, 32$. Modify your conjectures above if necessary. Test your conjecture for other values of n.

3.4 Determine whether the group $U(n)$ is cyclic for n = 12, 20, 28, 44, 52, 15, 21, 33, 39, 51, 57, 69, 35, 55, 55, 65, 85. Modify your conjectures above if necessary.

3.5 Must the centralizer of an element of a group be Abelian? If not, give an example in D_n for some n.

3.6 Must the center of a group be Abelian? If not, give an example in D_n for some n.

Recall the file "ulist" contains a function that lists all the elements in the group $U(n)$. Read this file into GAP. In the following we are going to investigate the relationship between the order of an element and the order of the inverse of that element. Consider $U(15)$, which is a **subset** of $\mathbf{Z}_{15}$.

```
gap> e := Elements(ulist(15));
```

```
[ ZmodnZObj( 1, 15 ), ZmodnZObj( 2, 15 ), ZmodnZObj( 4, 15 ), ZmodnZObj( 7, 15 ),
ZmodnZObj( 8, 15 ), ZmodnZObj( 11, 15), ZmodnZObj( 13, 15), ZmodnZObj( 14, 15) ]
gap> e[3];
ZmodnZObj( 4, 15 )
```

From the above output we see that $U(15)$ contains the numbers 1, 2, 4, 7, 8, 11, 13, and 14 (mod 15). The first command above assigns the name `e` to the list of elements in $U(15)$. Since 4 (mod 15) is the 3rd element in this list we can then refer to 4 as `e[3]`, as is done in last above GAP command. We can now determine the order of 4 in $U(15)$.

```
gap> Order(e[3]);
2
gap> Order(Inverse(e[3]));
2
```

Exercises

3.7 Compute the orders of the elements 3, 7, 53, and 61 in $U(100)$. Compute the orders of the inverses of these elements.

3.8 Compute the orders of the elements 3, 13, 153, and 317 in $U(430)$. Compute the orders of the inverses of these elements.

3.9 Pick a symmetric group S_n for some particular n and call it G in GAP. (The command to create the symmetric group S_5, for example, in GAP is `SymmetricGroup(5)`.) The command `Random(G)` will give you a random element in G. Find the order of a random element in G and the order of its inverse. Repeat this exercise for at least two other random elements of G and at least two other symmetric groups.

3.10 Make a conjecture about the relationship between the order of an element and the order of the inverse of that element.

3.11 Pick a symmetric group S_n for some particular n. Find the orders of two random elements in your group and the order of their product. Repeat this exercise for at least four other pairs of random elements of S_n and at least two other symmetric groups. Based on your results, what do you think we can say about the order of ab in terms of the orders of $a, b \in S_n$.

3.12 Repeat Exercise 3.11 for the groups $\mathrm{GL}(2, \mathbf{Z}_n)$ where n is some particular prime. (The command `GL(2,p)` in GAP sets up the group $\mathrm{GL}(2, \mathbf{Z}_p)$.)

The remainder of this chapter is not needed in the sequel. It is intended for students who would like to learn more about GAP.

If you would like to see how a predefined function GAP is being computed you can do the following. For example, suppose we want to see how GAP is executing the `IsCyclic` function. Type:

```
   gap> G:= DihedralGroup(IsPermGroup, 16);;
   gap> obj:=[G];;
   gap> code:= ApplicableMethod(IsCyclic,obj,1);
   #I  Searching Method for IsCyclic with 1 arguments:
   #I  Total: 7 entries
   #I  Method 5: ‘‘IsCyclic’’, value: 22
function( G ) ... end
```

You can then have the code for `IsCyclic` printed on the screen:

```
gap> Print(code);
function ( G )
    if Length( GeneratorsOfGroup( G ) ) = 1  then
        return true;
    else
        return TRY_NEXT_METHOD;
    fi;
    return;
end
gap>
```

Some predefined functions require more than one argument. All arguments need to be included in the third entered line above (the line starting with `gap> code:=`).

Appendix for Chapter 3

The following is the file “cyclic” which is used in this chapter.

```
cyclic:= function(n,a)
local x, b, o ;
x:= [];
b:= 1;
o:= One(Integers mod n);
  if Gcd(n,a) = 1 then
    repeat
      b:= b*a mod (n);
      Add(x,b);
    until b=1;
  fi;
return x*o;
end;
```

4 Chapter: Cyclic Groups

We can describe a cyclic group of order n, as the group of all powers of the n-cycle $(1, 2, \ldots, n)$. The following sets up the cyclic group of order 6 as the group of all powers of a 6-cycle.

```
gap> c6:= CyclicGroup(IsPermGroup, 6);
Group([ (1,2,3,4,5,6) ])
gap> Elements(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) ]
gap> a:= c6.1;
(1,2,3,4,5,6)
gap> Elements(Subgroup(c6,[a^2]));
[ (), (1,3,5)(2,4,6), (1,5,3)(2,6,4) ]
```

The third command above assigns the name a to the generator of the group `c6`. The fourth command is asking GAP for the elements in the subgroup of `c6` generated by the element a^2 (that is, the subgroup consisting of all powers of a^2). The output from this command says that this subgroup contains only the identity, a^2 and a^4. As was done in Chapter 3, we can also use GAP to determine a subgroup generated by two or more elements. For example the following shows the subgroup of `c6` generated by both a^2 and a^3:

```
gap> Elements(Subgroup(c6,[a^2, a^3]));
[ (), (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) ]
```

This is of course all of `c6`.

Exercises

4.1 Use GAP to list the subgroups of the following groups. *Hint:* The above explanation shows how to generate subgroups of a group. Find which subgroups are generated by one element, which by two elements, etc.
a) D_4
b) the cyclic subgroup of D_8 generated by $(1, 2, 3, 4, 5, 6, 7, 8)$.
Draw the subgroup lattice for each of the above groups.

4.2 Let G be the cyclic group generated by an element a of order n. By the Fundamental Theorem of Cyclic Groups there is exactly one subgroup of G of order k for each k that divides n. In addition, by the Fundamental Theorem of Cyclic Groups, every subgroup of a cyclic group is cyclic. So this subgroup of order k must be cyclic. Use GAP to find the smallest subgroup of G containing
a. a^4 and a^6 when $n = 30$
b. a^{10} and a^2 when $n = 30$
c. a^{15} and a^2 when $n = 30$
d. a^9 and a^{12} when $n = 30$
e. a^8 and a^{12} when $n = 30$
In each part a-e find an integer t such that this smallest subgroup is $\langle a^t \rangle$.

4.3 Repeat Exercise 4.2 for $n = 60$.

4.4 Formulate a conjecture that describes the smallest subgroup of a cyclic group G of order n that contains a^i and a^j for any positive integers i, j and n, where a is a generator of G and i and j are less than n. (You may have to do many more examples before you arrive at a conjecture.)

4.5 Again let G be the cyclic group generated by an element a of order n. Use GAP to find the smallest positive integer t such that $\langle a^t \rangle$ is the subgroup:
a. $\langle a^4 \rangle \cap \langle a^6 \rangle$ when $n = 30$
b. $\langle a^{10} \rangle \cap \langle a^2 \rangle$ when $n = 30$
c. $\langle a^{15} \rangle \cap \langle a^2 \rangle$ when $n = 30$
Hint: Type `?Intersection` at the GAP prompt to see how to find intersections in GAP.

4.6 Repeat Exercise 4.5 for $n = 60$.

4.7 Formulate a conjecture that describes the smallest subgroup of a cyclic group G of order n that contains $\langle a^i \rangle \cap \langle a^j \rangle$ for any integers i, j and n, where a is a generator of G and i and j are less than n. (You may have to do many more examples before you arrive at a conjecture.)

In the remainder of this chapter you will need the file "orderFrequency". Fetch this file off the website. This file contains the function `orderFrequency` which will tell you the number of elements of each order in a given group. For example:

```
gap> Read("orderFrequency");
gap> orderFrequency(c6);
[Order of element, Number of that order]=[ [ 1, 1 ], [ 2, 1 ], [ 3, 2 ],
[ 6, 2 ] ]
```

The output tells us that the cyclic group of order 6 has one element of order 1, one of order 2, two of order 3 and two of order 6.

Exercises

4.8 Find the number of elements of each order in the cyclic groups of order 75 and 90.

4.9 Find the number of elements of each order in the dihedral groups D_{17}, D_{25}, D_{33} and D_{49}. Make a conjecture about the number of elements of order 2 in D_n.

4.10 Find the number of elements of order 2 in the dihedral groups D_{18} D_{26}, D_{34} and D_{50}. Make a conjecture about the number of elements of order 2 in D_n. (Be careful that your conjectures for Exercises 4.9 and 4.10 are not contradictory.)

4.11 Do you see any relationship between the orders of elements in a group and the order of the group?

Let C_m denote the cyclic group of order m generated by an m-cycle. For any pair of positive

integers m and n, let $C_m \oplus C_n = \{(a, b) \mid a \in C_m, b \in C_n\}$. For any pair of elements (a, b) and (c, d) in $C_m \oplus C_n$, define $(a, b) * (c, d) = (a * c, b * d)$. This binary operation makes $C_m \oplus C_n$ into a group. We can set up groups of this form in GAP. For example the following creates the group $G = C_4 \oplus C_6$:

```
gap> c4:= CyclicGroup(IsPermGroup, 4);
Group([ (1,2,3,4) ])
gap> c6:= CyclicGroup(IsPermGroup, 6);
Group([ (1,2,3,4,5,6) ])
gap> G:= DirectProduct(c4, c6);;
gap> IsCyclic(G);
false
```

Note that even though C_4 and C_6 are cyclic groups, the group $C_4 \oplus C_6$ is not cyclic.

Exercises

4.12 Determine whether or not $C_m \oplus C_n$ is cyclic for $(m, n) = (2, 2), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (3, 6), (3, 7), (3, 8), (3, 9)$, and $(4, 8)$. On the basis of this output, guess how m and n must be related for $C_m \oplus C_n$ to be cyclic. [Gallian, Chapter 4, Computer Exercise 2]

Appendix for Chapter 4

The following is the file "orderFrequency" which is used in this chapter. (Thanks to Alexander Hulpke for providing a revised version.)

```
orderFrequency:= function(g)
local h,w;
w:= [];
w:= h -> Collected(List(Elements(h), Order));
Print("[Order of element, Number of that order]=");
return w(g);
end;
```

5 Chapter: Permutation Groups

Consider the puzzle in Figure 5.1 at the end of this chapter. Think of the space in the middle (without a number) as empty. You can slide the other numbers along any of the lines into an empty spot but you can not lift or jump over other numbers. One rearrangement we can do is move the disk in position 1 into the middle then move the disk in position 6 to position 1, then the disk in position 5 to position 6, then the disk in position 4 to position 5, then the disk in position 3 to position 4, and then the disk in the middle to position 3. Denote this rearrangement by r. We could also move the disk in position 1 to the middle, then move the disk in position 2 to position 1, then move the disk in position 3 to position 2 and then move the disk in the middle to the position 3. Convince yourself that all possible rearrangements of Figure 5.1 can be obtained from taking a finite combination of these two rearrangements. These rearrangements form a group. Enter this group into GAP.

```
gap> G:= SymmetricGroup(6);
Sym( [1 ..  6] )
gap> r:=(1,3,4,5,6);
(1,3,4,5,6)
gap> s:=(1,3,2);
(1,3,2)
gap> K:= Subgroup(G, [r,s]);
Group([(1,3,4,5,6), (1,3,2)]);
gap> Elements(K);
```

The elements in the subgroup K describe all the possible arrangements of this puzzle. If you want to know how to get the original arrangement of the puzzle into a particular arrangement, use the `Factorization` command in GAP. For example, to see how to change the original arrangement to the arrangement in Figure 5.2 type

```
gap> Factorization(K,(2,3,4));
x1^-1*x2*x1
```

The `x1` denotes the first generator of K and `x2` denotes the second generator of K. Thus the above output from GAP tells us $(2,3,4) = (1,3,4,5,6)^{-1} * (1,3,2) * (1,3,4,5,6)$. (Remember that GAP multiplies permutations from left to right!) The element $(1,3,4,5,6)^{-1}$ denotes the inverse of $(1,3,4,5,6)$. In terms of the puzzle, $(1,3,4,5,6)^{-1}$ means reversing the loop $(1,3,4,5,6)$, which is the rearrangement $(1,6,5,4,3)$. Note that $(1,3,4,5,6)^{-1} = (1,3,4,5,6)^4$ and $(1,3,2)^{-1} = (1,3,2)^2$.

Exercises

5.1 Indicate what arrangement of the puzzle in Figure 5.1 corresponds to the following permutations. Use GAP to determine how (if possible) to get the following arrangements from the arrangement in Figure 5.1.
a) (4,5,6)
b) (2,3)
c) (1,2)(3,4)

d) (1,2)(3,4)(5,6)
Check the answers to parts a and c by hand. The function `Factorization(K,a)` in GAP will return `fail` if a cannot be expressed in terms of the generators of K. [Gallian, Chapter 5, Example 9]

5.2 Repeat Exercise 5.1 for the puzzle in Figure 5.3. Hint: there are two permutations that generate all possible permutations of this puzzle. Check the answer to part d by hand.

5.3 Let G be S_{12}. The cycle structure of a permutation is the number of 2-cycles, 3-cycles, etc. it contains when it is written as the product of disjoint cycles. For example (1,2,3)(4,5) and (1,3,6)(2,7) have the same cycle structure. Let $a = (1, 5, 10)$ and $b = (1, 3, 5, 7, 9, 11)$. (Note a and b are elements of G.)
a) What do you think will be the cycle structure of b^2, b^3 and b^6? Check your answer using GAP.
b) Compute ab. What do you think will be the cycle structure of $(ab)^3$ and $(ab)^4$?

Make a cube out of paper or cardboard. Label the 8 vertices of the cube 1-8 as in Figure 5.4. What do you think is the order of the group of rotations of the cube? Call this group G. It is a subgroup of S_8 because each rotation can be described by noting the movement of the 8 vertices. Notice the element $a = (1, 2, 3, 4)(5, 6, 7, 8)$ is in G since it represents a 90 degree rotation about the axis passing through the centers of the front and back faces. Thus a^k is in G for every power k.

```
gap> S:=SymmetricGroup(8);
Sym([1 ..  8])
gap> a:=(1,2,3,4)(5,6,7,8);
(1,2,3,4)(5,6,7,8)
gap> H:= Subgroup(S,[a]);
Group([ (1,2,3,4)(5,6,7,8) ])
gap> Elements(H);
[ (), (1,2,3,4)(5,6,7,8), (1,3)(2,4)(5,7)(6,8), (1,4,3,2)(5,8,7,6) ]
```

Note that $H \neq G$ since, for example, $b = (1, 5, 8, 4)(2, 6, 7, 3)$ is in G.

```
gap> b:=(1,5,8,4)(2,6,7,3);
(1,5,8,4)(2,6,7,3)
gap> M:=Subgroup(S,[a,b]);;
gap> Elements(M);
[ (), (2,4,5)(3,8,6), (2,5,4)(3,6,8), (1,2)(3,5)(4,6)(7,8),
(1,2,3,4)(5,6,7,8), (1,2,6,5)(3,7,8,4), (1,3,6)(4,7,5),
(1,3)(2,4)(5,7)(6,8), (1,3,8)(2,7,5), (1,4,3,2)(5,8,7,6),
(1,4,8,5)(2,3,7,6), (1,4)(2,8)(3,5)(6,7), (1,5,6,2)(3,4,8,7),
(1,5,8,4)(2,6,7,3), (1,5)(2,8)(3,7)(4,6), (1,6,3)(4,5,7),
(1,6)(2,5)(3,8)(4,7), (1,6,8)(2,7,4), (1,7)(2,3)(4,6)(5,8),
(1,7)(2,6)(3,5)(4,8), (1,7)(2,8)(3,4)(5,6), (1,8,6)(2,4,7),
(1,8,3)(2,5,7), (1,8)(2,7)(3,6)(4,5) ]
gap> Size(M);
```

```
24
```

Convince yourself that every rotation of the cube is in M so $G = M$. Thus the rotational symmetries of a cube is a subgroup of S_8 of order 24.

Exercises

5.4 Use GAP to describe the symmetry $(1, 8, 3)(2, 5, 7)$ of the cube in terms of the generators a and b.

The function `CycleStructurePerm(a)` in GAP gives the cycle structure of the permutation a. The cycle structure of (1,2,3)(4,5)(6,7) for example, is denoted by [2,1]. (The first spot in the bracket notation denotes the number of 2-cycles, the second spot the number of 3-cycles, etc.) The cycle structure of (1,2,3)(4,5,6,7,8) is denoted by [,1,,1]. This means there are no 2-cycles, one 3-cycle, no 4-cycles and one 5-cycle. The absence of a number after a comma indicates there are no cycles of that length.

```
gap> CycleStructurePerm((1,2,3)(4,5)(6,7));
[ 2, 1 ]
gap> CycleStructurePerm((1,2,3)(4,5,6,7,8));
[ , 1, , 1 ]
```

The following function lists all the elements in a permutation group G that have the cycle structure s:

```
gap> cstruc:= function(G,s)
> return Filtered(Elements(G), x -> CycleStructurePerm(x) = s);
> end;
function( G, s ) ... end
```

(Thanks to Alexander Hulpke for providing this function.) Type this function into GAP or fetch it from the website. We can now use this function to, for example, find all the elements in S_6 that have one 2-cycle and one 4-cycle:

```
gap> cstruc(SymmetricGroup(6),[1,,1]);
[ (1,2)(3,4,5,6), (1,2)(3,4,6,5), (1,2)(3,5,6,4), (1,2)(3,5,4,6),
(1,2)(3,6,5,4), (1,2)(3,6,4,5), (1,2,3,4)(5,6), (1,2,3,5)(4,6),
(1,2,3,6)(4,5), (1,2,4,3)(5,6), (1,2,4,6)(3,5), (1,2,4,5)(3,6),
(1,2,5,3)(4,6), (1,2,5,6)(3,4), (1,2,5,4)(3,6), (1,2,6,3)(4,5),
(1,2,6,5)(3,4), (1,2,6,4)(3,5), (1,3,4,2)(5,6), (1,3,5,2)(4,6),
(1,3,6,2)(4,5), (1,3)(2,4,5,6), (1,3)(2,4,6,5), (1,3,2,4)(5,6),
(1,3,5,6)(2,4), (1,3,6,5)(2,4), (1,3)(2,5,6,4), (1,3)(2,5,4,6),
(1,3,2,5)(4,6), (1,3,4,6)(2,5), (1,3,6,4)(2,5), (1,3)(2,6,5,4),
(1,3)(2,6,4,5), (1,3,2,6)(4,5), (1,3,4,5)(2,6), (1,3,5,4)(2,6),
(1,4,3,2)(5,6), (1,4,6,2)(3,5), (1,4,5,2)(3,6), (1,4,2,3)(5,6),
(1,4,5,6)(2,3), (1,4,6,5)(2,3), (1,4)(2,3,5,6), (1,4)(2,3,6,5),
(1,4,6,3)(2,5), (1,4)(2,5,6,3), (1,4)(2,5,3,6), (1,4,2,5)(3,6),
```

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

Exercises

5.5 Use GAP to find the number of permutations in S_9 of the following forms:
a) A product of a 4-cycle, and two 2-cycles (for example $(1,2,3,4)(5,6)(7,8)$)
b) A product of a 5-cycle and a 4-cycle
c) A product of three 3-cycles
d) A product of four 2-cycles.

5.6 Recall the command `Centralizer(G,a)` finds the centralizer of an element a in a group G. Find the size of centralizer of each of the following elements in S_9:
a) $(1,2,3,4)(5,6)(7,8)$ and $(5,1,3,4)(2,6)(7,8)$
b) $(1,2,3,4,5)(6,7,8,9)$ and $(1,4,9,6,7)(2,3,5,8)$
c) $(1,2,3)(4,5,6)(7,8,9)$ and $(1,5,8)(2,4,9)(3,6,7)$
d) $(1,2)(3,4)(5,6)(7,8)$ and $(1,9)(2,8)(3,7)(4,6)$.
Based on these answers, for any element $a \in S_n$, make a conjecture about the number of elements in the centralizer of a and the number of element in the centralizer of any permutation in S_n with the same cycle structure as a. Test your conjecture out for some elements of S_7.

5.7 Find a relationship between the answers you obtained in each part of Exercises 5.5 and 5.6 and the order of S_9.

5.8 Pick an element in S_9 and call it a. Compare its cycle structure to the cycle structure of the permutation bab^{-1} for
a) $b = (1,2,3,4,5,6,7,8,9)$
b) $b = (1,2)(3,4)(5,6)(7,8)$
c) $b = (1,2,3,4)(5,6,7,8)$.

5.9 Repeat Exercise 5.8 for a different element a in S_9.

5.10 Make a conjecture about, given two elements a and b in a group of permutations G, how the cycle structure of a and bab^{-1} are related. Test your conjecture for a pair of elements in the dihedral group D_{50}.

5.11 Based on your conjecture in Exercise 5.10, make a conjecture about a relationship between the order of an element a and the order of bab^{-1}.

5.12 Recall the command for finding the order of an element a in GAP is `Order(a)`. Let $a = (1, 2)$. For the elements b in Exercise 5.8 compute the orders of ab and ba. In these three cases is it true that $|ab| = |ba|$?

The elements $r = (1, 3, 4, 5, 6)$ and $s = (1, 3, 2)$ above generated A_6:

```
gap> Size(Group([(1,3,4,5,6), (1,3,2)]));
360
gap> Factorial(6)/2;
360
```

Using the fact that A_n is the only subgroup of S_n of order $|S_n|/2$, we know that r and s generate A_6.

In the following exercises we investigate subgroups of S_n generated by two elements.

Exercises

5.13 Use GAP to help you conjecture what subgroup of S_n is generated by $a = (1, 2)$ and $b = (1, 2, \ldots, n)$.

5.14 For a fixed n, calculate the order of the subgroup of S_n generated by $(1, x)$ and $(1, 2, 3, \ldots, n)$ for various choices of x and n. What conditions on x and n are both necessary and sufficient for $(1, x)$ and $(1, 2, 3, \ldots, n)$ to generate S_n? (Thanks to Daniel Heath at Pacific Lutherian University for providing this exercise.) [Gallian, Chapter 5, Computer Exercise 1]

5.15 Using GAP for at least eight values of n determine the subgroup of S_n generated by the $(n-1)$-cycle $(1, 3, 4, 5, \ldots, n)$ and the 3-cycle $(1, 3, 2)$.

5.16 Using GAP for at least eight values of n determine the subgroup of S_n generated by the $(n-1)$-cycle $(1, 3, 4, 5, \ldots, n)$ and the 4-cycle $(1, 4, 3, 2)$.

5.17 Explain **why** you get different subgroups of S_n in Exercises 5.15 and 5.16 depending on whether n is even or odd and on whether the second generator is a 3 or 4-cycle.

5.18 For a fixed n, calculate the order of the subgroup of S_n generated by $(1, x, y)$ and $(1, 2, 3, \ldots, n)$ for various choices of x and y. Conjecture a necessary and sufficient condition involving x, y, and n so that $(1, x, y)$ and $(1, 2, 3, \ldots, n)$ generate S_n. (Thanks to Daniel Heath at Pacific Lutherian University for providing this exercise.)

At this point, you may find it interesting to note how GAP updates the information about a group (or other constructed object) as it determines characteristics of the group. For example, create the subgroup of S_5 generated by the permutations (1,3) and (1,4,5):

```
gap> g:= Group([(1,3), (1,4,5)]);
Group([ (1,3), (1,4,5) ])
```

The command `KnownAttributesOfObject` returns the current information GAP contains on your group:

```
gap> KnownAttributesOfObject(g);
[ "LargestMovedPoint", "GeneratorsOfMagmaWithInverses",
"MultiplicativeNeutralElement" ]
```

(You can use the the command line help, ? followed by the command, to see what any of these three characteristics mean.) If we now compute with `g` and then reuse the command `KnownAttributesOfObject`, we see that GAP now has more information about our group `g`:

```
gap> Size(g);
24
gap> KnownAttributesOfObject(g);
[ "Size", "OneImmutable", "LargestMovedPoint", "NrMovedPoints",
"MovedPoints", "GeneratorsOfMagmaWithInverses",
"MultiplicativeNeutralElement", "HomePcgs", "Pcgs", "GeneralizedPcgs",
"StabChainMutable", "StabChainOptions" ]
```

In determining the order of our group, GAP went through constructions that determined these 9 new attributes.

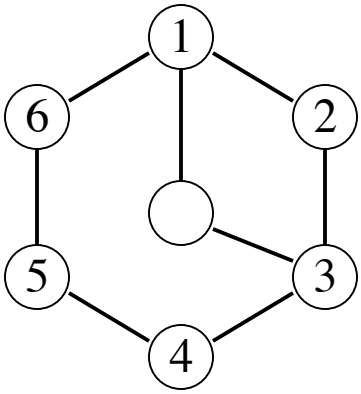


Figure 5.1

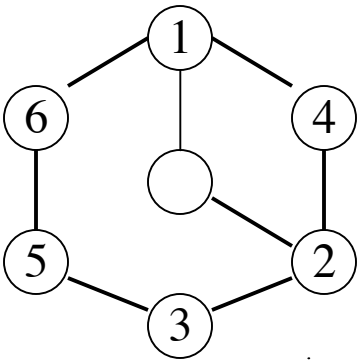


Figure 5.2

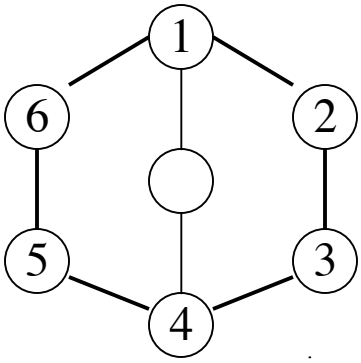


Figure 5.3

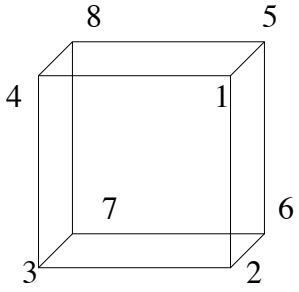


Figure 5.4

6 Chapter: Isomorphisms

An *automorphism* is a map $f : G \to G$ from a group G to itself that is operation preserving and both one-to-one and onto. We are going to examine the possible automorphisms of a finite cyclic group. Suppose we have a finite cyclic group G of order n generated by x. So $G = \{e, x, x^2, x^3, \ldots, x^{n-1}\}$. Define the map $f_k : G \to G$ by $f_k(x^i) = x^{ik}$ for $i = 1, 2, 3, \ldots, n-1$. (Thus for example $f_2(x^3) = x^6$.) Then f_k is a homomorphism. (Show this!) The question we are going to consider is when is f_k an automorphism? Since a finite cyclic group is finite and f_k is a map from G back to G, if we can show f_k is onto, it will have to be one-to-one. Consider the specific example where G is the cyclic group of order 8. As in Chapter 4 we will construct this group as all powers of an 8-cycle.

```
gap> G:= CyclicGroup(IsPermGroup, 8);
Group([ (1,2,3,4,5,6,7,8) ])
gap> Elements(G);
[ (), (1,2,3,4,5,6,7,8), (1,3,5,7)(2,4,6,8), (1,4,7,2,5,8,3,6),
(1,5)(2,6)(3,7)(4,8), (1,6,3,8,5,2,7,4), (1,7,5,3)(2,8,6,4), (1,8,7,6,5,4,3,2) ]
gap> a:= G.1;
(1,2,3,4,5,6,7,8)
gap> f:= x -> x^2;
function( x ) ...  end
gap> H:= Subgroup(G,[f(a)]);
Group([ (1,3,5,7)(2,4,6,8) ])
gap> Elements(H);
[ (), (1,3,5,7)(2,4,6,8), (1,5)(2,6)(3,7)(4,8), (1,7,5,3)(2,8,6,4) ]
gap> Size(H);
4
```

The third command above assigns a to the generator of the cyclic group of order 8. The fourth command defines a function f that takes an element x to x^2. H is the image of the map f. Note that H is a proper subgroup of G, so f is not an automorphism.

```
gap> f:= x -> x^3;
function( x ) ...  end
gap> H:= Subgroup(G,[f(a)]);
Group([ (1,4,7,2,5,8,3,6) ])
gap> Size(H);
8
```

In this case f is an automorphism, since $H = G$.

Exercises

6.1 In the cyclic group of order 10, use GAP to determine which f_k for $k = 1, 2, 3, 4, 5, 6, 7, 8, 9, 10$ are automorphisms. Based on your results, formulate a conjecture that describes when f_k is an automorphism of the cyclic group of order n.

6.2 Let G and $\bar{G}$ be groups and let $\Phi : G \to \bar{G}$ be an isomorphism. Then for all $a \in G$ the order of a equals the order of $\Phi(a)$ [Gallian, Theorem 6.2, Part 5]. Is there a connection between your conjecture and this fact? Explain.

7 Chapter: Cosets and Lagrange's Theorem

Let G be a group of permutations of a set S. For each $s \in S$ the *stabilizer* of s in G is the subgroup of G equal to $\{g \in G \mid g(s) = s\}$. The *orbit* of s under G is the subset of S equal to $\{g(s) \mid g \in G\}$.

The command `Orbit(G,s)` in GAP will give you the orbit of s under G. The command `Stabilizer(G,s)` creates the subgroup of G that is the stabilizer of s. For example:

```
gap> G:=SymmetricGroup(8);
Sym( [ 1 ..  8 ] )
gap> a:= (1,2,3)(4,5,6);;
gap> b:= (7,8);;
gap> H:=Subgroup(G,[a,b]);
Group([ (1,2,3)(4,5,6), (7,8) ])
gap> Elements(H);
[ (), (7,8), (1,2,3)(4,5,6), (1,2,3)(4,5,6)(7,8), (1,3,2)(4,6,5),
(1,3,2)(4,6,5)(7,8) ]
gap> Orbit(H,1);
[ 1, 3, 2 ]
gap> Orbit(H,7);
[ 7, 8 ]
gap> Stabilizer(H,1);
Group([ (7,8) ])
gap> Stabilizer(H,7);
Group([ (1,2,3)(4,5,6) ])
gap> Elements(Stabilizer(H,7));
[ (), (1,2,3)(4,5,6), (1,3,2)(4,6,5) ]
```

Careful: Notice that the command `Stabilizer(G,s)` returns a statement describing the stabilizer of s in G in terms of the generators of this stabilizer. To see all the elements in this group you need to use the commands `Elements(Stabilizer(G,s))`.

Exercises

7.1 Find the number of elements in Orbit(G, s) for $G = D_{10}$ and $s = 1, 2, 3$ and 4.

7.2 Find the number of elements in Stabilizer(G, s) for $G = D_{10}$ and $s = 1, 2, 3$ and 4.

7.3 Repeat Exercises 7.1 and 7.2 for D_{49} and D_{50}.

7.4 Make a conjecture about the number elements in Stabilizer(G, s) and in Orbit(G, s) for any $s \in \{1, 2, 3, \ldots, n\}$.

7.5 Explain, geometrically, why your conjecture in Exercise 7.4 is true.

7.6 Generalize the conjecture made in Exercise 7.4 to other finite permutation groups. Use GAP to help you formulate this conjecture.

8 Chapter: External Direct Products

In this chapter you will again need the file "orderFrequency". You will need to read this file into GAP in order to do the exercises in this section. The command to form external direct products is `DirectProduct`. For example:

```
gap> C4:= CyclicGroup(IsPermGroup,4);
Group([ (1,2,3,4) ])
gap> S3:=SymmetricGroup(3);
Sym( [ 1 ..  3 ] )
gap> D:= DirectProduct(S3,C4);
Group([ (1,2,3), (1,2), (4,5,6,7) ])
gap> Size(D);
24
gap> Read("orderFrequency");
gap> orderFrequency(D);
[Order of element, Number of that order]=[ [ 1, 1 ], [ 2, 7 ], [ 3, 2 ],
[ 4, 8 ], [ 6, 2 ], [ 12, 4 ] ]
```

The first command above assigns the name $C4$ to the cyclic group of order 4 (generated by (1,2,3,4)). The third command forms the direct product $S3 \oplus C4$. The next two lines tell us that $S3 \oplus C4$ has 24 elements. (The order of the external direct product of two finite groups G_1 and G_2 is $|G_1||G_2|$.) From the last output above we see that $S3 \oplus C4$ has one element of order 1, seven elements of order 2, two elements of order 3, eight elements of order 4, two elements of order 6, and four elements of order 12.

Exercises

8.1 Find the number of elements of order 5 in $\mathbf{Z}_{25} \oplus \mathbf{Z}_5$. [Gallian, Chapter 8, Example 4]

8.2 Find the number of cyclic **subgroups** of order 10 in $\mathbf{Z}_{100} \oplus \mathbf{Z}_{25}$. (Hint: First find the number of elements of order 10. How many elements of order 10 are in a cyclic subgroup of order 10? Do any of these cyclic subgroups have an element of order 10 in common?) [Gallian, Chapter 8, Example 5]

8.3 **By hand** find the number of elements of each order in $D_{10} \oplus \mathbf{Z}_2$.

8.4 Check your answer to Exercise 8.3 using `orderFrequency`.

8.5 Use `orderFrequency` to find the number of elements of each order in $D_5 \oplus \mathbf{Z}_4$.

8.6 Are $D_{10} \oplus \mathbf{Z}_2$ and $D_5 \oplus \mathbf{Z}_4$ isomorphic? Why or why not?

8.7 **By hand** find the number of elements of each order in D_{20}.

8.8 Check your answer to Exercise 8.7 using `orderFrequency`. Is D_{20} isomorphic to either $D_{10} \oplus \mathbf{Z}_2$

or $D_5 \oplus \mathbf{Z}_4$?

8.9 Find 4 nonisomorphic groups of order 40. How many nonisomorphic groups of order 40 can you find?

The command `AllSmallGroups(n)` gives a list of all groups of order n. (Type `?AllSmallGroups` while in GAP to see the limitations on the integers n that can be used in this command.) For example the following is a list of all groups of order 20:

```
gap> Gorder20:= AllSmallGroups(20);
[ <pc group of size 20 with 3 generators>,
  <pc group of size 20 with 3 generators>,
  <pc group of size 20 with 3 generators>,
  <pc group of size 20 with 3 generators>,
  <pc group of size 20 with 3 generators> ]
```

The output does not appear to be useful. But we see there are five groups of order 20. We can refer to each of these groups in the list. For example, `Gorder20[1]` refers to the first group of order 20 listed above. We can know explore properties of each of these five groups:

```
gap> Read("orderFrequency");
gap> orderFrequency(Gorder20[1]);
[Order of element, Number of that order]=[ [ 1, 1 ], [ 2, 1 ], [ 4, 10 ],
[ 5, 4 ], [ 10, 4 ] ]
gap> IsAbelian(Gorder20[1]);
false
gap> d10:= DihedralGroup(IsPermGroup,20);
Group([ (1,2,3,4,5,6,7,8,9,10), (2,10)(3,9)(4,8)(5,7) ])
gap> IsomorphismGroups(d10, Gorder20[1]);
fail
gap> IsomorphismGroups(d10, Gorder20[4]);
[ (1,2,3,4,5,6,7,8,9,10), (2,10)(3,9)(4,8)(5,7) ] -> [ f2*f3, f1*f2*f3^2 ]
```

From the output of the last and next to last above commands we see that the first group listed is not isomorphic to D_{10} but the fourth group listed is isomorphic to D_{10}.

By looping through a list we can actually get the "orderFrequency", for example, of each of these groups of order 20:

```
gap> List(AllSmallGroups(20) , x -> orderFrequency(x));
[ [ [ 1, 1 ], [ 2, 1 ], [ 4, 10 ], [ 5, 4 ], [ 10, 4 ] ],
  [ [ 1, 1 ], [ 2, 1 ], [ 4, 2 ], [ 5, 4 ], [ 10, 4 ], [ 20, 8 ] ],
  [ [ 1, 1 ], [ 2, 5 ], [ 4, 10 ], [ 5, 4 ] ],
  [ [ 1, 1 ], [ 2, 11 ], [ 5, 4 ], [ 10, 4 ] ],
  [ [ 1, 1 ], [ 2, 3 ], [ 5, 4 ], [ 10, 12 ] ] ]
```

In GAP a matrix can be entered as a list of row vectors. For example, the matrix

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

is entered by typing

```
gap> M:= [ [1,2,3], [4,5,6], [7,8,9]];
[ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ]
```

If you prefer to exhibit the matrix M as a 3 by 3 array use the command `PrintArray`:

```
gap> PrintArray(M);
[ [  1,  2,  3 ],
  [  4,  5,  6 ],
  [  7,  8,  9 ] ]
```

8.10 Let $G = \mathbf{Z}_3 \oplus \mathbf{Z}_3 \oplus \mathbf{Z}_3$ and let H be the subgroup of $\mathrm{SL}(3, \mathbf{Z}_3)$ consisting of

$$H = \{ \begin{bmatrix} 1 & a & b \\ 0 & 1 & c \\ 0 & 0 & 1 \end{bmatrix} \mid a, b, c \in \mathbf{Z}_3 \}.$$

Determine the number of elements of each order in G and H. Are G and H isomorphic? (This exercise shows that two groups with the same number of elements of each order need not be isomorphic.) [Gallian, Supplementary Exercises for Chapter 5-8, Exercise 5] Hint: Since H is generated by

$$\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \text{ and } \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

the following sets up this group H:

```
gap> z:= Z(3);;
gap> a:= [[z^0, z^0, 0*z], [0*z, z^0,  0*z], [0*z, 0*z, z^0]];;
gap> b:= [[z^0, 0*z, z^0], [0*z, z^0,  0*z], [0*z, 0*z, z^0]];;
gap> c:= [[z^0, 0*z, 0*z], [0*z, z^0,  z^0], [0*z, 0*z, z^0]];;
gap> H:= Subgroup(SL(3,3),[a,b,c]);
<matrix group with 3 generators>
```

9 Chapter: Normal Subgroups and Factor Groups

GAP can be used to compute right cosets and factor groups.

```
gap> S6:= SymmetricGroup(6);
Sym( [1..6] )
gap> A6:= AlternatingGroup(6);
Alt( [1..6] )
gap>D6:= DihedralGroup(IsPermGroup, 12);
Group([ (1,2,3,4,5,6), (2,6)(3,5) ])
gap> Z6:= Center(D6);
Group([ (1,4)(2,5)(3,6) ])
```

The above assigns the name `S6` to the symmetric group S_6, `A6` to the subgroup of even permutations, `D6` to the dihedral group D_6, and `Z6` to the center of D_6. To form factor groups we need **normal** subgroups. Test which subgroups are normal:

```
gap> IsNormal(S6,A6);
true
gap> IsNormal(S6,D6);
false
gap> IsNormal(D6,Z6);
true
```

Thus A_6 is normal in S_6 and $Z(D_6)$ is normal in D_6.

```
gap> RightCosets(S6,A6);
[RightCoset(AlternatingGroup[ 1..6 ]), ()),
RightCoset(AlternatingGroup[ 1..6 ]), (5,6))]
```

The above output tells us the right cosets of A_6 in S_6 are A_6 and $A_6(5,6)$. (**Remember:** GAP multiplies permutations from left to right. If your textbook multiplies permutations from right to left then a right coset in GAP will be a left coset using the notation of your textbook, when we are dealing with groups of permutations.) Thus the factor group S_6/A_6 has two elements.

```
gap> Size(FactorGroup(S6,A6));
2
```

Now consider the factor group $D_6/Z(D_6)$.

```
gap> RightCosets(D6,Z6);
[ RightCoset(Group([(1,4)(2,5)(3,6)] ), ()),
RightCoset(Group([(1,4)(2,5)(3,6)] ), (2,6)(3,5)),
RightCoset(Group([(1,4)(2,5)(3,6)] ), (1,5,3)(2,6,4)),
RightCoset(Group([(1,4)(2,5)(3,6)] ), (1,5)(2,4)),
```

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

Thus there are 6 right cosets: N, $N(2,6)(3,5)$, $N(1,3)(4,6)$, $N(1,3,5)(2,4,6)$, $N(1,5)(2,4)$ and $N(1,5,3)(2,6,4)$ where $N = Z(D_6)$. So the factor group has 6 elements. Which group of order 6 is $D_6/Z(D_6)$? We will now use GAP to help us answer this question.

```
gap> F:= FactorGroup(D6,Z6);
Group([ f1, f2 ])
gap> IsAbelian(F);
false
```

Since $D_6/Z(D_6)$ is non-Abelian, it can not be isomorphic to $\mathbf{Z}_6$.

```
gap> Read("orderFrequency");
gap> orderFrequency(F);
[Order of element, Number of that order]=[ [ 1, 1 ], [ 2, 3 ], [ 3, 2 ] ]
gap> orderFrequency(SymmetricGroup(3));
[Order of element, Number of that order]=[ [ 1, 1 ], [ 2, 3 ], [ 3, 2 ] ]
```

So $D_6/Z(D_6)$ and S_3 are non-Abelian groups of order 6 with the same number of elements of each order. Two groups that are isomorphic must have the same number of elements of each order. (The converse of this statement is false.) But in this case, this is enough to guarantee that $D_6/Z(D_6)$ and S_3 are isomorphic. [See for example Gallian, Theorem 7.3.])

GAP will also tell you which elements are in a particular coset. For example:

```
gap> Elements(RightCoset(Z6, (2,6)(3,5)));
[ (2,6)(3,5), (1,4)(2,3)(5,6)]
```

Thus the right coset $Z(D_6)(2,6)(3,5) = \{(2,6)(3,5), (1,4)(2,3)(5,6)\}$.

Exercises

9.1 Use GAP to find the right cosets of $Z(D_8)$ in D_8.

9.2 By hand, write out the Cayley table of the factor group $D_8/Z(D_8)$.

You can check your work to Exercise 9.2 by using the GAP command `MultiplicationTable`. For example, to find the Cayley table for S_3 type:

```
gap> e:= Elements(SymmetricGroup(3));
[ (), (2,3), (1,2), (1,2,3), (1,3,2), (1,3) ]
gap> PrintArray(MultiplicationTable( e ));
[ [  1,  2,  3,  4,  5,  6 ],
```

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

The GAP output of `PrintArray(MultiplicationTable( e ));` is an n by n array (where n is the order of the group) such that the integer in row i column j equal k if and only if the ith element in the list times the jth element equals the kth element.

9.3 The factor group $D_8/Z(D_8)$ is isomorphic to a group we have used often. Use GAP to help you determine which one.

9.4 Repeat Exercise 9.3 for the factor groups $D_{10}/Z(D_{10})$ and $D_{12}/Z(D_{12})$.

9.5 Based on your results in Exercises 9.3 and 9.4, make a conjecture about the factor group $D_n/Z(D_n)$ when n is even and greater than or equal to 8.

10 Chapter: Group Homomorphisms

The command `GroupHomomorphismByImages(G,H,[list of generators of G], [list of images of these generators])` in GAP will create the specified homomorphism. For example:

```
gap> S3:= SymmetricGroup(3);
Sym([1..3])
gap> f1:= GroupHomomorphismByImages(S3,S3, [(1,2,3),(1,3)], [(1,3,2),(1,2)]);
[(1,2,3), (1,3)] → [(1,3,2), (1,2)]
```

$f1$ is the homomorphism $f1 : S_3 \to S_3$ that maps (1,2,3) to (1,3,2) and (1,3) to (1,2).

```
gap> Image(f1, (2,3));
(2,3)
gap> Image(f1,(1,2));
(1,3)
```

The above tells us that $f1(2,3) = (2,3)$ and $f1(1,2) = (1,3)$ (as you can easily check).

```
gap> Size(Image(f1));
6
gap> Kernel(f1);
Group(())
```

Thus $f1$ is an automorphism. (Again this is easy to check by hand.) For another example consider the following. Read through the GAP commands and be sure you understand the output.

```
gap> f2:= GroupHomomorphismByImages(S3,S3, [(1,2,3),(1,3)], [(),(1,2)]);
[(1,2,3), (1,3)] → [ (), (1,2) ]
gap> Size(Image(f2));
2
gap> H:=Image(f2);
Group([ (), (1,2) ])
gap> Image(f2, (2,3));
(1,2)
gap> Kernel(f2);
Group([(1,2,3)]);
```

If you define a map that is not a homomorphism, GAP will return `fail`

```
gap> f3:= GroupHomomorphismByImages(S3,S3, [(1,2,3),(1,3)], [(1,3),(1,2)]);
fail
```

$f3$ maps (1,2,3) (an element of order 3) to (1,3) (an element of order 2) so $f3$ is not a homomorphism [Gallian, Theorem 10.1, Part 3].

Recall the group D_n is a subgroup of S_n which is generated by a rotation of order n and a reflection. Thus $(1, 2, 3, \ldots, n)$ and $(1, n)(2, n-1)\ldots(\frac{n}{2}, \frac{n}{2}+1)$ generate D_n when n is even and $(1, 2, 3, \ldots, n)$ and $(1, n)(2, n-1)\ldots(\frac{n-1}{2}, \frac{n-1}{2}+2)$ generate D_n when n is odd. So, for example, every element in D_6 can be written as products of powers of $(1, 2, 3, 4, 5, 6)$ and $(1, 6)(2, 5)(3, 4)$ and every element in D_7 can be written as products of powers of $(1, 2, 3, 4, 5, 6, 7)$ and $(1, 7)(2, 6)(3, 5)$.

One way to determine if a homomorphism from the finite group G to itself is an automorphism is to determine if it is onto. Thus, for example the homomorphism `f1` on the previous page is an automorphism because the image of `f1` is all of S_3. In the following exercises you will need to use the `GroupHomomorphismByImages` command in GAP to find homomorphisms from D_n to D_n. You will then need to check if they are automorphisms by checking to see if the kernel contains only the identity or by checking that the image is all of D_n. Since a homomorphism is completely determined by the image of the generators of a group, you only need to specify where you want to map the two generators of D_n. You may want to use the fact that the order of the homomorphic image of an element must divide the order of the original element [Gallian, Theorem 10.1, Part 3]. This will help you narrow the possibilities, before using GAP to test for automorphisms.

The files "autoDn" and "homoDn" contain functions that will list all the automorphisms and homomorphisms of a given dihedral group into itself. (Thanks to Alexander Hulpke for providing these functions.) Both files are on the website as well as in the appendix to this chapter. The following GAP output is all the automorphisms and then all the homomorphisms of D_6 into itself:

```
gap> Read("autoDn");
gap> Read("homoDn");
gap> d6:= DihedralGroup(IsPermGroup,12);
Group([ (1,2,3,4,5,6), (2,6)(3,5) ])
gap> autoDn(d6);
[ [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,2,3,4,5,6), (2,6)(3,5) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,2,3,4,5,6), (1,2)(3,6)(4,5) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,2,3,4,5,6), (1,3)(4,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,2,3,4,5,6), (1,4)(2,3)(5,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,2,3,4,5,6), (1,5)(2,4) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,2,3,4,5,6), (1,6)(2,5)(3,4) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (2,6)(3,5) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,2)(3,6)(4,5) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,3)(4,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,4)(2,3)(5,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,5)(2,4) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,6)(2,5)(3,4) ] ]
gap> homoDn(d6);
[ [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (), () ], [ (1,2,3,4,5,6), (2,6)(3,5) ] ->
  [ (), (2,6)(3,5) ], [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (), (1,2)(3,6)(4,5) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (), (1,3)(4,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (), (1,4)(2,3)(5,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (), (1,4)(2,5)(3,6) ],
```

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

```
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,5,3)(2,6,4), (1,4)(2,3)(5,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,5,3)(2,6,4), (1,5)(2,4) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,5,3)(2,6,4), (1,6)(2,5)(3,4) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (2,6)(3,5) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,2)(3,6)(4,5) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,3)(4,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,4)(2,3)(5,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,5)(2,4) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6,5,4,3,2), (1,6)(2,5)(3,4) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6)(2,5)(3,4), () ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6)(2,5)(3,4), (1,3)(4,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6)(2,5)(3,4), (1,4)(2,5)(3,6) ],
  [ (1,2,3,4,5,6), (2,6)(3,5) ] -> [ (1,6)(2,5)(3,4), (1,6)(2,5)(3,4) ] ]
gap> Size(autoDn(d6));
12
gap> Size(homoDn(d6));
64
```

As a homomorphism is completely determined by its image on a set of generators of the group, GAP only specifies the image of a set of generators of D_6. (Be patient when using these functions; they take awhile.)

Exercises

10.1 **By hand** find three automorphism of D_4. Using GAP determine the number of automorphisms of D_4.

10.2 **By hand** find three homomorphisms from D_4 to D_4 that are not automorphisms. Using GAP determine the number of homomorphisms from D_4 to D_4.

10.3 Repeat Exercises 10.1 and 10.2 for D_5.

10.4 **Using GAP** repeat Exercises 10.1 and 10.2 for D_{19}, D_{21}, D_{45} and D_{49}.

10.5 Make a conjecture about the number of homomorphisms and the number of automorphisms of D_n when n is odd.

10.6 **Using** GAP repeat Exercises 10.1 and 10.2 for D_{20}, D_{24}, D_{48} and D_{50}.

10.7 Make a conjecture about the number of homomorphisms and the number of automorphisms of D_n when n is even.

Appendix

The file autoDn:

```
autoDn:= function(G)
local a,b,aims,bims,maps,autos,abims;
a:=G.1; b:=G.2;
aims:=Filtered(Elements(G), i -> Order(a) = Order(i));
bims:=Filtered(Elements(G), i -> Order(b) = Order(i));
abims:= Cartesian(aims,bims);
maps:= List(abims, i -> GroupHomomorphismByImages(G,G,[a,b],i));
maps:= Filtered(maps, i -> i <> fail);
autos:= Filtered(maps, IsInjective);
return autos;
end;
```

The file homoDn:

```
homoDn:= function(G)
local a,b,aims,bims,maps,homos,abims;
a:=G.1; b:=G.2;
aims:=Filtered(Elements(G), i -> IsInt(Order(a) / Order(i)));
bims:=Filtered(Elements(G), i -> IsInt(Order(b) / Order(i)));
abims:= Cartesian(aims,bims);
maps:= List(abims, i -> GroupHomomorphismByImages(G,G,[a,b],i));
homos:= Filtered(maps, i -> i <> fail);
return homos;
end;
```

11 Chapter: Fundamental Theorem of Finite Abelian Groups

Recall from Chapter 8 of this manual, the command to form the direct product of two or more groups in GAP is `DirectProduct`. For example the below output creates a group G isomorphic to $\mathbf{Z}_6 \oplus \mathbf{Z}_6$:

```
gap> Z6:= CyclicGroup(IsPermGroup,6);
Group([ (1,2,3,4,5,6) ])
gap> G:=DirectProduct(Z6,Z6);
Group([ (1,2,3,4,5,6), (7,8,9,10,11,12) ])
```

The group $\mathbf{Z}_6$ is a cyclic group of order 6. The group `Z6` in the above GAP commands is also a cyclic group of order 6 (namely, the subgroup of S_6 generated by the permutation $(1,2,3,4,5,6)$). Thus $\mathbf{Z}_6$ is isomorphic to `Z6`.

Notice in the above output the elements in `Z6` that are in the second component are described as powers of the 6-cycle $(7,8,9,10,11,12)$ instead of the 6-cycle $(1,2,3,4,5,6)$. (The powers of $(7,8,9,10,11,12)$ also form a cyclic group of order 6, so $\langle(7,8,9,10,11,12)\rangle$ is isomorphic to $\mathbf{Z}_6$ as well.) GAP automatically changed the description of the elements in the second component of the direct summand from powers of $(1,2,3,4,5,6)$ to powers of $(7,8,9,10,11,12)$ since we are trying to represent an external direct product $H \oplus K$ as an internal direct product $H \times K$ and we need $H \cap K = \{e\}$.

By the Fundamental Theorem of Finite Abelian Groups every finite Abelian group is isomorphic to the direct product of cyclic groups of prime power order. We also know that a factor group G/H, where G is finite and Abelian, is also a finite Abelian group. Suppose $G = \mathbf{Z}_6 \oplus \mathbf{Z}_5 \oplus \mathbf{Z}_8$ and H is the subgroup of G generated by $(2,1,2)$. What finite Abelian group is G/H? The following steps in GAP resolve this question.

```
gap> Z5:= CyclicGroup(IsPermGroup,5);
Group([ (1,2,3,4,5) ])
gap> Z8:= CyclicGroup(IsPermGroup,8);
Group([ (1,2,3,4,5,6,7,8) ])
gap> G:= DirectProduct(Z6,Z5,Z8);
Group([ (1,2,3,4,5,6), (7,8,9,10,11), (12,13,14,15,16,17,18,19) ])
gap> H:= Subgroup(G, [(1,2,3,4,5,6)^2, (7,8,9,10,11),
>(12,13,14,15,16,17,18,19)^2]);
Group([ (1,3,5)(2,4,6), (7,8,9,10,11), (12,14,16,18)(13,15,17,19) ])
gap> F:= FactorGroup(G,H);
Group([ f1, f2 ])
gap> Size(F);
4
```

Note that $\mathbf{Z}_5$ is isomorphic to `Z5` and $\mathbf{Z}_8$ is isomorphic to `Z8`. Thus the `G` defined in the above GAP commands is isomorphic to $\mathbf{Z}_6 \oplus \mathbf{Z}_5 \oplus \mathbf{Z}_8$. Notice, in the above output, in the direct prod-

uct of `Z6, Z5` and `Z8`, the elements of `Z5` in the second component are written as powers of the permutation $(7, 8, 9, 10, 11)$. Similarly, the elements of `Z8` in the third component are written as powers of the permutation $(12, 13, 14, 15, 16, 17, 18, 19)$. The element $(1, 2, 3, 4, 5, 6)^2$ generates a subgroup of order 3 in `Z6`. Similarly $(12, 13, 14, 15, 16, 17, 18, 19)^2$ generates a subgroup of `Z8` or order 4. Thus `H` is isomorphic to the subgroup of $\mathbf{Z}_6 \oplus \mathbf{Z}_5 \oplus \mathbf{Z}_8$ generated by the element (2, 1, 2). The factor group is a finite Abelian group of order 4 so it must be isomorphic to either $\mathbf{Z}_4$ or $\mathbf{Z}_2 \oplus \mathbf{Z}_2$.

```
gap> Read("orderFrequency");
gap> orderFrequency(F);
[Order of element, Number of that order]=[ [ 1, 1 ], [ 2, 3 ] ]
```

Since the factor group has three elements of order 2 it must be isomorphic to $\mathbf{Z}_2 \oplus \mathbf{Z}_2$.

Let Zn denote a cyclic group of order n. If m divides n, then Zn contains a cyclic subgroup, Zm, of order m.

Exercises

11.1 **By hand**, describe the cosets of $(Z48/Z6)/(Z12/Z6)$. Since this group is a finite Abelian group it is a direct product of cyclic groups of prime power order. Describe which one.

Attempting to work Exercise 11.1 in GAP is a little tricky. We would like to thank Andy Miller of Belmont University for providing the following explanation on how to use GAP to work Exercise 11.1. First enter the cyclic groups $Z48, Z6$ and $Z12$ into GAP:

```
gap> Z48:= CyclicGroup(IsPermGroup,48);;
gap> Z48.1;
(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,
30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48)
gap> Z12:= Subgroup(Z48,[Z48.1^4]);;
gap> Z6:=Subgroup(Z48,[Z48.1^8]);;
```

Then construct the factor groups $Z48/Z6$ and $Z12/Z6$:

```
gap> g1:= FactorGroup(Z48,Z6);
Group([ f1, f2, f3 ])
gap> g2:= FactorGroup(Z12,Z6);
Group([ f1 ])
```

But when we then try to construct the factor group $(Z48/Z6)/(Z12/Z6)$ we get an error:

```
gap> FactorGroup(g1,g2);
Error, <N> must be a normal subgroup of <G> called from ...
```

The problem is the command `FactorGroup(G,H)` creates a group isomorphic (not equal) to G/H.

Thus `FactorGroup(Z12,Z6)` is not a subgroup of `FactorGroup(Z48,Z6)`. (`FactorGroup(Z12,Z6)` is only isomorphic to a subgroup of `FactorGroup(Z48,Z6)`). To express $Z12/Z6$ as a subgroup of $Z48/Z6$ we will use the fact that if N is a normal subgroup of a group G then the group G/N is the image of the homomorphism $\phi : G \to G/N$ defined by $\phi(g) = gN$. Thus, if H is a subgroup of G that contains N, the subgroup H/N of G/N is the image of H under ϕ. (That is, $H/N = \phi(H)$.) In the following GAP work we again have $g1 = Z48/Z6$ and $g2 = Z12/Z6$ but now $g1$ and $g2$ are constructed in such a way that $g2$ is a subgroup of $g1$.

```
gap> phi:= NaturalHomomorphismByNormalSubgroup(Z48,Z6);;
gap> g1:= Image(phi,Z48);
Group([ f1, f2, f3 ])
gap> g2:= Image(phi,Z12);
Group([ f3 ])
gap> IsSubgroup(g1,g2);
true
```

Since $g2$ is now a subgroup of $g1$ we can use the `FactorGroup` command and determine the isomorphism class of $g1/g2$.

```
gap> h:= FactorGroup(g1,g2);
Group([ f1, f2, <identity> of ...  ])
gap> Size(h);
4
gap> Read("orderFrequency");
gap> orderFrequency(h);
[Order of element, Number of that order]=[ [ 1, 1 ], [ 2, 1 ], [ 4, 2 ] ]
```

Since $g1/g2$ is an Abelian group of order 4 with an element of order 4, $g1/g2$ is isomorphic to $\mathbf{Z}_4$.

Exercises

11.2 Let $G = (Zr/Zs)/(Zt/Zs)$, where t divides r and s divides t. Make a conjecture about the isomorphism class of G. Prove your conjecture.

11.3 Let $G_1 = (Z24 \oplus Z8 \oplus Z12)/(Z2 \oplus Z2 \oplus Z2)$. Let $G_2 = (Z8 \oplus Z8 \oplus Z6)/(Z2 \oplus Z2 \oplus Z2)$. Use GAP to help you write G_1/G_2 as a direct product of cyclic groups of prime power order.

11.4 Repeat Exercise 11.3 for $G_1 = (Z40 \oplus Z10 \oplus Z24)/(Z2 \oplus Z5 \oplus Z2)$ and $G_2 = (Z20 \oplus Z10 \oplus Z6)/(Z2 \oplus Z5 \oplus Z2)$.

11.5 Generalize your conjecture given in Exercise 11.2. That is, make a conjecture about the factor group $(G/H)/(K/H)$ when G, H and K are finite Abelian groups, H is a subgroup of K and H is a subgroup of G.

12 Chapter: Introduction to Rings

The set of integers mod n, $\mathbf{Z}_n$, is a ring with binary operations addition and multiplication mod n. When n is a prime p, $\mathbf{Z}_p$ is a field. That is, $\mathbf{Z}_p$ is a commutative ring with 1 and every nonzero element is a unit.

Fact: The nonzero elements of $\mathbf{Z}_p$ form a cyclic group under multiplication mod p of order $p-1$.

The function `Z` in GAP creates a generator for this cyclic group. For example:

```
gap> z:= Z(7);
gap> R:=Ring([z]);
GF(7)
gap> Elements(R);
[ 0*Z(7), Z(7)^0, Z(7), Z(7)^2, Z(7)^3, Z(7)^4, Z(7)^5 ]
```

The nonzero elements of $\mathbf{Z}_7$ form a cyclic group (under multiplication) of order 6. The element `Z(7)` of `R` denotes a generator of this cyclic group. Note that mod 7, $3^1 = 3, 3^2 = 2, 3^3 = 6, 3^4 = 4, 3^5 = 5$ and $3^6 = 1$, so 3 is a generator of this cyclic group. Also note that mod 7, $5^1 = 5, 5^2 = 4, 5^3 = 6, 5^4 = 2, 5^5 = 3$ and $5^6 = 1$, so 5 is also generator of this cyclic group. Thus $Z(7)$ can be taken to mean either the 3 or 5 in $\mathbf{Z}_7$. (Mod 7, $2^3 = 1, 4^3 = 1$ and $6^2 = 1$ so $Z(7)$ cannot be equal to $2, 4$ or 6.) Recall cyclic groups of the same order n are isomorphic and are of the form $\{e, a, a^2, a^3, ..., a^{n-1}\}$, where a denotes a generator (using multiplicative notation). If you would like to have GAP show you which of 3 or 5 in $\mathbf{Z}_7$ is being used to generate the nonzero elements type:

```
gap> Int(Z(7));
3
```

The `Int` function translate elements of $\mathbf{Z}_p$ into integers. The function `One` is also useful. It gives the multiplicative identity in a ring and can be used to translate integers to elements in $\mathbf{Z}_p$. For example to see how 5 is denoted in $R = \mathbf{Z}_7$ type:

```
gap> 5*One(R);
Z(7)^5
```

So 3^5 mod 7 must be equal to 5 mod 7:

```
gap> 3^5 mod 7;
5
```

Exercises

12.1 Use GAP to help you find for which of the primes $p = 3, 5, 7, 11, 13$, and 17 the equation $a^2 + b^2 = 0$ has a nontrivial solution in $\mathbf{Z}_p$. Make a conjecture about the the existence of a nontrivial solution of this equation in $\mathbf{Z}_p$ for p a prime. [Gallian, Chapter 12, Computer Exercise 1] (If this exercise seems repetitive try writing a subroutine in GAP.)

The group of units in the ring of 2×2 matrices over $\mathbf{Z}_n$ is denoted in GAP by `GL(2, Integers mod`

n). The subgroup of matrices with determinant equal to 1 is denoted by SL(2, Integers mod n).

12.2 Find the orders of $\mathrm{GL}(2, \mathbf{Z}_n)$ and $\mathrm{SL}(2, \mathbf{Z}_n)$ for $n = 2, 3, 5, 7, 11$, and 13. What relationship do you see between the orders of $\mathrm{GL}(2, \mathbf{Z}_n)$ and $\mathrm{SL}(2, \mathbf{Z}_n)$ when n is a prime? Find the orders of $\mathrm{GL}(2, \mathbf{Z}_n)$ and $\mathrm{SL}(2, \mathbf{Z}_n)$ for $n = 16, 27, 25$, and 49. Make a conjecture about the relationship between the orders of $\mathrm{GL}(2, \mathbf{Z}_n)$ and $\mathrm{SL}(2, \mathbf{Z}_n)$ when n is a power of a prime. [Gallian, Chapter 12, Computer Exercise 4]

12.3 Find the orders of $\mathrm{GL}(2, \mathbf{Z}_n)$ and $\mathrm{SL}(2, \mathbf{Z}_n)$ for $n = 2, 4, 8, 16$, and 32. How do the orders of the two groups change each time you increase the power of 2 by 1? Find the orders of $\mathrm{GL}(2, \mathbf{Z}_n)$ and $\mathrm{SL}(2, \mathbf{Z}_n)$ for $n = 3, 9, 27$, and 81. How do the orders of the two groups change each time you increase the power of 3 by 1? Find the orders of $\mathrm{GL}(2, \mathbf{Z}_n)$ and $\mathrm{SL}(2, \mathbf{Z}_n)$ for $n = 5, 25, 125$, and 625. How do the orders of the two groups change each time you increase the power of 5 by 1? Make a conjecture about the relationship between the orders of $\mathrm{GL}(2, \mathbf{Z}_{p^i})$ and $\mathrm{GL}(2, \mathbf{Z}_{p^{i+1}})$. Make a conjecture about the relationship between the orders of $\mathrm{SL}(2, \mathbf{Z}_{p^i})$ and $\mathrm{SL}(2, \mathbf{Z}_{p^{i+1}})$. [Gallian, Chapter 12, Computer Exercise 4]

12.4 Find the order of $\mathrm{GL}(2, \mathbf{Z}_n)$ for $n = 12, 15, 20, 21$, and 30. Make a conjecture about the order of $\mathrm{GL}(2, \mathbf{Z}_n)$ in terms of the orders of $\mathrm{GL}(2, \mathbf{Z}_s)$ and $\mathrm{GL}(2, \mathbf{Z}_t)$ where $n = st$ and s and t are relatively prime. [Gallian, Chapter 12, Computer Exercise 4]

12.5 Fetch the subroutine intror2 from the manual website and read it into GAP. This function takes as input an integer n and returns a list of all the solutions to the equation $x^2 = -1$ in the ring $\mathbf{Z}_n$. For example intror2(5) will list all the solutions in $\mathbf{Z}_5$. In the ring $\mathbf{Z}_n$ find the number of solutions to the equation $x^2 = -1$ for n equal to each of the primes between 3 and 29. Make a conjecture about the number of solutions when n is an odd prime. In the ring $\mathbf{Z}_n$ find the number of solutions to the equation $x^2 = -1$ for n the square of each of the primes between 3 and 29. In the ring $\mathbf{Z}_n$ find the number of solutions to the equation $x^2 = -1$ for n the cube of each of the primes between 3 and 29. Make a conjecture about the number of solutions when n is a power of an odd prime. [Gallian, Chapter 12, Computer Exercise 5]

12.6 Using the subroutine mentioned in Exercise 12.5, find the number of solutions to the equation $x^2 = -1$ in $\mathbf{Z}_n$ for $n = 2^k, k = 1, 2, 3, 4, 5, 6$. Make a conjecture about the number of solutions when n is a power of two. [Gallian, Chapter 12, Computer Exercise 5]

12.7 Using the subroutine mentioned in Exercise 12.5, find the number of solutions to the equation $x^2 = -1$ for $n = 12, 20, 24, 28$, and 36. Make a conjecture about the number of solutions when n is a multiple of 4. [Gallian, Chapter 12, Computer Exercise 5]

12.8 Make a conjecture about the number of solutions to the equation $x^2 = -1$ in $\mathbf{Z}_n$ for $n = pq$ and $n = 2pq$ where p and q are odd primes. You should use the subroutine provided in Exercise 12.5 for many values of p and q to help you formulate your conjecture. [Gallian, Chapter 12, Computer Exercise 5]

12.9 Make a conjecture about the number of solutions to the equation $x^2 = -1$ in $\mathbf{Z}_n$ for $n = pqr$

and $n = 2pqr$ where p, q and r are odd primes. You should use the subroutine provided in Exercise 12.5 for many values of p, q and r to help you formulate your conjecture. What relationship do you see between the number of solutions when $n = p, n = q$ and $n = r$ and the case that $n = pqr$? [Gallian, Chapter 12, Computer Exercise 5]

12.10 Based on your answers to Exercises 12.5 - 12.9 formulate a conjecture on the number of solutions to $x^2 = -1$ in $\mathbf{Z}_n$.

Appendix for Chapter 12

The following is the file "intror2" which is used in this chapter.

```
intror2:= function(n)
local r, x, i;
x:= [];
i:= 1;
Print("The solutions to x^2 = -1 in Z_", n, " are ");
r:= Elements(Integers mod n);
  repeat
    if Int(r[i]^2) = n-1 then
    Add(x,r[i]);
    fi;
    i:= i+1;
  until i=n+1;
return x;
end;
```

13 Chapter: Integral Domains

In this chapter we will determine the number of idempotents and the number of nilpotent elements in the rings $\mathbf{Z}_n$. Recall that an *idempotent* in a ring R is an element r such that $r^2 = r$. A *nilpotent* element $r \in R$ is an element such that $r^m = 0$ for some positive integer m.

Fetch the file "nilpotentCount" off the website. This file contains a function that counts the number of nilpotent elements in a given ring. The appendix at the end of this chapter contains a print out of this file. (Thanks to Alexander Hulpke for providing a revised version of this function.) GAP has a built in function called `Idempotents` that lists the idempotents in a ring. For an example follow along with the following GAP output. (Recall GAP denotes a mod n in $\mathbf{Z}_n$ by `ZmodnZObj(a,n)`.)

```
gap> M:= Integers mod 6;
(Integers mod 6)
gap> Idempotents(M);
[ ZmodnZObj( 0, 6 ), ZmodnZObj( 1, 6 ), ZmodnZObj( 3, 6 ), ZmodnZObj( 4, 6 ) ]
gap> Size(Idempotents(M));
4
gap> N:= Integers mod 9;
(Integers mod 9)
gap> Size(Idempotents(N));
2
```

The above tells us that $\mathbf{Z}_6$ has 4 idempotents and $\mathbf{Z}_9$ has 2 idempotents.

```
gap> Read("nilpotentCount");
gap> nilpotentCount(M);
1
gap> nilpotentCount(N);
3
```

The above tells us that $\mathbf{Z}_6$ has 1 nilpotent element and $\mathbf{Z}_9$ has 3 nilpotents.

Exercises

13.1 Find the number of idempotents in $\mathbf{Z}_n$ for many values of n. Based on your results answer the following:
a) How many idempotents are in $\mathbf{Z}_n$ when n is a prime-power?
b) How many idempotents are in $\mathbf{Z}_n$ when n is equal to the product of two distinct primes?
c) In general, make a conjecture about the number of idempotents in $\mathbf{Z}_n$ as a function of n.
d) In the case where n is of the form pq where p and q are distinct primes can you see a relationship between the two idempotents that are not 0 and 1? [Gallian, Chapter 13, Computer Exercise 1]

13.2 Find the number of nilpotents in $\mathbf{Z}_n$ for many values of n. Based on your results answer the following:
a) How many nilpotents are in $\mathbf{Z}_n$ when n is a prime-power?

b) How many nilpotents are in $\mathbf{Z}_n$ when n is equal to the product of two distinct primes?
c) In general, make a conjecture about the number of nilpotents in $\mathbf{Z}_n$ as a function of n. [Gallian, Chapter 13, Computer Exercise 2]

13.3 Using GAP, find the number of units in $\mathbf{Z}_n$ for many values of n. Make a conjecture about the number of units in $\mathbf{Z}_n$ as a function of n. (The command `Elements(Units(R))` will list all the units in a given ring R.)

Appendix for Chapter 13

```
nilpotentCount:= function(R)
  local n;
  n:= Size(R);
  return Length(Filtered(Elements(R), i -> IsZero(i^n)));
end;
```

14 Chapter: Ideals and Factor Rings

The command in GAP that creates an ideal in the ring R is `Ideal(R,[`*list of generators*`])`. For example to create the ideal I in the ring $\mathbf{Z}_{20}$ generated by 3 and 5 type the following:

```
gap> R:= Integers mod 20;
(Integers mod 20)
gap> e:= Elements(R);
[ ZmodnZObj( 0, 20 ), ZmodnZObj( 1, 20 ), ZmodnZObj( 2, 20 ), ZmodnZObj( 3, 20 ),
  ZmodnZObj( 4, 20 ), ZmodnZObj( 5, 20 ), ZmodnZObj( 6, 20 ), ZmodnZObj( 7, 20 ),
  ZmodnZObj( 8, 20 ), ZmodnZObj( 9, 20 ), ZmodnZObj( 10, 20 ), ZmodnZObj( 11, 20 ),
  ZmodnZObj( 12, 20 ), ZmodnZObj( 13, 20 ), ZmodnZObj( 14, 20 ), ZmodnZObj( 15, 20 ),
  ZmodnZObj( 16, 20 ), ZmodnZObj( 17, 20 ), ZmodnZObj( 18, 20 ), ZmodnZObj( 19, 20 )
gap> I:= Ideal(R, [ e[4], e[6]]);
<two-sided ideal in (Integers mod 20), (2 generators)>
gap> Elements(I);
[ ZmodnZObj( 0, 20 ), ZmodnZObj( 1, 20 ), ZmodnZObj( 2, 20 ), ZmodnZObj( 3, 20 ),
  ZmodnZObj( 4, 20 ), ZmodnZObj( 5, 20 ), ZmodnZObj( 6, 20 ), ZmodnZObj( 7, 20 ),
  ZmodnZObj( 8, 20 ), ZmodnZObj( 9, 20 ), ZmodnZObj( 10, 20 ), ZmodnZObj( 11, 20 ),
  ZmodnZObj( 12, 20 ), ZmodnZObj( 13, 20 ), ZmodnZObj( 14, 20 ), ZmodnZObj( 15, 20 ),
  ZmodnZObj( 16, 20 ), ZmodnZObj( 17, 20 ), ZmodnZObj( 18, 20 ), ZmodnZObj( 19, 20 ) ]
```

In the third command line above `e[4]` denotes the element 3 in $\mathbf{Z}_{20}$ since 3 is the 4th element in the list of elements of $\mathbf{Z}_{20}$. Similarly, `e[6]` denotes the element 5 in $\mathbf{Z}_{20}$ since 5 is listed 6th. Note $I = R$.

Let I be an ideal in a commutative ring R. The *nilradical* of I is defined to be the set $N = \{r \in R \mid r^n \in I \text{ for some positive integer } n\}$.

Exercises

14.1 By hand, find all the ideals in $\mathbf{Z}_{24}$. Which ones are prime?

14.2 For two of the ideals in Exercise 14.1, call them I_1 and I_2, find the intersection of all prime ideals that contain I_1 and find the intersection of all prime ideals that contain I_2.

14.3 Write a short program in GAP that will determine the nilradical of an ideal.

14.4 Using your program from Exercise 14.3, find the nilradicals of I_1 and I_2.

14.5 Repeat Exercises 14.1, 14.2 and 14.4 for the ring $\mathbf{Z}_{900}$.

14.6 Based on your answers to 14.2, 14.4 and 14.5, make a conjecture about the nilradical of a ideal.

14.7 Use your program to find the nilradical of $\langle k \rangle$ in $\mathbf{Z}_n$ for $n = 8, 15, 24$ and for those k that divide n.

15 Chapter: Ring Homomorphisms

The map, f from $\mathbf{Z}_{10}$ to $\mathbf{Z}_{10}$ given by $f(x) = 2x$ is not a ring homomorphism. But the map g from $\mathbf{Z}_{10}$ to $\mathbf{Z}_{10}$ given by $g(x) = 5x$ is a ring homomorphism. (Convince yourself that these statements are true!) In this chapter we will investigate the question: Given a fixed n, for which m is the map $f : \mathbf{Z}_n \to \mathbf{Z}_n$ given by $f(x) = mx$ a ring homomorphism? Similarly, when is the map $f : \mathbf{Z}_n \to \mathbf{Z}_n$ given by $f(x) = x^m$ a ring homomorphism? Recall that any **group** homomorphism from $\mathbf{Z}_n$ to $\mathbf{Z}_n$ is completely determined by the image of 1 mod n. Since a ring homomorphism is also a group homomorphism, the image of any ring homomorphism from $\mathbf{Z}_n$ to $\mathbf{Z}_n$ is also completed determined by the image of 1 mod n.

Consider the following example in GAP:

```
gap> R:= Integers mod 10;
(Integers mod 10)
gap> e:= Elements(R);
[ZmodnZObj(0,10), ZmodnZObj(1,10), ZmodnZObj(2,10),
ZmodnZObj(3,10), ZmodnZObj(4,10), ZmodnZObj(5,10),
ZmodnZObj(6,10), ZmodnZObj(7,10), ZmodnZObj(8,10),
ZmodnZObj(9,10)]
gap> h:= x -> e[6]*x;
function(x) ...  end
gap> f:= MappingByFunction(R,R,h);
GeneralMappingByFunction( (Integers mod 10), (Integers mod
10), function(x)...end)
gap> IsRingHomomorphism(f);
true
```

The above output tells us the map f from $\mathbf{Z}_{10}$ to $\mathbf{Z}_{10}$ given by $f(x) = 5x$ is a ring homomorphism. The above command `h:= x -> e[6]*x;` creates a function that takes an element and multiplies it by 5 mod 10. (5 mod 10 is the sixth element in the list of elements of R.) The above command `f:= MappingByFunction(R,R,h);` creates this map. In general, the command is `MappingByFunction( <domain>, <range>, <function>)`. Now we can use `<ctl>-p` to redefine h and f and test for other homomorphisms:

```
gap> h:= x -> e[1]*x;
function(x) ...  end
gap> f:= MappingByFunction(R,R,h);
GeneralMappingByFunction( (Integers mod 10), (Integers mod
10), function(x)...end)
gap> IsRingHomomorphism(f);
true
gap> h:= x -> e[2]*x;
function(x) ...  end
gap> f:= MappingByFunction(R,R,h);
```

```
GeneralMappingByFunction( (Integers mod 10), (Integers mod
10), function(x)...end)
gap> IsRingHomomorphism(f);
true
gap> h:= x -> e[3]*x;
function(x) ...  end
gap> f:= MappingByFunction(R,R,h);
GeneralMappingByFunction( (Integers mod 10), (Integers mod
10), function(x)...end)
gap> IsRingHomomorphism(f);
false
```

We could continue testing all 10 possible cases for homomorphisms. Since the above steps are repetitive, it may be better to write a short program that will make GAP test all the cases. The subroutine "ringHoms" contains a function that takes as input a positive integer n. Fetch this subroutine from the manual website. The output is a list of m such that $f : \mathbf{Z}_n \to \mathbf{Z}_n$ given by $f(x) = mx \bmod n$ is a ring homomorphism. (Thanks to Russell Blyth for providing this function.) See the end of this chapter for a print out of this function.

```
gap> Read("ringHoms");
gap> ringHoms(10);
The map f: Z_10 -> Z_10 given by f(x)=mx is a homomorphism for m=[ 0, 1, 5, 6 ]
```

Thus $f : \mathbf{Z}_n \to \mathbf{Z}_n$ given by $f(x) = mx \bmod n$ is a ring homomorphism if and only if $m = 0, 1, 5$ or 6.

Exercises

15.1 Using GAP determine for which $m \leq 15$ the map $f : \mathbf{Z}_{15} \to \mathbf{Z}_{15}$ given by $f(x) = mx \bmod 15$ is a ring homomorphism.

15.2 Repeat Exercise 15.1 for the rings $\mathbf{Z}_{25}, \mathbf{Z}_{20}, \mathbf{Z}_{30}$ and $\mathbf{Z}_{40}$.

15.3 Make a conjecture that describes for which m the map $f : \mathbf{Z}_n \to \mathbf{Z}_n$ given by $f(x) = mx \bmod n$ is a ring homomorphism.

15.4 Prove your conjecture in Exercise 15.3.

15.5 Make a conjecture that describes for which k the map $f : \mathbf{Z}_m \to \mathbf{Z}_n$ given by $f(x) = kx \bmod n$ is a ring homomorphism. (The GAP commands that you used in Exercises 15.1 and 15.2 will not work here. Instead of using GAP, think about what conditions on k will be necessary to make f a ring homomorphism.)

15.6 Using GAP determine for which $m \leq 15$ the map $f : \mathbf{Z}_{15} \to \mathbf{Z}_{15}$ given by $f(x) = x^m \bmod 15$ is a ring homomorphism. You may want to write a program in GAP by modifying the program "ringHoms".

15.7 Using your program from Exercise 15.6, repeat Exercise 15.6 for the rings $\mathbf{Z}_{25}, \mathbf{Z}_{20}, \mathbf{Z}_{30}$ and $\mathbf{Z}_{40}$.

15.8 Make a conjecture that describes for which m the map $f : \mathbf{Z}_n \to \mathbf{Z}_n$ given by $f(x) = x^m \bmod n$ is a ring homomorphism.

Appendix for Chapter 15

The following is the file "ringHoms" which is used in this chapter. (Thanks to Russell Blyth for providing this function.)

```
ringHoms := function(n)
local R,e,h,f,l,j;
R := Integers mod n;
e := Elements(R);
l := [];
Print("The map f: Z_", n, " -> Z_", n, " given by f(x)=mx is a homomorphism for m=");
  for j in [1..Size(e)] do
    h := x -> e[j]*x;
    f := MappingByFunction(R,R,h);
    if IsRingHomomorphism(f) then
      Append(l,[Int(e[j])]);
    fi;
  od;
  return l;
end;
```

16 Chapter: Polynomial Rings

GAP will allow you to set up polynomials rings. For example the following GAP commands create the polynomial ring P1= $\mathbf{Z}_7[x]$:

```
gap> R1:= Integers mod 7;
GF(7)
gap> P1:= PolynomialRing(R1);
GF(7)[x_1]
```

Suppose we want to factor the polynomial $x^2 - 2 \in \mathbf{Z}_7[x]$. Recall from Chapter 12 of this manual that the nonzero elements in $\mathbf{Z}_7$ are denoted in GAP by powers of Z(7) where Z(7) denotes a generator of the cyclic group of nonzero elements:

```
gap> Elements(R1);
[ 0*Z(7), Z(7)^0, Z(7), Z(7)^2, Z(7)^3, Z(7)^4, Z(7)^5]
```

To see what integer $Z(7)$ represents (mod 7) type:

```
gap> Int(Z(7));
3
```

The command:

```
gap> x:= X(R1, "x");
x
```

creates the indeterminate x over the ring R1. We can now set up a polynomial in the ring P1 and factor it:

```
gap> g:= x^2-2;
x^2+Z(7)^5
gap> Factors(g);
[ x+Z(7), x+Z(7)^4 ]
```

Note that even though we entered the polynomial as x^2-2, GAP echoed the polynomial $x^2+Z(7)^5$. But $Z(7)^5 = 3^5 = 5 = -2$. So $x^2 + Z(7)^5 = x^2 - 2$. Notice the above GAP output tells us the factors of $x^2 - 2$ over $\mathbf{Z}_7$ are $x + Z(7) = x + 3$ and $x + Z(7)^4 = x + 4$.

As a second example, the following are the GAP commands and output used to find the factors of $x^2 - 2$ over $\mathbf{Z}_{11}$.

```
gap> R2:= Integers mod 11;
GF(11)
gap> y:= X(R2, "y");
 y
gap> h:= y^2-2;
```

```
y^2+Z(11)^6
gap> Factors(h);
[ y^2+Z(11)^6 ]
```

Note that $Z(11) = 2 \bmod 7$ and $Z(11)^6$ = -2:

```
gap> Int(Z(11));
2
gap> Int(-Z(11)^6);
2
```

Thus GAP echoes $y^2 - 2$ with $y^2 + Z(11)^6$. As expected, $x^2 - 2$ factors into linear factors in Z_7 because 2 is a square in Z_7, but $x^2 - 2$ does not factor in Z_{11} since 2 is not a square in Z_{11}.

GAP will also factor polynomials over the rationals. For example the following factors $f(x) = x^2 - 1 \in \mathbf{Q}[x]$:

```
gap> R:= Rationals;
Rationals
gap> z:= X(R, "z");
z
gap> f:= z^2-1;
z^2-1
gap> Factors(f);
[ z-1, z+1 ]
```

If you do not need to know the factors of a polynomial but only whether or not it is irreducible, you can use the `IsIrreducible` command:

```
gap> IsIrreducible(x^2-2);
false
gap> IsIrreducible(y^2-2);
true
gap> IsIrreducible(z^2-1);
false
```

Exercises

16.1 Use GAP to factor $x^{p-1} - 1$ in $\mathbf{Z}_p[x]$ for $p = 3, 5, 7$ and 11.

16.2 Using Exercise 16.1, make a conjecture about the factors of $x^{p-1} - 1$ in $\mathbf{Z}_p[x]$ for any prime p.

16.3 Find three monic irreducible polynomials in $\mathbf{Z}_3[x]$ of degree three and three of degree four.

16.4 For the polynomials found in Exercise 16.3 use GAP to determine if these polynomials are irreducible over the rational numbers. (Treat the coefficient 2 mod 3 in a polynomial over $\mathbf{Z}_3$, for example, as the coefficient 2 in a polynomial over the rational numbers.)

16.5 Repeat Exercises 16.3 and 16.4 for the field $\mathbf{Z}_5$.

16.6 What do you think the irreducibility of a polynomial in $\mathbf{Z}_p$ for p a prime tells you about the same polynomial over the rational numbers?

17 Chapter: Factorization of Polynomials

In this chapter we will investigate the factorization of $x^n - 1$ into its irreducibles over the rational numbers. Recall the GAP commands for creating a polynomial and for factoring this polynomial from the previous chapter. For example:

```
gap> R:= Rationals;
Rationals
gap> x:= X(R,"x");
x
gap> Factors(x^2-1);
[ x-1, x+1 ]
gap> Factors(x^4-1);
[ x-1, x+1, x^2+1 ]
```

Exercises

17.1 Factor $x^n - 1$ into its irreducibles over the rational numbers for $n = 6, 8, 12, 20$ and 30. On the basis of these data make a conjecture about the coefficients of the irreducible factors of $x^n - 1$. Test your conjecture for $n = 40, 50$ and 105.

17.2 Notice that your conclusion in Exercise 16.6 of this manual is the Mod p Irreduciblity Test [Gallian, Theorem 17.3]: *Let p be a prime and suppose that $f(x) \in \mathbf{Z}[x]$ with $degf(x) \geq 1$. Let $\bar{f}(x)$ be the polynomial in $\mathbf{Z}_p[x]$ obtained from $f(x)$ by reducing all the coefficients of $f(x)$ modulo p. If $\bar{f}(x)$ is irreducible over $\mathbf{Z}_p$ and $deg\bar{f}(x) = degf(x)$, then $f(x)$ is irreducible over $\mathbf{Q}$.* Use this theorem and the `IsIrreducible` command to determine if the following polynomials are irreducible over $\mathbf{Q}$:
a) $x^5 + 9x^4 + 12x^2$
b) $x^4 + x + 1$
c) $x^4 + 3x^2 + 3$
d) $x^5 + 5x^2 + 1$
e) $21x^3 - 3x^2 + 2x + 9$ [Gallian, Chapter 17, Computer Exercise 1]

18 Chapter: Divisibility in Integral Domains

Recall the ring of Gaussian integers, $\mathbf{Z}[i] = \{a + bi \mid a, b \in \mathbf{Z}\}$. The ring $\mathbf{Z}[i]$ is an Euclidean domain. In this chapter we will investigate the irreducible elements of $\mathbf{Z}[i]$. The command for creating $\mathbf{Z}[i]$ is `GaussianIntegers`:

```
gap> R:=GaussianIntegers;
GaussianIntegers
```

The $\sqrt{-1}$ is denoted in GAP by $E(4)$ (since $\sqrt{-1}$ is a primitive fourth root of one).

```
gap> i:=E(4);
E(4)
gap> i^2;
-1
```

We can now factor elements in $\mathbf{Z}[i]$ using the `Factors` command.

```
gap> Factors(R,4);
[ -1-E(4), 1+E(4), 1+E(4), 1+E(4) ]
gap> Factors(R,3+i);
[ 1-E(4), 1+2*E(4) ]
```

Thus we see the irreducible factors of 4 in $\mathbf{Z}[i]$ are $-1 - i$, $1 + i$, $1 + i$ and $1 + i$ and the irreducible factors of $3 + i$ are $1 - i$ and $1 + 2i$.

Careful: If you do not specify the ring, GAP will assume you want the factorization over the integers:

```
gap> Factors(4);
[ 2, 2 ]
```

Exercises

18.1 Make a list of the prime numbers in $\mathbf{Z}$ that are less than 60. For these primes determine whether or not they are irreducible elements in $\mathbf{Z}[i]$.

18.2 For all the primes $p \in \mathbf{Z}$ less than 60 compute p mod 4.

18.3 Make a conjecture stating which $p \in \mathbf{Z}$ are irreducible elements in $\mathbf{Z}[i]$.

18.4 For the primes $p \in \mathbf{Z}$, $p \leq 60$, that are **not** irreducible in $\mathbf{Z}[i]$ find positive integers $a, b \in \mathbf{Z}$ such that $a^2 + b^2 = p$. Is $a + bi$ irreducible in $\mathbf{Z}[i]$? Is $a - bi$ irreducible in $\mathbf{Z}[i]$?

A proposition that is often proved in more advanced algebra courses states that every irreducible element in $\mathbf{Z}[i]$ is one of the following:
i) the elements you found in Exercise 18.3 (assuming you did the problem correctly)
ii) the elements you found in Exercise 18.4 (assuming you did the problem correctly).

19 Chapter: Vector Spaces

Recall from Chapter 13 that $\mathbf{Z}_p$ is a field for every prime p. Let $GF(p)^n = \{(a_1, a_2, \ldots, a_n) \mid a_i \in \mathbf{Z}_p\}$. Then $GF(p)^n$ is an n dimensional vector space over $\mathbf{Z}_p$. In this chapter we will investigate subspaces of $GF(p)^n$. For example consider the vector space $GF(3)^2$:

```
gap> V:=GF(3)^2;
( GF(3)^2 )
gap> Elements(V);
[ [ 0*Z(3), 0*Z(3) ], [ 0*Z(3), Z(3)^0 ], [ 0*Z(3), Z(3) ],
[ Z(3)^0, 0*Z(3) ], [ Z(3)^0, Z(3)^0 ], [ Z(3)^0, Z(3) ], [ Z(3), 0*Z(3) ],
[ Z(3), Z(3)^0 ], [ Z(3), Z(3) ] ]
```

(Recall a generator of the multiplicative group of units in $\mathbf{Z}_p$ is denoted in GAP by $Z(p)$.)

Careful: Note that $GF(p)^n$ is the direct product of n copies of $\mathbf{Z}_p$ not the field of order p^n.

The vector space V is small enough that we can easily find all the 1-dimensional subspaces by hand. The `Display` command is useful here:

```
gap> Display(Elements(V));
. .
. 1
. 2
1 .
1 1
1 2
2 .
2 1
2 2
```

The zero in denoted by a dot and `Z(3)` is denoted by 2. The following GAP work shows the 1-dimensional subspaces:

```
gap> D:=Subspaces(V,1);
Subspaces( ( GF(3)^2 ), 1 )
gap> e:= Elements(D);;
gap> Display(Elements(e));
[ VectorSpace( GF(3), [ [ 0*Z(3), Z(3)^0 ] ] ),
  VectorSpace( GF(3), [ [ Z(3)^0, 0*Z(3) ] ] ),
  VectorSpace( GF(3), [ [ Z(3)^0, Z(3)^0 ] ] ),
  VectorSpace( GF(3), [ [ Z(3)^0, Z(3) ] ] ) ]
gap> Display(Elements(e[1]));
 . .
 . 1
 . 2
```

```
gap> Display(Elements(e[2]));
 . .
 1 .
 2 .
gap> Display(Elements(e[3]));
 . .
 1 1
 2 2
gap> Display(Elements(e[4]));
 . .
 1 2
 2 1
```

Thus, as expected, we see that the 1-dimensional subspaces are $\{(0,0),(0,1),(0,2)\}$, $\{(0,0),(1,0),(2,0)\}$, $\{(0,0),(1,1),(2,2)\}$, and $\{(0,0),(1,2),(2,1)\}$. If you just need to find the number of 1-dimensional subspaces you can type:

```
gap> Size(Subspaces(V,1));
4
```

Exercises

19.1 **By hand** find all the 1-dimensional subspaces of $GF(3)^3$.

19.2 Use GAP to check your answer to Exercise 19.1.

19.3 Use GAP to find the number of 1-dimensional subspaces of $GF(p)^3$ for $p = 2, 5, 7$ and 11.

19.4 Make a conjecture about the number of 1-dimensional subspaces of $GF(p)^3$. Prove your conjecture.

19.5 **By hand** find all the 2-dimensional subspaces of $GF(3)^3$.

19.6 Use GAP to check your answer to Exercise 19.5.

19.7 Use GAP to find the number of 2-dimensional subspaces of $GF(p)^3$ for $p = 5, 7$ and 11.

19.8 Make a conjecture about the number of 2-dimensional subspaces of $GF(p)^3$.

20 Chapter: Extension Fields

Consider the polynomial $g(x) = x^{3^2} - x \in \mathbf{Z}_3$. If we factor this polynomial in $\mathbf{Z}_3[x]$ we get $x(x+1)(x-1)(x^2+1)(x^2+x-1)(x^2-x-1)$:

```
gap> x:= X(GF(3),"x");
x
gap> Factors(x^9-x);
[ x, x+Z(3)^0, x-Z(3)^0, x^2+Z(3)^0, x^2+x-Z(3)^0, x^2-x-Z(3)^0 ]
```

The above shows that $x^9 - x$ does not factor over $GF(3)$ into linear factors. Alternatively you can have GAP list just the degrees of the factors of $x^9 - x$:

```
gap> factors:= Factors(x^9-x);;
gap> List(factors, DegreeOfLaurentPolynomial);
[ 1, 1, 1, 2, 2, 2 ]
```

This shows $x^9 - x$ factors over $GF(3)$ into 3 irreducible polynomials of degree one and 3 irreducible polynomials of degree 2. Let α denote a zero of an irreducible factor of $g(x)$ of degree 2. If we adjoin α to $\mathbf{Z}_3$ we get a field with 9 elements. You will see later that there is only one field (up to isomorphism) of order p^n for each prime p and each positive integer n. The field of order p^n for p a prime is denoted in GAP by `GF(p^n)`. The element $Z(p^n)$ in GAP denotes a generator of the cyclic group of nonzero elements in $GF(p^n)$. To see if $g(x)$ splits over this larger field use the command `Factors(P,g)` where `P` is the polynomial ring over this larger field and `g` is the polynomial. Notice that $(x^{3^2} - 1)$ splits into linear factors over $GF(9)$:

```
gap> polyring:= PolynomialRing(GF(9));
GF(3^2)[x]
gap> factors:= Factors(polyring, x^9-x);
[ x, x+Z(3)^0, x-Z(3)^0, x+Z(3^2), x+Z(3^2)^2, x+Z(3^2)^3, x+Z(3^2)^5,
  x+Z(3^2)^6, x+Z(3^2)^7 ]
gap> List(factors, DegreeOfLaurentPolynomial);
[ 1, 1, 1, 1, 1, 1, 1, 1, 1 ]
```

The above output shows that $(x^{3^2} - 1) = x(x+1)(x+2)(x+b)(x+b^2)(x+b^3)(x+b^5)(x+b^6)(x+b^7)$ where b is a generator of the cyclic group of nonzero elements in $GF(9)$.

Exercises

20.1 Factor the polynomial $f(x) = x^{p^n} - x \in \mathbf{Z}_p[x]$. For $p = 5$ and $n = 3$.

20.2 If you adjoin a zero of a nonlinear irreducible factor of the polynomial in Exercise 20.1 to $\mathbf{Z}_p$, what field do you get? Does $f(x)$ split in this extension field? If it does not split continue adjoining zeros until you get the splitting field.

20.3 Repeat Exercises 20.1 and 20.2 for $p = 7$ and $n = 2$.

20.4 Repeat Exercises 20.1 and 20.2 for $p = 7$ and $n = 4$.

21 Chapter: Algebraic Extensions

In this chapter we discuss how to create algebraic extensions in GAP. The polynomials $x^5 - 7$ is irreducible over $\mathbf{Q}$:

```
gap> x:= X(Rationals, "x");
x
gap> f:= x^5-7;
x^5-7
gap> Factors(x^5-7);
[ x^5-7 ]
```

Thus if we adjoin a zero of this polynomial to $\mathbf{Q}$ we get a field of degree five over $\mathbf{Q}$.

```
gap> F:=AlgebraicExtension(Rationals,f);
<algebraic extension over the Rationals of degree 5>
gap> a:= RootOfDefiningPolynomial(F);
a
```

The first command above defines a field F that is obtained by adjoining a zero of $x^5 - 7$ to $\mathbf{Q}$. The second command assigns the name a to a zero of f. (Thus $F = \mathbf{Q}(a)$.) Every element in $\mathbf{Q}(a)$ can be written in the form $q_0 + q_1a + q_2a^2 + q_3a^3 + q_4a^4$ for $q_i \in \mathbf{Q}$. We can now find the minimal polynomial of linear combinations of a over $\mathbf{Q}$. For example, the following finds the minimal polynomial of $4(7^{1/5}) + 10$ over $\mathbf{Q}$.

```
gap> MinimalPolynomial(Rationals, 4*a+10);
x^5-50*x^4+1000*x^3-10000*x^2+50000*x-107168
```

Exercises

21.1 Use GAP to find the minimal polynomial of $\sqrt[3]{2}+ \sqrt[3]{4}$ over $\mathbf{Q}$. [Gallian, Chapter 21, Exercise 16]

21.2 Use GAP to find the minimal polynomial of $5 + 4(\sqrt[3]{2}) + 10(\sqrt[3]{4})$ over $\mathbf{Q}$.

21.3 By hand find the minimal polynomial of $1 + i$ over $\mathbf{Q}$. Check your work using GAP.

We can also set up a finite field of order p^n by adjoining a root of an irreducible polynomial of degree n over $GF(p)$ to $GF(p)$. For example, the following creates the field of order 27 by adjoining a root of an irreducible cubic polynomial over $GF(3)$ to $GF(3)$:

```
gap> r:= GF(3);;
gap> x:= X(GF(3),"x");;
gap> f:= x^3 + 2*x^2 + 1;
x^3-x^2+Z(3)^0
gap> IsIrreducible(f);
true
gap> F:= AlgebraicExtension(r, f);
<field of size 27>
```

We can then use GAP to convert from multiplicative to additive notation in this field. [See Gallian, Chapter 22, Table 22.1]

```
gap> a:= RootOfDefiningPolynomial(F);
a
gap> a^3;
a^2-Z(3)^0
gap> a^4;
a^2-a-Z(3)^0
gap> a^6+Z(3)^0;
-a^2-a+Z(3)^0
```

Careful: Recall GAP denotes the number 1 in this field by $Z(3)^0$ and the number 2 by $Z(3)$.

22 Chapter: Finite Fields

For every prime p and every positive integer n there is one and only one field (up to isomorphism) of order p^n. [Gallian, Theorem 22.1] This field is denoted in GAP by `GF(p^n)`. The set of nonzero elements in $GF(p^n)$ form a cyclic group under multiplication of order $p^n - 1$. [Gallian, Theorem 22.2] GAP denotes a generator of this cyclic group by `Z(p^n)` and the remaining elements of $GF(p^n)$ are expressed in terms of $Z(p^m)$ for m a divisor of n. For example, the following defines F to be the field $GF(16)$ and then lists the elements in F:

```
gap> F:=GF(2^4);
GF(2^4)
gap> Elements(F);
[ 0*Z(2), Z(2)^0, Z(2^2), Z(2^2)^2, Z(2^4), Z(2^4)^2, Z(2^4)^3, Z(2^4)^4,
Z(2^4)^6, Z(2^4)^7, Z(2^4)^8, Z(2^4)^9, Z(2^4)^11, Z(2^4)^12, Z(2^4)^13,
Z(2^4)^14 ]
```

Careful: $Z(p^n)^m$ is not the same as $Z(p^{mn})$. The element $Z(p^n)$ is an element in $GF(p^n)$ of multiplicative order $p^n - 1$ and $Z(p^n)^m$ is the mth power of this element. The element $Z(p^{mn})$ is an element in $GF(p^{mn})$ of multiplicative order $p^{mn} - 1$.

```
gap> Order(Z(2^6));
63
gap> Order(Z(2^2)^3);
1
```

To understand the GAP notation think of the multiplicative group of nonzero elements in $GF(16)$ as generated by a. That is, $GF(16) = \{0, 1, a, a^2, \ldots, a^{14}\}$. The field $GF(p^n)$ has one and only one subfield of order p^m for every integer m that divides n. [Gallian, Theorem 22.3] Thus $GF(16)$ has a unique subfield of order 2 and a unique subfield of order 4. The subfield of order 2 is $\{0, 1\}$ and the subfield of order 4 is $\{0, 1, a^5, a^{10}\}$. In the GAP notation $Z(2^4) = a$, $Z(2^2) = Z(2^4)^5 = a^5$, and $Z(2^2)^2 = a^{10}$. We can use GAP to test this as follows:

```
gap> Z(2^2) = Z(2^4)^5;
true
gap> Z(2^2)^2 = Z(2^4)^10;
true
```

The command `DegreeFFE(Z(p^n)^m)`, for p a prime and m and n positive integers, returns the degree of the smallest field containing $Z(p^n)^m$ over $GF(p)$. For example:

```
gap> DegreeFFE(Z(2^4));
4
gap> DegreeFFE(Z(2^4)^3);
4
gap> DegreeFFE(Z(2^4)^5);
2
```

The GAP commands, discussed in Chapter 21 of this manual, for defining fields by adjoining zeros of irreducible polynomials also work over finite fields. For example, we can create a field of order 16 in GAP [See Gallian, Chapter 22, Example 1]:

```
gap> x:= X(GF(2), "x");;
gap> f:= x^4+x+1;
x^4+x+Z(2)^0
gap> IsIrreducible(f);
true
gap> F:= AlgebraicExtension(GF(2),f);
<field of size 16>
```

Exercises

22.1 Using GAP, find the degree of the smallest field containing $Z(2^4)^m$ over $GF(2)$ for $m = 1, 2, 3, \ldots, 10$. For which values of m is this degree strictly less than 4?

22.2 Find the multiplicative orders of $Z(2^4)^m$ for $m = 1, 2, 3, \ldots, 10$.

22.3 Using GAP, find the degree of the smallest field containing $Z(3^3)^m$ over $GF(3)$ for $m = 1, 2, 3, \ldots, 15$. For which values of m is this degree strictly less than 3?

22.4 Find the multiplicative orders of $Z(3^3)^m$ for $m = 1, 2, 3, \ldots, 15$.

22.5 Under what condition will the degree of the smallest field containing $Z(p^n)^m$ over $GF(p)$ equal n? Under what condition will this degree be less than n?

22.6 Using GAP factor the polynomial $x^{3^n} - x$ over $GF(3)$ for $n = 2, 3$ and 4. For each n, what was the largest degree of an irreducible factor?

22.7 Using GAP factor the polynomial $x^{5^n} - x$ over $GF(5)$ for $n = 2$ and 3. For each n, what was the largest degree of an irreducible factor?

22.8 Make a conjecture concerning the largest degree of any irreducible factor of $x^{p^n} - x$ over $GF(p)$.

22.9 Prove your conjecture in Exercise 22.8. [Gallian, Chapter 22, Exercise 26]

22.10 a) Construct a field of order 32 using GAP by adjoining a zero of an appropriate irreducible polynomial over $GF(p)$ to $GF(p)$ for some prime p.
b) Construct a field of order 81 using GAP by adjoining a zero of an appropriate irreducible polynomial over $GF(p)$ to $GF(p)$ for some prime p.

23 Chapter: Geometric Constructions

Exercises

23.1 Use GAP to determine whether or not $8x^3+4x^2-4x-1$ is irreducible. Note that $8\cos^3(2\pi/7)+4\cos^2(2\pi/7)-4\cos(2\pi/7)-1=0$. Use these two facts to help you show that a regular seven-sided polygon is not constructible with a straightedge and compass. [Gallian, Chapter 23, Exercise 14]

23.2 Use GAP to determine whether or not $4x^2+2x-1$ is irreducible. Note that $4\cos^2(2\pi/5)+2\cos(2\pi/5)-1=0$. Use these two facts to help you show that a regular pentagon is constructible with a straightedge and compass. [Gallian, Chapter 23, Exercise 18]

23.3 Use GAP to verify that $8x^3-6x-1$ is irreducible over $\mathbf{Q}$. (This fact can be used in the proof that a 60 degree angle can not be trisected using only a straightedge and compass.) [Gallian, Chapter 23, Exercise 13]

24 Chapter: Sylow Theorems

Let G be a finite group and let p be a prime that divides the order of G. Let p^k be the largest power of p that divides the order of G. A subgroup of G of order p^k is called a *Sylow p-subgroup* of G.

To do the exercises in this chapter you will need to fetch the file "sylows" from the website. This file contains a function that returns a list of all the Sylow p-subgroups of a group for a given group and a given prime.

```
gap> Read("sylows");
gap> G:=SymmetricGroup(6);
Sym( [ 1 ..  6 ] )
gap> sylows(G,3);
The Sylow 3-subgroups of SymmetricGroup( [ 1 ..  6 ] ) are:
[ Group([ (1,2,3), (4,5,6) ]), Group([ (1,2,4), (3,5,6) ]),
Group([ (1,2,5), (3,4,6) ]), Group([ (1,2,6), (3,4,5) ]),
Group([ (1,3,4), (2,5,6) ]), Group([ (1,3,5), (2,4,6) ]),
Group([ (1,3,6), (2,4,5) ]), Group([ (1,4,5), (2,3,6) ]),
Group([ (1,4,6), (2,3,5) ]), Group([ (1,5,6), (2,3,4) ]) ]
```

From the above output we see that S_6 has ten Sylow 3-subgroups. The first Sylow 3-subgroup in the list is the subgroup of S_6 generated by $(1,2,3)$ and $(4,5,6)$. Observe that all ten Sylow 3-subgroups are generated by two disjoint 3-cycles. Thus the Sylow 3-subgroups are Abelian because disjoint cycles commute.

Exercises

24.1 **By hand** find all the Sylow p-subgroups of S_4 for every prime p that divides the order of S_4.

24.2 Use GAP to check your answer to Exercise 24.1.

24.3 Use GAP to find the **number** of Sylow p-subgroups in A_6 for each prime p that divides $|A_6|$. (Recall the command for the alternating group is `AlternatingGroup(n);`.)

24.4 Repeat Exercise 24.3 for the group S_7.

24.5 Repeat Exercise 24.3 for a cyclic group of order 60.

24.6 Make a conjecture about the number of Sylow p-subgroups of a group mod p.

25 Chapter: Finite Simple Groups

The computer exercises in this section are based on material written by Christine Stevens at Saint Louis University. In this chapter we will use GAP to help us prove that A_5 and A_6 are simple groups.

The command `ConjugacyClasses(G)` lists all the conjugacy classes of a group G. For example:

```
gap> a4:=AlternatingGroup(4);
Alt( [ 1 ..  4 ]
gap> ConjugacyClasses(a4);
[ ()^G, (1,2)(3,4)^G, (1,2,3)^G, (1,2,4)^G ]
```

For $a \in G$, the notation $a\hat{}G$ above means the set of all conjugates of a in G. Thus we see that A_4 has four conjugacy classes: the conjugates of the identity, the conjugates of $(1,2)(3,4)$, the conjugates of $(1,2,3)$ and the conjugates of $(1,2,4)$. The command `ConjugacyClass(G,a)` creates the conjugacy class of G containing a:

```
gap> c:= ConjugacyClass(a4,(1,2,3));
(1,2,3)^G
gap> Elements(c);
[ (2,4,3), (1,2,3), (1,3,4), (1,4,2) ]
```

Exercises

25.1 Suppose you have disjoint sets T, U, V, W, X, Y and Z with cardinalities 1, 40, 40, 45, 72, 72 and 90 respectively. Suppose H is a set that is formed by taking the union of T with one or more of the other sets. List all the possible cardinalities of H. Which of these answers divide 360?

25.2 Use GAP to find all the conjugacy classes of A_6 and their cardinalities.

25.3 Let G be a group and H a normal subgroup of G. Let $h \in H$. Show the conjugacy class of h is a subset of H.

25.4 Use Exercises 25.1 - 25.3 to prove that A_6 is simple.

25.5 Use similar techniques as above to show A_5 is simple.

25.6 Show A_4 is not simple.

26 Chapter: Generators and Relations

Groups defined using generators and relations can be easily created in GAP. If you want to create an n-generated group, start with a free group on n generators. Then create the group by "moding out by" the relations. For example, D_4 is a 2-generated group with relations $a^4 = b^2 = (ab)^2 = e$, where a and b are the generators. [Gallian, Chapter 26, Examples 2 and 3] The following creates the group D_4 in GAP:

```
gap> f:=FreeGroup(2);
<free group on the generators [ f1, f2 ]>
gap> d4:=f/[f.1^4, f.2^2, (f.1*f.2)^2];
<fp group on the generators [ f1, f2 ]>
gap> Elements(d4);
[ <identity ...>, f2, f1^3*f2, f1, f1^3, f1*f2, f1^2*f2, f1^2 ]
```

The first command above creates a free group with two generators. The element `f.1` denotes the first generator and `f.2` denotes the second generator in the free group `f`:

```
gap> f.1;
f1
gap> f.2;
f2
```

The line `gap> d4:=f/[f.1^4, f.2^2, (f.1*f.2)^2];` creates D_4 as the free group on two generators mod the relations `f.1^4` = `f.2^2` = `( f.1 * f.2)^2`= e.

We can now use the group theory GAP commands discussed in previous chapters on `d4`. For example:

```
gap> IsAbelian(d4);
false
gap> Size(d4);
8
gap> Center(d4);
Group([ f1^2 ])
```

In addition, the `Factorization` command (see Chapter 5 of this manual) can be used to express the image of a word in the free group as an element in the factor group:

```
gap> a:=d4.1;;
gap> b:=d4.2;;
gap> Factorization(d4, a*b*a^3*b*a^5);
x1^-1
```

The first two above commands assign the letters a and b to the image of the two generators of the free group. The `Factorization` output tells us that aba^3ba^5 reduces to $a^{-1}(= a^3)$ in D_4. (GAP will denote the ith generator of the factor group by `xi`.)

You can also use GAP to help you classify a group that is defined using generators and relations. For example consider the group G defined by $G =< a, b \mid a^3 = b^9 = e, a^{-1}ba = b^{-1} >$. [Gallian, Chapter 26, Example 6]. Note that the relation $a^{-1}ba = b^{-1}$ can be rewritten as $a^2bab = e$. The below GAP commands create this group G:

```
gap> f:=FreeGroup(2);
<free group on the generators [ f1, f2 ]>
gap> G:=f/[f.1^3, f.2^9, f.1^2*f.2*f.1*f.2];
<fp group on the generators [ f1, f2 ]>
gap> Size(G);
3
```

Since there is only one group (up to isomorphism) of order 3, G must be the cyclic group of order 3.

Exercises

26.1 Use GAP to show that $< a, b \mid a^5 = b^2 = e, ba = a^2b >$ is isomorphic to $\mathbf{Z}_2$. [Gallian, Chapter 26, Exercise 4]

26.2 Let $G = < a, b \mid a^2 = b^4 = e, ab = b^3a >$.
a) Without using GAP express $a^3b^2abab^3$ in the form a^ib^j. Check your work using GAP.
b) Without using GAP express b^3abab^3a in the form a^ib^j. Check your work using GAP. [Gallian, Chapter 26, Exercise 12]

26.3 Let $G = < a, b \mid a^8 = b^2 = e, baba^3 = e >$.
a) Use GAP to find $|G|$.
b) Find the order of ab.
c) Find the center of G. [Gallian, Chapter 26, Exercise 15]

26.4 Let $G = < a, b \mid a^6 = b^3 = e, b^{-1}ab = a^3 >$. Use GAP to help you determine to which familiar group G is isomorphic. [Gallian, Chapter 26, Exercise 21]

26.5 Let $G = < a, b, c, d \mid ab = c, bc = d, cd = a, da = b >$. Use GAP to help you determine to which familiar group G is isomorphic.

26.6 Let $X_n = < a, b \mid a^n = b^2 = e, ab = ba^2 >$.
a) Find the order of X_n when $n = 3, 6, 24$ and 300.
b) Make a conjecture about the isomorphism type of X_n when n is a multiple of 3.
c) Make a conjecture about the isomorphism type of X_n when n and 3 are relatively prime. (First find the order of X_n for many appropriate values of n to help you formulate the conjecture.)

26.7 Let $G = < a, b \mid a^3 = b^3 = (ab)^2 = e >$. Use GAP to help you determine to which familiar group G is isomorphic.

Recall the command `IsomorphismGroups(G, H);` computes an isomorphism between the groups G and H. (If they are not isomorphic the command returns `fail`.) This command can also be used on groups defined using generators and relations.

```
gap> f:= FreeGroup(1);
<free group on the generators [ f1 ]>
gap> G:= f/[f.1^6];
<fp group on the generators [ f1 ]>
gap> H:= Subgroup(SymmetricGroup(6), [(1,2,3,4,5,6)]);
Group([ (1,2,3,4,5,6) ])
gap> K:= SymmetricGroup(3);
Sym( [ 1 .. 3 ] )
gap> IsomorphismGroups(H,K);
fail
gap> IsomorphismGroups(H,G);
[ (1,4)(2,5)(3,6), (1,3,5)(2,4,6) ] -> [ f1^-3, f1^2 ]
```

27 Chapter: Symmetry Groups

The first figure on the page of figures at the end of this chapter is a 3-prism. The front and back faces are equilateral triangles. We will use GAP to help us show the group of rotations in $\mathbf{R}^3$ of a 3-prism is isomorphic to D_3. [Gallian, Chapter 27, Exercise 4] Let G denote this group of rotations. Label the vertices of the facing triangle 1,2 and 3. Label the vertices of the other triangle in the prism 4,5 and 6. (See figure of the labeled 3-prism on last page of this chapter.) The group of rotations must be a subgroup of the group of permutations of the set $\{1, 2, 3, 4, 5, 6\}$. There are two types of rotations. We can rotate each triangle the same amount. Thus $(1, 2, 3)(4, 5, 6)$ is in G. We can also rotate the front facing triangle to the back. Thus the rotation $(1, 4)(2, 6)(3, 5)$ is in G.

```
gap> G:=Subgroup(SymmetricGroup(6), [(1,2,3)(4,5,6), (1,4)(2,6)(3,5)]);
Group([ (1,2,3)(4,5,6), (1,4)(2,6)(3,5) ])
gap> Elements(G);
[ (), (1,2,3)(4,5,6), (1,3,2)(4,6,5), (1,4)(2,6)(3,5), (1,5)(2,4)(3,6),
  (1,6)(2,5)(3,4) ]
```

The above exhibits G as a subgroup of S_6. We can now have GAP set up an isomorphism between D_3 and G. (We use here that $D_3 \cong S_3$.)

```
gap> d3:= SymmetricGroup(3);
Sym( [ 1 .. 3 ] )
gap> IsomorphismGroups(d3,G);
[ (1,2,3), (1,2) ] -> [ (1,2,3)(4,5,6), (1,5)(2,4)(3,6) ]
```

That is, the homomorphism that maps the generators $(1, 2, 3)$ and $(1, 2)$ of D_3 to the generators $(1, 2, 3)(4, 5, 6)$ and $(1, 5)(2, 4)(3, 6)$ respectively of G is an isomorphism. If a pair of groups are not isomorphic then this command returns `fail`:

```
gap> IsomorphismGroups(SymmetricGroup(3), AlternatingGroup(4));
fail
```

Exercises
For the exercises in this chapter see figures on the last page of this chapter.

27.1 Exhibit the group of rotations in $\mathbf{R}^3$ of a 4-prism as a subgroup of S_8. This group is isomorphic to which familiar group?

27.2 Exhibit the group of rotations in $\mathbf{R}^3$ of a 5-prism as a subgroup of S_{10}. This group is isomorphic to which familiar group?

27.3 Exhibit the group of rotations in $\mathbf{R}^3$ of a 6-prism as a subgroup of S_{12}. This group is isomorphic to which familiar group?

27.4 Make a conjecture about what the group of rotations in $\mathbf{R}^3$ of a n-prism is.

27.5 Prove your conjecture in Exercise 27.4.

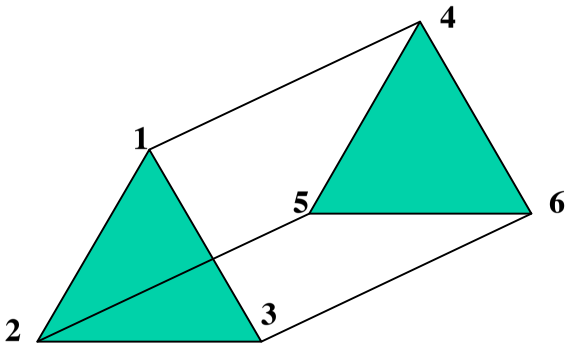
27.6 The order of the symmetry group (including both rotations and reflections) in $\mathbf{R}^3$ of a 3-prism is 12. Exhibit this symmetry group as a subgroup of S_6. This group is isomorphic to which familiar group of order 12?

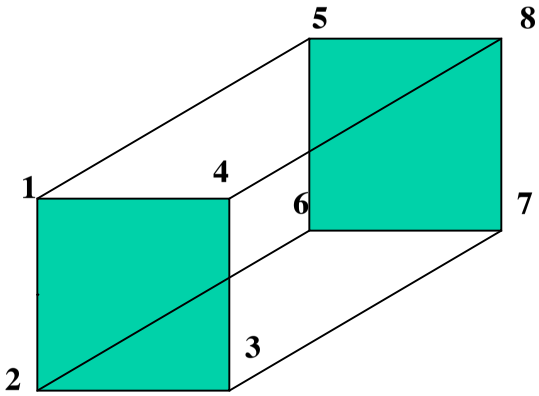
27.7 Exhibit the symmetry group in $\mathbf{R}^3$ of a 5-prism as a subgroup of S_{10}. This group is isomorphic to which familiar group?

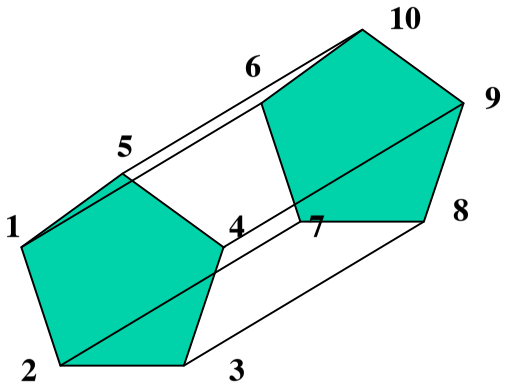
27.8 Make a conjecture about what the symmetry group in $\mathbf{R}^3$ of an n-prism is.

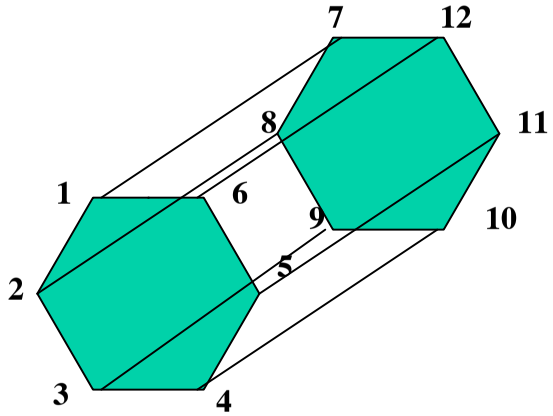
27.9 Test your conjecture in Exercise 27.8 for $n = 4$.

Chapter 27 Figures









28 Chapter: Frieze Groups and Crystallographic Groups

In order to do the exercises in this chapter you will first need to read the section on frieze groups in Chapter 28 of Gallian.

Exercises

28.1 In the frieze group F_6 let x denote a translation generator and y denote a horizontal reflection generator. [See Gallian Figure 28.9] Find a presentation of F_6 in terms of these generators. Enter the group F_6 into GAP using this presentation.

28.2 In the frieze group F_7 let x denote a translation generator, y denote a horizontal reflection generator and z denote a vertical reflection generator. [See Gallian Figure 28.9] Find a presentation of F_7 in terms of these generators. Enter the group F_7 into GAP using this presentation.

28.3 In the frieze group F_7 write x^2yzxz in the form $x^n y^m z^k$ by hand. Use GAP to check your work. [Gallian, Chapter 28, Exercise 3]

28.4 In the frieze group F_7 write $x^{-3}zxyz$ in the form $x^n y^m z^k$ by hand. Use GAP to check your work. [Gallian, Chapter 28, Exercise 4]

28.5 Use GAP to show that in the frieze group F_7 we have that $yz = zy$ and $xy = yx$ but $xz \neq zx$. [Gallian, Chapter 28, Exercise 5]

28.6 Use GAP to show that in the frieze group F_7 we have that $zxz = x^{-1}$. [Gallian, Chapter 28, Exercise 6]

29 Chapter: Symmetry and Counting

A group G is said to *act on* a set S if there is a homomorphism from G to sym(S) where sym(S) is the group of all permutations on S. In this chapter we will consider the action of G on itself given by conjugation. That is, given $g \in G$, define $\phi_g : G \to G$ by $\phi_g(h) = ghg^{-1}$ for all $h \in G$. Then ϕ_g is a permutation of the elements in G (that is, $\phi_g \in \text{sym}(G)$). The map $\Phi : G \to \text{sym}(G)$ given by $\Phi(g) = \phi_g$ is a homomorphism. Thus this map Φ gives an action of G on itself.

For elements a, b in a group G the command `a^b` in GAP computes $b^{-1}ab$. For example:

```
gap> (2,3,4)^(1,4,2);
(1,3,2)
gap> (1,2,4)*(2,3,4)*(1,4,2);
(1,3,2)
```

Note: Remember GAP multiplies permutations from left to right.

Given an action, Φ, of a group G on a set S, define the *kernel of the action* to be the set $\{g \in G \mid \Phi(g) = e\}$ where e denotes the identity in sym(S). The question we will investigate is: What is the kernel of the action when G acts on itself by conjugation? First we will consider this question when G is a cyclic group of order 4.

```
gap> G:=Group((1,2,3,4));
Group([ (1,2,3,4) ])
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3,4)],[(1,2,3,4)^()]);
[ (1,2,3,4) ] -> [ (1,2,3,4) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3,4)],[(1,2,3,4)^(1,2,3,4)]);
[ (1,2,3,4) ] -> [ (1,2,3,4) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3,4)],[(1,2,3,4)^(1,3)(2,4)]);
[ (1,2,3,4) ] -> [ (1,2,3,4) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3,4)],[(1,2,3,4)^(1,4,3,2)]);
[ (1,2,3,4) ] -> [ (1,2,3,4) ]
```

Since conjugation by any element in G maps the generator $(1, 2, 3, 4)$ back to $(1, 2, 3, 4)$, we see the kernel of the action is all of G.

Exercises

29.1 Prove that the maps ϕ_g and Φ defined above are homomorphisms.

29.2 Prove G is Abelian if and only if the kernel of the action of G on itself by conjugation is G.

In general, $g \in G$ will be in the kernel if and only if ϕ_g maps each element of a set of generators of G to itself. The below output investigates the kernel of this conjugation action of S_3 on itself.

```
gap> G:=SymmetricGroup(3);
Sym( [ 1 .. 3 ] )
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3),(1,2)],[(1,2,3)^(),(1,2)^()]);
[ (1,2,3), (1,2) ] -> [ (1,2,3), (1,2) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3),(1,2)],[(1,2,3)^(2,3),
> (1,2)^(2,3)]);
[ (1,2,3), (1,2) ] -> [ (1,3,2), (1,3) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3),(1,2)],[(1,2,3)^(1,2),
> (1,2)^(1,2)]);
[ (1,2,3), (1,2) ] -> [ (1,3,2), (1,2) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3),(1,2)],[(1,2,3)^(1,3),
> (1,2)^(1,3)]);
[ (1,2,3), (1,2) ] -> [ (1,3,2), (2,3) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3),(1,2)],[(1,2,3)^(1,2,3),
> (1,2)^(1,2,3)]);
[ (1,2,3), (1,2) ] -> [ (1,2,3), (2,3) ]
gap> f:=GroupHomomorphismByImages(G,G,[(1,2,3),(1,2)],[(1,2,3)^(1,3,2),
> (1,2)^(1,3,2)]);
[ (1,2,3), (1,2) ] -> [ (1,2,3), (1,3) ]
```

Thus we see the kernel of this action contains only the identity of S_3.

Exercises

29.3 Assume G is a group that is generated by two elements. Write a subroutine in GAP that lists the elements in the kernel of the action of a group on itself by conjugation.

29.4 Use your subroutine in Exercise 29.3 to find the kernel of the conjugation action when G is S_6, D_{12}, D_{19} and A_4.

29.5 For any group G the kernel of the conjugation action is a familiar subgroup. Use your answers to Exercise 29.4 to help you conjecture what the kernel is in general.

29.6 Prove your conjecture in Exercise 29.5

30 Chapter: Cayley Digraphs of Groups

The chapter assumes familiarity with the notation and material in Chapter 30 of [Gallian].

To work the exercises in this chapter you will first need to know how to create lists in GAP. A list is a collection of elements. The elements are enclosed within square brackets and are separated by commas. For example, the following creates the list consisting of the numbers 1, 2, 5 and 8.

```
gap> listexample:=[1,2,5,8];
[ 1, 2, 5, 8 ]
```

You can append additional elements to the end of the list using the command `Add`. For example, to add the number 11 to our list type:

```
gap> Add(listexample,11);
gap> listexample;
[ 1, 2, 5, 8, 11 ]
```

We can refer to the ith element in a list by typing the name of the list followed by `[i]`. For example:

```
gap> listexample[4];
8
gap> listexample[3]*20;
100
```

We can now use GAP to test whether we have a Hamiltonian circuit for a particular group and set of generators. In addition, we can get GAP to list the elements in this circuit in the order that they are traversed in the circuit (given a particular starting element).

A Hamiltonian circuit of D_4 with generators $R_{90} = (1,2,3,4)$ and $H = (1,2)(3,4)$ is obtained by applying $2 * (3 * R_{90}, H)$. [Gallian, Chapter 30, Example 3] We begin with a list containing only the identity:

```
gap> d:=[()];
[ () ]
```

Append elements to the list using the rule $2 * (3 * R_{90}, H)$.

```
gap> Add(d,(1,2,3,4)*d[1]);
gap> d;
[ (), (1,2,3,4) ]
gap> Add(d,(1,2,3,4)*d[2]);
gap> d;
[ (), (1,2,3,4), (1,3)(2,4) ]
gap> Add(d,(1,2,3,4)*d[3]);
gap> d;
[ (), (1,2,3,4), (1,3)(2,4), (1,4,3,2) ]
gap> Add(d,(1,2)(3,4)*d[4]);
gap> d;
```

```
[ (), (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4) ]
gap> Add(d,(1,2,3,4)*d[5]);
gap> d;
[ (), (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4), (1,4)(2,3) ]
gap> Add(d,(1,2,3,4)*d[6]);
gap> d;
[ (), (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4), (1,4)(2,3), (1,3) ]
gap> Add(d,(1,2,3,4)*d[7]);
gap> d;
[ (), (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4), (1,4)(2,3), (1,3), (1,2)(3,4) ]
gap> Add(d,(1,2)(3,4)*d[8]);
gap> d;
[ (), (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4), (1,4)(2,3), (1,3), (1,2)(3,4),
  () ]
```

Thus starting with the identity of D_4 and using the rule $2 * (3 * R_{90}, H)$ (where $R_{90} = (1,2,3,4)$ and $H = (1,2)(3,4)$) we get the Hamiltonian circuit:

$$\{e, (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4), (1,4)(2,3), (1,3), (1,2)(3,4), e\}.$$

We could reduce the number of repetitive operations needed to produce this circuit by writing a short program in GAP. The following program performs the same operations as the above but also allows you to choose which element you want to be the first element in the circuit. The first line of the program says it will take as input a single element `n`. This is the element of D_4 which we want to be the first element in the circuit. The next line defines local variables that will be used in the program. The next line starts the Hamiltonian circuit list. The list is called `s` and the first element in the list is set equal to the element `n` (the element that was input into the function). The next 9 lines of the program are a for-do loop that is performed 8 times (the order of D_4). The `elif` command means "else if". The `fi;` command ends the if-then-else statement. Similarly the `od;` command ends the for-do loop. The next to last line tells GAP to output the created list `s`.

```
CircuitCheck:= function(n)
local s,i;
s:=[n];
for i in [1..8] do
  if i = 4 then
     Add(s,(1,2)(3,4)*s[i]);
  elif i = 8 then
     Add(s,(1,2)(3,4)*s[i]);
  else
    Add(s,(1,2,3,4)*s[i]);
  fi;
od;
return s;
end;
```

Now we can read this program into GAP.

```
gap> Read("CircuitCheck");
gap> CircuitCheck(());
[ (), (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4), (1,4)(2,3), (1,3), (1,2)(3,4),
  () ]
gap> CircuitCheck((1,2,3,4));
[ (1,2,3,4), (1,3)(2,4), (1,4,3,2), (), (1,2)(3,4), (2,4), (1,4)(2,3), (1,3),
  (1,2,3,4) ]
```

Thus we see the Hamiltonian circuit $2 * (3 * R_{90}, H)$ starting with the identity is

$$\{e, (1,2,3,4), (1,3)(2,4), (1,4,3,2), (2,4), (1,4)(2,3), (1,3), (1,2)(3,4), e\}$$

and the circuit starting with $(1,2,3,4)$ is

$$\{(1,2,3,4), (1,3)(2,4), (1,4,3,2), e, (1,2)(3,4), (2,4), (1,4)(2,3), (1,3), (1,2,3,4)\}$$

.

Exercises

30.1 Let $D_4 =< r, f \mid r^4 = e = f^2, rf = fr^{-1} >$. Write a short GAP program to verify that $6 * [3 * (r,0), (f,0), 3 * (r,0), (e,1)]$ is a Hamiltonian circuit in $\mathrm{Cay}(\{(r,0), (f,0), (e,1)\} : D_4 \oplus \mathbf{Z}_6)$. [Gallian, Chapter 30, Exercise 9]

30.2 Use your program in Exercise 30.1 to find Hamiltonian circuits for $\mathrm{Cay}(\{(r,0), (f,0), (e,1)\} : D_4 \oplus \mathbf{Z}_6)$ starting with the elements $(r,0)$, $(f,0)$ and $(e,1)$.

30.3 Use GAP to find a Hamiltonian circuit in $\mathrm{Cay}(\{(a,0), (b,0), (e,1)\} : Q_4 \oplus Z_2)$. [Gallian, Chapter 30, Exercise 2]

30.4 Use GAP to find a Hamiltonian circuit in $\mathrm{Cay}(\{(a,0), (b,0), (e,1)\} : Q_4 \oplus Z_4)$. Find a Hamiltonian circuit in $\mathrm{Cay}(\{(a,0), (b,0), (e,1)\} : Q_4 \oplus Z_m)$ when m is even. [Gallian, Chapter 30, Exercise 3]

30.5 Use GAP to find a Hamiltonian circuit in $\mathrm{Cay}(\{(a,0), (b,0), (e,1)\} : Q_4 \oplus Z_3)$. [Gallian, Chapter 30, Exercise 23]

30.6 Use GAP to find a Hamiltonian circuit in $\mathrm{Cay}(\{(a,0), (b,0), (e,1)\} : Q_4 \oplus Z_5)$. Find a Hamiltonian circuit in $\mathrm{Cay}(\{(a,0), (b,0), (e,1)\} : Q_4 \oplus Z_m)$ when m is odd. [Gallian, Chapter 30, Exercise 24]

31 Chapter: Introduction to Algebraic Coding Theory

The chapter assumes familiarity with the notation and material in Chapter 31 of [Gallian].

A vector in GAP is entered by listing the components within square brackets. For example the vector $v = (1, 3, 7)$ is entered as:

```
gap> v:= [1,3,7];
[ 1, 3, 7 ]
```

A matrix can be entered as a list of row vectors. For example, the matrix

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

is entered into GAP by typing

```
gap> M:= [ [1,2,3], [4,5,6], [7,8,9]];
[ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ]
```

If you prefer to exhibit the matrix M as a 3 by 3 array use the command `PrintArray`:

```
gap> PrintArray(M);
[ [  1,  2,  3 ],
  [  4,  5,  6 ],
  [  7,  8,  9 ] ]
```

The notation `M[i][j]` denotes the entry in the ith row and jth column of M. Similarly, the notation `M[i]` denotes the ith row of M.

```
gap> M[2][3];
6
gap> M[1];
[ 1, 2, 3 ]
```

To multiply v and M type:

```
gap> v*M;
[ 62, 73, 84 ]
gap> M*v;
[ 28, 61, 94 ]
```

That is, $vM = (62, 73, 84)$ and $Mv = (28, 61, 94)$. Notice that GAP automatically treats v as a row vector in the multiplication vM but as a column vector in the multiplication Mv.

Read Example 9 of [Gallian, Chapter 31]. The software GAP easily performs the computations needed to decode received codes. First set up the parity check matrix given in [Gallian, Chapter 31, Example 9]:

```
gap> Elements(Integers mod 2);
[ 0*Z(2), Z(2)^0 ]
gap> z:= 0*Z(2);;
gap> a:= Z(2)^0;;
gap> H:= [ [a,a,z],
> [a, z, a],
> [a, a, a],
> [z, a, a],
> [a, z, z],
> [z, a, z],
> [z, z, a]];;
gap> PrintArray(H);
[ [  Z(2)^0,  Z(2)^0,  0*Z(2) ],
  [  Z(2)^0,  0*Z(2),  Z(2)^0 ],
  [  Z(2)^0,  Z(2)^0,  Z(2)^0 ],
  [  0*Z(2),  Z(2)^0,  Z(2)^0 ],
  [  Z(2)^0,  0*Z(2),  0*Z(2) ],
  [  0*Z(2),  Z(2)^0,  0*Z(2) ],
  [  0*Z(2),  0*Z(2),  Z(2)^0 ] ]
```

Next enter in the received the word $v = 0000110$, compute vH and note vH is the first row of H:

```
gap> v:=[z,z,z,z,a,a,z];
[ 0*Z(2), 0*Z(2), 0*Z(2), 0*Z(2), Z(2)^0, Z(2)^0, 0*Z(2) ]
gap> PrintArray(v*H);
[  Z(2)^0,  Z(2)^0,  0*Z(2) ]
gap> v*H = H[1];
true
```

Similarly, if the received word is $w = 1011111$, we can use GAP to compute wH:

```
gap> w:=[a,z,a,a,a,a,a];
[ Z(2)^0, 0*Z(2), Z(2)^0, Z(2)^0, Z(2)^0, Z(2)^0, Z(2)^0 ]
gap> PrintArray(w*H);
[  Z(2)^0,  0*Z(2),  Z(2)^0 ]
gap> w*H = H[2];
true
```

Thus, since vH equals the first row of H we decode $v = (0, 0, 0, 0, 1, 1, 0)$ as $(1, 0, 0, 0, 1, 1, 0)$. Since wH equals the second row of H we decode $w = (1, 0, 1, 1, 1, 1, 1)$ as $(1, 1, 1, 1, 1, 1, 1)$.

Read Example 11 of [Gallian, Chapter 31]. We will now use GAP to do syndrome decoding. First enter in the parity check matrix in [Gallian, Chapter 31, Example 11]. Here `a` and `z` are defined as in the example above:

```
gap> H:= [[a,a,z],
> [a,z,a],
```

```
> [z,a,a],
> [a,z,z],
> [z,a,z],
> [z,z,a]];;
gap> PrintArray(H);
[ [  Z(2)^0,  Z(2)^0,  0*Z(2) ],
  [  Z(2)^0,  0*Z(2),  Z(2)^0 ],
  [  0*Z(2),  Z(2)^0,  Z(2)^0 ],
  [  Z(2)^0,  0*Z(2),  0*Z(2) ],
  [  0*Z(2),  Z(2)^0,  0*Z(2) ],
  [  0*Z(2),  0*Z(2),  Z(2)^0 ] ]
```

Now set up the matrix `CL` whose rows are the coset leaders:

```
gap> CL:= [[z,z,z,z,z,z],
> [a,z,z,z,z,z],
> [z,a,z,z,z,z],
> [z,z,a,z,z,z],
> [z,z,z,a,z,z],
> [z,z,z,z,a,z],
> [z,z,z,z,z,a],
> [a,z,z,z,z,a]];;
gap> PrintArray(CL);
[ [  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2) ],
  [  Z(2)^0,  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2) ],
  [  0*Z(2),  Z(2)^0,  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2) ],
  [  0*Z(2),  0*Z(2),  Z(2)^0,  0*Z(2),  0*Z(2),  0*Z(2) ],
  [  0*Z(2),  0*Z(2),  0*Z(2),  Z(2)^0,  0*Z(2),  0*Z(2) ],
  [  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2),  Z(2)^0,  0*Z(2) ],
  [  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2),  Z(2)^0 ],
  [  Z(2)^0,  0*Z(2),  0*Z(2),  0*Z(2),  0*Z(2),  Z(2)^0 ] ]
```

The matrix `CL*H` will be the matrix whose rows are the syndromes.

```
gap> Synd:=CL*H;;
gap> PrintArray(Synd);
[ [  0*Z(2),  0*Z(2),  0*Z(2) ],
  [  Z(2)^0,  Z(2)^0,  0*Z(2) ],
  [  Z(2)^0,  0*Z(2),  Z(2)^0 ],
  [  0*Z(2),  Z(2)^0,  Z(2)^0 ],
  [  Z(2)^0,  0*Z(2),  0*Z(2) ],
  [  0*Z(2),  Z(2)^0,  0*Z(2) ],
  [  0*Z(2),  0*Z(2),  Z(2)^0 ],
  [  Z(2)^0,  Z(2)^0,  Z(2)^0 ] ]
```

To decode the word $v = 101001$ compute vH. Since vH equals the 5th row of `Synd` we decode v as v minus the 5th row of `CL`:

```
gap> v:=[a,z,a,z,z,a];;
gap> PrintArray(v*H);
[  Z(2)^0,  0*Z(2),  0*Z(2) ]
gap> v*H = Synd[5];
true
gap> decodev:= v - CL[5];;
gap> PrintArray(decodev);
[ Z(2)^0, 0*Z(2), Z(2)^0, Z(2)^0, 0*Z(2), Z(2)^0 ]
```

That is, we decode the word 101001 as 101101. Similarly, the following output shows we should decode the word $w = 011001$ as 111000:

```
gap> w:=[z,a,a,z,z,a];;
gap> PrintArray(w*H);
[  Z(2)^0,  Z(2)^0,  Z(2)^0 ]
gap> w*H = Synd[8];
true
gap> decodew:= w - CL[8];;
gap> PrintArray(decodew);
[ Z(2)^0, Z(2)^0, Z(2)^0, 0*Z(2), 0*Z(2), 0*Z(2) ]
```

Exercises

31.1 Find the parity check matrix of the binary linear code whose generator matrix is

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}.$$

[Gallian, Chapter 31, Exercise 13]

31.2 For the (7,4) binary linear code in Exercise 31.1, use GAP and the parity-check matrix method to decode each of the following received words

$$0001111, 0101011, 0111101, 0101110$$

.

31.3 Find the parity check matrix of the binary linear code whose generator matrix is

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

31.4 For the (6,3) binary linear code in Exercise 31.3, use GAP and the parity-check matrix method to decode each of the following received words

$$001001, 011000, 000110, 100001.$$

[Gallian, Chapter 31, Exercise 17]

31.5 Redo Exercise 31.2 using GAP and the syndrome decoding method.

31.6 Redo Exercise 31.4 using GAP and the syndrome decoding method.

32 Chapter: An Introduction to Galois Theory

Recall the GAP commands, discussed in Chapter 21 of this manual, for creating algebraic extensions of fields. For example, if we want to construct the field $\mathbf{Q}(\sqrt{2},\sqrt{3}) = \mathbf{Q}(\sqrt{2}+\sqrt{3})$ we adjoin a root of the minimal polynomial of $\sqrt{2}+\sqrt{3}$ over $\mathbf{Q}$ to $\mathbf{Q}$. The polynomial $x^4 - 10x^2 + 1$ is the minimal polynomial of $\sqrt{2}+\sqrt{3}$ over $\mathbf{Q}$. The below commands create the field $F = \mathbf{Q}(\sqrt{2}+\sqrt{3})$.

```
gap> x:= X(Rationals,"x");
x
gap> F:= AlgebraicExtension(Rationals, x^4-10*x^2+1);
<algebraic extension over the Rationals of degree 4>
```

We can now give this adjoined root, $\sqrt{2}+\sqrt{3}$, a name:

```
gap> a:=RootOfDefiningPolynomial(F);
a
gap> a^4;
10*a^2-1
```

A similar construction can be done over finite fields. Recall the finite field of order p^n is denoted in GAP by `GF(p^n)`. Also recall (see Chapter 20 of this manual) the splitting field of $x^{p^n} - x$ over $GF(p)$ is $GF(p^n)$.

```
gap> x:= X(GF(3), "x");
x
gap> Factors(x^9-x);
[ x, x+Z(3)^0, x-Z(3)^0, x^2+Z(3)^0, x^2+x-Z(3)^0, x^2-x-Z(3)^0 ]
gap> F:=AlgebraicExtension(GF(3),Z(3)^0+x^2);
<field of size 9>
```

The field F was constructed by adjoining a root of an irreducible factor of $x^9 - x$ of degree two. Since $|F| = 9$, F must be $GF(9)$.

Let E be an extension field of the field F. The *Galois group* of E over F, GAL(E/F), is the set of all automorphisms of E that map every element of F to themselves. GAP has a command for setting up Galois groups. For example the following creates the Galois group Gal$(GF(81)/GF(3))$:

```
gap> g:=GaloisGroup(AsField(GF(3),GF(81)));
<group with 1 generators>
gap> Elements(g);
[ IdentityMapping( GF(3^4) ), FrobeniusAutomorphism( GF(3^4) )^2,
  FrobeniusAutomorphism( GF(3^4) ), FrobeniusAutomorphism( GF(3^4) )^3 ]
```

Notice the GAP command `GaloisGroup` requires that the subfield of the extension field be listed first.

From the above output we see that the Galois group Gal$(GF(81)/GF(3))$ is a cyclic group of order 4.

The commands for listing the subfields of a field is `Subfield`. For example, the below output shows $GF(81)$ contains three subfields:

```
gap> Subfields(GF(81));
[ GF(3), GF(3^2), GF(3^4) ]
```

Let E be the splitting field of $x^{p^n} - x$ over $GF(p^m)$ for some positive integer m that divides n. That is, $E = GF(p^n)$. By the Fundamental Theorem of Galois Theory, there is a correspondence between the set of subfields of $GF(p^n)$ containing $GF(p^m)$ and the subgroups of $\mathrm{Gal}(GF(p^n)/GF(p^m))$.

Exercises

32.1 Determine the isomorphism class of $\mathrm{Gal}(GF(p^n)/GF(p^m))$ for $p = 2$, $m = 1$ and $n = 3, 5, 9$.

32.3 Repeat Exercise 32.1 for $p = 3$, $m = 1$ and $n = 2, 6$.

32.3 Repeat Exercise 32.1 for $p = 3$, $m = 2$ and $n = 4, 8$ and 10 and for $p = 5$, $m = 2$ and $n = 4$ and 6.

32.4 Repeat Exercise 32.1 for $p = 3$, $m = 3$ and $n = 6, 9$.

32.5 Make a conjecture about the isomorphism class of $\mathrm{Gal}(GF(p^n)/GF(p^m))$. *Careful:* Is it always the case that $GF(p^m)$ is a subfield of $GF(p^n)$ for $m \leq n$?

GAP has commands for determining when a group G is solvable and, in the case when G is solvable, for producing a series

$$\{e\} = H_0 \subset H_1 \subset \cdots \subset H_k = G$$

such that H_i is normal in H_{i+1} and H_{i+1}/H_i is Abelian for $0 \leq i < k$.

```
gap> S:=SymmetricGroup(3);
Sym( [ 1 .. 3 ] )
gap> DerivedSeries(S);
[ Sym( [ 1 .. 3 ] ), Group([ (1,3,2) ]), Group(()) ]
```

The above output is a series of subgroups $H_0 = \{e\}$, $H_1 =< (1,3,2) >= \{e, (1,3,2), (1,2,3)\}$ and $S_3 = H_k =< (2,3), (1,3,2) >$. This series shows S_3 is solvable. For another example, the following finds a series for D_4 which shows it is solvable:

```
gap> d4:=Group((1,2,3,4),(1,4)(2,3));
Group([ (1,2,3,4), (1,4)(2,3) ])
gap> DerivedSeries(d4);
[ Group([ (1,2,3,4), (1,4)(2,3) ]), Group([ (1,3)(2,4) ]), Group(()) ]
```

If the command `DerivedSeries(G)` is used on a group that is not solvable the last element in the series will not be the identity subgroup.

```
gap> S5:=SymmetricGroup(5);
Sym( [ 1 .. 5 ] )
gap> DerivedSeries(S5);
[ Sym( [ 1 .. 5 ] ), Group([ (1,3,2), (1,4,3), (1,4,5) ]) ]
gap> IsSolvable(S5);
false
```

Exercises

32.6 By hand find a series of subgroups of D_n that shows D_n is solvable for $n = 5$, 10, 30.

32.7 Rework Exercise 32.6 using GAP. Is there only one possible such series for a given dihedral group?

32.8 Determine if A_n is solvable for $n = 4$, 5 and 8.

32.9 Determine if $D_4 \oplus D_8$ is solvable.

32.10 Determine if $S_3 \oplus S_3$ is solvable.

32.11 Prove or disprove: The direct product of solvable groups is solvable.

33 Chapter: Cyclotomic Extensions

Let n be an integer greater than 1 and let $\phi(n)$ denote the number of positive integers less than n and relatively prime to n. For any positive integer n there are $\phi(n)$ primitive nth roots of unity. Denote these primitive nth roots of unity by $\omega_i, i = 1, \ldots, \phi(n)$. The nth cyclotomic polynomial over $\mathbf{Q}$ is the polynomial $\Phi_n(x) = (x - \omega_1)(x - \omega_2) \cdots (x - \omega_{\phi(n)})$. The command in GAP for the nth cyclotomic polynomial is `CyclotomicPolynomial(Rationals,n)`. For example the following commands output $\Phi_{15}(x)$:

```
gap> x:= X(Rationals, "x");;
gap> CyclotomicPolynomial(Rationals,15);
x^8-x^7+x^5-x^4+x^3-x+1
```

The nth cyclotomic extension of $\mathbf{Q}$ is the smallest extension field of $\mathbf{Q}$ that contains a primitive nth root of unity. The nth cyclotomic extension of $\mathbf{Q}$ is denoted in GAP by `CF(n)`. The element $\cos(2\pi/n) + i\sin(2\pi/n)$ in `CF(n)` is denoted by `E(n)`.

```
gap> f:= CF(8);
CF(8)
gap> E(8)^8;
1
gap> E(8)^2;
E(4)
gap> E(8)^4;
-1
```

Unfortunately polynomials can only be factored in GAP over finite fields or over the rationals. So we will not be able to factor polynomials over `CF(n)`. We can list the subfields of `CF(n)`:

```
gap> Subfields(f);
[ Rationals, GaussianRationals, CF(8), NF(8,[ 1, 3 ]), NF(8,[ 1, 7 ]) ]
```

The first three subfields listed are $\mathbf{Q}$, $\mathbf{Q}(i)$, and $\mathbf{Q}(\omega)$ where ω is a primitive 8th root of unity. The notation `NF(8,[ 1, 3 ])` means the subfield $\mathbf{Q}(\omega + \omega^3)$. Similarly `NF(8,[ 1, 7 ])` means the subfield $\mathbf{Q}(\omega + \omega^7)$.

GAP will also find the Galois groups of cyclotomic fields:

```
gap> g:=GaloisGroup(AsField(Rationals,CF(8)));
<group of size 4 with 2 generators>
gap> Elements(g);
 [ IdentityMapping( CF(8) ), ANFAutomorphism( CF(8), 3 ),
    ANFAutomorphism( CF(8), 5 ), ANFAutomorphism( CF(8), 7 ) ]
```

The above output tells us that $\mathrm{Gal}(\mathbf{Q}(\omega)/\mathbf{Q})$ has the four elements: the identity map, the automorphism of $\mathbf{Q}(\omega)$ that maps `E(8)` to `E(8)`^{3}, the automorphism that maps `E(8)` to `E(8)`^{5} and the automorphism that maps `E(8)` to `E(8)`^{7}.

Since $\mathbf{Q}(\omega)$ has five subfields, the Fundamental Theorem of Galois Theory says that $\mathrm{Gal}(\mathbf{Q}(\omega)/\mathbf{Q})$ must have five subgroups. Notice each nonidentity element of $\mathrm{Gal}(\mathbf{Q}(\omega)/\mathbf{Q})$ has order 2:

```
gap> e:=Elements(g);;
gap> Order(e[1]);
1
gap> Order(e[2]);
2
gap> Order(e[3]);
2
gap> Order(e[4]);
2
```

Thus the five subgroups of $\mathrm{Gal}(\mathbf{Q}(\omega)/\mathbf{Q})$ are the identity subgroup, the whole group, and three subgroups of order 2.

Exercises

33.1 a) Factor $x^{12} - 1$ as a product of irreducibles over $\mathbf{Z}$.
b) Factor $x^8 - 1$ as a product of irreducibles over $\mathbf{Z}_2, \mathbf{Z}_3$ and $\mathbf{Z}_5$. [Gallian, Chapter 33, Exercises 2 and 3]

33.2 Use GAP to show the Galois groups of $x^9 - 1$ and $x^7 - 1$ over $\mathbf{Q}$ are isomorphic. [Gallian, Chapter 33, Exercise 16]

33.3 Use GAP to show the Galois groups of $x^{10} - 1$ and $x^8 - 1$ over $\mathbf{Q}$ are not isomorphic. [Gallian, Chapter 33, Exercise 18]

33.4 Let G be the group $\mathrm{Gal}(\mathbf{Q}(\omega)/\mathbf{Q})$ where ω is a primitive 15th root of unity. Find the orders of all the elements in G.

33.5 Use GAP to determine whether or not the Galois groups of $x^{64} - 1$ and $x^{80} - 1$ over $\mathbf{Q}$ are isomorphic.

33.6 Find all the subfields of the 60th cyclotomic extension of $\mathbf{Q}$.

33.7 Find all the subgroups of the Galois group of $x^{60} - 1$ over $\mathbf{Q}$. List the correspondence (from the Fundamental Theorem of Galois Theory) with the fields obtained in Exercise 33.6.