

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

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

Exercises

Before starting these exercises type, at the gap prompt, `LogTo("assignment0");`. This will save your work in a file named “assignment0”. When you are done with the exercises type `LogTo();` and print “assignment0”. 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 $\text{gcd}(8701, 10057) = 8701s + 10057t$.

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 \rightarrow . 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. Be sure to save your work in a file so that you can print it.

0.4 Define a function `SumFirstnInt` which takes as input a positive integer n and outputs the sum $1 + 2 + 3 + \dots + n$. 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 $n(n + 1)/2$.)

There may be times that you would like to write a function in GAP and then be able to use that same function later on. To do this save your function using the `InputLogTo` command. This command is like the `LogTo` command but instead of saving both the input and output it just saves the input. For example, consider the situation where we want to create a function, call `newfunction`, that takes an integer t as input, cubes this value, adds 3 and then subtracts $2t$. (That is our function is $f(t) = t^3 + 3 - 2t$.) First we type the following GAP commands:

```
gap> InputLogTo("newfunction");  
gap> newfunction:= function(t)  
> local x;  
> x:= t^3 + 3 - 2*t;  
> return x;  
> end;  
function( t ) ... end  
gap> InputLogTo();
```

Now we can try out our new function:

```
gap> newfunction(3);
```

24

```
gap> newfunction(45);  
91038
```

Quit out of GAP and look in the folder where you have GAP installed. (This folder is probably called gap4r4.) You should see a file called “newfunction”. Open this file. Notice it contains the following:

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

Delete the line `InputLogTo();` in this file and then save the file. Now whenever we run GAP we can use this function by first reading it in:

```
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 43].) 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]