

Abstract Algebra with GAP

Section 2 Appendix: Creating Functions for Later Use

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, divides this total by 11 and then takes this value mod 5. (That is our function is $f(t) = (t^3 + 3)/11 \bmod 5$.) First we type the following GAP commands:

```
gap> InputLogTo("newfunction");
gap> newfunction:= function(t)
> local x;
> x:= (t^3+3)/11 mod 5;
> return x;
> end;
function( t ) ... end
gap> InputLogTo();
```

Now we can try out our new function:

```
gap> newfunction(3);
0
gap> newfunction(45);
3
```

Quit out of GAP and look in the folder where you have GAP installed. (This folder is probably call `gap4r3`.) You should see a file called “`newfunction`”. Open this file. Notice it contains the following:

```
newfunction:= function(t)
local x;
x:= (t^3+3)/11 mod 5;
return x;
end;
```

Now whenever we run GAP we can use this function by first reading it in:

```
gap> Read("newfunction");
gap> newfunction(6);
4
```

We can also easily edit this function by just opening and editing the file “newfunction”. If for example say we want another function, called `newfunction2`, to be $f(t) = (t^3+3)/11 \bmod 8$. Open up “newfunction.” Change `newfunction` to `newfunction2` and the 5 to an 8:

```
newfunction2:= function(t)
local x;
x:= (t^3+3)/11 mod 8;
return x;
end;
```

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

```
gap> Read("newfunction2");
gap> newfunction2(3);
2
```