Introduction to GAP

Part I

Alexander Hulpke

Department of Mathematics
Colorado State University
Fort Collins, CO, 80523-1874

`http://www.math.colostate.edu/~hulpke/lectures/tucson`

Arizona Summer Program 2008

Knowing the ropes

- Start/Stop
- Read-Eval-Print (Oops: View)-Loop
- Assignments, Variables
- Reading Files, Logging output
- Line editing
- Online help

Basic Objects

- Integers, Rationals (`5/3`), Cyclotomics (`E(n)`$=e^{\frac{2\pi i}{n}}$); Booleans `true`, `false` and `fail`
- Lists (and sets), Ranges: `l[n]`, `Length`, `IsBound`, `[1..10]`, `[1,3..7]`, `IsSSortedList`, `Set`
- (Row) Vectors and Matrices are lists, respectively lists of lists
- Assigning a list assigns a pointer to the list: `ShallowCopy`
- Lists can be immutable to permit properties be stored.

```
l:=[1,2,3];
m:=[l,[4,5,6]];
l[1]:=9; # changes m from sorted to unsorted
```

- Finite fields: Elements are represented as powers of a primitive element `Z(p^k)`. Conversion by multiplying an integer with `Z(p^k)^0` and `Int(elm)`. Compact matrix representations.
- Permutations in cycle form
- Words in abstract generators

List functions

The following extremely useful functions take as argument a list `l` and a one-argument function `f`, such functions often given by the shorthand `->` notation:

`List(l,f)` new list with `f` applied to all entries of `l`

```
List([1..10],i->i^2);
```

`Filtered(l,f)` list of elements of `l` for which `f` returns `true`.

Similar `First`, `Number`

```
Filtered([1..100],IsPrimeInt);
```

`ForAny(l,f)` are there elements of `l` for which `f` returns `true`?

Similar `ForAll`

More complicated constructions yield small programs:

```
Filtered(Tuples([1..5],4),
         i->1 in i
           and Number(i,j->j=2)=2
           and Length(Set(i))=3);
```

Caveat: direct loop programming can give better complexity.

Basic programming

- `function` assigned to function name. All non-local variables are assumed global. `return` the result.

```
f:=function(x,y)
local z;
  z:=x+y;
  return z;
end;
```

- Control constructs

```
if condition then ...fi;
while condition do ...od;
repeat ...until condition;
for var in list do ...od;
```

- Variables are passed *by reference*
- Recommended to write code in external editor, then `Read` into GAP.

Debugging

Unless you are able to write perfect code first time:

- `Print(...,"\n");`
- `Assert(`*level*`,` *condition*`);` and `SetAssertionLevel(`*level*`);`
- You enter the `brk>` loop (apart from errors) when pressing CTRL-C or by an explicit `Error("your text");` command in the code.
- You can inspect local variables in the `brk>` loop
- In the `brk>` loop you can find the position in the execution stack by `Where();`, you can move up by `DownEnv(`*nr*`);`.
- If you issue the command `GASMAN("message");` GAP will print information about memory use.
- `time;` returns the time of the last command, `Runtime();` the time since GAP started (in milliseconds).