JUSTUS-LIEBIG-
UNIVERSITÄT
GIESSEN

Mathematisches Institut, AG Algebra
Prof. Dr. Max Horn
Rechenkurs „Algebra mit GAP“

Worksheet 0

Version 2014-12-04

0 Getting started with GAP

0.1 What is GAP?

To quote the GAP homepage `http://www.gap-system.org`:

> GAP is a system for computational discrete algebra, with particular emphasis on Computational Group Theory. GAP provides a programming language, a library of thousands of functions implementing algebraic algorithms written in the GAP language as well as large data libraries of algebraic objects.

0.2 Installing GAP

We will be using GAP in the computer lab, under Linux. However, you probably will want to install it on your own computer so that you can work on exercises at other times, too.

So, here is what I recommend:

- If you are using Windows, then I recommend following the installation instructions presented here:

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

- If you are using Mac OS X, then I recommend using my (still under development) binary distribution. To this end, download

  `http://dev.quendi.de/GAP-4.7.5.dmg`

  then double-click the .dmg file, then copy GAP.app somewhere on your computer. You can start GAP.app as usual by double clicking it. (Note: This installer is work-in-progress; if you run into any issues, please let me know).

- If you are using Linux on computer with Intel CPU, then I recommend using Frank Lübeck's rsync installer from

  `http://www.math.rwth-aachen.de/~Frank.Luebeck/gap/rsync/`

- Anybody else will have to compile from source code. Feel free to ask me for help.

0.3 Documentation for GAP

Most texts dealing with GAP are in English. For an overview, see

`http://www.gap-system.org/Doc/doc.html`.

If you would like to learn more about GAP by yourself, check out the following list of learning materials:

`http://www.gap-system.org/Doc/Learning/learning.html`

0.4 First steps with GAP

In the following, we discuss how to

1. start GAP,
2. enter a simple command (e.g. compute the fifth power of 1234),
3. then exit GAP again.

For the lab sessions, we will do this on Linux.

1. Start GAP: On Windows, start GAP via the start menu.[1] On Linux or Mac OS X, first open a terminal,[2] then enter the command `gap` and press ↩ .

   In either case, the GAP command prompt should now show up, looking something like this:

```
 +-------+   GAP, Version 4.7.5 of 24-May-2014 (free software, GPL)
 |  GAP  |   http://www.gap-system.org
 +-------+   Architecture: x86_64-apple-darwin10.0.0-clang-default64
 Libs used:  gmp, readline
 Loading the library and packages ...
 Components: trans 1.0, prim 2.1, small* 1.0, id* 1.0
 Packages:   GAPDoc 1.5.1
 Try '?help' for help. See also  '?copyright' and  '?authors'
gap>
```

2. Enter a simple command: Enter `1234^5;` (note that every command sent to GAP must end with a semicolon), then press ↩ .

```
gap> 1234^5;
2861381721051424
```

3. Now exit GAP again. There are at least two ways to do this: By entering the command `quit;` or by pressing Ctrl - D (on a German keyboard, that is Strg - D).

0.5 Installing a text editor

Throughout the course, you will have to write more and more complex programs. For this, you should use a text editor. Below we suggest one for each of the three major systems covered by the course. This is just a suggestion, however, so if you already have a preferred editor, feel free to use it.

Linux

In the lab, we will use GEDIT, which we have preconfigured for you. It already comes with some basic support for GAP. For the initial setup, please go to the “Preferences” (found in the “Edit” menu) and set “Tab width” to 4, and enable “Insert spaces instead of tabs” as well as “Enable automatic indention”.

[1] I have no idea what to do on Windows 8, let me know if you succeed.

[2] On Mac OS X, start Terminal.app; for Linux we will demonstrate this in the first lab session

Windows

A popular free editor is Notepad++, which you can download from

http://notepad-plus-plus.org/

After installing it, you can also add a language module for GAP:

1. Download

   http://dev.quendi.de/gap/gap.xml

2. In Notepad++, select “Define your language...” from the “Language” menu.
3. In the resulting dialog, click the “Import...” button.
4. This opens a file selector dialog, with which you should download the gap.xml file from step 1.
5. Now close the “User Defined Language” dialog.

Mac OS X

A popular free editor is TextWrangler, which you can install via the Mac App Store, or download from

http://www.barebones.com/products/textwrangler/

After installing it, you can also add a language module for GAP: First quit TextWrangler. Then copy and paste the following command into a terminal session:

```
mkdir -p ~/Library/Application\ Support/TextWrangler/Language\ Modules && \
cd ~/Library/Application\ Support/TextWrangler/Language\ Modules && \
curl -O http://dev.quendi.de/gap/GAPLanguageModule.plist
```

0.6 Working with longer programs

Now that you have a working GAP and a text editor, let’s try to use it for writing a short program and then loading that program into GAP. For this, you should already have your editor and GAP running. Now open a new file in the editor and copy the following code into it (don’t worry if you do not yet understand what this does in detail):

test.g

```
fun := function(x)
   local n;
   n := x * (x+1) / 2;
   Print(x, " -> ", n, "\n");
end;
# Let's compute some example values:
fun(4);
fun(-4);
fun(42);
```

Now save the file somewhere (e.g. on the desktop) under the name `test.g`. To load the file into GAP, we can use the Read command. This takes as a single parameter the name (or rather, “path”) of your file. However, a naive attempt most likely will fail badly as in the following example:

```
gap> Read("test.g");
Error, file "test.g" must exist and be readable called from
<function "unknown">( <arguments> )
 called from read-eval loop at line 1 of *stdin*
you can 'quit;' to quit to outer loop, or
you can 'return;' to continue
brk>
```

GAP was not able to locate the file on your filesystem, and the prompt changed from “gap>” to “brk>”. To get out of that, enter `quit;` followed by ↩ , or press Ctrl - D .

Instead, we need to specify the full path. A trick that works on most systems is the following: First type just `Read("`. Then locate the file you want to read with a regular file browser (e.g. the Windows explorer), then drag and drop it onto the terminal window. This will insert the path. Now finish by entering `");` and pressing ↩ . You should see something like this:

```
gap> Read("/Users/mhorn/Desktop/test.g");
4 -> 10
-4 -> 6
42 -> 903
```

At this point, you can also directly invoke the function `fun`:

```
gap> fun(15);
15 -> 120
```

0.7 Some useful hints

There are a few techniques that help a lot while using the GAP command line interface. It is a good idea to learn these:

- The ↑ and ↓ key can be used to navigate through the history of things you previously entered.
- The TAB key can be used to complete partially entered commands. E.g. if you enter `Disp` and then press TAB , it will complete this to `Display`.
- Ctrl - A and Ctrl - E jump to the start resp. end of the line.

You should also run these commands once, to make GAP a bit friendlier to use. The `SaveAndRestoreHistory` setting is particularly helpful, as it tells GAP to remember what you typed across sessions.

```
SetUserPreference("UseColorPrompt", true);
SetUserPreference("HistoryMaxLines", 10000);
SetUserPreference("SaveAndRestoreHistory", true);
WriteGapIniFile();
```