Computeralgebra-Praktikum

Universität Siegen
Mohamed Barakat

WS 2018/19
Abgabe bis Do. 25.10.2018, 14:15 Uhr

1 Erste Schritte mit GAP

(i) Starte ein Terminal.

(ii) Starte GAP durch Eingabe von gap auf der Kommando-Zeile, gefolgt von einem ↩:

```
$ gap
 GAP 4.8.8
 https://www.gap-system.org
 Architecture: x86_64-linux-gcc--default64
 Libs used:  gmp, readline
 Loading the library and packages ...
 Components: trans 1.0, prim 2.1, small* 1.0, id* 1.0
 Packages:  Browse 1.8.7, FactInt 1.5.4, GAPDoc 1.6, IO 4.4.6, ...
 Try '??help' for help. See also '?copyright', '?cite' and '?authors'
gap>
```

(iii) Gebe folgende Befehle nach der gap>-Prompt ein. Jede Zeile endet mit ↩, jeder vollständige Befehl mit einem ; vor dem ↩:

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

Hinweis: Alle Befehlszeilen (samt gap>-Prompt) können mit der Maus kopiert und in GAP gepastet werden.

(iv) Um GAP zu beenden gebe folgendes ein:

```
gap> quit;
$
```

(v) Starte GAP nochmal und gebe folgende Befehle ein:

```
gap> 0 = 0;
true
gap> 0 = 1;
false
gap> 1 + 1;
2
gap> 2^(2^(2^2));
65536
gap> 2^(2^(2^(2^2)));
```

```
<integer 200...736 (19729 digits)>
gap> 2^(2^(2^(2^(2^2))));
Error, Integer operands: <exponent> is too large
not in any function at line 6 of *stdin*
you can replace the integer <exponent> via 'return <exponent>;'
brk>
```

Zum Verlassen der sogenannten break-loop: entweder `quit;` ↩ oder Strg - d

```
brk> quit;
gap> Factorial(50);
30414093201713378043612608166064768844377641568960512000000000000
gap> a := 0;
0
gap> for i in [ 1 .. 100 ] do
>      a := a + i;
>    od;
gap> a;
5050
```

(vi)
- Mit ↑ und ↓ kann man nun in der Zeilen-History rückwärts und vorwärts blättern. Durch Eingabe des exakten Anfangs eines früheren Befehls kann das rückwärts/vorwärts-Blättern eingeschränkt werden.
- Die Tab-Taste kann benutzt werden, um Befehle automatisch zu vervollständigen: Etwa wird der Befehlsanfang `LoadP` Tab zu `LoadPackage` vervollständigt.
- Mit Strg - a gelangt man an den Anfang der Zeile, mit Strg - e ans Ende.
- Mit Strg - k wird die restliche Zeile ab der Cursor-Position abgeschnitten.

(vii) GAP hat ein online verfügbares reference manual:

`http://www.gap-system.org/Manuals/doc/ref/chap0.html`

Das inline-Hilfesystem erreicht man mit `?`: Etwa

```
gap> ?r: if statement
gap> ?r: for loop
gap> ?function
gap> ?1
```

Man beendet das Blättern der Hilfe mit q .

Aufgabe 0. Unter Zuhilfenahme des GAP-Hilfesystems: Programmiere eine GAP-Funktion `treppe`, die bei Eingabe einer positiven natürlichen Zahl n die Summe $\sum_{i=1}^{n} i$ zurückgibt:

```
gap> List( [1..10], treppe );
[ 1, 3, 6, 10, 15, 21, 28, 36, 45, 55 ]
```

(viii)

```
gap> LoadPackage( "RingsForHomalg" );
true
gap> ZZ := HomalgRingOfIntegers( );
Z
gap> (-2)^2;
4
gap> IsFieldForHomalg( ZZ );
false
gap> QQ := HomalgFieldOfRationals( );
Q
gap> (1/2)^2;
1/4
gap> IsFieldForHomalg( QQ );
true
gap> Qx := QQ * "x";
Q[x]
gap> AssignGeneratorVariables( Qx );
#I  Assigned the global variables [ x ]
gap> (x+1)^2;
x^2+2*x+1
gap> IsFieldForHomalg( Qx );
false
gap> F5 := HomalgRingOfIntegers( 5 );
GF(5)
gap> IsFieldForHomalg( F5 );
true
gap> F5y := F5 * "y";
GF(5)[y]
gap> AssignGeneratorVariables( F5y );
#I  Assigned the global variables [ y ]
gap> (y+3)^2;
y^2+y-Z(5)^0
gap> IsFieldForHomalg( F5y );
false
gap> mat := HomalgMatrix( [ 1, 2, 3, 4, 5, 6 ], 2, 3, ZZ );
<A 2 x 3 matrix over an internal ring>
gap> R := HomalgRing( mat );
Z
gap> IsFieldForHomalg( R );
false
gap> Display( mat );
[ [  1,  2,  3 ],
  [  4,  5,  6 ] ]
```

Aufgabe 1. Sei $R \in \{\mathbb{Q}, \mathbb{F}_5, \mathbb{Z}, \mathbb{Q}[x], \mathbb{F}_5[y]\}$. Programmiere eine GAP-Funktion

`fully_divide_pair_trafo_R(a,b,R)`,

die bei Eingabe von $a, b \in R$ eine Matrix $U \in \mathrm{GL}_2(R)$ zurückgibt, mit

$$U \begin{pmatrix} a \\ b \end{pmatrix} = \begin{cases} \left(\begin{smallmatrix} 0 \\ 0 \end{smallmatrix}\right) & a = b = 0, \\ \left(\begin{smallmatrix} d \\ 0 \end{smallmatrix}\right) & \text{sonst}, \end{cases}$$

mit $d = 1$ falls R ein Körper ist und sonst $d = \mathrm{ggT}(a, b)$.
Hinweis: `?HomalgIdentityMatrix`, `?IsZero`, `?EuclideanQuotient` und Algorithmus 1 falls R ein Euklidischer Ring, der kein Körper ist. Benutze für den Körper fall folgende Prozeduren

```
##
mulmat := function( a, K )
    return HomalgMatrix( [ a^-1, 0, 0, 1 ], 2, 2, K );
end;

##
addmat := function( b, K )
    return HomalgMatrix( [ 1, 0, -b, 1 ], 2, 2, K );
end;
```

Algorithm 1: Erweiterter Euklidischer Algorithmus

Input:

- Zwei Ringelemente $a, b \in R$ und der Ring R

Output: Eine Matrix $U \in \mathrm{GL}_2(R)$ mit $U \begin{pmatrix} a \\ b \end{pmatrix} = \begin{cases} \left(\begin{smallmatrix} 0 \\ 0 \end{smallmatrix}\right) & a = b = 0, \\ \left(\begin{smallmatrix} \mathrm{ggT}(a,b) \\ 0 \end{smallmatrix}\right) & \text{sonst.} \end{cases}$

fully_divide_pair_trafo_R_R *(a, b, R)*

1. Initialisiere $U := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \in R^{2\times 2}$
2. **while** *not* `IsZero`(b) **do**
3. $\quad q := $ `EuclideanQuotient`(R, a, b)
4. $\quad U := \begin{pmatrix} 0 & 1 \\ 1 & -q \end{pmatrix} U$
5. $\quad r := a - qb$
6. $\quad a := b$
7. $\quad b := r$
8. **return** U

Vgl. Algorithmus 2.1.18 im LA I Skript:

`http://www.algebra.mathematik.uni-siegen.de/barakat/Lehre/WS17/LAI/Skript/LA_I.pdf`