Príloha A

Magma – stručný úvod a základné ovládanie

Magma `http://magma.maths.usyd.edu.au/magma/` je výkonný softvérový nástroj optimalizovaný pre riešenie problémov z oblasti algebry, teórie čísel algebraickej geometrie a kombinatoriky. Manuál obsahujúci opis všetkých dostupných funkcií jazyka Magma je dostupný na stránke `https://magma.maths.usyd.edu.au/magma/handbook/`. PDF verzia manuálu má takmer 5 500 strán a je súčasťou inštalačného balíka. Aj keď softvérový balík Magma je dostupný len za poplatok, vzhľadom na licenčné podmienky je možné manuál šíriť, a tak je dostupný na stránkach niektorých univerzít ako napr. `http://www.math.uzh.ch/sepp/magma-2.19.8-cr/Handbook.pdf`. Ďalšia dokumentácia vrátane krátkeho úvodu [5] a knihy so súborom riešených problémov s využitím Magmy [59] je dostupná na stránke `http://magma.maths.usyd.edu.au/magma/documentation/`. Pre podrobnejšie štúdium o programovaní v Magme je tiež zaujímavá publikácia `https://magma.maths.usyd.edu.au/magma/pdf/intro.pdf`, v ktorej je uvedený detailný opis jazyka a programovania v Magme.

V nasledujúcom opise sa obmedzíme len na opis základných príkazov, konštrukcií a operátorov pomocou ktorých je možné vytvárať zložitejšie príkazy (**Magma skripty**) a riešiť tak aj zložité úlohy. Použité Magma funkcie sú vysvetľované v texte pomocou jednoduchých riešených príkladov v jednotlivých kapitolách učebnice. Hlavným cieľom tejto prílohy je poskytnúť čitateľovi veľmi stručný úvod k jazyku Magma, vysvetliť základnú filozofiu a nasmerovať ho na relevantné zdroje, kde môže hľadať ďalšie potrebné informácie. Väčšinu informácií v tejto prílohe je možné využiť aj pri práci v **kalkulačke Magma** (ďalej len kalkulačka)(`http://magma.maths.usyd.edu.au/calc/`). Kalkulačka je dostupná cez **voľne dostupné** webovské rozhranie, pomocou ktorého môžeme spúšťať časovo aj veľkosťou kódu limitované Magma skripty. Pokiaľ bude v opise uvedená informácia, ktorú je možné využiť len v plnej verzii balík Magma, tak na to v texte explicitne upozorníme.

A.1 Spustenie programu a zadávanie príkazov

Plná verzia programu Magma sa spúšťa z príkazového riadku operačného systému príkazom

```
magma
```

a stlačením klávesy enter. Plná verzia programu Magma je následne dostupná pomocou terminálového okna, ktoré je zobrazené na obr. A.1. Pripravenosť programu prijímať príkazy od užívateľa indikuje znak `>`. Túto konvenciu platnú pre plnú verziu budeme využívať pri zápise príkazov tak, že vstupy užívateľa budú zobrazené vždy za znakom `>` a výstupy (výsledky príkazov) budú zobrazované bez znaku `>`. Kalkulačka spracováva vstupné dáta dávkovým spôsobom (teda všetky príkazy musí užívateľ potvrdiť spolu) a výsledky sú následne zobrazované v samostatnom výstupnom okne tak, ako to je znázornené na obr. 1.2. Konvenciu plnej verzie sme využili pre jej väčšiu názornosť v celej učebnici.

```
Magma V2.20-10    Sun Aug 20 2017 14:03:48 on upc9     [Seed = 1149527960]
Type ? for help.  Type <Ctrl>-D to quit.
>
```

Obr. A.1: Terminálové rozhranie plnej verzie programu Magma

„Help“ informácie

Podrobnejší opis o funkcionalite konkrétneho príkazu môžeme získať priamo z programu Magma pomocou zadania `?` a slova, ktorého význam chceme zistiť. Napríklad význam príkazu **`mod`** zistíme takto:

```
> ?mod
    17 matches:
    1   0   /magma/ring-field-algebra
    2   S   /magma/geometry/modular-curves/mod-crv-quotient
    ...
   10   I   /magma/ring-field-algebra/function-field/elements/\
             elements-arithmetic/mod
    ...
   17   I   /magma/ring-field-algebra/univariate-polynomial/\
             operation-element/division/mod
To view an entry, type ? followed by the number next to it.
```

Informáciu o konkrétnej položke (napr. č. 10) získame v kalkulačke takto:

```
> ?mod
> ?10
================================================================================
PATH: /magma/ring-field-algebra/function-field/elements/elements-
arithmetic/mod
```

```
KIND: Intrinsic
================================================================================
a mod I : RngFunOrdElt, RngFunOrdIdl -> RngFunOrdElt

Return the element a belonging to the order O as an element of O/I.
================================================================================
```

Často sú vo výstupe zobrazené aj príklady použitia príkazu (pre zobrazenú položku 10 príklad v „Magma help“ systéme nie je), takže základné použitie príkazu je možné pomerne rýchlo zvládnuť.

Základné aritmetické operácie a výstup funkcií

Výpis hodnoty realizujeme príkazom **print**, pričom každý príkaz **musí byť ukončený** znakom **;** napr. takto:

```
> print 12 + 22;
34.
```

Slovo **print** môžeme pri zápise vynechávať, čo budeme v nasledujúcich príkladoch aj využívať.

Magma používa pre označenie aritmetických operátorov (súčet, rozdiel, násobenie, umocnenie) operátory **+**, **-**, *****, **^**. Operátor pre delenie závisí od typu objektu. Napríklad pre celé čísla má „delenie“ význam nájdenia kvocientu a zvyšku, čo nájdeme pomocou **div** a **mod** operátorov napr. takto:

```
> 27 div 7, 27 mod 5;
3 2.
```

Pre reálne a racionálne čísla používame klasický operátor delenia **/**, ktorý Magma interpretuje tak, že v prípade racionálnych čísel vypočíta presný výsledok:

```
> (2/3) / (4 + 7/9);
6/43
> 22.0/7.0
3.14285714285714285714285714286.
```

Poradie operácii je vyhodnocované podľa bežných pravidiel, ktoré v aritmetike platia, takže napr. násobenie sa realizuje pred sčítaním. Samozrejme, iné poradie vyhodnocovania je možné vnútiť pomocou zátvoriek.

Magma obsahuje veľké množstvo funkcií, ktoré môžu mať aj niekoľko argumentov. Argumenty sú oddelené čiarkou **,**. Názvy takmer všetkých interných funkcií začínajú veľkým písmenom – napr. **Factorial**, najväčší spoločný deliteľ **GCD**, a pod. Výstup funkcie môžeme vypísať na obrazovku takto:

```
> GCD(15130, 3162);
34.
```

Hodnotu môžeme vypísať aj vo forme **print** príkazu (slovo **print** netreba uvádzať):

```
> "najvacsi spolocny delitel 15130 a 3162 je:", GCD(15130, 3162);
najvacsi spolocny delitel 15130 a 3162 je: 34.
```

Výstup funkcie tiež, samozrejme, môžeme uložiť do premennej s použitím operátora priradenia `:=`

```
> res := GCD(15130, 3162);
```

S premennou môžeme pracovať podobne, ako sme zvyknutí z podobných nástrojov a programovacích jazykov, premenná sa môže nachádzať aj na oboch stranách operátora priradenia:

```
> res := GCD(15130, 3162);
> res := res + 10;
> res;
44.
```

V plnej verzii programu Magma môžeme príkazom `load "menosuboru"` načítať obsah súboru do prostredia Magma. Obsah je interpretovaný tak, ako by sme ho písali na klávesnici. V kalkulačke môžeme súbor skopírovať do vstupného okna „ručne“, po spustení jeho obsahu však všetky predtým vypočítané a uložené premenné budú vymazané. Takže v kalkulačke nie je možné priamo „akumulovať“ výsledky predchádzajúcich výpočtov. Funkcie `ShowIdentifiers()` a `ShowValues()` zobrazia aktuálne definované premenné a ich hodnoty.

A.2 Relačné operátory a riadenie slučiek

Boolovské operátory

Základné relačné a logické operátory, ktoré môžeme využívať pri vytváraní a vyhodnocovaní zložitejších podmienok sú zobrazené v tab. A.1.

Tabuľka A.1: Základné relačné a logické operátory jazyka Magma

Operátor	Použitie	Význam
`eq`	`x eq y`	**pravda** ak x je rovné y, inak **nepravda**
`ne`	`x ne y`	**pravda** ak x nie je rovné y, inak **nepravda**
`lt`	`x lt y`	**pravda** ak x je menšie ako y, inak **nepravda**
`le`	`x le y`	**pravda** ak x je menšie alebo rovné y, inak **nepravda**
`gt`	`x gt y`	**pravda** ak x je väčšie ako y, inak **nepravda**
`ge`	`x ge y`	**pravda** ak x je väčšie alebo rovné y, inak **nepravda**
`not`	`not a`	**pravda** ak a je **nepravda**, inak **nepravda**
`and`	`a and b`	**pravda** ak a aj b sú **pravda**, inak **nepravda**
`or`	`a or b`	**nepravda** ak a aj b sú **nepravda**, inak **pravda**
`xor`	`a xor b`	**pravda** ak iba jedno z a, b je **pravda**, inak **nepravda**

V Magme okrem relačných a logických operátorov existuje aj veľký počet funkcií, ktoré vracajú hodnotu `true` alebo `false`. Tieto funkcie najčastejšie majú v názve

na začiatku **Is**. Príkladom je funkcia **IsPrime** na testovanie prvočíselnosti. Napríklad pre štvrté a piate Fermatove čísla dostávame výsledok testu prvočíselnosti:

```
> IsPrime(2^2^4+1);
true
> IsPrime(2^2^5+1);
false.
```

Podmienené príkazy

Príkaz **if (podmienka) then** umožňuje vykonávať príkazy v závislosti od vyhodnotenia podmienky (**pravda**, **nepravda**) a má tieto formáty:

```
> if (podmienka) then
> prikazy
> end if
```

```
> if (podmienka) then
> prikazy;
> else
> prikazy;
> end if
```

```
> if (podmienka) then
> prikazy;
> elif (podmienka) then
> prikazy;
> elif ...
> else
> prikazy;
> end if
```

Príklad

```
> n := Random(1, 100);
> if not IsPrime(n) then
if> print n, Factorization(n);
if> end if;
98 [ <2, 1>, <7, 2> ]
```

keďže platí $98 = 2^1 * 7^2$.

Množiny a postupnosti

Množiny a postupnosti sú objekty, pričom množina je neusporiadaná a prvok sa môže v množine vyskytovať len raz. Postupnosť je usporiadaná a prvky v postupnosti sa môžu opakovať. Prvky množiny sú zadávané pomocou { } a prvky v postupnosti pomocou [ ], napríklad takto:

```
> t := {(-11)^2, (-7)^2, (-5)^2, (-3)^2, 3^2, 5^2, 3^2, 11^2};
> q := [(-11)^2, (-7)^2, (-5)^2, (-3)^2, 3^2, 5^2, 3^2, 11^2];
> t, q;
{ 9, 25, 49, 121 }
[ 121, 49, 25, 9, 9, 25, 9, 121 ].
```

i-ty prvok postupnosti q je $q[i]$, pričom indexovanie začína od hodnoty 1, čo využijeme napr. takto:

```
> q :=[ 121, 49, 25, 9, 9, 25, 9, 121 ];
> q[10] := q[5] - 4*q[1]; q[2]:= 1000;
> q;
[ 121, 1000, 25, 9, 9, 25, 9, 121, undef, -475 ].
```

Magma využíva pre množiny a postupnosti zápisy $\{i..j$ **by** $k\}$ a $[i..j$ **by** $k]$, ktoré reprezentujú hodnoty celých čísel $i, i+k, i+2k, \ldots, j$ v množine alebo postupnosti a {"výraz s x": x **in** D | podmienka} alebo ["výraz s x": x **in** D | podmienka], ktoré znamenajú množinu alebo postupnosť hodnôt "výraz s x" vyjadrenú pre všetky x z D také, že Boolovska podmienka je **pravda**. Napr množinu t môžeme vyjadriť takto:

```
> t := { n^2 : n in [-11..11 by 2] | IsPrime(n) };
{ 9, 25, 49, 121 }.
```

Symbol `&` nasledovaný binárnym operátorom a množinou alebo postupnosťou S skombinuje všetky prvky S použitím binárneho operátora. Napríklad nasledujúci príkaz vypíše hodnotu $\sum_{i=1}^{10} (i!)^2$:

```
> &+[ Factorial(i) ^ 2 : i in [1..10] ];
13301522971817.
```

Vytváranie slučiek

Pre vytváranie slučiek Magma používa konštrukcie **`while`**, **`repeat`** a **`for`** podobne, ako ich využívajú bežné programovacie jazyky. Tieto konštrukcie majú tvar:

```
> while (Boolovsky vyraz) do
> prikazy;
> end while;

> repeat
> (Boolovsky vyraz) do
> prikazy;
> until Boolovsky vyraz

> for premenna in oblasti  do
> prikazy;
> end for;
```

Užívateľom vytvorené funkcie

Môžeme využiť dva spôsoby zápisu. Skrátený s **func** príkazom. Napríklad funkciu $f(n,q) = \prod_{i=1}^{n} q^n - q^{i-1}$ môžeme deklarovať a následne použiť takto:

```
> f := func< n, q | &*[q^n - q^(i-1) : i in [1..n]] >;
> f(5, 3);
475566474240.
```

Ak potrebujeme viac návratových hodnôt, môžeme využiť dlhší zápis:

```
> counting := function(n, r)
> x := Factorial(n);
> y := Factorial(r);
> z := Factorial(n-r);
> p := x div z;
> c := p div y;
> return p, c;
> end function;
> per, com := counting(5, 2);
> per, com;
20 10.
```

A.3 Algebraické systémy – Galoisove polia a eliptické krivky

Magma podporuje prácu s veľkým množstvom rôznych typov algebraických systémov. Práve veľmi prepracovaná podpora rôznych algebraických systémov je silnou stránkou Magmy. V učebnici využívame len Galoisove polia $\mathbb{GF}(p)$, $\mathbb{GF}(2^k)$ a **aditívnu komutatívnu (abelovskú) grupu**, ktorú tvoria body P na eliptickej krivke (EC) definovanej nad $\mathbb{GF}(p)$.
Vytvorenie $\mathbb{GF}(p)$ a realizovanie aritmetických operácií s $a, b \in \mathbb{GF}(p)$ je v Magme veľmi jednoduché, čo demonštruje tento príklad práce s prvkami z $a, b \in \mathbb{GF}(7)$:

```
> gf:= GF(7);                 // vytvorenie GF(7)
> a := gf!2;                  // prvok 2 v GF(7);
> b := gf!6;                  // prvok 6 v GF(7);
> c := a + b;                 // sucet v GF(7);
> c;
1
> c := a - b;                 // rozdiel v GF(7)
> c;
3
> c := a*b;                   // sucin v GF(7)
> c;
5
> c := a^-1;                  // inverzia v GF(7)
> c;
```

```
4
> c := b/a;                    // "delenie" v GF(7)
> c;
3.
```

Pre vytvorené GF môžeme pomocou dostupných funkcií zistiť a vypísať aj rôzne vlastnosti ako napr. zoznam primitívnych prvkov takto:

```
> gf := GF(7);
> print gf;
Finite field of size 7
> Characteristic(gf);
7
> PrimitiveElement(gf);        // generator
3
> print Set(gf);
{ 0, 1, 2, 3, 4, 5, 6 }.
```

Samozrejme, rozmer $\mathbb{GF}(p)$ nie je obmedzený na malé hodnoty p a môžeme použiť aj veľké rozmery, napr. NIST prvočíslo $p_{521} = 2^{521} - 1$ dostaneme:

```
> p := 2^(521) - 1;
> IsPrime(p);
true
> gf := GF(p);                 // vytvorenie GF(2^(521)-1)
> Characteristic(gf);
6864797660130609714981900799081393217269435300143305409394463459185543183\
3
9765605212255964066145455497729631139148085803712198799971664381257402829\
1
115057151
> PrimitiveElement(gf);        // generator
3.
```

pričom vypisovať prvky tak veľkého pola nemá, samozrejme, zmysel. V Magme môžeme podobne priamo vytvárať aj $\mathbb{GF}(2^k)$, napríklad takto:

```
> p := 256;
> gf<w> := GF(p);              // vytvorenie GF(2^8)
> print gf;
Finite field of size 2^8
> Characteristic(gf);
2
>PrimitiveElement(gf);         // generator
w.
```

V prípade, že potrebujeme pracovať v polynomiálnej aritmetike, môžeme vytvoriť AES GF ako okruh polynómov stupňa 7 nad $\mathbb{GF}(2)$ takto:

```
> gf2 := GF(2);
> Pgf2<x> := PolynomialRing(gf2);
> m := x^8 + x^4 + x^3 +x + 1;
> print IsIrreducible(m);
```

```
true
> gfaes<x> := ext<gf2 | m>;
> a := x^6 + x^4 + x^2 + x + 1;
> b := x^7 + x + 1;
> c := a * b;
> c;
x^7 + x^6 + 1.
```

pričom výsledok násobenia $c = a \cdot b$ je zhodný s „ručne vypočítaným“ výsledkom (4.4) v riešenom príklade na str. 42.

Podobne jednoducho môžeme v Magme vytvoriť **aditívnu grupu** bodov P na eliptickej krivke nad $\mathbb{GF}(p)$. V učebnici využívame len EC v tvare

$$y^2 \equiv x^3 + ax + b \pmod{p},$$

pričom

$$4a^3 + 27b^2 \neq 0 \pmod{p}.$$

Napríklad EC nad $\mathbb{GF}(23)$ s parametrami $a = -3$ a $b = 11$ môžeme vytvoriť takto:

```
> K := GF(23);
> a := K!(-3);                // -3 = -3 mod 23 = 20
> b := K!10;
> E := EllipticCurve([a, b]);
> E;
Elliptic Curve defined by y^2 = x^3 + 20*x + 10 over GF(23)
> Order(E);                   // pocet bodov na krivke E
18
> Points(E);                  // zoznam bodov v projektivnych suradniciach
{@ (0 : 1 : 0), (1 : 10 : 1), (1 : 13 : 1), (2 : 9 : 1), (2 : 14 : 1), (4 :
4 : 1), (4 : 19 : 1), (6 : 1 : 1), (6 : 22 : 1), (12 : 0 : 1), (13 : 11 :
1), (13 : 12 : 1), (19 : 2 : 1), (19 : 21 : 1), (21 : 10 : 1), (21 : 13 :
1), (22 : 9 : 1), (22 : 14 : 1) @}.
```

Použité GF môže mať, samozrejme, aj podstatne väčšie rozmery, napr. už použité $\mathbb{GF}(2^{521} - 1)$. S bodmi, ktoré ležia na eliptickej krivke, môžeme realizovať operácie súčtu, násobenia, opačného bodu, ... napr. takto:

```
> K := GF(23);
> a := K!(-3);                // -3 = -3 mod 23 = 20
> b := K!10;
> E := EllipticCurve([a, b]);
> P := E![1,10,1];            // zvoleny bod P leziaci na EC
> P;
(1 : 10 : 1)
> Q := E![19,21,1];           // zvoleny bod Q leziaci na EC
> Q;
```

```
(19 : 21 : 1)
> P+Q;                          // sucet bodov na EC
(6 : 1 : 1)
> P+P;
(21 : 13 : 1)
> 2*P;                          // to iste ako  P + P
(21 : 13 : 1)                   // vysledok je bod nekonecno
> P-P;
(0 : 1 : 0)
> k := 2^2^4+1;
> k*P;                          // nasobenie zvolenym sklarom k
(1 : 13 : 1)
> Order(P);                     // rad bodu P
9
> Order(P)*P;                   // vysledok musi byt bod nekonecno
(0 : 1 : 0)
> Order(Q);                     // rad bodu Q
18
> Order(Q)*Q;                   // vysledok musi byt bod nekonecno
(0 : 1 : 0).
```

Z uvedených príkladov vidno, že s pomocou programu Magma môžeme realizovať operácie v $\mathbb{GF}(p)$, $\mathbb{GF}(2^k)$ a na EC veľmi jednoduchým spôsobom. Pomocou opísaných konštrukcií pre riadenie cyklov tak môžeme realizovať aj pomerne zložité výpočty. Ďalšie užitočné funkcie a príklady použitia sú opísané v **riešených príkladoch** v jednotlivých kapitolách učebnice. V prípade potreby realizovať zložitejšie výpočty je možné využiť voľne dostupné podrobné manuály k jazyku Magma. Na sledovanie riešených príkladov v jednotlivých kapitolách by však tieto úvodné informácie mali čitateľovi postačovať.