5 Übung 5

5.1 Aufgabe 1

5.2 Aufgabe 2

5.3 Aufgabe 3

5.4 Aufgabe 4

5.5 Aufgabe 5

Siehe `Krypto/gap/ElGamal.g`

6 Übung 6

6.1 Aufgabe 1

6.2 Aufgabe 2

- öffentlicher Schlüssel: $(n, e) = (299, 79)$.
- geheimer Schlüsel: $d = 127$.

$$\begin{aligned} ed - 1 &= 79 \times 127 - 1 \\ &= 10033 - 1 \\ &= 10032 \\ &= 2^s \times t \\ &= 2^4 \times 627 \end{aligned}$$

Also $s = 4$ und $t = 627$. Für (2.1.26) müssen wir dann ein $a \in \{1, \ldots, n-1\}$ finden mit

$$\mathrm{ggT}(a, n) = 1 \tag{1a}$$
$$\mathrm{ord}[a^t]_p \neq \mathrm{ord}[a^t]_q \tag{1b}$$

wobei wir p und q ja gerade noch nicht kennen. Wir ignorieren also die zweite Bedingung (1b) und gehen probabilistisch vor, d.h. wir wählen ein zufälliges $a \in \{1, \ldots, n-1\}$, welches (1a) erfüllt. Mit den so bestimmten Variablen und einem $i \in \{0, \ldots, s-1\} = \{0, 1, 2, 3\}$ haben wir also a, i, t, n sodass

$$\mathrm{ggT}(a^{2^i t} - 1, n) \in \{p, q\}. \tag{2}$$

Zum Beispiel für $a = 224$, welches $\mathrm{ggT}(a, n) = 1$ erfüllt, ergibt sich

$$i = 0 : \mathrm{ggT}(224^{2^0 \times 627} - 1, 299) = 13$$

Also haben wir 13 als Teiler von 299 erkannt, und tatsächlich ist $299/13 = 23$.

Als Beispiel, dass auch $i > 0$ nötig sein kann, z.B. $a = 44$, dann ergibt sich

$$\begin{aligned} i = 0 &: \mathrm{ggT}(44^{2^0 \times 627} - 1, 299) = 1 \\ i = 1 &: \mathrm{ggT}(44^{2^1 \times 627} - 1, 299) = 23 \end{aligned}$$

Als Algorithmus nach (2.1.28) habe ich `Krypto/gap/RSAFactoring.g` geschrieben, dem man n, e, d übergibt (der also bereits annimmt, dass man d aus n und e effizient errechnet hat) und der daraus einen Faktor von n bestimmt.

6.3 Aufgabe 3

ElGamal $\mathbb{Z}_p^*$, $p = 17, a = 3, k = 10$.

a) öffentlicher Schlüssel

$$\begin{aligned} (p, a, k) &= (17, 3, 10) \\ z &= a^k \mod p \\ &= 3^{10} mod 17 \end{aligned}$$

Schnelle Expo mit $a = 3, n = 10, m = 17$:

k	pk	nk	ak	nk mod 2
0	1	3	10	0
1	1	9	5	1
2	9	13	2	0
3	9	-1	1	1
4	-9	0	(Ende)	

$a^3 \mod 17 = -9 \mod 17 = 8 \mod 17$.

Also $z = 8$ und damit ist der öffentliche Schlüssel $(p, a, z) = (17, 3, 8)$.

b) $m = 5, y = 5$.

(langsame Expo im Kopf):

$$\begin{aligned} c &= a^y \mod p \\ &= 3^5 \mod 17 \\ &= 3^4 \cdot 3 \mod 17 \\ &= 81 \cdot 3 \mod 17 \\ &= 13 \cot 3 \mod 17 \\ &= -43 \mod 17 \\ &= -12 \mod 17 \\ &= 5 \mod 17 \end{aligned}$$

$$\begin{aligned} d &= m \cdot z^y \mod p \\ &= 5 \cdot 8^5 \mod 17 \\ &= 5 \cdot 64 \cdot 64 \cdot 8 \mod 17 \\ &= 5(-4)(-4)8 \mod 17 \\ &= 5 \cdot 2 \cdot 64 \mod 17 \\ &= -40 \mod 17 \\ &= -6 \mod 17 \\ &= 11 \mod 17 \end{aligned}$$

$(c, d) = (5, 11)$.

c) $(12, 12) = (c, d)$.

$$\begin{aligned} s &= 12^{-1} \mod 17 \\ 12 \cdot 3 &= 37 \cong 2 \mod 17 \\ 2 \cdot 9 &= 18 \Rightarrow 2^{-1} = 9 \mod 17 \\ (12 \cdot 3)^(- 1) &= 9 \mod 17 \\ 12^{-1} \cdot 3^{-1} &= 9 \mod 17 \\ 12^{-1} &= 3 \cdot 9 = 27 = 10 \mod 17 \\ s &= 10 \end{aligned}$$

$$ds^k \mod p = 12 \cdot 10^{10} \mod 17$$

k	pk	nk	ak	nk mod 2
0	1	10	10	0
1	1	5	5	1
2	-2	2	2	0
3	-2	1	1	1
4	2	0	(Ende)	

6.4 Aufgabe 4

Siehe `Krypto/gap/ElGamal.g`.

Zuerst habe ich den Algorithmus zur schnellen Exponentiation names `fex` geschrieben. Dann habe ich den Miller-Rabin-Test implementiert, einmal als Primzahl-Test `MillerRabinTest, MRT` mit Eingaben n, k wobei k die Anzahl der Wiederholungen des Tests ist, und einmal als Primzahl-Generator `MillerRabinGenerator, MRG` mit Eingaben `low, high` das Intervall, in dem nach einer Primzahl gesucht werden soll, und k erneut wie oben.

Mit diesen Hilfsmitteln habe ich dann das Programm `ElGamal` geschrieben, das zu einer Eingabe $n \in \mathbb{N}$ ein Schlüssel-Paar `key.public` und `key.private` (als gap-Records) erzeugt, wobei `key.public` $= (p, a, z)$ und `key.private` $= k$ wobei für die Primzahl p gilt $p \geq n$, und für den geheimen Schlüssel k gilt $k \geq \frac{1}{2}n$.

Der Algorithmus geht nach dem in (2.2.3) Beschriebenen vor mit $f = 2$, d.h. anstatt Primzahl p und Primitivwurzel a getrennt zu wählen, wird ein Paar (m, a) bestimmt, für das m eine Primzahl und a eine Primitivwurzel modulo m ist. Hierbei wird für die erstmalige Primzahl q das Suchintervall $[n, 2n]$ nach Betrand bestimmt, und die Wiederholungen bei 10. Ebenso wird m mit 10 Wiederholungen auf Primzahl getestet.

Sobald man das Paar (p, a) hat, wird ein zufälliger geheimer Schlüssel $k \geq \frac{n}{2}$ gebildet. Anschließend wird noch das für den öffentlichen Schlüssel fehlende $z = a^k \mod p$ berechnet, wobei wiederum die Schnelle Exponentiation zum Einsatz kommt. Damit ist auch der öffentliche Schlüssel `key.public` $= (p, a, z)$ bestimmt, und beide werden zusammen ausgegeben.

6.5 Aufgabe 5

Satz 6.5.1 (Lagrange)**.** Sei G eine endliche Gruppe und $U \leq G$ eine Untergruppe von G. Dann gilt:

(1) $|U|$ teilt $|G|$.

(2) $\left|U\backslash^G\right| = \left|{}^G/_U\right| = \frac{|G|}{|U|}$.

Beweis. Weil G endlich ist, ist U eine endliche Untergruppe von G. Setze $|U| := m \in \mathbb{Z}_{\geq 1}$. Zu jedem $g \in G$ sei $gU := \{gu : u \in U\}$ eine Linksnebenklasse von U. Dann ist die Abbildung

$$g\cdot : U \to gU,\ u \mapsto gu$$

eine Bijektion: Weil $g \in G$ invertierbar ist, definiert g^{-1} die Abbildung

$$g^{-1}\cdot : gU \to U,\ gu \mapsto g^{-1}gu = u$$

und es gilt

$$g \cdot (g^{-1} \cdot (gu)) = g \cdot (u) = gu$$

sowie

$$g^{-1} \cdot (g \cdot (u)) = g^{-1} \cdot (gu) = u$$

also $(g\cdot)\cdot(g^{-1}\cdot) = \mathrm{Id}_{gU}$ und $(g^{-1}\cdot)\cdot(g\cdot) = \mathrm{Id}_U$. Also ist $g\cdot$ bijektiv. Insbesondere gilt $|gU| = |U|$, weil U endlich ist. Damit folgt $|U| = |gU| \; \forall g \in G$.

Es sei ${}^G/_U := \{gU : g \in G\}$ die Menge der Linksnebenklassen von U. Für $g, h \in G$ untersuchen wir den Fall, dass $gU = hU$. Unter der Äquivalenzrelation

$$g \sim h :\Leftrightarrow gU = hU$$

betrachte die Menge ${}^G/_\sim := \{[g]_\sim : g \in G\}$ mit $[g]_\sim := \{h \in G | h \sim g\}$. Offensichtlich gilt ${}^G/_U \simeq {}^G/_\sim$. Im Fall $gU = hU$ gibt es also zu jedem $u_1 \in U$ ein $u_2 \in U$ sodass $gu_1 = hu_2$, also $h^{-1}g = u_1^{-1}u_2 \in U$. Damit stellen wir fest, dass

$$g \sim h \Leftrightarrow gU = hU \Leftrightarrow h^{-1}g \in U \Leftrightarrow g^{-1}h \in U.$$

Die Anzahl der Nebenklassen von U ist also ein Vielfaches von $|U|$, d.h. $\exists n \in \mathbb{Z}_{\geq 1} : \left|{}^G/_U\right| = n \cdot |U| = n \cdot m$.

□

6.6 Aufgabe 6

Siehe `Krypto/gap/EllipticCurve.g`.

Zunächst habe ich eine Funktion `defines_ellipse` geschrieben, die bei Eingabe von a_4, a_6 und dem Körper F überprüft, ob durch $E_0(a_4, a_,6, F)$ eine elliptische Kurve i.S.v. (2.3.10) definiert ist, wobei die Existenz des neutralen Elements ignoriert wird, also nur die Eigenschaften in (2.3.6) überprüft werden, d.h.

$$F \text{ ist ein Körper} \tag{3}$$
$$\mathrm{Char}\, F \neq 2 \tag{4}$$
$$\mathrm{Char}\, F \neq 3 \tag{5}$$
$$4a_4^3 + 27a_6^2 \neq 0 \tag{6}$$

wobei die letzte Gleichung über F gesehen werden sollte.

Für die endlichen Körper benutze ich das gap-Paket `GaussForHomalg` vom homalg_project (`https://github.com/homalg-project/homalg_project`).

a) $a_4 = -7, a_6 = -6, F = \mathbb{Z}_5, P = (3, 0)$

Zunächst betrachte ich die Zahlen als Elemente von $\mathbb{Z}_5$, d.h.

$$\begin{array}{rll} a_4 & \mod 5 = -7 & \mod 5 = 3 \\ a_6 & \mod 5 = -6 & \mod 5 = 4 \\ P & \mod 5 = (3,0) & \mod 5 = (3,0) \end{array}$$

In gap wird das durch Multiplizieren mit 1_F erledigt, d.h. (6) wird so überprüft:

$$1_F * (4 * a_4^3 + 27 * a_6^2) \neq 1_F * 0$$

Auch die Koeffizienten in (6) können wir über $\mathbb{Z}_5$ betrachten:

$$\begin{array}{rl} 4 & \mod 5 = 4 \\ 27 & \mod 5 = 2 \end{array}$$

Nun ist aber $4 \cdot 3^3 + 2 \cdot 4^2 \mod 5 = 140 \mod 5 = 0$, also gilt (6) nicht, d.h. mit $a_4 = -7, a_6 = -6, F = \mathbb{Z}_5$ handelt es sich nicht um eine Ellipse (was alle weiteren Rechnungen erübrigt).

Das ergebnis liefert auch unsere Funktion `defines_ellipse`:

Example

```
gap> Z5 := HomalgRingOfIntegers( 5 );
GF(5)
gap> defines_ellipse( -7, -6, Z5 );
false
```

b) $a_4 = -5, a_6 = 5, F = \mathbb{Z}_{11}, P = (4,7)$.

Über $\mathbb{Z}_{11}$ ergibt das

$$\begin{array}{rll} a_4 & \mod 11 = -5 & \mod 11 = 6 \\ a_6 & \mod 11 = 5 & \mod 11 = 5 \\ P & \mod 11 = (4,7) & \mod 11 = (4,7) \end{array}$$

Ebenso die Koeffizienten

$$\begin{array}{rl} 4 & \mod 11 = 4 \\ 27 & \mod 11 = 5 \end{array}$$

Wir stellen fest, $\mathbb{Z}_{11}$ ist ein Körper, nicht von Charakteristik 2 oder 3. Bleibt also (6) zu überprüfen:

$$4 \cdot 6^3 + 5 \cdot 5^2 \mod 11 = 989 \mod 11 = 10 \neq 0 \mod 11.$$

Das ergebnis liefert auch unsere Funktion `defines_ellipse`:

Example

```
gap> Z11 := HomalgRingOfIntegers( 11 );
GF(11)
gap> defines_ellipse( -5, 5, Z11 );
true
```

Nun haben wir also eine elliptische Kurve $E_0(-5,5,\mathbb{Z}_{11})$ und wollen die Anzahl der Elemente bestimmen. Da sie als x-y-Graph eine Teilmenge der Zahlenebene $\mathbb{Z}_{11}^2$ ist, brauchen wir also nur für endlich viele Punkte zu überprüfen, ob sie auf der elliptischen Kurve liegen. Dazu bestimmen wir zunächst die Gleichung der Kurve:

$$y^2 = x^3 + a_4 x + a_6, \tag{7}$$

also

$$y^2 = x^3 + 6x + 5$$

Dazu habe ich eine Funktion `ellipse_membership` geschrieben, die bei Eingabe eines Punktes $xy \in F^2$ und den Werten a_4, a_6 und F ausgibt, ob xy die Gleichung (7) erfüllt.

Diesen Filter kann ich nun auf die Zahlenebene F^2 anwenden, wenn F endlich ist. Zusammen mit der Funktion `defines_ellipse` habe ich dann die Funktion `ellipticCurve` geschrieben, die zuerst überprüft, ob es sich um eine ellipse handelt, dann ob der Körper endlich ist, und dann die Punkte auf der elliptischen Kurve als Teilmenge von F^2 ausgibt:

Example

```
gap> E0 := ellipticCurve( -5, 5, Z11 );
[ [ 0*Z(11), Z(11)^2 ], [ 0*Z(11), Z(11)^7 ], [ Z(11)^0, Z(11)^0 ],
[ Z(11)^0, Z(11)^5 ], [ Z(11), Z(11)^4 ], [ Z(11), Z(11)^9 ],
```

```
[ Z(11)^2, Z(11)^2 ], [ Z(11)^2, Z(11)^7 ], [ Z(11)^3, Z(11) ],
[ Z(11)^3, Z(11)^6 ], [ Z(11)^5, Z(11)^3 ], [ Z(11)^5, Z(11)^8 ],
[ Z(11)^7, Z(11)^2 ], [ Z(11)^7, Z(11)^7 ], [ Z(11)^9, Z(11) ],
[ Z(11)^9, Z(11)^6 ] ]
gap> Length( E0 );
16
```

Wir dürfen aber auch nicht den Fernpunkt $O \in E_0$ vergessen, der das neutrale Element der Gruppe ist. Die Gruppe hat also insgesamt 17 Elemente.

Damit können wir auf zwei Arten überprüfen, ob der Punkt $P = (4,7)$ auf E_0 liegt:

Example

```
gap> P := [ 4, 7 ];
[ 4, 7 ]
gap> ellipse_membership( P, -5, 5, Z11 );
true
gap> One( Z11 )*P in E0;
true
```

c) $a_4 = -43, a_6 = 166, F = \mathbb{R}, P = (3,8)$.

Da $\mathbb{R}$ ein unendlicher Körper ist, gilt $\operatorname{Char}\mathbb{R} = 0$, und wir brauchen auch nicht in Restklassen zu rechnen. Es gilt also wieder nur die Gleichung (6) zu überprüfen, also

$$4 \cdot (-43)^3 + 27 \cdot 166^2 = 425984 \neq 0$$

Statt über $\mathbb{R}$ rechnen wir in gap über den berechenbaren Körper $\mathbb{Q}$, was am Ergebnis nichts ändert:

Example

```
gap> QQ := HomalgFieldOfRationals();
Q
gap> defines_ellipse( -43, 166, QQ );
true
```

Da es sich bei $E_0(-43, 166, \mathbb{R}) \subset (\mathbb{R}^2 \cup \{O\})$ um einen stetigen Graph handelt, ist die Gruppe unendlich groß.

Um einen Punkt P zu sich selbst zu addieren, also ihn zu verdoppeln in der Gruppen-Operation auf E_0 bildet man die Tangente an der elliptischen Kurve durch P. Wenn die Tangente parallel zur y-Achse verläuft, schneidet sie die elliptische Kurve in keinem weiteren Punkt. In diesem Fall gilt dann $P + P = O$ und die Ordnung von P wäre zwei.

Wie man am Graph der Kurve

$$y^2 = x^3 - 43x + 166,$$

sehen kann, ist der äußerst linke Punkt derjenige, an dem $y = 0$ gilt, dann gilt also

$$\begin{aligned} 0 &= x^3 - 43x + 166 \\ \Rightarrow x &\cong -7.9865 \end{aligned}$$

Wir müssen also die Tangente an E_0 im Punkt $P = (x, y)$ berechnen.

Für unseren Punkt $P = (3,8)$ hingegen gilt:

$$P + P =$$