Elementare Zahlentheorie mit Maple (01202) **WS 15/16**

Klausur am 05.03.2016:

Musterlösungen

Aufgabe 1

Da $\mathrm{ggT}(m,n) = 1$ gilt, gibt es ganze Zahlen s,t so dass die Gleichung $1 = sm + tn$ erfüllt ist. Aus $a^m = b^n$ folgt $a = a^{sm+tn} = a^{sm}a^{tn} = b^{sn}a^{tn} = (b^s a^t)^n$. Die Zahl a ist also die n-te Potenz von $b^s a^t$. Da a eine natürliche Zahl ist, muss auch $b^s a^t$ eine natürliche Zahl sein. Wir setzen also $b^s a^t = k, k \in \mathbb{N}$. Daraus folgt $a = k^n$, und dann $b^n = a^m = k^{nm}$, also $b = k^m$.

Aufgabe 2

1. Seien $a, b, c \in \mathbb{Z}$ ungleich 0. Eine ganze Zahl d heißt größter gemeinsamer Teiler von a, b und c, wenn gilt:
   (a) $d \mid a, d \mid b$ und $d \mid c$, und
   (b) wenn $g \in \mathbb{Z}$ ein gemeinsamer Teiler von a, b, c ist, dann gilt $g \leq d$.
2. Wir setzen $d = \mathrm{ggT}(a,b,c), e = \mathrm{ggT}(a,b)$ und $f = \mathrm{ggT}(e,c)$. Da d ein Teiler von a, b und c ist, gilt $d \mid as + bt$ für alle $s, t \in \mathbb{Z}$, insbesondere teilt d die Darstellung $e = as + bt$. Somit gilt $d \mid c$ und $d \mid e$, und damit folgt $d \mid f$. Dann muss $d \leq f$ gelten. Umgekehrt gilt $f \mid e$ und $f \mid c$. Da e ein Teiler von a und b ist, gilt $f \mid a$ und $f \mid b$. Daraus folgt, dass f ein Teiler von a, b und c ist. Damit gilt $f \leq d$, und insgesamt folgt daraus $d = f$.
3. Wir setzen $d = \mathrm{ggT}(\mathrm{ggT}(a,b),c)$ und $e = \mathrm{ggT}(a,b)$. Es gibt ganze Zahlen x, y mit $e = xa + yb$. Ferner gibt es ganze Zahlen v, z mit $d = ve + zc$. Daraus folgt $d = v(xa + yb) + zc = vxa + vyb + zc$. Wir setzen $s = vx, t = vy, u = z$, woraus die Behauptung folgt.

Aufgabe 3

Aus $a \equiv b \,(\mathrm{mod}\, m)$ folgt $m \mid a - b$. Wegen $d \mid m$ folgt dann auch $d \mid a - b$.

⇒: Wir nehmen zunächst an, dass $d \mid a$ gilt. Dann folgt $d \mid a - b - a = -b$. Da b und $-b$ dieselben Teiler besitzen, gilt also $d \mid b$, und es folgt die Behauptung.

⇐: Hier gehen wir analog vor. Wir nehmen also $d \mid b$ an. Dann folgt sofort $d \mid a - b + b = a$.

Damit ist insgesamt die Behauptung gezeigt.

Aufgabe 4

Für alle Primzahlen p gilt $\sigma(p) = p + 1$, $\varphi(p) = p - 1$ und $\tau(p) = 2$. Damit gilt

$$\sigma(p) - \varphi(p) = p + 1 - (p - 1) = 2 = \tau(p).$$

Dann gilt dies auch für die Summe, also $\sum_{p\leq x}\sigma(p)-\sum_{p\leq x}\varphi(p)=\sum_{p\leq x}(\sigma(p)-\varphi(p))=\sum_{p\leq x}\tau(p)$.

Aufgabe 5

Sei d durch 4 teilbar. Für alle $y\in\mathbb{Z}$ ist dy^2 durch 4 teilbar, es gilt also $dy^2\equiv 0(\mathrm{mod}4)$. Sei $x\in\mathbb{Z}$ gerade. Dann gilt $x=2n$ für ein $n\in\mathbb{Z}$, und es folgt $x^2=4n\equiv 0(\mathrm{mod}\,4)$. Damit gilt $x^2-dy^2\equiv 0(\mathrm{mod}4)$. Es ist aber $-1\equiv 3(\mathrm{mod}4)$, und es folgt, dass die Gleichung $X^2-dY^2=-1$ für gerade $x\in\mathbb{Z}$ nicht erfüllbar ist. Ist x ungerade, so gilt $x=2n+1$ für ein $n\in\mathbb{Z}$. Dann ist $x^2=(2n+1)^2=4n^2+4n+1\equiv 1(\mathrm{mod}4)$. Dann ist $x^2-dy^2\equiv 1(\mathrm{mod}4)\not\equiv 3(\mathrm{mod}4)$, und es folgt, dass die Gleichung $X^2-dY=-1$ auch für ungerade x nicht lösbar ist. Somit hat sie keine Lösungen in $\mathbb{Z}$.

Aufgabe 6

Gesucht ist eine Gauß'sche Zahl z mit $z+\bar{z}=8$ und $z\cdot\bar{z}=80$. Wir setzen $z=a+bi$ für $a,b\in\mathbb{Z}$. Dann gilt $\bar{z}=a-bi$ und wir erhalten

$$z+\bar{z}=a+bi+a-bi=2a=8,$$

woraus $a=4$ folgt. Weiter können wir damit aus

$$z\cdot\bar{z}=(4+bi)(4-bi)=16+b^2=80$$

schließen, dass $b^2=64$ gilt, und damit $b=\pm\sqrt{64}=\pm 8$. Wir nehmen $b=8$ und erhalten damit eine Gauß'sche Zahl $z=4+8i$, die die beiden Gleichungen erfüllt. ($z=4-8i$ wäre ebenso richtig.)

Aufgabe 7

Wir legen zunächst die Variablen fest, in denen die Werte a',b' und c' der reduzierten Gleichung gespeichert werden sollen: `aa, bb, cc` sowie eine Variable `d` für den größten gemeinsamen Teiler von a und b. Anschließend wird das Vorliegen der Voraussetzungen einer linearen Diophantischen Gleichung geprüft: a und b sollen nicht beide 0 sein. Nur falls diese Bedingung erfüllt ist, wird der größte gemeinsame Teiler von a und b berechnet. Ist die Bedingung nicht erfüllt, wird die Prozedur durch den Befehl `return("keine Lösung")` mit der übergebenen Ausgabe am Bildschirm verlassen.

Ist der ggT d berechnet, so wird weiter geprüft, ob d ein Teiler von c ist. Nur in diesem Fall hat die übergebene lineare Diophantische Gleichung überhaupt Lösungen, und nur dann kann die reduzierte Gleichung berechnet werden. Ist die Bedinung also erfüllt, so werden $a'=$ `aa`, $b'=$ `bb`, $c'=$ `cc` berechnet sowie die Rückgabe dieser Werte in einer Liste eingeleitet. Ist d nicht Teiler von c, so wird die Prozedur mit der Rückgabe `"keine Lösung"` beendet.

Wir verwenden den Befehl `return()`, da die Koeffizienten der reduzierten Gleichung in einer anderen Prozedur Verwendung finden könnten. Eine mögliche Prozedur könnte folgendermaßen aussehen:

```
DiophantSolution := proc(a::integer, b::integer, c::integer)
# gibt zu einer linearen Diophantischen Gleichung die
# reduzierte Gleichung aus
  local aa, bb, cc, d;
  if (a <> 0) or (b <> 0) then
    d := gcd(a,b);
  else
    return("keine Lösung");
  fi;
  if c mod d = 0 then
    aa := a/d;
    bb := b/d;
    cc := c/d;
    return([aa, bb, cc]);
  else
    return("keine Lösung");
  fi;
end:
```

Die Ausgabe für z. B. $[12, 18, 24]$ lautet:

```
DiophantSolution(12, 18, 24);
```

$$[2, 3, 4] \qquad \textbf{(1)}$$

Aufgabe 8

Zuerst deklarieren wir die benötigten lokalen Variablen `p` für das Produkt der Primzahlen, `L` für die Liste, `i` als Laufvariable der `for`-Schleife und `m` als die Zahl, die auf ihre Primzahleigenschaft hin untersucht wird. Anschließend initialisieren wir `p` mit 1 und `L` als leere Liste. Danach steigen wir in eine `for`-Schleife ein, in der wir zunächst für jedes `i` von 1 bis n nacheinander die Zahlen $\texttt{p} = p_1 \cdots p_i$ und $\texttt{m} = p+1$ berechnen. Mit der `if...then...else`-Abfrage prüfen wir, ob `m` prim ist und fügen in die Liste `L` entweder eine 1 oder eine 0 ein, je nachdem ob die Abfrage wahr oder falsch ist. Nachdem die `for`-Schleife beendet ist, wird die Liste `L` am Bildschirm ausgegeben.

Eine mögliche Prozedur könnte folgendermaßen aussehen:

```
Primzahl := proc(n::posint)
# prüft für alle Zahlen i von 1 bis n, ob
# p_1***p_i + 1 eine Primzahl ist oder nicht
  local p, L, i, m;
  p := 1;
  L := [];
  for i from 1 to n do
    p := p*ithprime(i);
    m := p + 1;
    if isprime(m) then
      L := [op(L), 1];
    else
      L := [op(L), 0];
    fi;
  od;
  print(L);
end:
```

Die Ausgabe für z. B. $n = 20$ lautet:

```
Primzahl(20);
```

$$[1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0] \qquad \textbf{(1)}$$