FernUniversität in Hagen

FernUniversität in Hagen
FAKULTÄT für Mathematik und Informatik

Bitte hier unbedingt Matrikelnummer und Adresse eintragen, sonst keine Bearbeitung möglich.

Postanschrift: FernUniversität ·58084 Hagen

(Name, Vorname)

(Straße, Nr.)

(PLZ, Wohnort)

KLAUSUR zum Kurs Elementare Zahlentheorie mit Maple (01202)
WS 2013/14

DATUM: 01.03.2014
UHRZEIT: 10.00 - 12.00 Uhr
KLAUSURORT:

Bearbeitungshinweise
(Bitte vor Arbeitsbeginn durchlesen!)

1. Schreiben Sie Ihre Klausur bitte nicht mit Bleistift und nicht mit einem roten Stift.
2. Füllen Sie bitte das Adressfeld leserlich und vollständig aus, und schreiben Sie Ihren Namen und Ihre Matrikelnummer auf jedes Lösungsblatt, das Sie abgeben.
3. Die Reihenfolge, in der Sie die Aufgaben/Teilaufgaben lösen, ist Ihnen freigestellt. Kreuzen Sie in der Tabelle (s.u.) an, welche Aufgaben Sie bearbeitet haben.
4. Bei jeder Aufgabe ist die erreichbare Höchstpunktzahl vermerkt. Sie haben die Klausur bestanden, wenn Sie **40** Punkte erreichen.
5. Als Hilfsmittel erlaubt ist ein **beidseitig handbeschriebenes DIN A4-Blatt**.
6. Weitere Hilfsmittel wie Bücher, Taschenrechner, Studienbriefe, weitergehende eigene Aufzeichnungen, (Tablet-)PCs, eBookreader etc. dürfen während der Klausur nicht benutzt werden. Ihre Benutzung sowie andere Täuschungsversuche führen dazu, dass Ihre Klausur mit 5 bewertet wird.

	Bemerkungen:

Aufgabe	1	2	3	4	5	6	7	Summe
Bearbeitet								
max. Punktezahl	8	10	10	10	12	10	20	80
erreichte Punktezahl								
Korrektur								

Prüfergebnis/Note	

Elementare Zahlentheorie mit Maple (01202) WS 2013/14

Klausur am 01.03.2014:

Aufgabenstellungen

Die Lösungen aller Aufgaben müssen Sie begründen.

Aufgabe 1

(a) Definieren Sie bitte den größten gemeinsamen Teiler zweier ganzer Zahlen $a, b \in \mathbb{Z}$.

(b) Verwenden Sie nun den erweiterten Euklidischen Algorithmus, um zwei Zahlen $x, y \in \mathbb{Z}$ zu bestimmen, für die gilt:

$$\mathrm{ggT}(56, 72) = 56x + 72y.$$

[*2 + 6 = 8 Punkte*]

Aufgabe 2

Seien $a, b \in \mathbb{Z}$ mit $\mathrm{ggT}(a, b) = p$, wobei p eine Primzahl ist. Finden Sie alle möglichen Werte für

(a) $\mathrm{ggT}(a^2, b)$

(b) $\mathrm{ggT}(a^2, b^2)$

und begründen Sie Ihre Antwort.

[*5 + 5 = 10 Punkte*]

Aufgabe 3

Seien $x, a, n \in \mathbb{N}$ und sei $n > 1$.

Beweisen Sie: Unter der Voraussetzung $x \equiv a \pmod{n}$ gilt $x \equiv a \pmod{2n}$ oder $x \equiv a + n \pmod{2n}$.

[*10 Punkte*]

Aufgabe 4

Zeigen Sie: Ist $p \in \mathbb{N}$ eine Primzahl, so kann $p^n, n \in \mathbb{N}$, keine vollkommene Zahl sein.

[*10 Punkte*]

Aufgabe 5

Sei (a, b, c) ein primitives pythagoreisches Tripel.

Beweisen Sie: $a + b$ ist entweder zu 1 oder zu 7 kongruent modulo 8.

Hinweis: Sie dürfen ohne Beweis verwenden, dass das Quadrat jeder ungeraden Zahl die Form $8k + 1$ für ein $k \in \mathbb{N}$ hat.

[*12 Punkte*]

Aufgabe 6

Seien $z, w \in \mathbb{Z}[i]$.

Beweisen Sie: Ist $\mathrm{ggT}\big(N(z), N(w)\big) = 1$, dann gilt $\mathrm{ggT}(z, w) = e$, und e ist eine Einheit in $\mathbb{Z}[i]$.

[*10 Punkte*]

Aufgabe 7

Folgende Maple-Prozedur ist gegeben:

```
>  # Die Zeilennummerierung dient zu Ihrer Orientierung

 1.   DoppelListe:=proc(L::list,M::list)
 2.   local N, i;
 3.   if nops(L) = nops(M) then
 4.     N:=[];
 5.     for i from 1 to nops(L) do
 6.       N:=[op(N),L[i],M[i]];
 7.     od;
 8.   else
 9.     return("Listen sind nicht gleich lang");
10.   print(N);
11.   end:
```

(a) Wenn Sie diese Prozedur in Maple eingeben, bekommen Sie eine Fehlermeldung. Wo steckt der Fehler?

(b) Nehmen Sie an, der Fehler in der Prozedur ist beseitigt. Vollziehen Sie die Arbeitsweise der Prozedur an der Eingabe der beiden Listen $[1, 2, 3]$ und $[4, 5, 6]$ nach.

(c) Wie lauten die Elemente `N[2]` und `N[5]` unter der Eingabe aus 2.?

(d) Gegeben ist folgender Beginn einer Prozedur:

```
> Ergebnis:=proc()
  local p, i, L;
  p:=unapply(x^6-24*x^4+72*x+12,x);
  ...
```

Vervollständigen Sie die Prozedur nach folgender Maßgabe:

Die Prozedur soll eine Liste ausgeben, in der alle Funktionswerte $p(x)$ mit $-5 \leq x \leq 2$ enthalten sind.

[*2 + 6 + 4 + 8 = 20 Punkte*]

Anhang

Maple-Befehl	Erläuterung
`#`	Kommentar
`{}`	leere Menge in Maple
`[]`	leere Liste
`!`	Fakultät
`->`	Definition von Abbildungen
`<>`	ungleich, $\neq$
`abs()`	Absolutbetrag der übergebenen Zahl
`add()`	Addition von mehreren Summanden
`chrem()`	Algorithmus des Chinesischen Restsatzes
`conjugate()`	komplex konjugierte Zahl
`divisors()`	Teiler der übergebenen Zahl
`do`	Beginn der Anweisungen einer Schleife
`elif`	Teil der if-then-else-Anweisung
`else`	Teil der if-then-else-Anweisung
`end:`	Ende der Maple-Prozedur
`evalf()`	Auswertung von Ausdrücken mit Konstanten/Funktionen
`even`	gerade ganze Zahl
`factorial()`	Fakultät der übergebenen Zahl
`floor()`	größte ganze Zahl, die kleiner oder gleich ist
`for`	for-Schleife
`GaussInt`	Gauß'sche Zahlen-Programmbibliothek
`I`	imaginäre Einheit
`if`	if-Anweisung
`ifactor()`	faktorisiert die übergebene Zahl
`igcd()`	Berechnung des ggT der übergebenen Zahlen
`igcdex()`	Berechnung des ggT mit erweitertem Euklidischen Algorithmus
`ilcm()`	Berechnung des kgV der übergebenen Zahlen
`Im()`	Imaginärteil der übergebenen komplexen Zahl
`infinity`	Befehl für ∞
`integer`	ganze Zahl
`iquo()`	Quotient bei Division mit Rest
`irem()`	Rest bei Division mit Rest
`isolve()`	ganzzahliges Lösen von Gleichungen
`isprime()`	Primzahltest
`ithprime()`	i-te Primzahl
`local`	lokale Variablen

`minus`	Komplement von Mengen
`mod`	Rest bei Division mit Rest
`mods`	symmetrische Form beim modulo-Rechnen
`msolve()`	modulares Lösen von Gleichungen
`mul()`	Multiplizieren mehrerer Faktoren
`nextprime()`	nächstgrößere Primzahl
`nops()`	Anzahl der Elemente einer Liste
`numtheory`	Zahlentheorie-Programmbibliothek
`od`	Ende der Anweisungen einer Schleife
`odd`	ungerade ganze Zahl
`op()`	Extraktion der Elemente z.B. einer Liste
`phi()`	Eulersche φ-Funktion
`pi()`	Primzahlfunktion
`plot()`	Erstellen von Grafiken
`polynom`	Polynom
`posint`	natürliche Zahl
`prevprime()`	nächstkleinere Primzahl
`print()`	Ausgabe auf dem Bildschirm
`proc()`	Maple-Prozedur
`product()`	Produkt
`rand()`	Erzeugung einer Pseudozufallszahl
`rational`	rationale Zahl
`Re()`	Realteil der übergebenen komplexen Zahl
`return()`	Beendigung der Prozedur und Speicherung des übergebenen Wertes
`round()`	Rundung auf nächste ganze Zahl
`sigma()`	Teilersummenfunktion
`sum()`	Summe
`sum2sqr()`	Summe aus zwei Quadraten der übergebenen Zahl
`tau()`	Teileranzahlfunktion
`then`	Teil der if-Anweisung
`type()`	Datentyp der übergebenen Variable
`unapply()`	Definition von Abbildungen
`union`	Vereinigung von Mengen in Maple
`while`	while-Schleife
`with(GaussInt)`	Aufruf der Programmbibliothek `GaussInt`
`with(numtheory)`	Aufruf der Programmbibliothek `numtheory`