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 2014/15

DATUM: 28.02.2015
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	8	Summe
Bearbeitet									
max. Punktezahl	7	10	10	11	12	10	12	8	80
erreichte Punktezahl									
Korrektur									

Prüfergebnis/Note	

Elementare Zahlentheorie mit Maple (01202) WS 14/15

Klausur am 28.02.2015:

Aufgabenstellungen

Die Lösungen aller Aufgaben müssen Sie begründen.

Aufgabe 1

Seien $a, b, c \in \mathbb{Z}$. Beweisen Sie: Aus $a \mid b$ und $a \mid c$ folgt $a^2 \mid bc$.

[*7 Punkte*]

Aufgabe 2

Sei p eine Primzahl und seien $a, b \in \mathbb{N}$. Beweisen Sie: Aus $p \mid a$ und $p \mid (a^2 + b^2)$ folgt $p \mid b$.

[*10 Punkte*]

Aufgabe 3

Bestimmen Sie den Rest von 15! bei Division durch 17.

(Hinweis: Satz von Wilson.)

[*10 Punkte*]

Aufgabe 4

Bestimmen Sie alle möglichen Formen der Primfaktorzerlegung von $n \in \mathbb{N}$, für die $\tau(n) = 10$ gilt.

[*11 Punkte*]

Aufgabe 5

Beweisen Sie: Ist (a, b, c) ein primitives pythagoreisches Tripel, so gilt $12 \mid ab$.

[*12 Punkte*]

Aufgabe 6

(a) Welche der folgenden Zahlen ist Summe von zwei Quadraten?

(i) 22

(ii) 26

(b) Ist die Zahl 5 eine Gauß'sche Primzahl?

Begründen Sie Ihre Antworten! Insbesondere bei Teilaufgabe (a) bedeutet dies nicht, dass Sie die Lösung durch Ausprobieren finden sollen.

[*3 + 3 + 4 = 10 Punkte*]

Aufgabe 7

Charakterisierung der geraden vollkommenen Zahlen: Eine gerade natürliche Zahl n ist genau dann vollkommen, wenn $n = 2^{p-1}(2^p - 1)$ gilt und $2^p - 1$ eine Mersenne'sche Primzahl ist.

Schreiben Sie eine Prozedur – der **kein** Parameter übergeben werden soll –, die für alle geraden Zahlen von 6 bis einschließlich 500 prüft, ob die Zahl vollkommen ist. Verwenden Sie dazu die angegebene Charakterisierung und **nicht** die σ-Funktion. Alle vollkommenen Zahlen sollen in einer Gesamtliste ausgegeben werden.

[*12 Punkte*]

Aufgabe 8

Gegeben ist folgende Prozedur:

```
klausur1:=proc(n::posint)
  local z, i;
  z := 0;
  for i from 1 to n do
    if isprime(i) then
      z := z + 1;
    fi;
  od;
  print(z);
end:
```

(a) Beschreiben Sie in einem Satz, was diese Prozedur berechnet.

(b) Die Prozedur benötigt viel Rechenzeit, da in der `for`-Schleife jede Zahl auf ihre Primzahleigenschaft hin geprüft wird. Schreiben Sie die Prozedur um, indem Sie eine `while`-Schleife verwenden, die nur die Primzahlen durchläuft.

[*2 + 6 = 8 Punkte*]

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`