IN-G 2015 NT Loes

Ministerium für Bildung und Kultur
SAARLAND

Lösungshinweise: Nur für die Hand der Lehrperson

Schriftliche Abiturprüfung 2015

Fach:	**Informatik**
Kurstyp:	G-Kurs
Bearbeitungszeit:	3 Zeitstunden
Hilfsmittel:	nicht-programmierbarer Taschenrechner
Seitenzahl:	Die Lösungshinweise umfassen mit Deckblatt, Punkteverteilungstabelle und Bewertungstabelle 11 Seiten.

1. Aufgabe (20 Punkte)

1.1 (9,5 Punkte)

1.1.1 (3 Punkte) (Lehrplan: "Strukturiertes Programmieren")

Berechnung von A(1,0):
A(1,0) = A(0,1) = 1 + 1 = 2

Berechnung von A(1,3):
A(1,3) = A(0,(A(1,2)))
= A(0,A(0,A(1,1)))
= A(0,A(0,A(0,A(1,0))))
= A(0,A(0,A(0,A(0,1))))
= A(0,A(0,A(0,2)))
= A(0,A(0,3))
= A(0,4)
= 5

1.1.2 (4 Punkte) (Lehrplan: "Strukturiertes Programmieren")

Java-Code:

```
public static long ack(long m, long n)
{
     if (m == 0) return n + 1;
     if (n == 0) return ack(m - 1, 1);
     return ack(m - 1, ack(m, n - 1));
}
```

PASCAL/Delphi-Code:

```
function ack(m, n: integer): integer;
begin
   if (m = 0) then result:= n+1
   else
   if (n = 0) then result:= ack(m - 1, 1)
   else result:= ack(m - 1, ack(m, n - 1));
end;
```

1.1.3 (2,5 Punkte) (Lehrplan: "Algorithmenanalyse, Grenzen der Berechenbarkeit")

Das **Halteproblem** ist ein Beispiel für einen nicht berechenbaren Algorithmus.
Es handelt sich hierbei um die Frage, ob es einen Algorithmus gibt, der bei Eingabe eines beliebigen Algorithmus zusammen mit beliebigen Eingabedaten entscheiden kann, ob dieser terminiert (also hält) oder nicht.

1.2 (10,5 Punkte) (Lehrplan: "Objektorientiertes Modellieren und Programmieren")

1.2.1 (3,5 Punkte)

TKomplex
realteil: real imaginaerteil: real
gibReal(): real gibImaginaer(): real gibBetrag(): real add(z: TKomplex): TKomplex mult(z: TKomplex): TKomplex

1.2.2 (2 Punkt)

Java-Code:

```
public TKomplex(double re,im)
{
      this.realteil = re;
      this.imaginaerteil = im;
}
```

PASCAL/Delphi-Code:

```
constructor TKomplex.create(re,im: real);
begin
      inherited create();
      realteil:=re;
      imaginaerteil:=im;
end;
```

1.2.3 (3 + 2 Punkte)

Java-Code:

```
public TKomplex mult(TKomplex z)
{
      double a1 = this.realteil;
      double b1 = this.imaginaerteil;
      double a2 = z.gibReal();
      double b2 = z.gibImaginaer();
      TKomplex res = new TKomplex( a1*a2-b1*b2, a1*b2+b1*a2 );
      return res;
}

public double gibBetrag()
{
      double a = this.realteil;
      double b = this.imaginaerteil;
      return Math.sqrt(a*a + b*b);
}
```

PASCAL/Delphi-Code:

```
function TKomplex.mult(z: TKomplex): TKomplex;
var a1, a2, b1, b2:real;
begin
      a1:= realteil;
      b1:= imaginaerteil;
      a2:= z.gibReal();
      b2:= z.gibImaginaer();
      result:= TKomplex.create (a1*a2-b1*b2, a1*b2+b1*a2);
end;

function TKomplex.gibBetrag():real;
begin
      result:= sqrt(sqr(realteil)+sqr(imaginaerteil));
end;
```

2. Aufgabe (20 Punkte)

2.1 (7 Punkte) (Lehrplan: "Automaten und Formale Sprachen")

2.1.1 (2 Punkte)

a) $z_0 \xrightarrow{a} z_1 \xrightarrow{a}$ kein Übergang möglich

$z_0 \xrightarrow{a} z_0 \xrightarrow{a} z_1 \xrightarrow{a}$ kein Übergang möglich

$z_0 \xrightarrow{a} z_0 \xrightarrow{a} z_0 \xrightarrow{a} z_1 \xrightarrow{b} z_2$ kein akzeptierender Zustand

$z_0 \xrightarrow{a} z_0 \xrightarrow{a} z_0 \xrightarrow{a} z_0 \xrightarrow{b}$ kein Übergang möglich → Wort nicht akzeptiert

b) $z_0 \xrightarrow{a} z_1 \xrightarrow{a}$ kein Übergang möglich

$z_0 \xrightarrow{a} z_0 \xrightarrow{a} z_1 \xrightarrow{b} z_2 \xrightarrow{b}$ kein Übergang möglich

$z_0 \xrightarrow{a} z_0 \xrightarrow{a} z_0 \xrightarrow{b}$ kein Übergang möglich → Wort nicht akzeptiert

c) $z_0 \xrightarrow{a} z_1 \xrightarrow{b} z_2 \xrightarrow{a} z_2 \xrightarrow{a} z_3$ akzeptierender Zustand → Wort akzeptiert

d) $z_0 \xrightarrow{b}$ kein Übergang möglich → Wort nicht akzeptiert

2.1.2 (1 Punkt)

$r = a^* a\, b\, a^* a = a^+ b a^+$

2.1.3 (1 Punkte)

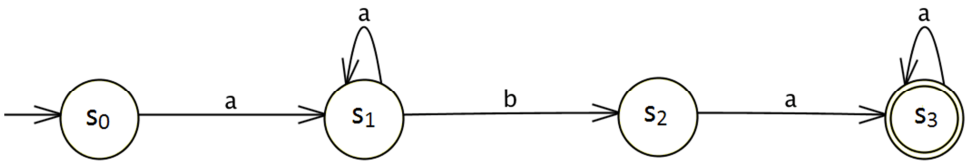
Sowohl im Zustand z_0, als auch im Zustand z_2 gibt es je zwei Übergänge mit dem Eingabezeichen a in verschiedene Folgezustände.

2.1.4 (3 Punkte)

Teilmengenkonstruktion:

δ	a	b
$s_0 := \{z_0\}$	$\{z_0; z_1\} = s_1$	{ }
s_1	s_1	$\{z_2\} = s_2$
s_2	$\{z_2; z_3\} = s_3$	{ }
s_3	s_3	{ }

Übergangsgraph:



2.2 (3 Punkte) (Lehrplan: "Automaten und Formale Sprachen")

2.2.1 (1 Punkt)

Ein Automat, der die Sprache L akzeptiert, müsste zählen, wie viele a's er vor dem b liest, um anschließend die gleiche Zahl an a's nochmals zu lesen. Er muss sich also merken, wie viele a's er schon gelesen hat. Endliche Automaten können sich nur in ihren Zuständen etwas merken. Daher ist die Anzahl der a's, die ein endlicher Automat zählen kann, durch die Anzahl der Zustände begrenzt.

2.2.2 (2 Punkte)

Grammatik G = (N, T, P, S) mit

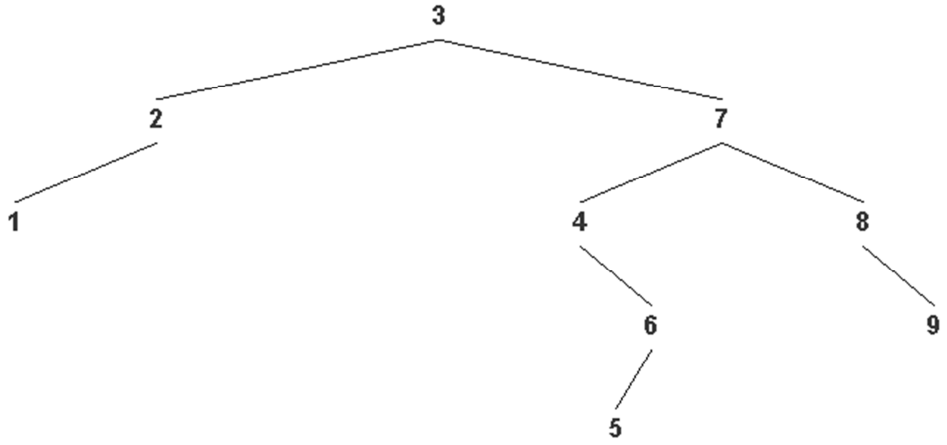
N = {S}

T = {a; b}

P = {S → aSa | aba}

2.3 (10 Punkte) (Lehrplan: "Datentypen und Datenstrukturen")

2.3.1 (2 Punkte)



2.3.2 (2 Punkte)

Maximale Höhe bei linearer Struktur, also bei vorsortiertem Feld. Höhe: 999

Minimale Höhe: 9. Liegt vor, wenn ein annähernd vollständiger Baum entsteht.

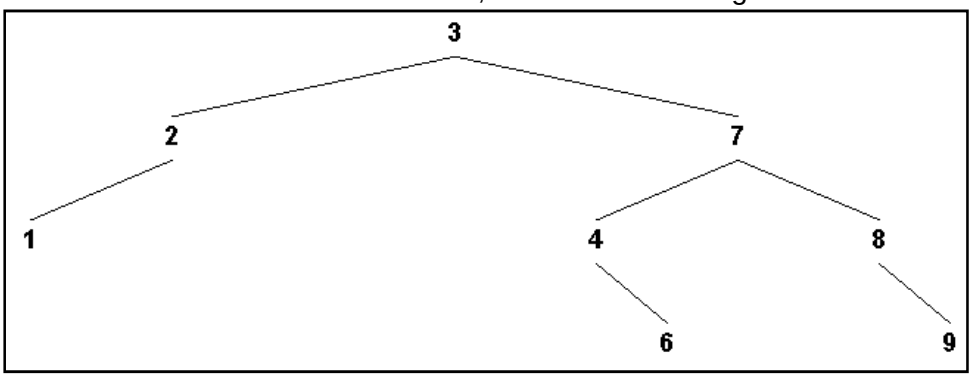
Ein vollständiger Baum der Höhe 9 enthält 2^{9+1}-1=1023 Knoten.

2.3.3 (1 Punkt)

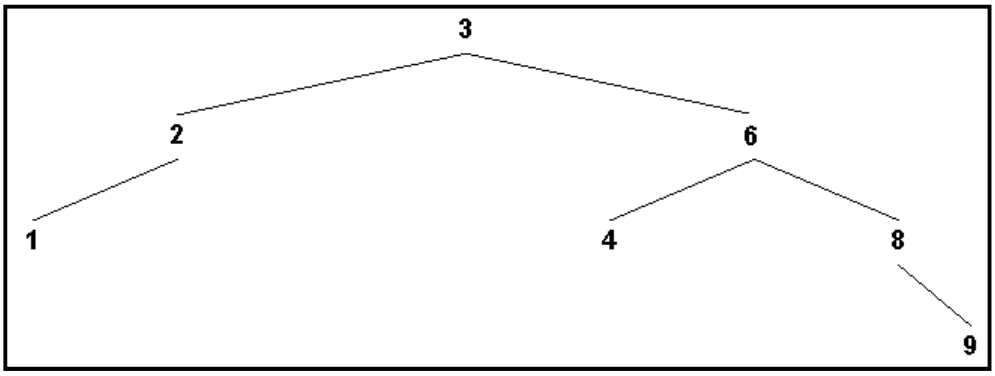
3, 2, 1, 7, 4, 6, 5, 8, 9

2.3.4 (3 Punkte)

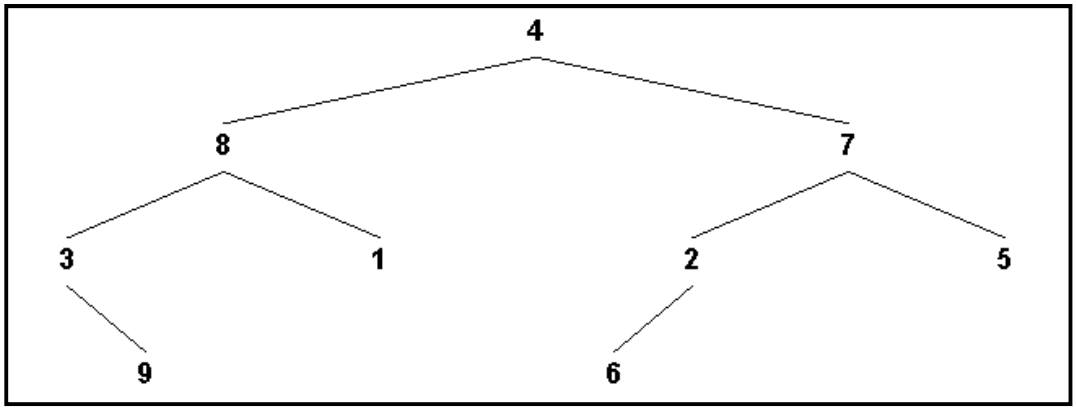
Ist der zu löschende Knoten ein Blatt, so wird er einfach gelöscht.



Besitzt der zu löschende Knoten zwei Kinder, so wird er durch den nächstkleineren Knoten (größter Knoten des linken Teilbaums) oder den nächstgrößeren Knoten (kleinster Knoten des rechten Teilbaums) ersetzt.



2.3.5 (2 Punkte)



3. Aufgabe (20 Punkte)

3.1 (12 Punkte) (Lehrplan: "Kryptographie")

3.1.1 (3 Punkte)

$$192 = 6 \cdot 29 + 18$$
$$29 = 1 \cdot 18 + 11$$
$$18 = 1 \cdot 11 + 7$$
$$11 = 1 \cdot 7 + 4$$
$$7 = 1 \cdot 4 + 3$$
$$4 = 1 \cdot 3 + 1$$
$$1 = 4 - 3$$
$$1 = 4 - (7 - 4)$$
$$1 = 2 \cdot 4 - 7$$
$$1 = 2 \cdot (11 - 7) - 7$$
$$1 = 2 \cdot 11 - 3 \cdot 7$$
$$1 = 2 \cdot 11 - 3 \cdot (18 - 11)$$
$$1 = 5 \cdot 11 - 3 \cdot 18$$
$$1 = 5 \cdot (29 - 18) - 3 \cdot 18$$
$$1 = 5 \cdot 29 - 8 \cdot 18$$
$$1 = 5 \cdot 29 - 8 \cdot (192 - 6 \cdot 29)$$
$$1 = 53 \cdot 29 - 8 \cdot 192$$
$$\Rightarrow 29^{-1}\ mod\ 192 = 53\ mod\ 192$$

3.1.2 (2 Punkte)

$c = 2^{29} \bmod 221 = 32$

3.1.3 (4 Punkte)

Faktorisieren von n durch einfaches Probieren (z.B. Sieb des ERATOSTHENES); es gilt: $n = p \cdot q = 13 \cdot 17$.

Daraus folgt $\varphi(221) = (p-1) \cdot (q-1) = 12 \cdot 16 = 192$.

Aus der Berechnung des modularen Inversen von 29 modulo 192 (vgl. Aufgabe 3.1.1) erhält man unmittelbar den geheimen Schlüssel: (d, n) = (53, 221).

Diese RSA-Verschlüsselung kann relativ leicht gebrochen werden, da n problemlos faktorisiert werden kann. Bei einem hinreichend großen Wert für n ist die Faktorisierung in zwei Primzahlen jedoch praktisch unlösbar, d.h. der Rechenaufwand ist zu groß.

3.1.4 (1 Punkt)

$m = 172^{53} \bmod 221$

3.1.5. (2 Punkte)

Erzeugung (ohne Hashwert gemäß Lehrplan):
Der Absender codiert die Nachricht m mit seinem geheimen Schlüssel d:
$c = m^d \bmod n$

Verifikation:
Der Empfänger decodiert die Nachricht mit dem öffentlichen Schlüssel des Absenders: $m = c^e \bmod n$

3.2 (8 Punkte) (Lehrplan: "Funktionsweise von Computersystemen")

3.2.1 (5 Punkte)

```
0 JMP 4
1 DEF 0 ; x
2 DEF 1 ; Zähler k von 1 bis x
3 DEF 1 ; Quadratzahl von k
4 INM 1 ; Eingabe von x
5 OUT 3 ; Ausgabe des Quadrats von k
6 LDA 3 ; Berechnung der nächsten Quadratzahl
7 ADD 2
8 ADD 2
9 INC
10 STA 3 ; Speichern der nächsten Quadratzahl
11 LDA 2 ; k = k + 1
12 INC
13 STA 2
14 SUB 1
15 JNP 5 ; Wenn x >= k dann wiederhole
16 END
```

3.2.2 (3 Punkte)

Die Eingaben 4 und 3 liefern das Ergebnis 12. Das Programm berechnet allgemein das Produkt von `zahl1` und `zahl2`. Die eingegebene Zahl `zahl2` muss dabei größer als 0 sein.

Punkteverteilungstabelle:

Aufgabe 1:

Teilaufgaben	Summe	Punkte in den Anforderungsbereichen		
		I	II	III
1.1	9,5	3,5	5	1
1.2	10,5	1,5	6	3
	20	**5**	**11**	**4**
		25%	**55%**	**20%**

Aufgabe 2:

Teilaufgaben	Summe	Punkte in den Anforderungsbereichen		
		I	II	III
2.1	7	3	4	
2.2	3		3	
2.3	10	2	6	2
	20	**5**	**13**	**2**
		25%	**65%**	**10%**

Aufgabe 3:

Teilaufgaben	Summe	Punkte in den Anforderungsbereichen		
		I	II	III
3.1	12	1	11	
3.2	8	1	4	3
	20	**2**	**15**	**3**
		10%	**75%**	**15%**

Punkteverteilungstabelle (insgesamt):

Aufgaben	Summe	Punkte in den Anforderungsbereichen		
		I	II	III
1	20	5	11	4
2	20	5	13	2
3	20	2	15	3
	60	**12**	**39**	**9**
		20%	**65%**	**15%**

Bewertungstabelle

Prozent der maximal erreichbaren Roh-punktzahl	Note	Punktzahl
ab 97% bis 100% der max. Punktzahl	sehr gut	15 P
ab 93% bis weniger als 97%		14 P
ab 90% bis weniger als 93%		13 P
ab 85% bis weniger als 90%	gut	12 P
ab 80% bis weniger als 85%		11 P
ab 75% bis weniger als 80%		10 P
ab 70% bis weniger als 75%	befriedigend	09 P
ab 65% bis weniger als 70%		08 P
ab 60% bis weniger als 65%		07 P
ab 55% bis weniger als 60%	ausreichend	06 P
ab 50% bis weniger als 55%		05 P
ab 45% bis weniger als 50%		04 P
ab 38% bis weniger als 45%	mangelhaft	03 P
ab 32% bis weniger als 38%		02 P
ab 25% bis weniger als 32%		01 P
weniger als 25% der max. Punktzahl	ungenügend	00 P

- Ende der Lösungshinweise -