

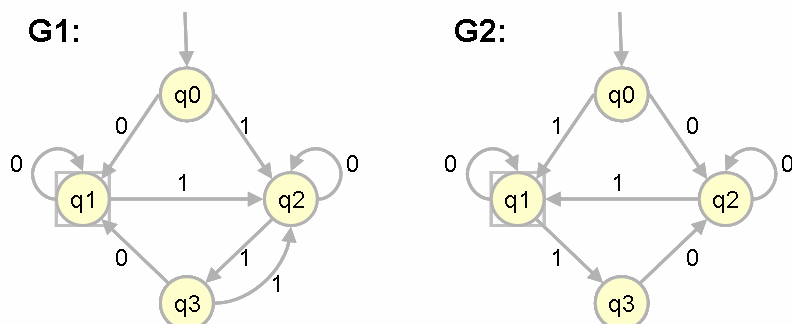
# **Lösungshinweise Nachklausur INFORMATIK – I**

## 1 Wissensfragen und Graphen (14 Punkte)

- a) Welche der folgenden Aussagen sind korrekt? (5 Punkte)  
 Verwenden Sie in folgender Tabelle J für ja und N für nein.  
 Korrekte Antworten werden mit +1 Punkt, falsche Antworten mit -1 Punkt und nicht beantwortete Fragen mit 0 Punkten bewertet.

Jede for-Schleife kann in eine entsprechende while-Schleife umgewandelt werden.	J
Die Binärzahl 110001 entspricht der Dezimalzahl 47	N
Die beiden Automaten G1 und G2 in Teilaufgabe b) sind deterministisch.	J
Ist die aus der Vorlesung bekannte Rechenstruktur NAT termerzeugt?	J
In Java gilt folgende Typhierarchie: byte $\supset$ short $\supset$ int $\supset$ long	N

- b) Gegeben seien folgende zwei Automaten:



- Überprüfen Sie welche der folgenden Worte von den Automaten jeweils erkannt werden.  
 Verwenden Sie in folgender Tabelle J für ja und N für nein.  
 Korrekte Antworten werden mit +0,5 Punkten, falsche Antworten mit -0,5 Punkt und nicht beantwortete Fragen mit 0 Punkten bewertet. (4 Punkte)

	Eingabewort	Erkannt von G1	Erkannt von G2
1.	0100100	J	N
2.	0010101	N	J
3.	1101101	N	J
4.	1010101	N	N

- c) Geben Sie je ein Wort der Länge 4 an, dass nur von G1, nur von G2, von beiden und von keinem der Automaten aus Teilaufgabe b) akzeptiert wird. **(2 Punkt)**

	Erkannt von G1	Erkannt von G2	Eingabewort
1.	J	N	0000, 0110, 1010, 1100
2.	N	J	0001, 0010, 0100, 1000, 1101
3.	J	J	Es gibt keine Eingabeworte für diese Kombination.
4.	N	N	0011, 0101, 0111, 1001, 1011, 1110, 1111

- d) Sind die beiden Automaten G1 und G2 aus Teilaufgabe b) vollständig? **(1 Punkt)**  
Kreuzen Sie die richtigen Antworten an.

G1 ist vollständig.	<input checked="" type="checkbox"/>
G1 ist nicht vollständig.	<input type="checkbox"/>
G2 ist vollständig.	<input type="checkbox"/>
G2 ist nicht vollständig.	<input checked="" type="checkbox"/>

- e) In der folgenden Wertetabelle sind die beiden Funktionen # und ~ beschrieben. Sind die beiden Funktionen strikt? **(2 Punkte)**  
Kreuzen Sie die richtigen Antworten an.

a	b	a # b	a ~ b
0	0	0	⊥
0	1	1	⊥
0	⊥	⊥	0
1	0	1	⊥
1	1	1	⊥
1	⊥	⊥	1
⊥	0	⊥	0
⊥	1	⊥	1
⊥	⊥	⊥	⊥

Die Funktion # ist strikt.	<input checked="" type="checkbox"/>
Die Funktion # ist nicht strikt.	<input type="checkbox"/>
Die Funktion ~ ist strikt.	<input type="checkbox"/>
Die Funktion ~ ist nicht strikt.	<input checked="" type="checkbox"/>

## 2 Termersetzung (10 Punkte)

Gegeben sei der Markov-Algorithmus M über den Zeichenvorrat  $V = \{a, b\}$  und folgender Regelmenge T:

$$T = \left\{ \begin{array}{ll} ba \rightarrow ab & (1) \\ aa \rightarrow \varepsilon & (2) \\ bb \rightarrow \varepsilon & (3) \\ a \rightarrow a & (4) \\ b \rightarrow b & (5) \\ \varepsilon \rightarrow \cdot \varepsilon & (6) \end{array} \right\}.$$

- a) Wenden sie den Markov-Algorithmus M auf die Eingabewörter ba und abaaba an. Geben Sie bei jeder Regelanwendung die verwendete Regel (1) - (6) an. **(2 Punkte)**

ba  $\xrightarrow{1}$  ab  $\xrightarrow{4}$  ab ... (terminiert nicht) (ohne diese Angabe -0,5)

abaaba  $\xrightarrow{1}$  aababa  $\xrightarrow{1}$  aaabba  $\xrightarrow{1}$  aaabab  $\xrightarrow{1}$  aaaabb  $\xrightarrow{2}$  aabb  $\xrightarrow{2}$  bb  $\xrightarrow{3}$   $\varepsilon$   $\xrightarrow{6}$   $\cdot \varepsilon$

Abzug: -1 Pkt. für falsche Regelreihenfolge od. fehlende Regeln.

-0,5 Pkt. für fehlende Regelnummern,

- b) Geben Sie die Menge L1 aller Eingabewörter  $S \subset V^*$  an, für welche M mit dem Ergebnis  $\varepsilon$  terminiert. Sie dürfen die Menge L1 auch informell beschreiben. **(2 Punkte)**

$L1 = \{v \in V^* : \text{in } v \text{ ist sowohl die Anzahl der } a \text{ als auch die Anzahl der } b \text{ gerade}\}$

- c) Für welche Menge L2 von Eingabwörtern terminiert M, wenn die Regel  $aa \rightarrow \varepsilon$  entfernt wird? Begründen Sie kurz Ihre Antwort. **(2 Punkte)**

$L2 = \{b^{2n} : n \in \mathbb{N}\}$

Begründung:

Die Regel  $bb \rightarrow \varepsilon$  ist die einzige, die zur Terminierung führen kann und durch keine Regel wird ein a vernichtet. Somit muss  $L2 = \{b^{2n} : n \in \mathbb{N}\}$  gelten

Hinweis:

- c) Wenn das Eingabewort ein a enthält, terminiert der Algorithmus wegen der Regel  $a \rightarrow a$  nicht mehr. Die Regel  $bb \rightarrow \varepsilon$  ist die einzige, die zur Terminierung führen kann. Also terminiert der Algorithmus für die Wortmenge  $S' = \{b^{2n} : n \in \mathbb{N}\}$ .

d) Geben Sie einen Markov-Algorithmus M2 an, der eine gegebene Zeichenfolge aus 0 und 1 mit einer Länge  $\geq 0$  invertiert, d.h. jedes Zeichen 0 wird zu 1 und jede 1 wird zu 0. **(2 Punkte)**

Beispiel:

Das Wort 111001 wird zu 000110.

Die Terminierung des Algorithmus soll durch eine haltende Regel erfolgen!

$$V = \{0, 1\}$$
$$T = \{ \alpha 0 \rightarrow 1\alpha$$
$$\quad \alpha 1 \rightarrow 0\alpha$$
$$\quad \alpha \rightarrow .\varepsilon$$
$$\quad \varepsilon \rightarrow \alpha \}$$

e) Geben Sie einen Markov-Algorithmus M3 an, der die Zeichen der Eingabeworte über dem Zeichenvorrat  $\{a, b, c\}$  nach der Relation  $B = \{a < b, a < c, b < c\}$  sortiert. **(2 Punkte)**

Beispiel:

Eingabewort = bcabaa

Ausgabewort = aaabbc

Die Terminierung des Algorithmus soll durch eine haltende Regel erfolgen!

$$V = \{a, b, c\}$$
$$T = \{ ba \rightarrow ab$$
$$\quad ca \rightarrow ac$$
$$\quad cb \rightarrow bc$$
$$\quad \varepsilon \rightarrow .\varepsilon \}$$

### 3 Java-Programm: Verständnis (10 Punkte)

- a) Welcher Wert wird durch die Zuweisung in method1() an der Stelle `//*1*` aus `y` berechnet? Was berechnet die Methode `method1`? **(3 Punkte)**

**Hinweis:**

Berücksichtigen Sie den in der Methode verwendeten Datentyp!

```
static long method1(long x){
    long q = 0;
    long y = x;
    long h;
    while (y > 0){
        h = y - (y / 10) * 10;           // *1*
        q = q + h;
        y = y / 10;
    }
    return l;
}
```

`// *1*:` `h = y mod 10.`

`method1(x)` = die Quersumme von `x`

- b) Welche Ausgabe wird durch den Methodenaufruf `method2(x)` erzeugt? Wie sieht die Ausgabe beim Aufruf von `method2(3)` aus? **(3 Punkte)**

```
static method2(int x){
    if (x<1) {
        Out.println(x);
    }
    else {
        method2(x - 1);
        Out.println("1");
    }
}
```

`method2(x)` = `01x` für `x >= 1`, Zeichen untereinander  
`x` für `x < 1`

`method2(3)` =  
 0  
 1  
 1  
 1

- c) Was berechnet die Methode `method3()`?  
Welche Ausgabe erzeugt das Programm?

**(4 Punkte)**

```
public class TestProgram{

    static long method3(double x){
        // assertion x >= 0
        double y = x / 10;
        if (y < 10) return 1;
        else return 1 + method3(y);
    }

    public static void main(String[] args){
        double x = 1234.123;
        Out.print(x);
        Out.print(method3(x));
    }
}
```

<pre>method3(x) :    log(x) - 1 abgerundet    für x &gt; 10                 1                      sonst                 Alternativ:                 (Die Anzahl der Stellen vor dem Komma von x) - 1 für x &gt; 10                 1                      sonst</pre>
--

Ausgabe des Programms:  
1234.1233

## 4 EBNF und Grammatiken (10 Punkte)

- a) Beschreiben Sie mit Hilfe einer EBNF die Menge der Palindrome auf dem Alphabet  $\{a, b, c\}$ , deren mittleres Zeichen ein  $c$  ist. **(2 Punkte)**

Zur Erinnerung:

Ein Palindrom  $P$  ist ein Wort, welches von vorne und hinten gelesen gleich ist.

Beispiele:

abcba oder bcacacb

$P = "a" P "a" | "b" P "b" | "c" P "c" | "c"$ .

- b) Gegeben sei folgende Chomsky-Grammatik  $G$ :

$G = (\{0, 1, +\}, \{S, A, B, C\}, P, S)$

$P = \{$

- $S \rightarrow A | +A \quad (1)$
- $A \rightarrow 0B | 0 \quad (2)$
- $B \rightarrow 1C \quad (3)$
- $C \rightarrow 1C | A \quad (4)$

Liegen die folgenden Wörter in der durch  $G$  beschriebenen Sprache?

Wenn ja, geben Sie jeweils die Ableitung unter Angabe der verwendeten Ableitungsregeln an. **(3 Punkte)**

+001	Nein
010	Ja, $S \rightarrow^1 A \rightarrow^2 0B \rightarrow^3 01C \rightarrow^4 01A \rightarrow^2 010$
+101	Nein
01110	Ja, $S \rightarrow^1 A \rightarrow^2 0B \rightarrow^3 01C \rightarrow^4 011C \rightarrow^4 0111A \rightarrow^2 01110$

- c) Geben Sie eine EBNF-Regel an, welche die gleiche Sprache wie die Grammatik  $G$  aus Teilaufgabe b definiert. **(3 Punkte)**

Wort =  $["+" ] \{ "01" \}^* "0"$ .



## 5 Java-Programm: Kalender (9 Punkte)

- a) Ergänzen Sie die Methode `countDaysOfMonth(int month, int year)`, damit sie die Anzahl der Tage des übergebenen Monats `month` im Jahr `year` zurückgibt.  
Hierbei gilt für den Februar folgende (vereinfachte) Regel:  
Wenn die Jahreszahl durch 4 teilbar ist, hat der Februar 29 Tage.

Die Monate werden durchnummeriert (Januar entspricht 1, Februar entspricht 2, ...). Wenn die Zahl `month` nicht zwischen 1 und 12 einschließlich liegt, soll 0 zurückgegeben werden. Verwenden Sie für diese Methode zur Fallunterscheidung mindestens eine `switch`-Anweisung. **(4 Punkte)**

Hinweise:

- Bei dieser Aufgabe dürfen Sie den Modulo-Operator `%` verwenden. Dieser Operator liefert den Rest einer ganzzahligen Division zurück.  
z.B. `9 % 4 == 1`
- Kommentieren Sie Ihr Programm entsprechend den Vorgaben aus der Vorlesung.

```
static int countDaysOfMonth(int month, int year){
    switch (month) {
        case 1: case 3: case 5: case 7: case 8: case 10: case 12:           // all months with 31 days
            return 31;
        case 4: case 6: case 9: case 11:                                 // all months with 30 days
            return 30;
        case 2:                                                         // for February we need to calculate
            if ((year % 4) == 0) {                                       // if it is a leap year return 29
                return 29;
            }
            else {                                                       // else return 28
                return 28;
            }
        default:                                                         // if month is not between 1 and 12
            return 0;                                                   // return 0
    }
}
```

- b) Ergänzen Sie die Methode `weekDay(int dayOfYear, int firstWeekDay)`, damit sie als Rückgabewert den Wochentag des übergebenen Tages (`dayOfYear`) ausgibt. **(3 Punkte)**

`firstWeekDay` ist eine Zahl zwischen 0 und 6, die den ersten Wochentag des Jahres angibt. 0 steht dabei für Montag, 1 für Dienstag, etc.

`dayOfYear` wird als Zahl zwischen 1 und 365 übergeben.

Beispiel:

1. Januar : `dayOfYear` = 1

2. Januar : `dayOfYear` = 2

etc.

Hinweise:

- Wie in Teilaufgabe a) dürfen Sie den Modulo-Operator `%` verwenden.
- Kommentieren Sie Ihr Programm entsprechend den Vorgaben aus der Vorlesung.

```
static int weekDay(int dayOfYear, int firstWeekDay){  
  
    return ((dayOfYear + firstWeekDay - 1) % 7) // add firstWeekDay as offset to the dayOfYear  
                                                // as dayOfYear starts with 1 and the weekdays with 0 we need to  
                                                // subtract 1  
                                                // afterwards we apply modulo 7 to get the weekday  
  
    alternativ:  
  
    int temp = (dayOfYear - 1) % 7;           // apply modulo 7 on dayOfYear - 1  
    temp = temp + firstWeekDay;             // add the weekday as "offset"  
    temp = temp % 7;                        // as temp might be bigger than 6 we need to modulo again  
    return temp;                            // return the weekday  
}
```

- c) Die folgende Methode `mystique()` verwendet die Methoden aus den vorangegangenen Teilaufgaben. Was berechnet diese Methode? **(2 Punkte)**

```
static int mystique(int year, int month, int day, int firstWeekDay){  
    int d = day;  
    for (int i = 1; i < month; i++){  
        d = d + countDaysOfMonth(i, year);  
    }  
    return weekDay(d, firstWeekDay);  
}
```

`mystique()` : berechnet den Wochentag des angegebenen Datums.

## 6 Rechenstrukturen (7 Punkte)

Gegeben sei die aus der Vorlesung bekannte Rechenstruktur INT und einige ihrer Funktionen:

Menge S der Sorten:  $S = \{\text{bool}, \text{int}\}$

Menge F einiger Funktionssymbole:  $F = \{\text{true}, \neg, \wedge, \text{zero}, \text{succ}, \text{add}, \leq\}$   
mit den Funktionalitäten

fct zero = int  
fct succ = (int) int

fct pred = (int) int  
fct add = (int, int) int

Wir erweitern die Rechenstruktur INT folgendermaßen zu der Rechenstruktur **K** der Punkte in einem kartesischen Koordinatensystem mit ganzzahligen Koordinaten: Die Menge S der Sorten der Rechenstruktur **K** sei gegeben durch  $S_K = \{\text{bool}, \text{int}, \text{point}\}$

Zusätzlich zu den Funktionssymbolen der Rechenstruktur INT besitze die Rechenstruktur **K** die Symbole  $F_K = F_{\text{INT}} \cup \{\text{make}, \text{projX}, \text{projY}, \text{moveto}\}$

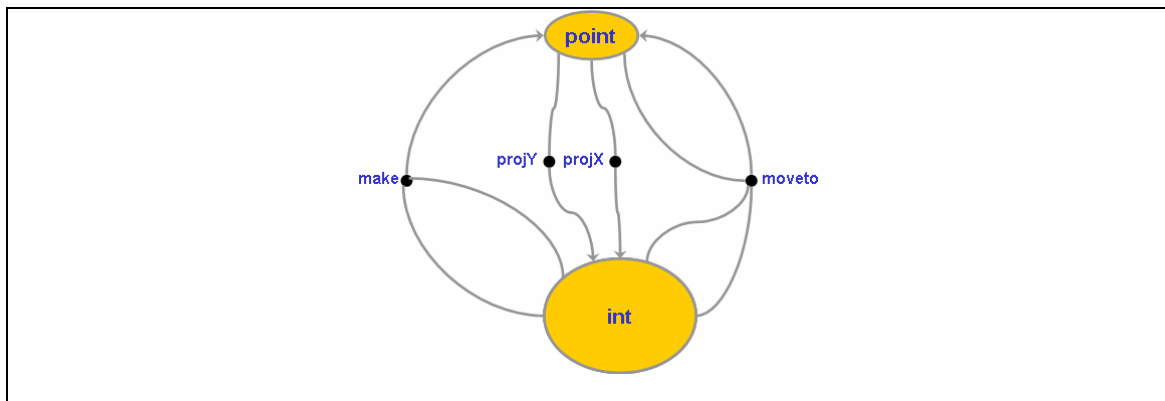
mit den Funktionalitäten

fct make = (int, int) point  
fct projX = (point) int  
fct projY = (point) int  
fct moveto = (point, int, int) point

- a) Geben Sie das Signaturdiagramm der Rechenstruktur **K** für die vier zusätzlichen Funktionssymbole make, projX, projY und moveto an. **(2 Punkte)**

Hinweis:

Die 6 Funktionssymbole der Rechenstruktur INT sollen im Signaturdiagramm dieser Teilaufgabe weggelassen werden!



- b) Welche der folgenden Terme sind bzgl. der Rechenstruktur  $\mathbf{K}$  mit allen 8 Funktionssymbolen syntaktisch korrekt? **(5 Punkte)**

Kreisen Sie die fehlerhafte Stelle ein.

Verwenden Sie in folgender Tabelle J für ja und N für nein.

Korrekte Antworten werden mit +1 Punkt, falsche Antworten mit -1 Punkt und nicht beantwortete Fragen mit 0 Punkten bewertet.

make(succ(zero), projX(make(zero, succ(succ(zero))))))	J
moveto(make(pred(succ(zero)) [PARAMETER] ), succ(succ(zero)), succ(zero))	N
make(pred(projX(make(succ(zero), zero))), pred(add(succ(zero), zero)))	J
succ(projX(make(projX(zero), succ(succ(zero))))))	N
moveto(makepoint(succ(succ(zero)), succ(zero)), add(succ(zero), zero), zero)	N