



Informatik I

Hauptklausur WS 06/07

Vorname:

Name:

Matrikelnummer:

Zur Klausur sind keine Hilfsmittel zugelassen. Die Bearbeitungszeit beträgt 60 Minuten. Bitte tragen Sie Ihren Namen und Ihre Matrikelnummer auf dieser und allen folgenden Seiten ein.

Schreiben Sie Ihre Lösungen in den jeweils für die entsprechende Aufgabe vorgesehenen Kasten. Die Größe des Kastens steht nicht zwangsweise in Zusammenhang mit dem Umfang der für volle Punktzahl erforderlichen Antwort. Sollte der Platz im Kasten nicht ausreichen, können Sie die Rückseite des jeweiligen Blattes verwenden. Vermerken Sie im Kasten, wo der Rest Ihrer Antwort zu finden ist.

Für alle Programmieraufgaben gilt, dass Sie Variablennamen, Kommentare, etc. wahlweise in deutscher oder in englischer Sprache angeben dürfen. Ansonsten gelten die aus der Vorlesung bekannten Programmierrichtlinien.

Die Klausur ist komplett und geheftet abzugeben. Sie dürfen die Rückseiten der Aufgabenblätter als Konzeptpapier benutzen. Verwenden Sie kein eigenes Papier. Sollten Sie zusätzliches Papier benötigen, so fordern Sie dieses bei der Klausuraufsicht an.

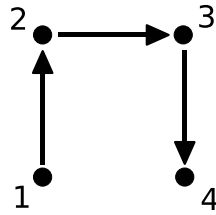
Die unten stehende Tabelle sowie das Summenfeld unten auf jeder Seite werden von uns bei der Korrektur ausgefüllt.

Aufgabe	1	2	3	4	5	6	7	Summe
Maximal	10	9	6	8	10	7	10	60
Erreicht								

Note:

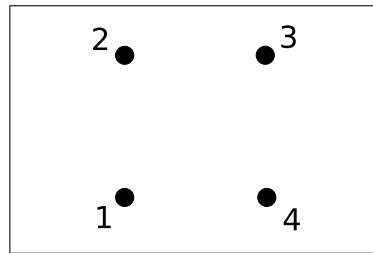
1 Graphen (10 Punkte)

a) Gegeben sei folgender Graph:

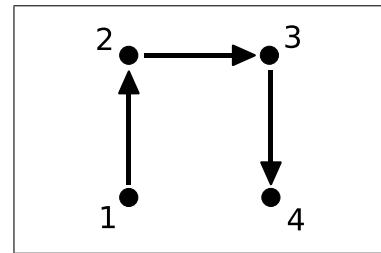


Geben Sie die zugehörige Adjazenzmatrix an: (1 Punkt)

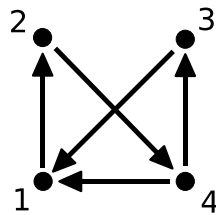
Zeichnen Sie die Kanten des dualen Graphen ein: (1 Punkt)



Ergänzen Sie den Graphen zur reflexiven transitiven Hülle: (2 Punkte)



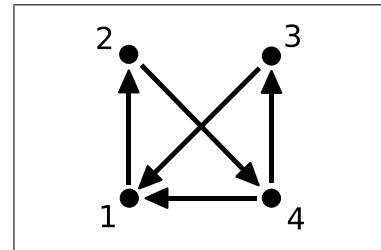
b) Gegeben sei nun folgender Graph:



Der Graph enthält einen Hamiltonschen Kreis. Geben Sie diesen an: (1 Punkt)

Geben Sie alle Knoten an, die von Knoten 4 erreichbar sind: (1 Punkt)

Ergänzen Sie Kanten, so dass ein ungerichteter Graph entsteht: (1 Punkt)



Kreuzen Sie an: (1 Punkt, falsche Antwort führt nicht zu Abzug)

Der Graph enthält einen Eulerschen Weg: ja nein

c) Gesucht ist ein gerichteter Baum:

- Der Baum besteht aus 7 Knoten.
- Er hat 4 Blätter.
- Die Wurzel hat den Ausgangsgrad 2.
- Alle Knoten, die keine Blätter sind, haben verschiedenen Ausgangsgrad.

Zeichnen Sie den Baum in den nebenstehenden Kasten. (2 Punkte)

2 Imperative Programmierung (9 Punkte)

- a) Das folgende Java-Programm soll die Summe der Zahlen von 1 bis einschließlich 10 berechnen und anschließend ausgeben. Das Programm enthält Fehler. Korrigieren Sie die Fehler im Quellcode. Geben Sie außerdem zu jedem Fehler die Ursache in Form eines Java-Kommentars an. Korrigieren Sie jedoch eine Stelle im Programm, die nicht fehlerhaft war, führt dies zu Punktabzug. (5 Punkte)

```
class Sum {
    private static void main(String[] args) {
        string text = "sum: ";
        int sum = 1;
        for (i = 1; i < 10; i++) {
            sum += i;
        }
        Out.println(text, sum);
    }
}
```

- b) Nennen Sie zwei aus der Vorlesung bekannte Datentypen in Java, die zum Arbeiten mit Zeichenfolgen gut geeignet sind. (1 Punkt)

- c) Gegeben sei die folgende Methode in Java: (3 Punkte)

```
static int mystery(int x, int y) {
    int r;
    do {
        r = x % y;
        x = y;
        y = r;
    } while (r != 0);
    return x;
}
```

Geben Sie an, was folgende Aufrufe als Ergebnis zurückliefern:

mystery(6, 9)

mystery(28, 12)

Was berechnet die Methode für zwei Zahlen $x, y \in \mathbb{N}$?

3 Boolesche Algebra (6 Punkte)

Gegeben sei der Term $\mathcal{C}(a \wedge (a \vee b))$

a) Geben Sie den zugehörigen Kantorowitsch-Baum an. (2 Punkte)

b) Geben Sie den Term in Postfix-Schreibweise an. (1 Punkt)

c) Füllen Sie die Wertetabelle aus: (1 Punkt)

a	b	$\mathcal{C}(a \wedge (a \vee b))$
O	O	
O	L	
L	O	
L	L	

d) Geben Sie den Term in konjunktiver Normalform an. (Rechnung ist nicht verlangt.) (2 Punkte)



4 Chomsky–Grammatiken (8 Punkte)

Gegeben sei folgende Grammatik G_1 :

$G_1 = (\Sigma, N, P, A)$ mit $\Sigma = \{a, b, c, d, e\}$, $N = \{A, X, Y, Z\}$

$$P = \left\{ \begin{array}{l} A \rightarrow XY \\ X \rightarrow aXb|c \\ Y \rightarrow dY|Z \\ Z \rightarrow e \end{array} \right\}$$

a) Geben Sie den maximalen Chomsky–Typ der Grammatik an. (1 Punkt)

b) Leiten Sie das Wort $aacbbe$ Schritt für Schritt aus dem Axiom ab. (2 Punkte)

c) Geben Sie die von der Grammatik erzeugte Sprache $L(G_1)$ an. (1 Punkt)

d) Kreuzen Sie Zutreffendes an: (1 Punkt)

Es existieren Grammatiken vom Typ

CH-0 CH-1 CH-2 CH-3

welche die selbe Sprache wie G_1 erzeugen.

e) Gegeben sei nun der folgende reguläre Ausdruck: $a(a + b)^*c$

Geben Sie eine Grammatik G_2 maximalen Chomsky–Typs an, welche die durch den regulären Ausdruck definierte Sprache erzeugt. Geben Sie den Chomsky–Typ Ihrer Grammatik an. (3 Punkte)

5 Objektorientierte Programmierung (10 Punkte)

Gegeben sei nebenstehendes Klassendiagramm. Die Klasse `Month` modelliere einen Monat. Das Attribut `name` beschreibe den Monatsnamen, `numDays` die Anzahl der Tage des Monats und `avgTemp` die Durchschnittstemperatur des Monats. Sämtliche Attribute seien privat.

Month
<code>name: String</code>
<code>numDays: int</code>
<code>avgTemp: float</code>
<code>getName(): String</code>
<code>setName(s: String)</code>
<code>getNumDays(): int</code>
<code>setNumDays(n: int)</code>
<code>calcAvgTemp(t: float[])</code>

- a) Implementieren Sie die Klasse mit ihren Attributen sowie den im Diagramm angegebenen get- und set-Methoden in Java. Die Methode `calcAvgTemp()` implementieren Sie erst im nächsten Aufgabenteil. (5 Punkte)



Name:

Matrikelnummer:

- b) Die Methode `calcAvgTemp()` bekommt als Parameter ein Array mit Elementen vom Typ `float` übergeben. Nehmen Sie an, dass dieses Array für jeden Tag des Monats einen Temperaturwert enthält. Die Methode soll nun den Durchschnitt dieser Temperaturwerte berechnen und dann dem Attribut `avgTemp` zuweisen. Implementieren Sie die Methode `calcAvgTemp()` in Java. (3 Punkte)

- c) Ergänzen Sie untenstehende `main()`-Methode so, dass sie ein Objekt vom Typ `Month` erzeugt und dann dem Attribut `name` den Text "January" und dem Attribut `numDays` den Wert 31 zuweist. Verwenden Sie dazu die `set`-Methoden. (2 Punkte)



Name:

Matrikelnummer:

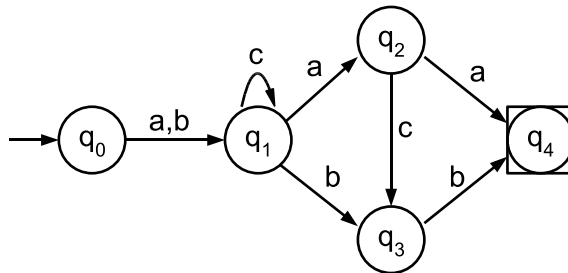
```
public static void main(String[] args) {
```

```
}
```



6 Akzeptoren (7 Punkte)

Gegeben sei folgender Akzeptor über dem Zeichenvorrat $\Sigma = \{a, b, c\}$:



a) Geben Sie vier verschiedene Wörter aus Σ^* an, die der Akzeptor akzeptiert. (1 Punkt)

b) Geben Sie einen regulären Ausdruck für die Sprache des Akzeptors an. (1 Punkt)

c) Formulieren Sie eine EBNF-Regel, mit der jedes Wort aus der Sprache des Akzeptors aus dem Nicht-terminal *Wort* abgeleitet werden kann. Worte, die der Akzeptor nicht akzeptiert, sollen nicht abgeleitet werden können. (1 Punkt)

d) Kreuzen Sie an: (pro richtiges Kreuz 0,5P; pro falsches Kreuz -0,5P; insgesamt nicht weniger als 0P)

Der Akzeptor ist deterministisch: ja nein

Der Akzeptor ist vollständig: ja nein

e) Untenstehende Methode soll den Zustand q_1 in Java implementieren. Die Methode ist unvollständig. Ergänzen Sie die Methode durch Eintragen der richtigen Anweisungen auf den Linien. (3 Punkte)

```

static void q1() {

    char c;

    Out.print("Input: ");
    c = In.readChar();

    _____ {

        case 'a': _____

        case 'b': _____

        _____ : q1();

    }
}
  
```



7 Verständnis- und Wissensfragen (10 Punkte)

a) Kreuzen Sie jeweils an, ob die Aussage wahr oder falsch ist. (6 Punkte)

Hinweis: Korrekte Antworten werden mit 0,5 Punkten, falsche Antworten mit einem Abzug von 0,5 Punkten und nicht beantwortete Fragen mit 0 Punkten bewertet. Sollte daraus in der Summe ein negatives Ergebnis für die Teilaufgabe a) resultieren, wird die Teilaufgabe mit 0 Punkten bewertet.

Aussage	wahr	falsch
Jede durch einen regulären Ausdruck beschriebene Sprache kann auch in EBNF beschrieben werden.		
Das Problem, dass Daten und Befehle dem Prozessor von Speicher und Bus nicht schnell genug bereitgestellt werden können, bezeichnet man als „Zuse’schen Flaschenhals“.		
Ein Byte besteht aus 32 Bit.		
Um in Java einer Variablen vom Typ <code>double</code> einen Wert vom Typ <code>int</code> zuzuweisen, ist keine explizite Typumwandlung (<code>type cast</code>) erforderlich.		
Im Gegensatz zu Speicherseiten müssen Speichersegmente alle die selbe Größe haben.		
Eine Klasse kann in Java beliebig viele Schnittstellen (Interfaces) implementieren.		
Grammatiken vom Typ Chomsky-2 bezeichnet man als kontextsensitiv.		
Markov-Algorithmen sind nichtdeterministisch.		
Von einer abstrakten Klasse lassen sich keine Unterklassen ableiten.		
Eine Variable ist lebendig, sobald sie Speicherplatz belegt.		
Bei einer Load-Store-Architektur können sämtliche Operationen mit Operanden im Speicher arbeiten.		
Das Betriebssystem wird grundsätzlich im nicht privilegierten Modus ausgeführt, damit die Anwendungen vor dem Betriebssystem geschützt sind.		

b) Nennen Sie die beiden Eigenschaften von Speicherzugriffsmustern, die ein Cache ausnützt. (1 Punkt)

c) Nennen Sie eine Schedulingstrategie, bei der die mittlere Wartezeit möglichst gering ist. (1 Punkt)

d) Gegeben sei ein Stapel (Stack). Darauf werden nacheinander folgende Operationen ausgeführt:

`push(3); push(5); push(7); a=pop(); push(9); b=pop(); c=pop(); push(a+pop()); d=pop();`

Geben Sie an, welche Werte die Variablen nach Ausführung der letzten Operation haben. (2 Punkte)

a:

b:

c:

d: