

Vorname

Nachname

Matrikel-Nr

Schreiben Sie jede Lösung möglichst auf die Vorderseite eines *neuen* Blattes (und lassen Sie die Rückseiten Ihrer Lösungsblätter *leer*).

**Aufgabe 1 (16 Punkte):** Geben Sie eine (kontextfreie) Grammatik an für die Menge aller Zahlen im 5-er-System ("Pentalzahlen"), die (ohne Rest) durch 4 teilbar sind.

**Aufgabe 2 (16 Punkte):** In einer neuen Programmiersprache gelten folgende Regeln:

Ein *Bezeichner* muss mit einem großen oder kleinen Buchstaben beginnen und darf ansonsten nur kleine Buchstaben enthalten.

Beispiele für solche Bezeichner: a B carl Carl summe

Gegenbeispiele: CARL zwischenSumme summe03

Ein (Ganzzahl-) *Literal* muss mit einer Ziffer zwischen 1 und 9 beginnen und darf ansonsten nur Ziffern zwischen 0 und 9 enthalten. Ausserdem soll 0 als Literal erlaubt sein.

Beispiele für solche Literale: 0 1 9 30 12300 2737475 102030

Gegenbeispiele: 00 0123 007

Eine *BL-Folge* besteht aus Bezeichnern und Literalen, die durch Kommas voneinander getrennt sind. Eine BL-Folge muss aus mindestens einem Bezeichner oder Literal bestehen (d.h. eine BL-Folge darf nicht leer sein), darf ansonsten aber beliebig viele Bezeichner und/oder Literale enthalten. **Achtung:** Eine Komma darf (und muss) nur *zwischen zwei Elementen* einer Folge stehen. Nach dem letzten Element einer Folge ist ein Komma *nicht* erlaubt.

Beispiele für BL-Folgen (hier beginnt jede Folge auf einer neuen Zeile):

```
carl
12300
carl, 12300
a, B, 0, summe, 12300, Carl
```

Gegenbeispiele:

```
carl, 12300,           // falsches Komma (am Ende)
, carl, 12300          // falsches Komma (am Anfang)
carl 12300             // Komma fehlt
CARL, 12300            // CARL ist kein Bezeichner und kein Literal
carl, 0123             // 0123 ist kein Bezeichner und kein Literal
```

Geben Sie eine (kontextfreie) Grammatik für die Menge aller BL-Folgen an. Benutzen Sie dabei möglichst oft das in den Übungen behandelte Muster für die *Beschreibung von Folgen*.

**Aufgabe 3 (16 Punkte):** In einem Gentle-Programm sei folgender Typ definiert:

```
1 'type' BAUM
2   leer
3   k(Zahl1: INT, Zahl2: INT, LinkerBaum: BAUM, RechterBaum: BAUM)
```

Jeder Wert dieses Typs ist ein *binärer Baum*, der aus null oder mehr Knoten besteht. Jeder Knoten enthält als "Nutzen" zwei *Zahlen* (vom Typ INT).

Definieren Sie ein Aktions-Prädikat (action predicate) namens *anzZwillinge*, welches *die Anzahl der Knoten mit zwei gleichen Zahlen* in einem Baum ermittelt. Hier die Spezifikation des Prädikates:

```
4 'action' anzZwillinge(B: BAUM -> AnzahlZwillingeInB: INT)
```

**Aufgabe 4 (16 Punkte):** Definieren Sie ein Bedingungs-Prädikat (condition predicate) namens `keineNegativen` mit einem Eingabeparameter vom Typ `BAUM` (dieser Typ wurde in der vorigen Aufgabe definiert). Ein Aufruf `keineNegativen(B)` soll genau dann glücken, wenn alle `INT`-Zahlen im Baum `B` größer oder gleich 0 (d.h. nicht negativ) sind. Hier die Spezifikation des Prädikates:

```
1 'condition' keineNegativen(BAUM)
```

**Aufgabe 5 (16 Punkte):** Beantworten Sie die folgenden Fragen möglichst kurz, aber genau.

- 5.1. Welche Prädikate eines Gentle-Programms werden einem Parsergenerator (yacc bzw. bison) übergeben, damit der daraus einen Parser generiert?
- 5.2. Welche Prädikate eines Gentle-Programms werden einem Lexergenerator (lex bzw. flex) übergeben, damit der daraus einen Lexer generiert?
- 5.3. Welche weiteren "kleinen Bestandteile eines Gentle-Programms" werden dem Lexergenerator übergeben, damit der daraus einen Lexer generiert?
- 5.4. Was ist (in Gentle) der wesentliche Unterschied zwischen einem Aktionsprädikat (action predicate) und einem Bedingungsprädikat (condition predicate) ?
- 5.5. Wie definiert man (in Gentle) ein Tokenprädikat? Welche Sprachen benutzt man dazu? Beschreiben Sie nur kurz die grundsätzliche Vorgehensweise, keine "fitzeligen Einzelheiten".
- 5.6. Die Sprachen Prolog und Gentle ähneln sich stark, unterscheiden sich aber auch an einigen wichtigen Stellen. Welcher Unterschied bewirkt vor allem, dass Gentle-Programme viel schneller sind als Prolog-Programme?
- 5.7. In Programmen, die in den Sprachen C, C++ oder Gentle geschrieben sind, unterscheidet man zwei Arten von Vereinbarungen: Deklarationen und Definitionen. Was ist dabei eine Deklaration? Geben Sie möglichst nur die in der Vorlesung behandelte Kurzdefinition (für den Begriff Deklaration) an.
- 5.8. Was ist eine Klasse? Geben Sie möglichst nur die in der Vorlesung behandelte Kurzdefinition (für den Begriff Klasse) an.

**Lösung 1 (16 Punkte):** Eine (kontextfreie) Grammatik für die Menge aller Zahlen im 5-er-System, die (ohne Rest) durch 4 teilbar sind.

Das Startsymbol der folgenden Grammatik ist RK0:

```

RK0 -> 0      RK0 -> RK0 0      RK1 -> RK1 0      RK2 -> RK2 0      RK3 -> RK3 0
RK0 -> 4      RK1 -> RK0 1      RK2 -> RK1 1      RK3 -> RK2 1      RK0 -> RK3 1
RK1 -> 1      RK2 -> RK0 2      RK3 -> RK1 2      RK0 -> RK2 2      RK1 -> RK3 2
RK2 -> 2      RK3 -> RK0 3      RK0 -> RK1 3      RK1 -> RK2 3      RK2 -> RK3 3
RK3 -> 3      RK0 -> RK0 4      RK1 -> RK1 4      RK2 -> RK2 4      RK3 -> RK3 4

```

**Lösung 2 (16 Punkte):** Eine Grammatik für die Menge aller BL-Folgen:

```

KB -> 'a'      GB -> 'A'      ZIFF_1_9 -> '1'      ZIFF_0_9 -> '0'
KB -> 'b'      GB -> 'B'      ZIFF_1_9 -> '2'      ZIFF_0_9 -> ZIFF_1_9
...
KB -> 'z'      GB -> 'Z'      ZIFF_1_9 -> '9'

```

```

ZIFF_FO -> ZIFF_0_9
ZIFF_FO -> ZIFF_0_9 ZIFF_FO

```

```

KB_FO -> KB
KB_FO -> KB KB_FO

```

```

BEZEICH -> GB
BEZEICH -> KB
BEZEICH -> BEZEICH KB_FO

```

```

LITERAL -> 0
LITERAL -> ZIFF_1_9
LITERAL -> ZIFF_1_9 ZIFF_FO

```

```

BL -> BEZEICH
BL -> LITERAL

```

```

BL_FO -> BL
BL_FO -> BL, BL_FO

```

**Lösung 3 (16 Punkte):** Ein Aktions-Prädikat namens *anzZwillinge*, welches *die Anzahl der Knoten mit zwei gleichen Zahlen* in einem Baum ermittelt.

```

1 'action' anzZwillinge(BAUM -> INT)
2   'rule' anzZwillinge(leer -> 0): .
3   'rule' anzZwillinge(k(Z1, Z2, B1, B2) -> 1+N1+N2):
4     ne(Z1, Z2)
5     anzZwillinge(B1 -> N1)
6     anzZwillinge(B2 -> N2)
7   'rule' anzZwillinge(k(Z1, Z2, B1, B2) -> N1+N2):
8     anzZwillinge(B1 -> N1)
9     anzZwillinge(B2 -> N2)

```

**Lösung 4 (16 Punkte):** Ein Bedingungs-Prädikat (condition predicate) namens *keineNegativen* mit einem Eingabeparameter vom Typ BAUM. Ein Aufruf *keineNegativen(B)* glückt genau dann, wenn alle INT-Zahlen im Baum B größer oder gleich 0 (d.h. nicht negativ) sind.

```

1 'condition' keineNegativen(BAUM)
2   'rule' keineNegativen(leer): .
3   'rule' keineNegativen(k(Z1, Z2, B1, B2)):
4     ge(Z1, 0)
5     ge(Z2, 0)
6     keineNegativen(B1)
7     keineNegativen(B2)

```

**Lösung 5 (16 Punkte):** Beantworten Sie die folgenden Fragen möglichst kurz, aber genau.

5.1. Welche Prädikate eines Gentle-Programms werden einem Parsergenerator (yacc bzw. bison) übergeben, damit der daraus einen Parser generiert?

**Die Zwischensymbolprädikate (nonterm predicates).**

5.2. Welche Prädikate eines Gentle-Programms werden einem Lexergenerator (lex bzw. flex) übergeben, damit der daraus einen Lexer generiert?

**Die Endsymbolprädikate (token predicates).**

5.3. Welche weiteren "kleinen Bestandteile eines Gentle-Programms" werden dem Lexergenerator übergeben, damit der daraus einen Lexer generiert?

**Alle String-Literale, die in den Regeln von Zwischensymbolprädikaten vorkommen, und alle .t-Dateien und alle .b-Dateien.**

5.4. Was ist der wesentliche Unterschied zwischen einem Aktionsprädikat (action predicate) und einem Bedingungsprädikat (condition predicate) in Gentle?

**Wenn ein Aufruf eines Aktionsprädikats mißlingt, dann ist das ein schwerwiegender Programmierfehler und bewirkt, dass die Ausführung des betreffenden Gentle-Programms (des Compilers) sofort abgebrochen wird. Wenn ein Aufruf eines Bedingungsprädikats mißlingt, dann ist das "ganz normal" und führt nicht zum Abbruch der Compilerausführung.**

5.5. Wie definiert man ein Endsymbolprädikat (token predicate)? Welche Sprachen benutzt man dazu? Beschreiben Sie nur kurz die grundsätzliche Vorgehensweise, keine "fitzeligen Einzelheiten".

**Ein Endsymbolprädikat (token predicate) definiert man, indem man eine .t-Datei erstellt. In diese Datei schreibt man einen regulären Ausdruck und C-Befehle.**

5.6. Die Sprachen Prolog und Gentle ähneln sich stark, unterscheiden sich aber auch an einigen wichtigen Stellen. Welcher Unterschied bewirkt vor allem, dass Gentle-Programme viel schneller sind als Prolog-Programme?

**Ein Prolog-Ausführer arbeitet mit tiefem Wiederaufsetzen (deep backtracking), eine Gentle-Ausführer arbeitet dagegen mit flachem Wiederaufsetzen (shallow backtracking).**

5.7. In Programmen, die in den Sprachen C, C++ oder Gentle geschrieben sind, unterscheidet man zwei Arten von Vereinbarungen: Deklarationen und Definitionen. Was ist dabei eine Deklaration? Geben Sie möglichst nur die in der Vorlesung behandelte Kurzdefinition (für den Begriff Deklaration) an.

**Eine Deklaration ist ein Versprechen (des Programmierers an den Ausführer), eine bestimmte Größe irgendwo in einem Programm zu definieren.**

5.8. Was ist eine Klasse? Geben Sie möglichst nur die in der Vorlesung behandelte Kurzdefinition (für den Begriff Klasse) an.

**Eine Klasse ist ein Modul und ein Bauplan für Module.**