Übungsblatt 09

Ausgabe: 17.12. Abgabeschluss: Mittw., 7.1., 9:45 Uhr, `eClaus.informatik.uni-stuttgart.de`

Abgabe erfolgt ausschließlich elektronisch über `eClaus.informatik.uni-stuttgart.de` – versuchen Sie nach Möglichkeit die Abgabe nicht in der letzten Minute zu machen!

Suchen Sie sich aus diesem Aufgabenblatt einen geeigneten Teil der Aufgaben aus.

Von jedem Aufgabenblatt werden maximal 20 Punkte auf den Schein angerechnet.

---

1. (2+3 Punkte, mittel) **Kontextfreie Grammatiken:**

   (a) (2 Punkte) Gegeben sei die Sprache L, die alle korrekten Klammerungen mit runden und eckigen Klammern enthält, mit der Einschränkung, dass innerhalb einer `[]` keine `()` stehen darf. Beispiele: `(())([[]])`$\in L$, aber `([()])`$\notin L$ und ebenso `(()))(()`$\notin L$. Geben Sie die Produktionsregeln einer kontextfreien Grammatik $G = (V, \{\texttt{(},\texttt{)},\texttt{[},\texttt{]}\}, P, S)$ an, die L erzeugt.

   (b) (3 Punkte) Geben Sie die Produktionsregeln einer kontextfreien Grammatik an, welche die Sprache L' der „natürlichen Polynome zweiten Grades„ erzeugt. Ein „natürliches Polynom zweiten Grades„ ist von der Form `ax^2+bx+c` mit `b`, `c` $\in \mathbb{N} \cup \{$`0`$\}$, `a` $\in \mathbb{N}$, `a`>`0`. Dabei soll auf die vorgegebene Reihenfolge geachtet werden und ein Summand nur auftreten, wenn dessen Koeffizient $\neq$`0` ist. Der Faktor `1` muss weggelassen werden. Beispiele: `x^2+1`$\in L'$, `5x^2+6x`$\in L'$, `7x`$\notin L'$, `5x+6x^2`$\notin L'$, `1x^2+3x+1`$\notin L'$, `x^2+0x+3`$\notin L'$.

2. (3+5 Punkte, schwer–sehr schwer) **Eindeutigkeit von Grammatiken:** Gegeben sei die Sprache $L = \{a^i b^j a^k b^l \mid i = j \ \vee \ k = l, \ i, j, k, l \geq 0\}$, bei der entweder zu Beginn eine gleich lange Folge von a und b auftritt (und danach beliebig viele a und danach beliebig viele b) oder zu Beginn beliebig viele a und dann b gefolgt von gleichvielen a wie b.

   Eine mögliche Grammatik für diese Sprache hat die Produktionsregeln
   $P = \{S \to AB, \ S \to BA, \ A \to aAb, \ A \to \varepsilon, \ B \to aB, \ B \to C, \ C \to bC, \ C \to \varepsilon\}$.

   (a) (3 Punkte) Zeigen Sie, dass diese Grammatik mehrdeutig ist (z.B. durch zwei wesentlich verschiedene Ableitung eines Wortes). Geben Sie eine Mengencharakterisierung der Wörter an, die bei dieser Grammatik mehrdeutig sind.

   (b) (5 Punkte) Geben Sie eine eindeutige Grammatik für die Sprache L an. Beschreiben Sie Ihre Idee und begründen Sie, warum die Grammatik eindeutig ist.

3. (2+5 Punkte, mittel–schwer) **Funktionale Programmierung:** Ada 95 ist eine imperative (befehls- oder anweisungsorientierte) Programmiersprache. Im dritten Semester werden Sie auch funktionale Programmiersprachen kennenlernen. Bei diesen kann man auf Zuweisungen und Schleifen verzichten – die einzigen Sprachelemente sind die Fallunterscheidung und die Rekursion. Bei den Beispielen zur Berechnung der Fakultät und der Fibonacci-Zahlen haben wir bereits ein wenig funktional programmiert. Prinzipiell lässt sich jede berechenbare Funktion auch funktional programmieren, d.h. nur mit Fallunterscheidung und Rekursion.

   - (2 Punkte, mittel) Versuchen Sie zunächst `for`-Schleifen und `while`-Schleifen durch Rekursion zu simulieren (die Fallunterscheidung wird dabei nur zum Rekursionsabbruch benutzt).
   - (5 Punkte, knifflig) Programmieren Sie eine `function istPrim(n:natural) return boolean`, die testet, ob der formale Parameter `n` eine Primzahl ist. Verwenden Sie auch hier ausschließlich Fallunterscheidung und Rekursion.

     Alternativaufgabe für Tüftler: (noch etwas schwerer) Programmieren Sie eine `function ntePrimzahl(n:natural) return natural`, die zum formalen Parameter `n` die `n`-te Primzahl berechnet. Verwenden Sie auch hier ausschließlich Fallunterscheidung und Rekursion.

4. (5(+3+1) Punkte, mittel) **Das n-Damen-Problem:** Auf einem Schachbrett kann eine Dame waagerecht, senkrecht und diagonal auf jedes Feld, das in diesen Richtungen liegt, wechseln. Eine Dame A bedroht eine andere Dame B, wenn die Dame B auf einem der Felder steht, auf das die Dame A wechseln kann.

   (5 Punkte) Schreiben Sie ein Programm, das alle Stellungen von 8 Damen auf einem 8×8-Schachbrett berechnet, so dass diese 8 Damen sich paarweise nicht bedrohen. Geben Sie auch die Gesamtzahl der gefundenen Lösungen an. Welche Anzahlen ergeben sich für n von 1 bis 10, wenn man n Damen auf einem $n \times n$-Schachbrett platzieren soll? Überlegen Sie sich, welche Laufzeit ihr Programm hat.

   Zusatzaufgabe (+3 Punkte): Modifizieren Sie das Programm so, dass Lösungen, die aus anderen Lösungen durch Drehung hervorgehen, nicht ausgegeben werden.

   Zusatzaufgabe für Freaks (+1 Punkt): Modifizieren Sie das Programm weiter, so dass auch Lösungen, die aus anderen Lösungen durch Spiegelung hervorgehen, nicht ausgegeben werden.

Fragen können im Forum `www.autip.de/forum/viewforum.php?f=236` diskutiert werden. Weitere Informationen zur Vorlesung und Übung unter `www.fmi.uni-stuttgart.de/fk/lehre/ws08-09/autip1/`