Praktische Informatik I WS 2008/09

Übungsblatt 7

Abgabe: 15.12.2008

Aufgabe 1 Syntaxcheck von Hand für JAVA (30%)

Überprüft mit der JAVA Grammatik aus Kapitel 18 in
J. Gosling,B. Joy, G. Steele, and G. Bracha:
The Java Language Specification - Third Edition,
ob die folgenden Quellcode-Ausschnitte syntaktisch korrekt sind:
a) Ist diese Zeichenfolge ein *MethodBody*?

```
{ while (7 <= 5) {
    r *= (j - k) >> i++ % 5;
    return;
  }
}
```

b) Ist diese Zeichenfolge ein *MethodOrFieldDecl*?

```
private int sumPosInt(int i, int j){
    int sum = 0;
    if(i >= 0 && j >= 0){
        sum = i+++++j;
    }
    return sum;
}
```

c) Ist diese Zeichenfolge ein *MethodOrFieldDecl*?

```
static int _(int $){
       return $ + $ == $ ? --$  : _($);
}
```

Dabei soll die genaue Ableitung angegeben werden, bzw. (falls es kein korrekter Code ist) die vollständige Auflistung der fehlgeschlagenen Pfade.

Hinweis: Die Grammatikbeschreibung in Kapitel 18 verweist für *Identifier* auf *IDENTIFIER*, die zugehörige lexikalische Grammatik für *IDENTIFIER* ist dort nicht explizit angegeben. Was als *IDENTIFIER* erlaubt ist, ist in Kapitel 3.8 erklärt.

Aufgabe 2 Syntaxcheck mit JAVA (70%)

Aufgabe 2.1 Syntaxüberprüfung für Taschenrechner mit polnischer Notation

Die polnische Notation (PN) ist eine Schreibweise für mathematische Terme, die z.B. von manchen Taschenrechnern als Eingabeform verwendet wird. Der Verknüpfungsoperator steht dabei immer vorne, und es folgen die zu verknüpfenden Zahlen oder Ausdrücke danach. Um die Terme leichter lesbar zu machen, verwenden wir Klammern. Den mathematischen Term „$4 + 8 - 2$“ schreibt man z.B. als „`+(4,-(8,2))`“.

Schreibt ein Java-Programm, welches vollständig geklammerte mathematische Terme, gegeben in PN, auf ihre syntaktische Korrektheit bezüglich der unten angegebenen Grammatik überprüft.

Euer Programm soll den zu überprüfenden Ausdruck als eine Kommandozeilenparameterzeichenkette übergeben bekommen, d. h. es soll später z.B. in der Form

```
java synCheck "+(4,-(8,2))"
```

aufgerufen werden. Der zu prüfende String befindet sich dann in dem Übergabeparameter der `main`-Methode Eures Programms, und dort in dem Array-Element Nummer 0.

Als Ergebnis soll auf dem Bildschirm nur ausgegeben werden, ob es sich bei einem Ausdruck um einen korrekten `TERM` handelt oder nicht; es soll dabei *nicht* auch noch das Ergebnis eines übergebenen Ausdrucks berechnet werden.

Die Ausdrücke sollen aus positiven Zahlen inklusive der 0, den mathematischen Grundoperationen `+,-,*,/`, runden Klammern `()` sowie Kommas zusammengesetzt sein. Der Einfachheit halber sind auch Zahlen mit führenden Nullen zugelassen. Daraus ergibt sich für die Token-Definitionen die folgende lexikalische Grammatik:

```
binop = '+' | '/' | '*'
minop = '-'
lb    = '('
rb    = ')'
komma = ','
nums = num {num}
num   = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
```

Aufbauend auf der lexikalischen Grammatik besteht die PN-Grammatik aus fünf Regeln:

```
TERM    = BINTERM
        | MINTERM
        | nums

BINTERM = binop ARGS

MINTERM = minop MINREST

MINREST = ARGS
        | nums

ARGS    = lb TERM komma TERM rb
```

Schreibt ein Programm `synCheck`, dass seine Eingabe auf korrekte PN-Syntax überprüft.

Trennt in eurem Programm die Ermittlung der Token aus dem Eingabestring als eigene Methode ab. Hier ist es hilfreich, den von Java in dem Package `java.util` zur Verfügung gestellten `StringTokenizer` zu verwenden. Zur Erkennung, ob es sich um ein `num`-Token handelt, bieten sich die Funktionen `charAt` der `String`-Klasse sowie `isDigit` aus der `Character`-Klasse an.

Schreibt für jede der fünf Regeln der obigen PN-Grammatik eine eigene Methode zur Überprüfung. Dadurch spiegelt sich die rekursive Natur der Grammatikregeln direkt in einer entsprechenden Aufrufstruktur eurer Methoden wieder.

Aufgabe 2.2 Überprüfung der Syntaxüberprüfung

Entwickelt eine Menge von Testfälle, die nachweisen, dass euer Programm korrekte und fehlerhafte PN-Terme unterscheiden kann. Erläutert zu jedem Testfall neben dem erwarteten Ergebnis, welcher Aspekt der obigen Grammatik damit überprüft wird.