

## Lexikalischer Analysator (Scanner)

(24 Punkte)

Schreiben Sie einen lexikalischen Analysator (*Scanner*) für die Sprache *MicroJava*. Die lexikalische Struktur vom *MicroJava* finden Sie im VO-Skriptum in Kapitel 4.2 auf Seite 7.

In ihrem Repository finden Sie vorgegebene Klassen für die Implementierung. Für diese Übung brauchen Sie die Klassen *Scanner*, *Token* und *Errors* im Package *ssw.mj*. Die Klassen *Token* und *Errors* sind bereits fertig implementiert. Die Klasse *Scanner* müssen Sie für diese Übung vervollständigen.

*Scanner.next()* muss das jeweils nächste erkannte Terminalsymbol (*Token*) zurückgeben. Wenn das Ende des Eingabestroms erreicht ist, muss *eof* zurückgegeben werden.

Schlüsselwörter, Bezeichner, (positive ganze) Zahlen, Zeichenkonstanten sowie Operatoren müssen erkannt werden. Leerzeichen, Tabulatoren, Zeilenumbrüche und Kommentare (von `"/**` bis `"/`", auch geschachtelt) müssen überlesen werden.

Folgende lexikalische Fehler müssen erkannt werden:

- Das Auftreten ungültiger Zeichen (*INVALID\_CHAR*)
- Fehlende schließende Anführungszeichen bei Zeichenkonstanten (*MISSING\_QUOTE*)
- Leere Zeichenkonstanten (*EMPTY\_CHARCONST*)
- Ungültige Escapesequenzen (*UNDEFINED\_ESCAPE*)
- Zeilenumbrüche in Zeichenkonstanten (*ILLEGAL\_LINE\_END*)
- Ungeschlossene Kommentare (*EOF\_IN\_COMMENT*)
- Zu große Zahlenkonstanten (*BIG\_NUM*)

Der Wertebereich von *int* ist von `-2147483648` bis `2147483647` definiert. Der lexikalische Analysator liefert aber nur positive ganze Zahlen (also von `0` bis `2147483647`). Bei einer negativen Zahl werden zwei Tokens geliefert, ein Minuszeichen und eine positive ganze Zahl. Die Zahl `-2147483648` kann daher nicht als Konstante im Programm angegeben werden.

Im Vorlesungs-Skriptum sind alle Methoden der Klasse *Scanner* *static* deklariert. In der Übung verwenden wir dynamische Methoden, um die JUnit-Testfälle zu vereinfachen. Daher müssen Objekte der Klasse *Scanner* und *Errors* angelegt werden. Die Initialisierung findet in den jeweiligen Konstruktoren statt.

## JUnit-Tests

Implementieren Sie Ihren Compiler so, dass er alle vorgegebenen JUnit-Testfälle aus *ScannerTest* besteht. Auch die Tutoren verwenden diese Testfälle. Sie können die gegebenen JUnit-Testfälle auch als Vorlage für eigene Tests verwenden, um so Ihren Compiler noch ausführlicher zu testen.

## Abgabe und Hinweise

Die Abgabe der Übungen muss elektronisch erfolgen. Geben Sie folgende Dateien ab:

- Elektronisch in das Repository: **Alle** Quellcode-Dateien, die zum **Ausführen** des Compilers benötigt werden (Packages *ssw.mj*, *ssw.mj.codegen* und *ssw.mj.symtab*), also auch alle Klassen der Angabe. Die Verzeichnis-Struktur muss erhalten bleiben.
- `svn://ssw.jku.at/2012W/UB/k<MatrNr>/branches/UE2`

JUnit Testfälle: ScannerTest