Konkrete Syntax

Eine **kontext-freie Grammatik** definiert die **Struktur der Programme** der Sprache. Hieraus wird der Parser generiert. Sie muß deshalb eindeutig sein (und die LALR(1)-Bedingung erfüllen).

Notation an Beispielen:

```
Prog    ::= Block .
Block   ::= '{' Decl* Stmt+ '}' .
Stmt    ::= Block / Assign ';' / Expr ';' .
Decl    ::= 'int' VarDecls ';' .
VarDecls::= VarDecl // ',' .
VarDecl ::= Ident .
```

Benannte Nichtterminale: `Prog Block Decl`
Benannte Terminale: `Ident`
Literale Terminale: `'{'  '}'  ';'`

EBNF-Konstrukte:

Folge: `Decl*`
nicht-leere Folge: `Stmt+`
nicht-leere Folge mit Trenner: `VarDecl // ','`
Alternativen: `Block / Assign ';'`
optional: `[ Parameter ]`
Klammerung: `( ':=' / ':-' )`

Hinweise zum Entwurf:

- Syntax einer vorgegebenen Sprache nicht unnötig ändern.
- Syntaktische Struktur soll der semantischen entsprechen.
- EBNF nur, wo keine Berechnungen im Baum erwartet werden.
- Namensanalyse erfordert syntaktische Unterscheidung von Bezeichnern: definierendes und angewandtes Auftreten.

Praktikum Sprachimplementierung mit Werkzeugen WS 1999/ 2000 / Folie 201

Ziele:

Grundbegriffe kontext-freier Grammatiken

im Vorlesungsteil:

Erläuterung der EBNF-Konstrukte und ihrer Anwendung

nachlesen:

Eli: Syntactic Analysis (Context-Free Grammars and Parsing)

Verständnisfragen:

- Wie unterscheidet man bei gegebener Grammatik zwischen benannten Terminalen und Nichtterminalen?
- Wie erkennt man bei gegebener Grammatik das Startsymbol?

Abstrakte Syntax

Eine **kontext-freie Grammatik** definiert **Strukturbäume** als Programmrepräsentation (**Baum-Grammatik**). Der Strukturbaum wird vom Parser aufgebaut.

Analyse und Transformation werden als **Berechnungen in** bestimmten **Strukturbaum-Kontexten** spezifiziert.

Konkrete Produktion:

```
Decl    ::= 'int' VarDecls ';' .
```

abstrakte Produktion, Baum-Kontext:

```
RULE: Decl ::= 'int' VarDecls ';' COMPUTE
        Berechnungen für diesen Kontext
END;
```

Die Knoten des Strukturbaumes repräsentieren Nichtterminale der Baum-Grammatik. Terminale stehen nicht im Baum. Literale Terminale dienen nur zur Identifikation der Produktion.

EBNF-Konstrukte gibt es in der Baum-Grammatik nur sehr eingeschränkt (`LISTOF`).

Es brauchen nur die Baum-Kontexte angegeben zu werden, zu denen es tatsächlich Berechnungen gibt. Eli-Werkzeuge vervollständigen die Baum-Grammatik automatisch.

Ebenso ergänzen sie Produktionen zur konkreten Syntax, die nur in der Baum-Grammatik stehen.

Der Aufbau des Strukturbaumes durch den Parser wird von Eli automatisch generiert.

Praktikum Sprachimplementierung mit Werkzeugen WS 1999/ 2000 / Folie 202

Ziele:

Rolle der abstrakten Syntax

im Vorlesungsteil:

Zusammenhang: konkrete - abstrakte Syntax

nachlesen:

Eli: Syntactic Analysis (The Relationship Between Phrases and Tree Nodes)

Verständnisfragen:

Warum können Schreibfehler in Produktionen der Baum-Grammatik zu unerwünschten Effekten führen? Wie erkennt man sie?

Überflüssige Kettenproduktionen

Konkrete Produktionen mit gleicher Struktur und Bedeutung zu einem Baum-Kontext zusammenfassen.
Kettenproduktionen zwischen äquivalenten Symbolen entfallen.

Konkrete Ausdrucksgrammatik:

```
Expr    ::= Expr AddOpr Fact / Fact .
Fact    ::= Fact MulOpr Operand / Operand .
Operand ::= '(' Expr ')' .
Operand ::= Number .
AddOpr  ::= '+' / '-' .
MulOpr  ::= '*' / '/' .
```

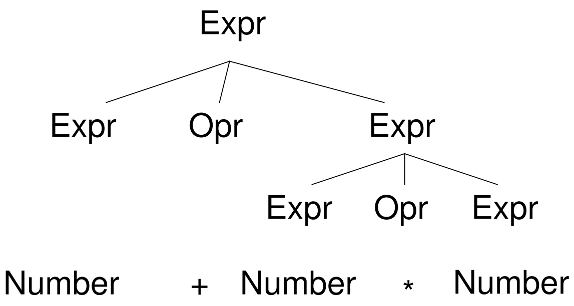
Äquivalente Symbole:

```
Expr ::= Fact Operand .
Opr  ::= AddOpr MulOpr .
```

Baum-Grammatik dazu:

```
RULE: Expr ::= Expr Opr Expr  END;
RULE: Expr ::= Number         END;
RULE: Opr  ::= '+'            END;
RULE: Opr  ::= '*'            END;
...
```

Strukturbaum dazu:



Ein Symbolknoten gehört zu zwei benachbarten Kontexten: oberer und unterer Kontext.

Praktikum Sprachimplementierung mit Werkzeugen WS 1999/ 2000 / Folie 203

Ziele:

Baum-Grammatik vereinfachen

im Vorlesungsteil:

- eindeutige konkrete Ausdrucksgrammatik definiert Präzedenzen der Operatoren
- abstrakte Syntax beschreibt Ausdrucksstruktur
- Symbol-Äquivalenzen und Kettenproduktionen

nachlesen:

Eli: Syntactic Analysis (The Relationship Between Phrases and Tree Nodes)

Verständnisfragen:

- Warum darf die Baum-Grammatik mehrdeutig sein?
- Geben Sie Beispiele für sinvolle Symbol-Äquivalenzen ausserhalb von Ausdrucksgrammatiken.