5. Die syntaktische Analyse

yacc-Eingabeende/Fehlersituation:

- Ende des Eingabetextes: Parser erwartet 0 oder neg. Zahl
  (muss lex liefern!)
- an Parser gelieferte Token bilden Folge, die durch Startregel abgedeckt wird -> Parser akzeptiert Eingabe, meldet an Aufrufer Wert 0
- im Fehlerfall meldet Parser „syntax error“, liefert Rückgabewert 1
- Wünschenswert:
  - Parser liefert Informationen über Art/Ort des Fehlers
  - Parser arbeitet weiteren Eingabetext ab
  - Aufräumarbeiten vor Programmende möglich

5. Die syntaktische Analyse

error-Symbol:

- erkennt Parser Eingabefehler: Umschalten in Fehlerbehandlungs-Modus
- falsches Eingabesymbol wird durch *error*-Symbol abgedeckt
- Meldung „syntax error“ wird ausgegeben
- Aktion hinter *error*-Symbol wird ausgeführt
- weiterer Fehler im Fehlerbehandlungs-Modus: wird ignoriert
- Parser liest weiter
- 3 erlaubte Eingabesymbole hintereinander: Umschalten in „Normal-Modus“ (default)
- besser: Aufsetzpunkt nach Fehler und Umschalten in Normal-Modus durch Entwickler festlegen (yyerrok;)
- Beispiel: satz1, satz2, satz3, satz4

5. Die syntaktische Analyse

Rückgabewerte von lex an yacc:

- yychar
- yylval Rückgabevariable des Scanners, Typ int
- anderer Typ für yylval: mittels

```
%union{
        typ1 name1;
        typ2 name2;
        …
}
```

- Neue Rückgabetypen in yacc-Datei Token zuordnen (<*name*> *token*)

5. Die syntaktische Analyse

Beispiel: yylval soll mal String, mal Realwert liefern

datei.y

```
%union{
        float  reelleZahl;
        char string[30];
}
%token <reelleZahl> ZAHL
%token <string> NAME
%%
…
bez  :   NAME {printf(“Bez. ist %s\n“,$1);}
```

datei.l

```
 %%
[a-zA-Z]*      {strcpy(yylval.string,yytext); return(NAME);}
[1-9][0-9]*[.]?[0-9]*   {yylval.reelleZahl = atof(yytext); return(ZAHL);}
…
```

5. Die syntaktische Analyse

Rückgabewerte in yacc:

- $$, $1, $2,..Rückgabewerte der Aktionen, Typ int
- anderer Typ für $i:

  %type <kompname> symbolname
- Beispiel:

```
%type <reelleZahl> eingabe ausdruck
…
%%
eingabe        :        ausdruck   {printf("   = %f\n",$1);}
               |        …;
ausdruck       :        …;
```

Aufgabe: Schreiben Sie einen yacc-Parser für die switch-Anweisung (siehe Aufgabe Folie 107: BNF)

5. Die syntaktische Analyse

benutzerdefinierte Routinen:

- meist Definition von C-Funktionen, Variablendefinitionen, Präprozessoranweisungen
- statt komplexer Aktionen im Regelteil besser dort nur Funktionsaufruf
- mittels #include“lex.yy.c“ lex-Scanner einbinden
- vordefinierte Routinen (aus yacc-Bibliothek), hier überschreibbar:
  - main() ; ruft yyparse() auf
  - yylex() ; wird von yyparse() aufgerufen
  - yyerror() ; wird bei Fehler von yyparse() aufgerufen

5. Die syntaktische Analyse

Arbeitsweise des Parsers:

- yacc –v datei.y -> y.output, enthält Beschreibung der Arbeitsweise
- 4 Operationen: schiebe, reduziere, annehmen, fehler
- Zustandsübergang mittels „gehe zu“

- Einbinden der Standard-Yacc-Bibliothek:

  gcc –o zielname y.tab.c –ly -lfl

5. Die syntaktische Analyse

Java-Parser:

- JavaCC (LL(k)-Parser), jay, BYacc/J,…
- hier: BYacc/J:
  - Analog zu yacc
  - Im Deklarationsteil typischerweise package-/import-Anweisungen
  - Aktionen in Java-Quellcode (nur Methodenaufrufe)
  - Methodendeklarationen im Code-Teil
  - yylex() bzw. yyerror() müssen definiert werden
  - Aufruf mittels yacc –J ….
  - Vordefinierte Klasse ParserVal mit int-, double-, String- u. Object-Attribut (für Tokenwert yylval)

```
…yylval = new ParserVal(gelesenemDouble);…
…/*in Regel*/ { System.out.println(„Wert“ +$1.dval);}
```

5. Die syntaktische Analyse

Fragestellungen zu Kapitel 5:

- Aufgaben der syntaktischen Analyse?
- Was erzeugt die syntaktische Analyse?
- Welche Analyse-Verfahren (Prinzip) gibt es? Funktionsweise?
- Womit lassen sich Sprachen syntaktisch beschreiben?
- Wie ist ein Kellerautomat definiert?
- Eigenschaften von kfG (deterministisch, eindeutg/mehrdeutig, LL(k),..)
- Bestandteile einer yacc-Datei
- Syntax/Semantik der Grammatik-Regeln
- Erstellen und Übersetzen einer yacc-Datei am Beispiel
- „Fehlererkennung/Behebung“ bei yacc?