

LV Compilerbau WS 2006/07: Lernziele

Kap. 1 - Einleitung

Sie sollen erklären können

- Definition eines Compilers
- Analyse-Synthesemodell eines Compilers
- Begriffe lexikal. Analyse, syntakt. Analyse, semant. Analyse
- Begriff Coderzeugung
- Begriff Compiler-Phase
- Begriffe Frontend/Backend
- Begriff Compiler-Paß

LV Compilerbau WS 2006/07: Lernziele

Kap. 2 - Sprachanalyse

Sie sollen Bescheid wissen über

- Grundlagen von Sprachen und Grammatiken
- Terminal - und Nichtterminalsymbole
- Einteilungen von Grammatiken anhand des Typs der Produktionen
- wichtige Eigenschaften von Grammatiken wie Eindeutigkeit
- verschiedene Darstellungsmöglichkeiten von Produktionen
- den Sinn von Modellsprachen für den Compilerbau
- die Art der Darstellung der Semantik von Programmiersprachen

Sie sollen

- zu einfachen Sprachen eine Grammatik aufstellen können

LV Compilerbau WS 2006/07: Lernziele

Kap. 3 – Lexikalische Analyse

Sie sollen wissen,

- welche theoretischen Grundlagen die lexikalische Analyse hat
- wo die lexikal. Analyse innerhalb des Compilers angesiedelt ist
- welche Aufgabe der Scanner hat
- warum lexikalische Analyse und syntakt. Analyse getrennt sind
- was wichtige Begriffe (Lexem, Symbole, etc.) bedeuten
- welche Möglichkeiten man prinzipiell zur Implementierung von Scanner hat

Sie sollen sich

- im PL/0 - Scanner auskennen

Sie sollen fähig sein,

- einen Scanner mit Hilfe des Übergangsdigramms des endlichen Automaten zu entwickeln

Kap. 4 – Syntaktische Analyse

Sie sollen wissen,

- wo innerhalb eines Compilers die Syntaxanalyse sitzt
- welche theoretischen Grundlagen die Syntaxanalyse hat
- welche prinzipiellen Ansätze für die Syntaxanalyse existieren
- wie die Top-Down-Analyse prinzipiell vorgeht
- welche Probleme bei der Top-Down-Analyse auftauchen können
- wie man Grammatiken transformiert, damit sie die LL(1)-Eigenschaft erfüllen
- was LL(1)-Grammatiken sind
- wie ein Recursive-Descent-Parser arbeitet
- wie ein tabellengesteuerter Top-Down-Parser funktioniert

LV Compilerbau WS 2006/07: Lernziele

Sie sollen

- den PL/0-Parser verstehen

Sie sollen fähig sein,

- einen Recursive-Descent-Parser für eine einfache kontextfreie Grammatik zu schreiben

Kap. 5 – Semantische Analyse

Sie sollen wissen,

- welche Sprachregeln die semantische Analyse überprüft,
- wie sich statische und dynamische semantische Analyse unterscheiden
- wo die Stellung der statischen semantischen Analyse ist,
- was bei Typprüfungen geschieht,
- was Sinn von Gültigkeitsprüfungen ist,
- was die Bedeutung der Symboltabelle für die semantischen Analyse ist,
- wo im PL/0-Compiler semant. Prüfungen vorkommen

LV Compilerbau WS 2006/07: Lernziele

Kap. 6 – Fehlerbehandlung

Sie sollen wissen,

- was Fehlerbehandlung im Compiler bedeutet,
- wo Fehlerbehandlung auftritt,
- nach welchen Prinzipien beim rekursiven Abstieg Fehlerbehandlung möglich ist,
- wo im PL/0-Compiler die Fehlerbehandlung möglich wäre.

LV Compilerbau WS 2006/07: Lernziele

Kap. 7 – Laufzeit-Speicherverwaltung

Sie sollen wissen,

- für welche Datenobjekte Speicherplatz bei der Codegenerierung angelegt werden muß,
- wie sich dynamische und statische Speicherverwaltung unterscheiden,
- wie der Laufzeitspeicher sich aufteilt,
- welche Speichersegmente es gibt und was sie beinhalten,
- warum das Stack-Segment besonders wichtig ist,
- was ein Activation Record ist und was er enthält,
- was der Sinn von Dynamic-Link und Static-Link-Ketten ist,
- was das Environment ist,
- wie Static-Links initialisiert werden,
- was das Display für eine Bedeutung hat.

LV Compilerbau WS 2006/07: Lernziele

Sie sollen die Fähigkeit entwickeln,

- bei der Codegenerierung die Variablen richtig zu adressieren.

LV Compilerbau WS 2006/07: Lernziele

Kap. 8 – Code- und Zwischencodgenerierung

Sie sollen wissen,

- wo im Gefüge des Compilers die Code- bzw. die Zwischencodgenerierung liegt
- was syntax-orientierte Übersetzung ist
- welche Eigenschaftenem Zwischencode hat
- welche einfachen Beispiele es für Zwischencode gibt
- wie Postfixcode funktioniert
- semantische Aktionen am Beispiel Zwischencode erzeugen
- was Interpretation von Zwischencode ist
- wie man aus Zwischencode Assemblercode erzeugen kann

Sie sollen

- sich in der Zwischencodgenerierung des PL/0-Compilers auskennen

LV Compilerbau WS 2006/07: Lernziele

Sie sollen fähig sein,

- einen Zwischencodegenerator für Postfixcode zu schreiben

Kap. 9 - Compilergeneratoren

Sie sollen wissen,

- für welche Aufgaben Compilerbau-Tools eingesetzt werden
- welche die meistverwendeten Tools sind
- wie der Scannergenerator Lex funktioniert
- wie die Eingabedatei von Lex aussieht
- wie der Parsergenerator Yacc funktioniert
- in welcher Form die Grammatik bei Yacc spezifiziert wird
- wo Grenzen eines Parsergenerators liegen

Kap. 10 - Erweiterung und Portierung

Sie sollen die Begriffe

- T-Diagramme
- Bootstrapping
- Erweiterung von Compilern
- Portierung von Compilern

kennen.