Übungsblatt 4 zur Vorlesung Compilerbau

Abgabe: 12.01.2025, 23:59

Prof. Dr. Michael Kuhn (michael.kuhn@ovgu.de)
Michael Blesel (michael.blesel@ovgu.de)
Parallel Computing and I/O • Institut für Intelligente Kooperierende Systeme
Fakultät für Informatik • Otto-von-Guericke-Universität Magdeburg
https://parcio.ovgu.de

Auf diesem Übungsblatt werden wir mir Hilfe von bison einen Parser für Bminor erstellen und den dazugehörigen AST produzieren.

1. Bison Parser (200 Punkte)

Implementieren Sie einen grundlegenden Parser für Bminor mit Hilfe von bison. In den Materialien finden sie eine grundlegende Struktur mit den benötigten Dateien und einem dazugehörigen Makefile. Die in der Vorlesung besprochene Grammatik finden sie in grammar.md. Sie können sich an dieser orientieren, es steht ihnen aber frei die Grammatik zu modifizieren.

Der Parser soll in diesem Schritt erstmal nur Bminor Programme validieren können und ausgeben ob es sich um ein Programm handelt welches von der Grammatik erzeugt werden könnte. In dem tests Ordner finden Sie eine Sammlung an korrekten und inkorrekten Bminor Programmen. Diese sollten von ihrem Parser korrekt analysiert werden können.

Zur Implementierung eines Parsers mit bison empfiehlt es sich das "Parsing in Practice" Kapitel des Buches nachzuholen.

Die folgenden Schritte sind notwendig:

1. Füllen Sie die beigelete bminor.flex Datei mit passenden Scanning Regeln für Bminor aus. (**Tip:** probieren sie den Scanner Teil an den beigelegten Beispielprogrammen aus um sicherzugehen, dass die richtigen Token erzeugt werden.)
2. Fügen Sie in bminor.bison die von Flex erzeugten Token ein.
3. Übertragen sie die Grammatik für Bminor in ein korrektes bison Format.
4. Testen Sie ihren Parser mit den beigelegten Bminor Beispielprogrammen.

2. AST Erstellung (500 Punkte)

Erweitern Sie ihren Bminor Parser um die Funktionalität einen AST für geparste Programme zu erstellen. Hierfür finden sie in den Materialien bereits einige Header-Dateien die der im Buch gezeigten AST Syntax entsprechen. Erweitern Sie diese Dateien mit den fehlenden Elementen und fügen Sie in der bison Datei den benötigten Code hinzu um damit einen AST generieren zu lassen.

Implementieren Sie eine geignete Visualisierung für den erstellten AST. Dies könnte entweder die Ausgabe in der `dot` Sprache für `Graphviz` sein, oder eine geeignete textuelle Darstellung via `printf`. Wichtig ist, dass die Baumstruktur des AST eindeutig erkennbar ist.

Schreiben Sie eine kurze Erklärung zu den wichtigsten Aspekten ihrer Implementation und wie diese verwendet werden kann. Haben Sie Modifikationen an der Grammatik durchgeführt?

Abgabe

Als Abgabe werten wir den letzten Commit vor der Abgabefrist in Ihrem Git-Repository. Im Hauptverzeichnis des Repositories wird ein Verzeichnis `CB-2024-Uebung-04-Materialien` mit folgendem Inhalt erwartet:

- Eine Datei `gruppe.md` mit den Gruppenmitgliedern (eines je Zeile) im folgenden Format:

  ```
  Erika Musterfrau <erika.musterfrau@example.com>

  Max Mustermann <max.mustermann@example.com>
  ```

- Der modifizierete Quellcode für den grundlegenden Parser (Aufgabe 1)
- Der modifizierete Quellcode für den AST erzeugenden Parser (Aufgabe 2)
- Eine Datei `parser.md` mit den geforderten Erklärungen (Aufgabe 2)