Blatt 04: Semantische Analyse

Carsten Gips, BC George (HSBI)

Zusammenfassung

Ziel dieses Aufgabenblattes ist die Erstellung einer Symboltabelle und eines einfachen Type-Checkers für eine fiktive statisch typisierte Sprache mit Expressions, Kontrollstrukturen und Funktionen.

Methodik

Sie finden im Sample Project eine Grammatik, die (teilweise) zu der Zielsprache auf diesem Blatt passt. Analysieren Sie diese Grammatik und vervollständigen Sie diese bzw. passen Sie diese an.

Erstellen Sie mit dieser Grammatik und ANTLR wieder einen Lexer und Parser. Definieren Sie einen AST und konvertieren Sie Ihren Parse-Tree in einen AST.

Es ist empfehlenswert, den Type-Checker dreiphasig zu realisieren:

1. Aufbauen der Symboltabelle und Prüfen von z.B. Deklaration/Definition vs. Benutzung (Variablen) usw.
2. Prüfen bei Funktionsaufrufen auf vorhandene/sichtbare Funktionsdefinitionen
3. Prüfen der verwendeten Typen

Sprachdefinition

Ein Programm besteht aus einer oder mehreren Anweisungen (*Statements*).

Anweisungen (*Statements*)

Eine Anweisung ist eine Befehlsfolge, beispielsweise eine Deklaration (Funktionen), Definition (Variablen, Funktionen), Zuweisung, ein Funktionsaufruf oder eine Operation. Sie muss immer mit einem Semikolon abgeschlossen werden. Eine Anweisung hat keinen Wert.

```
int a;            # Definition der Integer-Variablen a
int b = 10 - 5;   # Definition der Integer-Variablen b und Zuweisung des Ausdruckes 10-5 (In
c = "foo";        # Zuweisung des Ausdrucks "foo" (String) an die Variable c (diese muss daz
func1(a, c);      # Funktionsaufruf mit Variablen a und c
```

Kontrollstrukturen und Code-Blöcke sowie `return`-Statements zählen ebenfalls als Anweisung.

Code-Blöcke und Scopes

Code-Blöcke werden in geschweifte Klammern eingeschlossen und enthalten eine beliebige Anzahl von Anweisungen.

Jeder Code-Block bildet einen eigenen Scope - alle Deklarationen/Definition in diesem Scope sind im

Aufgaben

Grammatik und AST (2P)

Erstellen Sie eine ANTLR-Grammatik für die Zielsprache. Sie können dabei die Grammatik im Sample Project als Ausgangspunkt nutzen und diese anpassen und vervollständigen.

Definieren Sie einen AST für die Zielsprache. Welche Informationen aus dem Eingabeprogramm müssen repräsentiert werden?

Programmieren Sie eine Traversierung des Parse-Trees, die den AST erzeugt. Testen Sie dies mit den obigen Beispielprogrammen und definieren Sie sich selbst weitere Programme unterschiedlicher Komplexität für diesen Zweck.

Aufbau der Symboltabelle (2P)

Bauen Sie für den AST eine Symboltabelle auf. Führen Sie dabei die im ersten Lauf möglichen Prüfungen durch, beispielsweise ob referenzierte Variablen tatsächlich bereits definiert und sichtbar sind oder ob eine Variable oder Funktion in einem Scope mehrfach definiert wird oder ob Variablen als Funktion genutzt werden. Geben Sie erkannte Fehler auf der Konsole aus.

Symboltabelle: Funktionsaufrufe (1P)

Implementieren Sie einen zweiten Lauf. Dabei soll für Funktionsaufrufe geprüft werden, ob diese

Last modified: *f7ac9d2 (reformat using shorter lines, 2025-08-09)*