Fachhochschule
Dortmund
University of Applied Sciences and Arts
Fachbereich Informatik

Compilerbau und Formale Sprachen

Projekt

Fips -- Files and Paths

Überblick

- Programm (Compiler) in Java (70 P) >35P
  - Lexikalische Analyse 10P + 5P > 3P
  - Syntaxanalyse 15P + 5P > 5P
  - Semantische Analyse 20P + 5P > 5P
  - Fehlerbehandlung 10P + 5P > 3P
  - Codeerzeugung/Interpreter 15P + 5P > 5P
  - Kreativität
- Ausarbeitung + Selbstständigkeitserklärung (20P) >10P
- Kurze Präsentation (10P) >5P

Überblick

- Gruppen bis max. 3 Personen
- Programm (Compiler) in Java
  - Kommentierung
  - „Per Hand“ / JavaCC / AntLR
- Ausarbeitung + Selbstständigkeitserklärung
  - Beschreibung Token
  - Beschreibung Grammatik
  - Beschreibung Semantik + Ergänzungen zur Sprache
  - Beschreibung: Installation + Aufruf des Compilers/Interpreters
  - Bei Gruppenarbeit: Aufteilung der Teile auf Gruppenmitglieder / evtl. Schnittstellen

FIPS

FARE

Hintergrund

Scriptsprache für den Umgang mit Dateien und Pfaden

- Es sollen Variablen verschiedener Typen (s.u.) deklariert und Anweisungen ausgeführt werden können
- Es sollen Funktionen definiert (und aufgerufen werden können)
- Die main-Methode soll wie bei C oder Java als Einsprungspunkt dienen
- Ein- bzw. Mehrzeilenkommentare sollten möglich sein

FARE

Beispiel

```
int i = 0;
Path p = /home/robert/hello;

int anzahl(String neu) {
   for (File f:p.files) f.name = neu + f.name;
}

void main() {
    System.print(“Bitte geben Sie den Pfad ein:“);
    String neu = System.readString();
    System.println(anzahl(neu));
}
```

FIPS - DATENTYPEN

FARE

Datentypen

- String
  - Ops: .length(), .charAt(i) ++ (vgl. Java)
  - Literale: ”wort” (vgl. Java)
- char
  - Ops: +,- (vgl. Java)
  - Literale: ’c’ (vgl. Java)
- int:
  - Ops: +,-,*,/,% (vgl. Java)
  - Literale: 25 (vgl. Java)
- boolean:
  - Ops: &&, ||, ! (vgl. Java)
  - Literale: true, false (vgl. Java)

Fare

Datentypen

- Path: (Pfad)

  Ops:

  - files als Array, toString, …
  - name als Array von Strings
  - remove(), copyTo(Path), moveTo(Path)

- Files: (Datei/Verzeichnis)

  Ops:

  - type (dir, file, …), name, path, …
  - remove(), rename(String), moveTo(Path)
  - Zugriff auf Inhalt

Fare

Zusammengesetzte Datentypen

- Set<typ>: (Menge)
  - Ops: +, -, ^ (Vereinigung, Differenz, Schnitt)
    .contains(element)
  - Literale: { elemente } Bspl.: { ’a’, ’b’, ’c‘ }
- Map<typ1,typ2>: (HashMap)
  - Ops: map[element], .containsKey(element)
  - Literale: [[key1: value1, key2: value2, …]]
- Array: (Felder)
  - Ops: feld[index]
  - Literale: [[element1, element2, …]]

Weitere Operationen: Vergleichsoperationen >,<, >=, <=, ==, !=, wenn es Sinn macht.

FIPS - ANWEISUNGEN

Fare

Anweisungen

- if / if– else
- return
- while
- for
- Blöcke { }
- Leere Anweisung ;
- Ausdrücke Bspl.: f() + y;

FIPS - CODEERZEUGUNG

Fare

Codeerzeugung

- Interpreter:
  - Der Code kann einfach auf dem AST interpretiert werden oder Sie definieren sich eine eigene Zwischensprache.
- Compiler:
  - Sie erzeugen aus der Quelldatei Java- oder C-Code, der den Code ausführt.
  - Hier wäre auch die Nutzung von asm oder llvm für die direkte Übersetzung in Bytecode/Objektcode denkbar