Univerzitet u Beogradu, Elektrotehnicki Fakultet

MiCrojava COmpiler

Programski Prevodioci 1

Autor: Stefan Petrovic

# Opis Zadatka

Cilj zadatka je realizacija malog, ali potpuno funkcionalnog prevodioca za programski jezik Mikrojava, koji, kroz cetiri faze, analizira napisan kod, proverava njegovu sintakticku I semanticku ispravnost, I generise odgovarajuci bajtkod za pokretanje na Mikrojava virtuelnoj masini. Te faze su leksicka analiza, sinkaksna analiza, semanticka analiza I generisanje koda.

# Uputstvo za pokretanje.

Preporucuje se pokretanje iz komandne linije terminala, unutar foldera projekta, koristeci PowerShell. Pokretanje iz build.xml, unutar Eclipse okruzenja, je isto opcija, doduse nece komanda read raditi u tom slucaju.

se naziva lexerGen u Ant build fajlu. Alternativno moze se pokrenuti sledeca komanda u terminalu:

java -jar lib/JFlex.jar -d src/rs/ac/bg/etf/pp1/ spec/mjlexer.lex

Za generisanje parsera potrebno je izabrati repackage unutar build.xml fajla ili izvrsiti sledecu komandu:

java -jar lib/cup\_v10k.jar -destdir src/rs/ac/bg/etf/pp1/ -parser MJParser spec/mjparser.cup

Za generisanje koda potrebno je pokrenuti Compiler.java klasu kao java aplikaciju. Ime programa je potrebno navesti u odgovarajucoj pathname promenljivoj. Izvrsni kod se nalazi u program.obj fajlu. Za pokretanje koda neophodno je u konzoli izvrsiti sledecu komandu:

java -cp lib\mj-runtime.jar rs.etf.pp1.mj.runtime.Run test\program.obj

Ako zelite prikaz izgenerisanog bajtkoda, trebate disasembler pokrenuti preko sledece komande:

java -cp lib\mj-runtime.jar rs.etf.pp1.mj.runtime.disasm test\program.obj

Svim komandama se moza dodati >test/izlaz.out 2>test/izlaz.err na kraj, radi ispisa izlaza programa u fajl sa .out ekstenzijom, i mogucih gresaka u .err fajl.

# Test primeri

Test primeri se sastoje od vise testprimera koji proveravaju funkcionalnosti koda, ukljucujuci pristup nizu I matrici, aritmeticke operacije, citanje I stampanje podataka preko standardnog ulaza/izlaza; kao I otpornost na greske samog kompajlera u slucaju kompajlovanja lose napisanog koda.

# Alati

Radi efikasne realizacije resenja, koriscene su biblioteke:

1. AST-CUP - Construction of useful parsers sa AST prosirenjem. Sluzi za generisanje klasa za sintaksno stablo parsiranja.
2. JFlex - Biblioteka za generisanje leksickih analizatora
3. mj-runtime - Virtuelna masina za mikrojavu
4. Log4J – Biblioteka za izvestavanje o izvrsavanju programa.
5. symboltable – JAR arhiva sa klasama neophodnim za generisanje tabele simbola

# Klase

* **SemanticAnalyzer** – Klasa izvedena iz VisitorAdaptor klase, koja sluzi da se utvrdi semanticka ispravnost prilozenog koda, odnosno proveru tipova podataka I staranje o njihovom adekvatnom koriscenju, kao I formiranje tabele simbola za dati program
* **CodeGenerator –** Klasa izvedena iz VisitorAdaptor klase, koja, na osnovu izgenerisanog stabla parsiranja, sama generise bajtkod za microJava virtuelnu masinu.
* **Compiler** – Klasa koja pokrece ceo sistem I generise .obj fajl koji se moze pokrenuti u microJava virtuelnoj masini.