Programski prevodioci 1

PROJEKAT

Student:

Luka Matović

2017/0722

1. **Opis zadatka**

Cilj projektnog zadatka je realizacija kompajlera (programskog prevodica) za programski jezik Mikrojavu. Programski prevodilac za Mikrojavu ima četiri osnovne funkcionalnosti: leksičku analizu, sintaksnu analizu, semantičku analizu i generisanje koda.

Leksički analizator treba da prepoznaje jezičke lekseme i vrati skup tokena izdvojenih iz izvornog koda, koji se dalje razmatraju u okviru sintaksne analize.

Sintaksni analizator ima zadatak da utvrdi da li izdvojeni tokeni iz izvornog koda programa mogu da formiraju gramatički ispravne sekvence i formira apstraktno sintaksno stablo.

Semantički analizator se formira na osnovu apstraktnog sintaksnog stabla koje je nastalo kao rezultat sintaksne analize radi provere isravnosti definicija promenljivih i metoda i njihovog korišćenja.

Generator koda - nakon ispravnje semantičke analize, generator koda takođe obilazi stablo na osnovu kog generše bytecode za MikroJava virtuelnu mašinu.

1. **Opis komandi**

* **Za generisanje java koda alatima**

desni klik na build.xml -> Run As -> Ant Build : za kreiranje sym.java i MjParser.java, a za kreiranje Yylex.java u build.xml u targetu compile promeniti depends u “lexerGen” i izvrsiti istu komandu

* **Prevodjenje , pokretanje i testiranje resenja**

Desni klik na Compiler.java ->Run As -> Java Application

Prikaz bytekoda objektnog fajla: *java -cp lib/mj-runtime-1.1.jar rs.etf.pp1.mj.runtime.disasm test/test.obj*

Izvršavanje programa: *java -cp lib/mj-runtime-1.1.jar rs.etf.pp1.mj.runtime.Run test/test.obj*

1. **Opis prilozenih testova**

**testForIncDecAndRevVal.mj** – test za inkrement, decrement I obrnutu vrednost

**testForPrintAndExpr.mj** – test za print I read funkcije

**test301.mj** – javni test za nivo A (projekat je uradnjen za nivo A)

**testArray.mj** – pristup ugljezdenom elementu niza

**Program.mj** – javni test

1. **Opis novouvedenih klasa**

**SemaniticAnalyzer.java** - klasa izvedena iz VisitorAdaptor, implementira visit metode klasa nastale kreiranjem sintaksnog apstraktnog stabla i kreira tabelu simbola naseg programa.

**ExtendedTab.java** - klasa izvedena iz Tab klase, overajduje potrebne metode I uvodi ovi bool tip.

**CustomVisitor.java** - klasa izvedena iz VisitorAdapter klase I sadrzi potrebne klase koje sluze za obilazak stable I skupljanje odgovarajucih podataka koji se kasnije koriste.

**CodeGenerator.java** - klasa izvedena iz VisitorAdaptor, implementira visit metode klasa nastale kreiranjem sintaksnog apstraktnog stabla i uz pomoc vec sada kreirane tabele simbola kreira instrukcije specificne za odredjenu klasu.

**Compiler.java** - klasa preko koje se pokrece prevodjenje programa, kao i generisanje koda u slucaju da je sve dobro prevedeno.