Programski prevodioci 1

PROJEKAT

Student:

Lara Vuković

2017/0319

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**

Prilikom izvršavanja leksičke i sintaksne analize, potrebno je u korenom direktorijumu pokrenuti **build.xml** fajl, na sledeći način

(desni klik na build.xml -> Run as -> Ant Build -> označiti **compile[default]**)

U nastavku, za izvršavanje semantičke analize i generisanja koda, pokreće se **MJParserTest.java** fajl koji se nalazi u direktorijumu src/rs/ac/bg/etf/pp1. Potrebno je dodati 2 argumenta komandne linije, a to su nazivi test programa koji obavezno mora imati ekstenziju **.mj**, kao i naziv **.obj** fajl programa koji nastaje kao rezultat generisanja koda.

Oba fajla se moraju naći u direktorijumu **test**.

Primer argumenata: test/program.mj test/program.obj

Kada se izvrše svi prethodni koraci, moguće je pokrenuti projekat u više modova:

* Run debug – gde se mogu tačno videti koraci izvršsavanja kao i stanje steka programa
* Run release – klasično izvršavanje programa
* disasm – prikaz mašinskih instrukcija koje saćinjavaju prevedeni program

1. **Opis prilozenih testova**

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

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

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

**program.mj** – javni test

**testErrLexSin** – test sa leksičkom greškom, sintaksnom greškom sa oporavkom i sintaksnom greškom koja je fatalna

**testErrSem** – test sa semantičkim greškama ispisanim u izlaz.err fajl

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.

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

**CustomObj.java** - klasa izvedena iz DumpSymbolTableVisitor klase, overajduje potrebne metode za obilazak I daje novi ispis.

**CodeGenerator.java** – klasa koja služi za obilazak sintaksnog stabla i generisanje koda. Implementira VisitorAdaptor.

**CounterVisitor.java** – klasa koja sa svoje 2 statičke klase FormParamCounter i VarCounter koje su izvedene iz CounterVisitor-a, prebrojavaju formalne parametre i lokalne promenljive metoda.

Implementira VisitorAdaptor.

**MJParserTest.java** – klasa koja u stvari predstavlja klasu sa main metodom, i ona se pokreće u cilju kompajliranja programa. Main metoda ove klase sadrži argumente naziva ulaznog i izlaznog fajla, kao i da li je potrebno rezultat kompajliranja ispisati u fajl/fajl sa greškama.

Implementira Compiler interfejs, i u njenoj metodi compile vrši sav traženi posao.

**CompilerError.java** – klasa koja je data kao deo postavke projekta uz interfejs Compiler. Objekti ove klase u stvari predstvaljaju objekte sa informacijama o različitim tipovima grešaka.