Elektrotehnički fakulet – Beograd

**Programski prevodioci 1**

Kompajler za Mikrojavu

Nastavnici: Student:

Prof. dr. Dragan Bojić Nebojša Savić

As. Maja Vukasović 2016/0078

As. Kristijan Žiža

Postavka zadatka

Cilj projektnog zadatka iz predmeta Programski Prevodioci 1, na smeru Softversko Inženjerstvo, je realizacija kompajlera za programski jezik Mikrojava. Kompajler omogućava prevodjenje sintaksno i semantički ispravnih Mikrojava programa u Mikrojava bajtkod koji se izvršava na virtuelnoj mašini za 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. Ukoliko se tokom leksičke analize detektuje leksička greška, potrebno je ispisati odgovarajuću poruku na izlaz.

**Sintaksni analizator**ima zadatak da utvrdi da li izdvojeni tokeni iz izvornog koda programa mogu formiraju gramatički ispravne izraze. Tokom parsiranja Mikrojava programa potrebno je na odgovarajući način omogućiti i praćenje samog procesa parsiranja na način koji je opisan u postavci projektnog zadatka. Nakon parsiranja sintaksno ispravnih Mikrojava programa potrebno je obavestiti korisnika o uspešnosti parsiranja. Ukoliko izvorni kod ima sintaksne greške, potrebno je izdati adekvatno objašnjenje o detektovanoj sintaksnoj grešci, izvršiti oporavak i nastaviti parsiranje.

**Semantički analizator**se formira na osnovu apstraktnog sintaksnog stabla koje je nastalo kao rezultat sintaksne analize. Semantička analiza se sprovodi implementacijom metoda za posećivanje čvorova apstraktnog sintaksnog stabla. Stablo je formirano na osnovu gramatike implementirane u prethodnoj fazi. Ukoliko izvorni kod ima semantičke greške, potrebno je prikazati adekvatnu poruku o detektovanoj semantičkoj grešci.

**Generator koda** prevodi sintaksno i semantički ispravne programe u izvršni oblik za odabrano izvršno okruženje Mikrojava VM. Generisanje koda se implementira na sličan način kao i semantička analiza, implementacijom metoda koje posećuju čvorove.

Opis rešenja

Leksičke provere su napisane u fajlu: ***src/spec/mjlexer.flex****.*

Implementacija parsera je u fajlu: ***src/spec/mjparser.cup****.*

Testovi se nalaze u folderu ***test****.*

Korišćene konfiguracije:

1. **parser-gen** – generisanje leksera i parsera
2. **repackage –** prepakivanje paketa generisanih za parser
3. **disasm** – disasembliranje kreiranog mikrojava koda
4. **runObj**– pokretranje objetnog fajla koji smo generisali
5. **runObjDebug**– pokretranje objetnog fajla koji smo generisali sa opcijom debagovanja

**Pokretanje generisanog objektnog fajla iz komandne linije:**

java -cp ../../lib/mj-runtime.jar rs.etf.pp1.mj.runtime.Run javniA.obj

Test primeri

1. ***javniA.mj*** – Javni test za fazu A
2. ***javniB.mj*** – Javni test za fazu B
3. ***test\_conditions.mj*** – Test koji proverava proveru i izracunavanje uslova
4. ***test\_for.mj*** – Test koji proverava proverava rad for petlji i nizove
5. ***test\_functions.mj*** – Test koji proverava globalne funkcije (deklarisanje i pozive)
6. ***test\_if.mj*** – Test koji proverava if grananja
7. ***test\_recovery.mj*** – Test koji proverava oporavak od greške
8. ***test\_scope.mj*** – Test koji proverava scope globalnih i lokalnih varijabli i konstanti
9. ***test\_semantic.mj*** – Test koji proverava razne semantičke greške

Novouvedene klase

1. **TableExtensions** – klasa služi kao proširenje postojeće implementacije tabele simbola. U konkretno našem slučaju koristimo je da bismo u tabelu simbola dodali bool tip kao i za neke pomoćne metode.