Универзитет у Београду

Електротехнички факултет



**Programski prevodioci 1**

**Microjava Parser**

Jovan Spasojević 0118/2017

Sadržaj

[Sadržaj 2](#_Toc61692597)

[1. UVOD 3](#_Toc61692598)

[2. Genersianje koda 4](#_Toc61692599)

[3. TESTOVI 5](#_Toc61692600)

[4. Novo Uvedene Klase 6](#_Toc61692601)

1. UVOD

Cilj projektnog zadatka je realizacija kompajlera za programski jezik Mikrojavu. Kompajler omogućava prevodjenje sintaksno i semantički ispravnih Mikrojava programa u Mikrojava bajtkod koji se izvršava na virtuelnoj mašini za Mikrojavu. Sintaksno i semantički ispravni Mikrojava programi su definisani specifikacijom [MJ].

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 sentence. Tokom parsiranja Mikrojava programa potrebno je na odgovarajući način omogućiti i praćenje samog procesa parsiranja na način koji će biti u nastavku dokumenta detaljno opisan. 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. Svi relevantni pomoćni materijali za izradu projekta se mogu pronaći na sajtu predmeta ili u okviru sekcije Prilog ovog dokumenta.

1. Genersianje koda

Da bi moglo da se generiše kod mora da se prvo prebaci u root direktorijum projekta i otvori u tom folderu komanda linija.

Komande**:**

**ant lexerGen-** generise sve potrebne klase potrebne za lekser

**ant compile** -generise sve potrebne klase potrebne za parsiranje koda

**ant testRun-** parsiraće defaultni program koji koristi test\test301.mj kao putanju ka fajlu i kreira fajl program.obj ako uspešno parsira

**ant testRun -Darg0=’’program\_path’’-** parsira program na zadatoj putanji i kreira fajl program.obj ako uspešno parsira

**ant runObj-** izvršava fajl program.obj kreiran prethodnim komandama

1. TESTOVI

Sintaksni testovi:

**syntax\_err1.mj** – testira ko se unese deo teksta koji ne može biti prepoznat kao kod

**syntax\_recovery.mj**-testira sve načine oporavka od grešaka.

Semantički testovi:

**sem\_err1.mj**-testira semantiče uslove na strani **5** mikrojave dodatka pdf-a

**sem\_err2.mj**-testira semantiče uslove na strani **6** mikrojave dodatka pdf-a

**sem\_err3.mj**-testira semantiče uslove na strani **7** mikrojave dodatka pdf-a

**sem\_detection.mj**-testira detekciju korišćenja promenljivi, pozivanje funkcija, pravljenje objekata klasa etc.

Generisanje koda:

**fibonacci.mj-** program koji za dat unos vraća fibonacijevu vrednost koja je realizovana

rekurzivno.

**program\_p.mj,programDomaci1.mj,programDomaci2.mj –** primeri dati na pdf-ovima o postavci i specijikaciji projekta.

**test\_zadatakx.mj**- 4 primera sa dodacima koji simuliraju zadatke koji je profesor radio na vežbama.

**test30x**.**mj**- 3 primera sa dodacima koji su uzeti sa sajta predmeta.

**tutorijal\_test.mj-** testira razne delove generisanja koda.

1. Novo Uvedene Klase

Paket **src**\**rs.ac.bg.etf.pp1:**

**CaseNameList.**java: koristi se da se pamti informacije o korišćenim celobrojnim vrednostima grana i proverava da li je neka celobrojna vrednost već bila iskorišćena

**CodeGenerator**.java: proširuje visitorAdapter i iplementira visit metode u kojima će se generisati kod.

**DesignatorObj**.java: služi da sadrži dodatne informacije vezane sa objekt čvorove ako ikad zatreba.

**MJDumpSymbolTableVisitor.java:** proširuje DumpSymbolTableVisitor da bi se dodalo obradu bool tipova i da se razreše neki problemi u prošloj implementaciji poput rekurzije ako klasa ima polje tipa samog sebe.

**MJTab.**java:proširuje Tab klasu da se dodaju bool tipovi u univerzum i implementiraju funkcije neophodne za proveru kompatibilnosti pri dodeljivanju, provere jednakosti tipova etc.

**StructList.java:** koristi se za skladištenje tipova stvarnih argumenta funkcije koji će se posle upoređivati sa formalnim argumentima da se proveri da li su kompatibilni pri dodeli.

Paket **test**\**rs.ac.bg.etf.pp1:**

**MJTest.java:** klasa koja izvršava leksičku analizu ulaznog programa.

**Compiler.**java: klasa koja izvršava leksičku ,sintaksnu i semantičku analizu ulaznog programa i u slučaju uspešnog parsiranja vraća program.obj koji može da se pokrene Mikrojava virtualnom mašinom.

Paket **rs.ac.bg.etf.pp1.visitor:**

**CountVisitor.java:** klasa koja se koristi za prebrojavanje lokalnih parametara funkcije ili formalnih parametara funkcije etc.