**Elektrotehnički fakultet u Beogradu Katedra za računarsku tehniku i informatiku**

*Predmet: Programski prevodioci 1*

*Nastavnik: dr Dragan Bojić, redovni profesor*

*Asistenti: mast. inž. Maja Vukasović, mast. inž. Mihajlo Ogrizović*

*Školska: 2022/2023.*

*Ispitni rok: Januarsko-februarski ispitni rok*

*Datum: 8.12.2022*

**Projekat**

**-Kompajler za Mikrojavu-**

Student: Mihailo Milenković

Opis zadatka

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. 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 korišćenih komandi

1.Obrada prve faze i generisanje leksičkog analizatora za prepoznavanje tokena navedenih u mjlexer.flex fajlu.

**<target** **name=***"lexerGen"* **depends=***"delete"***>**

**<java** **jar=***"lib/JFlex.jar"* **fork=***"true"***>**

**<arg** **value=***"-d"***/>**

**<arg** **value=***"./src/rs/ac/bg/etf/pp1"***/>**

**<arg** **value=***"spec/mjlexer.flex"***/>**

**</java>**

**</target>**

2. Obrada druge faze na osnovu mjparser.cup fajla proveravav se sintaksna ispravnost programa i generiše buildtree. Ukoliko je parsiranje neuspešno prijavljuje se greška.

**<target** **name=***"parserGen"* **depends=***"delete"***>**

**<java** **jar=***"lib/cup\_v10k.jar"* **fork=***"true"***>**

**<arg** **value=***"-destdir"***/>**

**<arg** **value=***"src/rs/ac/bg/etf/pp1"***/>**

**<arg** **value=***"-ast"***/>**

**<arg** **value=***"src.rs.ac.bg.etf.pp1.ast"***/>**

**<arg** **value=***"-parser"***/>**

**<arg** **value=***"MJParser"***/>**

**<arg** **value=***"-buildtree"***/>**

**<arg** **value=***"spec/mjparser.cup"***/>**

**</java>**

**</target>**

3. Kompajliranje koda

**<target** **name=***"compile"* **depends=***"repackage"***>**

**<javac** **srcdir=***"src/rs/ac/bg/etf/pp1"* **includeantruntime=***"false"***>**

**<classpath>**

**<pathelement** **path=***"lib/JFlex.jar"***/>**

**<pathelement** **path=***"lib/cup\_v10k.jar"***/>**

**<pathelement** **path=***"lib/log4j-1.2.17.jar"***/>**

**<pathelement** **path=***"lib/symboltable-1-1.jar"***/>**

**<pathelement** **path=***"lib/mj-runtime.jar"***/>**

**</classpath>**

**</javac>**

**</target>**

4. Pokretanje generisanog .obj fajla iz konzole sa standardnim ulazom preusmerenim na .txt fajl, pozicionirati se u lib fajl projekta

java -cp mj-runtime.jar rs.etf.pp1.mj.runtime.Run ../test/testing302.obj < ../test/input302.txt

5.Debagovanje rešenja uz pomoć prikaza expression stack-a nakon izvršavanja svake instrukcije

java -cp mj-runtime.jar rs.etf.pp1.mj.runtime.Run ../test/testing302.obj < ../test/input302.txt -debug

Opis test primera

1. A1 – testiranje osnovnih računskih operacija, read i print funkcije i osnovnih operacija sa nizovima (kreiranje i dodela vrednosti).

2. A2 – testiranje nizova i višestruke dodele vrednosti elementima pomoću niza.

3. A3 – testiranje prekoracenja opsega prilikom visestruke dodele vrednosti elementima pomoću niza.

4. B1 – testiranje while i foreach petlji, ugneždavanje petlji.

5. B2 – testiranje continue, break, if i else naredbi.