**Programski prevodioci 1**

Kompajler za programski jezik Mikrojavu

**Boško Ćurčin 549/16**

**PROJEKAT:**

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.

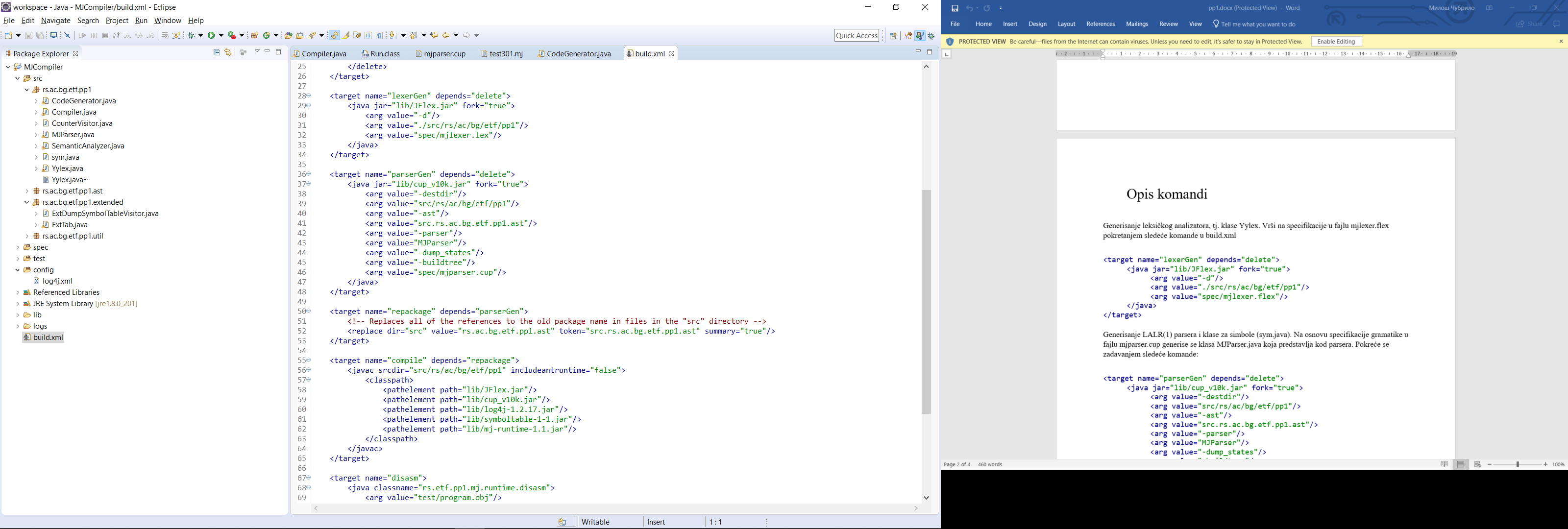
**Sintaksni analizator** ima zadatak da utvrdi da li izdvojeni tokeni iz izvornog koda programa mogu formiraju gramatički ispravne sentence. Tokom parsiranja Mikrojava programa potrebno je na odgovarajući način omogućiti i praćenje samog procesa parsiranja. Nakon parsiranja sintaksno ispravnih Mikrojava 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.

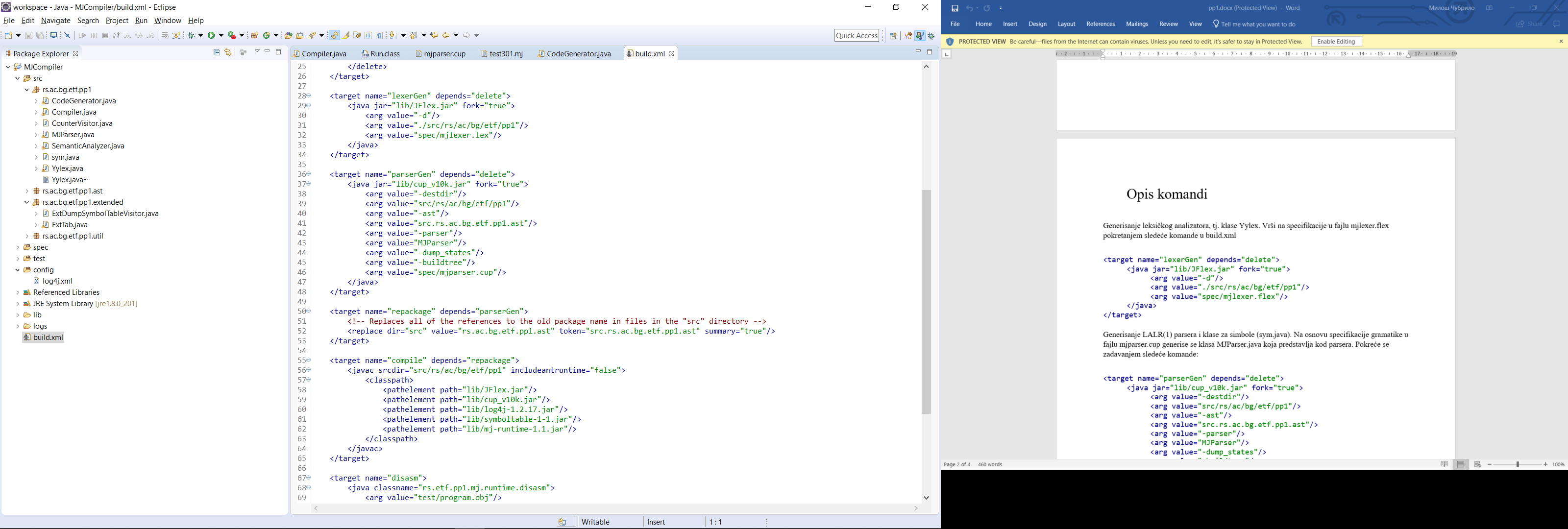
**Generator koda** prevodi sintaksno i semantički ispravne programe u izvršni oblik za odabrano izvršno okruženje Mikrojava VM. Generisanje koda se implementira na sličan način kao i semantička analiza, implementacijom metoda koje posećuju čvorove.

**KOMANDE**

Generisanje klase **Yylex** koja predstavlja leksicki analizator, na osnovu mjlexer.flex specifikacije:

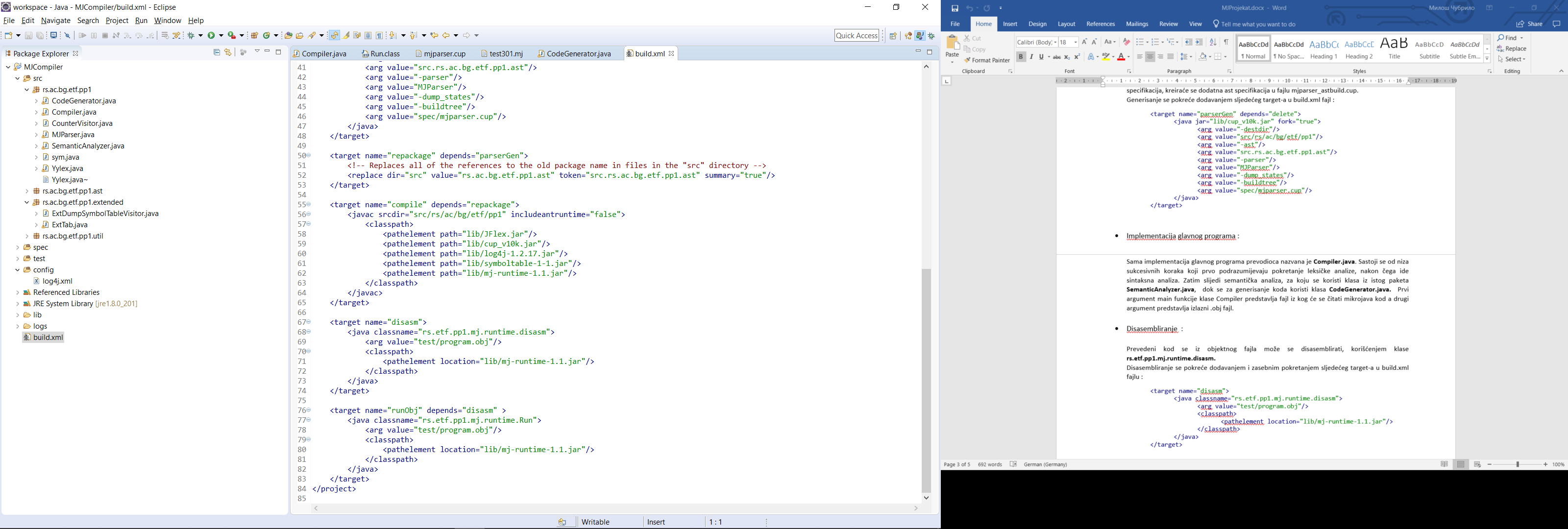


Generisanje klase **MJParser** (LALR(1) parser) I **sym** (klase za simbole), na osnovu specifikacije **mjparser.cup**:

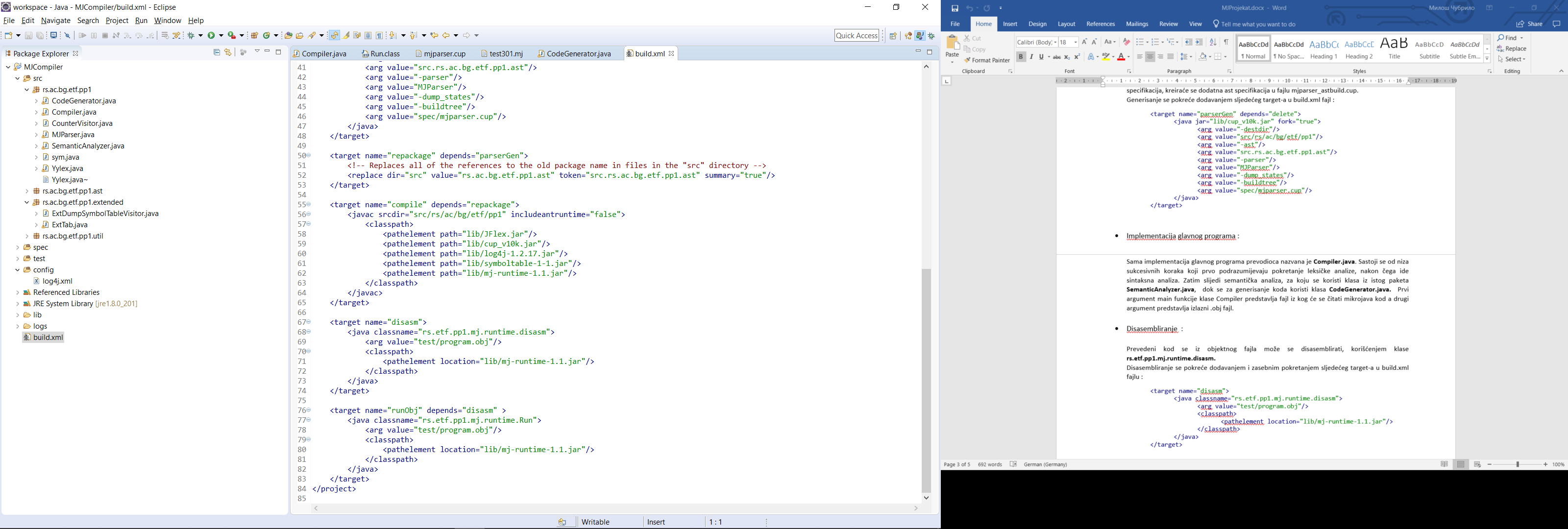


Nakon toga, vrši se leksička i sintaksna analiza. Ukoliko uspešno prođu, prelazi se na semantičku analizu obilaskom apstraktnog sintaksnog stabla generisanog u prethodnom koraku. Obilazak je realizovan u klasi **SemanticPass**. Nakon toga se vrsi generisanje koda, za sta se koristi klasa CodeGenerator.

Dobijeni kod se moze disasemblirati pomocu naredbe:



Na kraju se moze I pokrenuti program (na MJ virtuelnoj masini) naredbom:



**TESTOVI**

Dodata 3 test primera :

* Test1 – potpuno sintaksno i semantički ispravan program
* Test2 – sintaksno neispravan kod koji demonstrira oporavak od grešaka
* Program – javni test preuzet sa sajta predmeta

Svi test primeri se odnose na nivo A projektnog zadatka.