|  |
| --- |
| Programski prevodioci 1 |
| Projekat |
| Kompajler za Mikrojavu |

|  |
| --- |
| Draga Tišma 2019/0415 |

**Postavka zadatka:**

Zadatak je zamišljen kao realizacija kompajlera za programski jezik Mikrojava, korišćenjem alata JFlex i AST-CUP. On ima 4 osnovne funkcionalnosti u vidu leksičke analize, sintaksne analize, semantičke analize i generisanja koda.

**Opis komandi za generisanje java koda alatima, prevođenje koda kompajlerom, pokretanje i testiranje rešenja**

Za pokretanje aplikacije koristi se ant alat. Za njegovo korišćenje potrebno je konfigurisati Environment Variables. U User delu dodaju se promenljive ANT\_HOME i JAVA\_HOME koje ukazuju na odgovarajuće foldere. U system delu u okviru promenljive Path treba dodati iste putanje do bin foldera.

U okviru samog projekta fajl build.xml se koristi za pokretanje svih potrebnih alata.

Komande za generisanje Yylex klase, koristeci fajl mjlexer.flex i JFlex alate.

**<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>**

Komande za generisanje klase MJParser, na osnovu mj-cup specifikacije i AST-CUP alata

**<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=***"-dump\_states"***/>**

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

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

**</java>**

**</target>**

Nakon ovoga sledi semanticka analiza koja je ostvarena pomocu klase SemanticAnalyzer. Analiza se vrši obilaskom apstraktnog sintaksnog stabla koje je generisano prethodnim koracima.

Sledeca faza je generisanje koda, implementirano pomocu klase CodeGenerator.

Kod koji iskucamo se pokrece iz klase Compiler, disasemblira iz objektnog fajla i konacno pokrece komandama:

**<target** **name=***"disasm"***>**

**<java** **classname=***"rs.etf.pp1.mj.runtime.disasm"***>**

**<arg** **value=***"test/program.obj"***/>**

**<classpath>**

**<pathelement** **path=***"lib/mj-runtime-1.1.jar"***/>**

**</classpath>**

**</java>**

**</target>**

**<target** **name=***"runObj"* **depends=***"disasm"* **>**

**<java** **classname=***"rs.etf.pp1.mj.runtime.Run"***>**

**<redirector** **input=***"test/resources/input.txt"* **output=***"test/resources/output.txt"* **alwayslog=***"true"***/>**

**<arg** **value=***"test/program.obj"***/>**

**<classpath>**

**<pathelement** **path=***"lib/mj-runtime-1.1.jar"***/>**

**</classpath>**

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

**</java>**

**</target>**

**Kratak opis priloženih test primera**