\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Електротехнички факултет у Београду

Катедра за рачунарску технику и информатику

***Предмет****:* Програмски преводиоци 1 (13Е114ПП1)

***Наставници***: др Драган Бојић

***Асистенти:*** Маја Вукасовић, маст. инж.

Кристијан Жижа, маст. инж.

***Испитни рок:*** јануар 2021.

***Датум:*** 11.1.2021.

***Кандидат:*** Марко Кујунџић

***Број индекса:*** 418/2016

***Тема:*** Компајлер за програмски језик МикроЈава

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Опис решења

Решење представља имплементацију малог, али функционалног преводиоца за програмски језик МикроЈава, који представља упрошћење програмског језика Јава.

Преводилац се састоји из 4 независне целине: Лексичке анализе, Синтаксне анализе, Семантичке анализе и Генерисања кода, о којима ће бити више речи у наставку текста.

Као излаз преводиоца добија се МикроЈава бајткод, односно објектни фајл, који се може извршити на МикроЈава виртуелној машини.

# Структура решења

Превођење програма написаног на језику МикроЈава се састоји из 4 дела.

Лексичка анализа служи да се улазни програмски фајл издели на токене, чија се исправност проверава у фајлу mjlexer.lex. Фајл представља низ лексичких правила, попут дозвољених токена у овом програмском језику, уз механизан детектовања потенцијалних грешака, као што су унос непостојећег токена.

Синтаксна анализа служи за генерицаље синтаксног стабла програма, односно стабла које се добија из mjparser.cup спецификације парсера. У том фајлу се налази цела граматика програма, односно смене којима се замењује свака линија програмског кода. Постоји и опоравак од синтаксних грешака, који служи да обједини грешке истог типа, па тек онда да прекине извршавање програма.

Семантичка анализа служи за семантичку проверу улазног кода, односно за испитивање логике написаног кода над истим генерисаним синтаксним стаблом.

Генерисање кода служи за генерисање програмских инструкција које се потом извршавају на МикроЈава виртуелној машини.

Превођење и покретање програма

У циљу превођења програма користи се алат Ant, који служи да изгенерише Лексер, Парсер а потом и Синтаксни анализатор. Улазни програмски фајл има екстензију .mj, а после Синтаксне анализе генерише се .obj фајл, који се прослеђује Генератору кода. Аргументи програма представљају путању до улазног програмског фајла, као и путању до излазног објектног фајла, нпр. *"test/program.mj" "test/program.obj".*

После тога, потребно је покренути и Ant target ProgramObj или main методу класе Run.java. У случају да се покреће main метода те класе, као аргумент је потребно проследити путању до објектног фајла програма који се извршава, нпр. *"test/program.obj"*.

Класе коришћене у решењу

Приликом решавања задаток проблема, коришћене су додатне класе:

* CounterVisitor.java : за бројање коришћења сваког типа података
* CodeGenerator.java : генератор кода за МикроЈава виртуелну машину
* MJParser.java : парсер добијен коришћењем Cup алата
* MyDumpSymbolTableVisitor.java : класа која служи за обилазак стабла
* SemanticAnalyzer.java : за семантичку анализу стабла
* Yylex.java : лексер добијен коришћењем алата Jflex
* SymTab.java : проширење класе Tab.java, због рада са типом bool

Тестови

У реализацији решења коришћени су следећи тестови:

* program.mj : јавни тест за А ниво
* syntaxErrorRecovery1..4.mj : тестови који служе да представе опоравак од различитих врста грешака у процесу синстаксне анализе
* semanticErrors.mj : тест са већим бројем неисправних и исправник језичких семантичких конструкција