**Електротехнички факултет – Универзитет у Београду**

A picture containing logo

Description automatically generated**Програмски преводиоци 1**

**Студент: Михајло Никитовић 2018/0164**

Кратак опис пројектног задатка

Основни задатак овог пројекта је био да се изради компајлер за програмски језик Микројава (MJCompiler). Пројекат је био подељен у четири фазе: лексичка анализа, синтаксна анализа, семантичка анализа и генерисање кода. Резултат овог пројекта био је компајлер који је омогућавао превођење синтаксно и семантички исправних Микројава програма у Микројава бајткод који се извршава на виртуелној машини за Микројаву.

Лексичка анализа

Задатак лексичког анализатора био је препознавање токена у облику кључних речи наведених у фајлу mjlexer.flex. Спецификација .flex је претворена у имплементацију лексера на програмском језику Јава користећи алат JFlex. Сваки токен тражи свог идентичног пара, а ако лексичка анализа прође без грешке, прелази се на следећу фазу, тј. синтаксну анализу. У случају грешке, исписује се знак који није препознат, редни број линије у програму где је откривена грешка и колона у којој је грешка идентификована.

Синтаксна анализа

Задатак синтаксног анализатора био је провера да ли написани програм одговара граматици наведеној у фајлу mjparser.cup. За имплементацију парсера користи се алат синтаксних анализатора AST-CUP. Након успешне извршене лексичке анализе, у овој фази сваки токен мора се подударати са граматиком написаном у mjparser.cup фајлу. На тај начин се проверава да ли написани скуп језичких лексема може формирати исправну граматику. Уколико компајлер наиђе на грешку у овој фази, може се извршити опоравак од грешке.

# Семантичка анализа

Задатак ове фазе био је обилазак претходно генерисаног синтаксног стабла из фазе 2. Нова класа SemanticAnalyzer.java биће обезбеђена методама visit() које ће при обиласку стабла вршити проверу да ли је дошло до неке грешке или је потребно да одређени чвор у стаблу буде одређеног типа. Та писана правила су дата у самој поставци пројектног задатка и поштована су у овој имплементацији компајлера. У овој фази се врши и попуњавање табеле симбола.

Генерисање кода

Ово је уједно и последња фаза у којој се подразумева да је написан лексички, семантички и синтаксно исправан програм. Задатак ове фазе био је поново обилазак апстрактног стабла и поновно писање метода унутар класе CodeGenerator.java. Резултат ове фазе је генерисан бајткод.